



**REPUBLIC OF TURKEY
AKSARAY UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES**

**DEPARTMENT OF ELECTRICAL-ELECTRONIC AND COMPUTER
ENGINEERING**

**DAMAGE DETECTION IN AIRCRAFT ENGINE BORESCOPE
INSPECTION USING DEEP LEARNING**

Ph.D. THESIS

İsmail UZUN

SUPERVISOR

Prof. Dr. Mehmet Reşit TOLUN

AKSARAY, 2023

Aksaray University, Graduate School of Natural and Applied Sciences Approval,
İsmail UZUN, Doctor of Philosophy with thesis, Student No. 162353602 successfully
presented the Doctor of Philosophy Thesis titled “**DAMAGE DETECTION IN
AIRCRAFT ENGINE BORESCOPE INSPECTION USING DEEP
LEARNING**”, prepared after satisfying all the requirements identified in the
concerned bylaws, before the jury whose signatures are affixed below.

Advisor: Prof. Dr. Mehmet Reşit TOLUN

Çankaya University

I approve that this thesis is Doctor of Philosophy in scope and quality.....

Member: Prof. Dr. Ferda Nur ALPASLAN

Middle East Technical University

I approve that this thesis is Doctor of Philosophy in scope and quality.....

Member: Assoc. Prof. Dr. İrfan SÜRAL

Osman Gazi University

I approve that this thesis is Doctor of Philosophy in scope and quality.....

Member: Asst. Prof. Dr. Filiz SARI

Aksaray University

I approve that this thesis is Doctor of Philosophy in scope and quality.....

Member: Asst. Prof. Dr. Hasan DEMİR

Aksaray University

I approve that this thesis is Doctor of Philosophy in scope and quality.....

Date of Defence: 16 / 01 / 2023

I agree that this thesis which is accepted by the jury, has fulfilled the requirements for
being a Doctor of Philosophy Thesis.

.....

Assoc. Prof. Dr. Mehmet Ali HINIS

Director of Natural and Applied Sciences

PLAGIARISM PAGE

I hereby declare that all information in this PhD thesis has been obtained and presented in accordance with academic rules ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

I declare that I will bear all moral and legal responsibility that may arise in the event that a situation contrary to this statement I have made about my thesis is detected by the Institute without being bound by a certain time.

İsmail UZUN

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest gratitude to my supervisor, Prof. Dr. Mehmet Reşit TOLUN, for his trust, his continuous guidance throughout the research, his patience as well as for his motivating behavior to go an extra mile even in difficult situations. In addition, I would like to express my sincere thanks to the committee members Prof. Dr. Ferda Nur ALPASLAN and Asst. Prof. Dr. Filiz SARI for their valuable ideas and guidance throughout the research journey, as well as Assoc. Prof. Dr. İrfan SÜRAL and Asst. Prof. Dr. Hasan DEMİR for their valuable contributions.

Finally, I would like extend my thanks to my brothers Yusuf UZUN and Naim UZUN for their encouragment and for being a role model to me.

The numerical calculations reported in this paper were fully performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources).

I dedicate this dissertation to my little Zeynep Eva, my wife Ipek and to my big family.

İsmail UZUN
AKSARAY, 2023

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	i
TABLE OF CONTENTS	ii
ABSTRACT	iii
ÖZET	iv
LIST OF FIGURES	v
LIST OF TABLES	vi
ABBREVIATIONS	vii
GLOSSARY	ix
1. INTRODUCTION	1
1.1 Turbofan Engines	5
1.2 Borescope Inspection and Engine Damages	8
1.3 Literature Review	10
2. DEEP CONVOLUTIONAL NEURAL NETWORKS	17
2.1 Neural Networks.....	17
2.2 Convolutional Neural Networks.....	21
2.3 Deep Convolutional Neural Networks	22
3. MATERIALS and METHODS	24
3.1 Data Collection and Cleaning.....	24
3.2 Data Processing and Labeling	26
3.3 Deep Learning Model.....	32
3.4 Training	34
3.5 Overfitting Handling	35
3.5.1 Data augmentation.....	35
3.5.2 Dropout.....	36
3.5.3 L2 regularization.....	36
3.6 Evaluation Metrics	37
3.7 Hardware and Software Environment	39
4. RESULTS	41
5. CONCLUSION and FUTURE WORK	51
REFERENCES	54
APPENDIX LIST	60
APPENDIX A-1: TFOD API pipeline config file faster_rcnn_inception.config .	61
APPENDIX A-2 : TFOD API labelmap.pbtxt	66
APPENDIX B : SLURM script file	67
APPENDIX C-1 : Complex image and label samples	68
CURRICULUM VITAE	71

Ph.D. THESIS

**DAMAGE DETECTION IN AIRCRAFT ENGINE BORESCOPE
INSPECTION USING DEEP LEARNING**

İsmail UZUN

**Aksaray University
Graduate School of Natural and Applied Sciences
Department of Electrical-Electronic and Computer Engineering**

Supervisor: Prof. Dr. Mehmet Reşit TOLUN

ABSTRACT

Aircraft engine inspection is a key pillar of aviation safety by maintaining adequate performance standards to ensure the airworthiness of an engine. In addition, it is vital for asset value retention. Borescope inspection is currently the most widely used aircraft engine visual inspection method. However, borescope inspection is a time consuming, subjective, and complex process which heavily depends on the experience of the inspector as well as on their concentration level during inspection. On the other hand, cost saving of airlines and maintenance, repair, and overhaul (MRO) centers expose pressure and workload on inspectors. These make an automated system to support damage detection during borescope inspection necessary to avoid potential risks.

Deep learning has found very wide application and has proven to be very successful during the last 10-15 years in the image recognition domain. In this research, we suggest a deep learning based automated damage detection framework from aircraft engine borescope inspection images. Faster R-CNN based deep learning model with Inception v2 feature extractor is utilized for the architecture. Due to the limited number of images, data augmentation and other overfitting methods are employed. The framework supports crack, burn, nick and dent damage types across all modules of turbofan engines. It is trained and validated with moderate to complex borescope images obtained from the field.

The framework achieves 92.05% accuracy for nick or dent, 92.64% for crack and 81.14% for burn damage classes. Moreover, it achieves 88.61% average accuracy.

Keywords: Borescope inspection, aircraft engine, damage detection, defect detection, deep learning, Faster R-CNN, crack, nick, dent, burn.

January, 2023; 71 pages

DOKTORA TEZİ

UÇAK MOTORLARININ BAROSKOPİK İNCELEMESİNDE DERİN ÖĞRENME KULLANILARAK HASAR TESPİTİ

İsmail UZUN

**Aksaray Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı**

Danışman: Prof. Dr. Mehmet Reşit TOLUN

ÖZET

Uçak motorlarının muayenesi motorun uçuşa elverişlilik için gerekli standartların sağlanmasını garanti etmesi sebebiyle uçuş güvenliğinin temel direklerinden biri sayılır. Ayrıca uçakların değerinin korunmasında da hayati bir öneme sahiptir. Baroskop ile muayene, günümüzde uçak motoru muayenelerinde en sık kullanılan görsel muayene yöntemidir. Ancak baroskopik muayenenin başarısı muayeneyi yapan denetimcilerin tecrübesine ve o anki konstanstrasyonlarına oldukça bağlı, zaman alan, sübjektif ve karmaşık bir işlemdir. Diğer yandan, havayolu şirketleri ve bakım, onarım merkezlerinin tasarruf tedbirleri denetimciler üzerinde baskı ve iş yükü oluşturmaktadır. Bu sebeplerden dolayı, insan faktöründen kaynaklı potansiyel riskleri en aza indirmek için baroskopik incelemelerde hasar tespitine destek olacak otomatik hasar tespit sistemlerine ihtiyaç vardır.

Derin öğrenme son 10-15 yıl içerisinde farklı disiplinlerde çok geniş kullanım alanı bulmuş ve özellikle görüntü tanıma alanında oldukça başarılı sonuçlar vermiştir. Bu çalışmada, uçak motorlarının baroskopik incelemelerinden elde edilmiş fotoğraflardan otomatik hasar tespiti yapan, derin öğrenmeye dayalı bir sistem önermekteyiz. Sistem, Faster R-CNN mimarisine dayalı ve Inception v2 öznitelik çıkarıcıya sahip bir model kullanmaktadır. Veri miktarının yetersizliğinden dolayı çalışmada veri çoğaltma ve diğer aşırı uyum önleme yöntemleri kullanılmıştır. Önerilen sistem turbofan motorların her bölümünde çatlak, çentik, oyuk ve yanık hasar türleri tespitini desteklemektedir. Sistem sahadan elde edilmiş, orta ve yüksek seviye karmaşıklıkta görüntülerle eğitilmiş ve test edilmiştir.

Önerilen sistem çentik, oyuk hasar sınıfı için %92.05, çatlak için %92.64 ve yanık hasar sınıfı için %81.14 ortalama hassasiyet (doğruluk) değerlerine erişmiştir. Sistemin genel ortalama hassasiyeti ise %88.61'e ulaşmıştır.

Anahtar Kelimeler: Baroskopik inceleme, uçak motoru, hasar tespiti, derin öğrenme, Faster R-CNN, çatlak, çentik, oyuk, yanık

Ocak, 2023; 71 sayfa

LIST OF FIGURES

Figure 1.1. Fiberscope on the left and video borescope on the right	2
Figure 1.2. Turbofan engine	3
Figure 1.3. High level overview of the workflow with the proposed framework.....	5
Figure 1.4. Schematic diagram of a 2 spool turbofan engine.....	7
Figure 1.5. Sample borescope images with a) Burn and Crack b) Coating loss c) Dent d) Nick e) Corrosion f) Erosion damages.....	9
Figure 2.1. Biological neuron and artificial neuron model	18
Figure 2.2. Common activation functions.....	19
Figure 2.3. Simple neural network with input, one hidden and an output layer.....	21
Figure 2.4. A sample CNN for an image classification task.....	22
Figure 3.1. Overview of proposed automated damage detection framework development and deployment process.....	24
Figure 3.2. Sample borescope image with two cracks	26
Figure 3.3. Engine serial numbers are removed from the top left corner of both images for data anonymization	27
Figure 3.4. Sample borescope inspection image labeled with “nick” label and a bounding box around the damage using labelImg software.....	28
Figure 3.5. Sample images with multiple damages and overlapping damage areas..	29
Figure 3.6. A big crack is labeled with two bounding boxes in the upper images (a) vs one bounding box in the below image (b).	30
Figure 3.7. Tensorflow 1 Object Detection API detection speed and mAP comparisational results table for various models trained on COCO set. ...	33
Figure 3.8. Faster R-CNN based damage detection model architecture	33
Figure 3.9. Intersection Over Union (IOU).....	38
Figure 3.10. Sample Precision x Recall curve with all points interpolation..	39
Figure 4.1. Without transfer learning (a) average precision (AP) of burn, crack, dent, nick damage classes and mean average precision (mAP).....	41
Figure 4.2. (a) AP and mAP of 4 damage classes crack, burn, nick dent at IoU=0.25 and confidence score=0.50.....	42
Figure 4.3. 4 image groups with nick/dent misclassifications.	43
Figure 4.4. (a) AP and mAP of 3 damage classes crack, burn, nick_or_dent at IoU=0.25 and confidence score=0.50 (b) Training loss at the left and validation (total) loss at the right.....	44
Figure 4.5. Detection samples of single damage per damage type.....	45
Figure 4.6. Multiple detection samples of the same damage type.....	46
Figure 4.7. Detection samples for multiple damages of different types.....	47
Figure 4.8. (a) AP and mAP of 3 damage classes crack, burn, nick_or_dent at IoU=0.25 and confidence score=0.50 when random vertical flip added (b) Training loss at the left and validation (total) loss at the right.....	48
Figure 4.9. Impact of image preprocessing and data augmentation methods on accuracy individual AP and mean AP values.....	49
Figure 4.10. Average precision when the model is trained/validated per class individually.....	50

LIST OF TABLES

Table 1.1. Top 10 delivered commercial turbofan engines in civil aviation history ...	6
Table 1.2. Component base learners of the three stacked ensembles.....	13
Table 3.1. Number of images that include each damage class and total number of labels per damage class in the data set.	31
Table 3.2. Image/Label Statistics	32
Table 3.3. Number of images and number of labels per training and validation set.	34
Table 3.4. TRUBA compute nodes used for development (Levrek) and training/validation (Barbun-cuda and Akya-cuda).....	40
Table 5.1. Comparision with similar studies	52



ABBREVIATIONS

AD	airworthiness directives
AI	artificial intelligence
AMM	aircraft maintenance manual
ANAC	Agência Nacional de Aviação Civil
AP	average precision
API	application programming interface
BMP	bitmap
BI	borescope inspection
CAA	civil aviation authority
CC	combustion chamber
CLACHE	contrast limited adaptive histogram equalization
CMS	channel mean subtraction
CNN	convolutional neural network
COCO	common object in context
CPU	central processing unit
DCNN	deep convolutional neural network
DL	deep learning
EASA	European Union Aviation Safety Agency
FAA	Federal Aviation Administration
FCNN	fully connected neural network
fwIU	frequency weighted intersection over union
GE	General Electric
GPU	graphics processing unit
GUI	graphical user interface
HPC	high pressure compressor
HPT	high pressure turbine
ILSVRC	ImageNet large scale visual recognition challenge
IAE	International Aero Engines
IoU	intersection over union
JPG	Joint Photographic Experts Group
LPC	low pressure compressor
LPT	low pressure turbine
mAP	mean average precision
MP	maintenance program
MPD	maintenance planning document
MRO	maintenance, repair and overhaul
NMS	non-maximum suppression
PA	pixel accuracy
RCNN	region-based convolutional neural network
P&W	Pratt & Whitney
RGB	red, green, blue
ReLU	rectified linear units
ROI	region of interest
RPN	region proposal network
SSD	singe shot multibox detector
TC	Transport Canada
TFOD API	Tensorflow Object Detection API
TPU	Tensor processing unit

TRUBA	Turkish National e-Science e-Infrastructure
VGG	Oxford Vision Geometry Group
YOLO	You Look Only Once



GLOSSARY

Burn: surface damage due to excessive heat. It is usually caused by improper fit, defective lubrication, or over-temperature operation.

Coating loss: chipping, flaking, blistering, peeling or oxidation of blade confined to coating.

Corrosion: loss of metal by a chemical or electrochemical action such as iron rust.

Crack: A line on the surface of material along which it has partial separation without breaking. Crack is usually caused by vibration, overloading, internal stresses, defective assembly, or fatigue. Depth may be a few thousandths, to the full thickness of the piece.

Cut (or missing piece)—loss of metal, usually to an appreciable depth over a relatively long and narrow area, by mechanical means, as would occur with the use of a saw blade, chisel, or sharp-edged stone striking a glancing blow.

Dent—a small, rounded depression in a surface usually caused by the part being struck with a rounded object. In dent there is no material loss, only deformation.

Erosion—loss of metal from the surface by mechanical action of foreign objects, such as grit or fine sand. The eroded area is rough and may be lined in the direction that the foreign material moved relative to the surface.

Nick—a sharp-sided gouge or depression with a V-shaped bottom, that is generally the result of careless handling of tools and parts. There is material loss with nick damage type.

1. INTRODUCTION

Civil aviation has been recently very popular with the advent of the technology, easy access to the expanding infrastructure, increasing will of travel due to growing middle class in the world's population, emergence of low-cost airlines and finally increasing shipment of goods via airlines. As a result of these factors, the size of the world's aircraft fleet has reached 25.900 aircraft in 2019 and is expected to reach 47.080¹ aircraft in 2041 (URL-1). Due to this huge number of aircraft flying, not only the traffic but also the technical wellbeing (airworthiness) of these aircraft needs to be under strict control for maintaining aviation safety. Civil Aviation Authority (CAA) of each country (called Sivil Havacılık Genel Müdürlüğü in Türkiye) performs technical and operational annual edits on the airline operators to ensure proper maintenance and operation of the aircraft. Additionally, the manufacturers, major players being Boeing, Airbus, Embraer, Bombardier, are dependent on the CAAs of the countries in which they produce their aircraft. Major CAAs are Federal Aviation Administration (USA), European Union Aviation Safety Agency (EU), Agência Nacional de Aviação Civil (Brasil), and Transport Canada (Canada).

The manufacturers specify certain maintenance actions to be followed for proper operations. This is generally called the Maintenance Program. Also, the Civil Aviation Authorities, which authorized the release of the aircraft product into the market, mandate certain inspections or modifications to correct the discrepancies found in the design. These are called Airworthiness Directives (AD). The Maintenance Program and Airworthiness Directives are the major requirements that the operators have to fulfil.

Aircraft engine (like CFM56, V2500-A5, PW 4000 etc.) inspection is a key pillar of inspection/maintenance procedure to ensure the airworthiness of an engine. This is performed by airlines, maintenance organizations, engine owners etc. for the visual assessment of the internal condition of aircraft engines to avoid any potential failures and incidents or accidents due to these. Current visual engine inspections are mainly performed using borescopes. Borescope itself is a tool used to inspect the internal parts

¹ Includes only commercial aviation which means aircraft for hire operated by companies to transport passengers and/or cargo. It does not include military or private aircraft

of the engine for damages such as cracks, stress fractures, corrosion, etc. without disassembling the engine (Vembar et al., 2005). The most common form of the borescope is the video borescope that gets the visual recordings through a camera at its top and conveys the visual image to a display (Figure 1.1). For this, borescope flexible tube with a camera on its top is injected through designated engine ports and moved across the engine parts whereas the captured video feed is analyzed (and recorded) by experts in real time for potential anomalies.



Figure 1.1. Fiberscope (Courtesy iFlex Fiberscope) (URL-2) on the left and video borescope (Courtesy iRis DVR 5) (URL-3) on the right.

The engines being inspected typically have multiple stages and each stage has many blades that must be inspected (Figure 1.2). Due to the high stress exerted on the material used in the engines, particularly the blades, any damage carefully needs to be detected and rectified. If not, the internal damage can escalate into engine shut down or any other catastrophic accident might happen. For this purpose, the manufacturers specify areas to be inspected, acceptable and non-acceptable damage limits, and any repair procedures as applicable. These are included in the Aircraft Maintenance Manual (AMM) created and periodically updated by the manufacturer. The operator cannot deviate from the AMM.

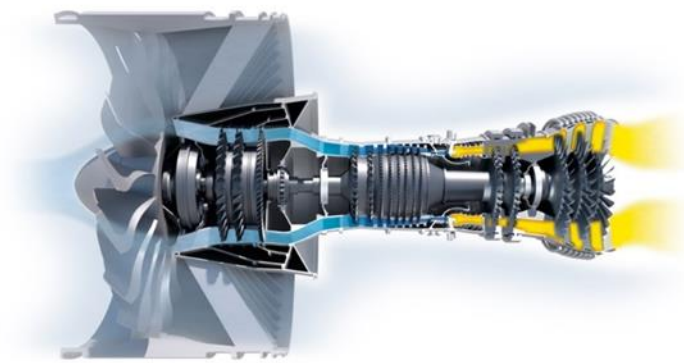


Figure 1.2. Turbofan engine (Henrich, 2021).

During the borescope inspection, in case a possible damage with non-acceptable damage limits is identified then the engine is sent to maintenance, operation, repair (MRO) facility where the engine shall be disassembled for further inspection. In some cases, this is a costly operation with considerable fraction of the engine list price (Wong et al., 2021).

Currently, borescope inspections are performed by experts who are trained on the inspection procedures, identification and classification of damages. Success of the inspection depends too much on the training, on the job experience as well as skills of the expert, therefore any damage might be missed or incorrectly assessed as a result of possible errors coming from human factors (Drury and Watson, 2001).

Beside these, in performing the inspection task, inspectors have to be sensitive to the efficiency (speed measure) and effectiveness (accuracy measure) (Nickles et al., 2019). This means inspectors frequently have to run against time to discover critical defects at early phases of the inspection. Typically, inspection teams do overtime to cope with this pressure. In addition, some inspection steps are performed at night shifts during no flight hours. On the other hand, inspectors are expected to accurately detect and classify all damages with different severities and different locations in the engine. Missing a critical damage can lead to a fatal result. All these can cause stress on the inspectors as well as low concentration levels due to physical tiredness (Nickles et al., 2019).

The purpose of this study is to develop an automated framework to support inspectors detect damages from borescope images in a quick and reliable way. Quick, in the sense that taking borescope inspection images it shall mark detected damages with bounding boxes within seconds, reliable in the sense that it shall detect as many damages as possible in given images. So, the objective is to try not to miss damages for the cost of false positives. The advantage of this framework would be to minimize errors due to human factors, speeding up the inspection process and in case of uncertainty to represent confidence score. The framework takes borescope images as input and provides bounding boxes along with confidence scores for detected damages. In a single image there can be multiple damages of the same damage class and/or different class. The framework supports crack, burn, nick or dent damage types. Initially, it was aimed to support more damage classes, however the lack of enough data for the other classes led to dropping support for other classes at this phase of the research. The framework supports damage detection across all aircraft engine modules, namely high pressure compressor (HPC), low pressure compressor (LPC), combustion chamber (CC), high pressure turbine (HPT) and low pressure turbine (LPT).

Deep learning (DL), in particular Deep Convolutional Neural Networks (DCNN) in the recent years have proven to be successful in object detection tasks. Inspired by this success, we suggest a Faster Region-based Convolutional Neural Network (Faster R-CNN) (Ren et al., 2015) based framework using Inception V2 (Szegedy et al., 2016) feature extractor. The network is trained on top of the weights pre-trained with COCO data set (Lin et al., 2014) and fine-tuned using borescope images. Tensorflow Object Detection API (URL-4) is used for creating, training and validating the model.

Borescope images, videos and inspection reports used for obtaining training as well as validation images/data are obtained from three private airlines in and outside Türkiye, from the internet and through contacts in the industry.

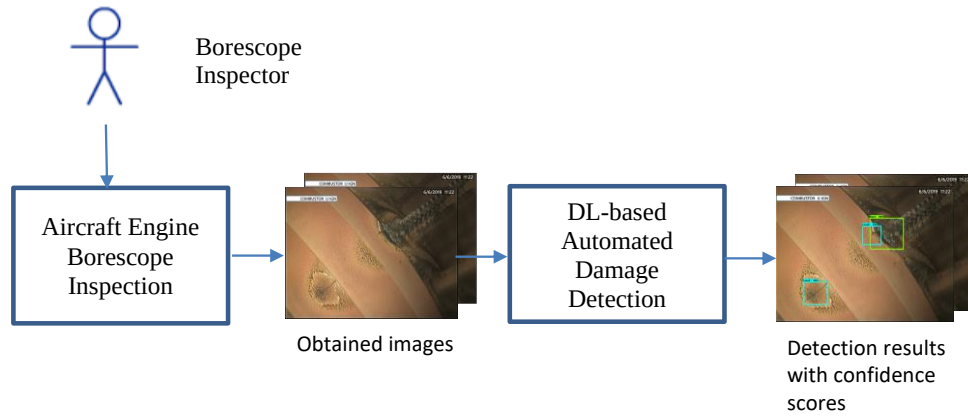


Figure 1.3. High level overview of the workflow with the proposed framework.

1.1 Turbofan Engines

Turbofan engines are the most widely used engine types in commercial aircraft fleets. “A turbofan engine, sometimes referred to as a fanjet or bypass engine, is a jet engine variant which produces thrust using a combination of jet core efflux and bypass air which has been accelerated by a ducted fan that is driven by the jet core.” (URL-5)

Early version of aircraft engines were simple piston engines with propellers which are found today exclusively in some small private aircraft (Henrich, 2021). Then, the powered version, turboprop engine arrived which is composed of a propeller driven by a gas turbine through a gearbox. That was followed by the revolutionary turbojet engine which made the foundation for the future jet engines. Turbojet engines are air breathing engines with a turbine. In turbojet engines all in-taken air is compressed and passed through the combustion chamber as well as turbines to produce thrust, the force that pushes the aircraft forward. This is a key difference between turbojet and turbofan engines. In turbofan engines, part of the in-taken air is bypassed and then has been accelerated by a ducted (channeled) fan driven by the jet core. In this way, both bypassed (cold) air and burned (hot) air contribute the thrust. Bypassed cold air also helps to reduce the noise by surrounding the hot gases. The ratio between the amount of bypassed air vs air passed through the engine core is called bypass ratio. Bypass ratio is an important parameter to describe a turbofan engine. A bypass ratio of 6 (6:1), for example, means that 6 times more air passes through the bypass duct than the amount that passes through the combustion chamber (URL-6). Modern commercial aircrafts use high bypass ratio turbofan engines.

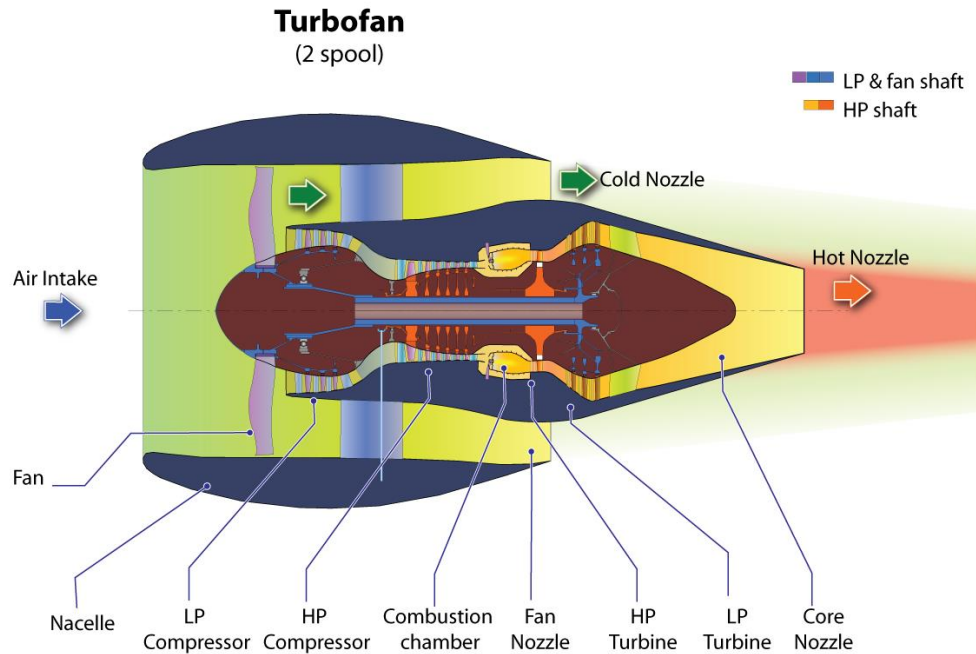
The major aircraft engine manufacturers are CFM International which is a joint venture between General Electric (GE) and Safran Group, Pratt & Whitney (PW), General Electric, International Aero Engines (IAE) which is a partnership of Pratt & Whitney, Rolls-Royce, Japanese Aero-engine Corporation and MTU Aero Engines of Germany and finally Rolls Royce (URL-7) (Morrison, 2015.). According to the report released by Statista (Burgueño Salas, 2022), CFM International had 39%, Pratt & Whitney 35%, GE Aviation 14% and finally Rolls-Royce 12% market share of the commercial aircraft engines in 2020.

Table 1.2 provides top 10 delivered commercial turbofan engines in civil aviation history up to 2015 based on Flightglobal's Ascend Fleets database (Morrison, 2015)

Table 1.1. Top 10 delivered commercial turbofan engines in civil aviation history.

Manufacturer	Engine	Aircraft using the engine
CFM International	CFM56	Boeing 707, 737 Airbus A320, A340
P&W	JT8D	Boeing 727-100, 737 McDonnell Douglas MD-80, DC-9
GE	CF6	Douglas DC-10, Boeing 747-400, 767, Airbus A330
IAE	V2500	Airbus A320
GE	CF34	Bombardier CRJ200, CRJ700, CRJ900, Comac ARJ21
P&W	JT3D	Boeing 707, McDonnell Douglas DC-8
Soloviev	D-30	Ilyushin Il-62M, Il-76, Tupolev Tu-154
P&W	PW4000	Boeing 747-400, 767, 777, MD-11, Airbus A300, A310 and A330
Motor Sich	AI-25	Yakovlev Yak-40
Rolls Royce	RB211	Boeing 747-400, 757 767, Tupolev Tu-204

Although the number of spools (for example Rolls-Royce RB211 chose a three-spool design whereas CFM56 has two spools), number of stages at each module etc differ between vendors and engine types the generic working principle and modules of a turbofan engine are illustrated in Figure 1.4.



kashkhan/tfan-schematic-kk-20090106.png

Figure 1.4. Schematic diagram of a 2 spool turbofan engine (URL-8).

The air intake happens through the fan rotor, the blade wheel at the front. Intaken air follows two paths: as mentioned above, majority of the cold air is not combusted, instead it bypasses the engine core and is expanded through the cold nozzle. The rest of the air goes through the low pressure compressor (LPC) which does a precompression. Then compressed air goes into the high pressure compressor (HPC) which compresses the air further. Both LPC and HPC consist of blades and vanes to direct the air. The compressed air then goes into the combustion chamber (CC) which has nozzles that mix the air with the fuel. In CC, igniter triggers burning of the fuel and air mixture which causes an explosion type of gas expansion. CC reaches the temperature of approximately 1,700 degrees Celsius. The expanded gas leaves the CC with high energy towards the high pressure turbine (HPT). CC consists of nozzles, liner, bulkhead and reflectors. When the hot gas stream escapes the CC it passes through HPT, then LPT and rotates several turbine wheels/blades. The stream also rotates LPC, HPC wheels as well as the fan through the shafts. Then, finally the gas stream leaves the core nozzle causing additional thrust.

1.2 Borescope Inspection and Engine Damages

Borescope inspections are generally required under circumstances including but not limited to:

- Periodical inspections coming from maintenance planning document (MPD) issued by the manufacturer, or airworthiness directives (AD) issued by the civil aviation authority of the aircraft operation.
- Before and after aircraft engine lease period in order to make sure that the engine is airworthy and has no major degradation in value
- After some incidents that may occur during the operation of the aircraft like foreign object damage (such as a bird strike or volcanic ash ingestion) where action according to the maintenance manual should be taken

Borescope inspection enables internal visual checks of the inaccessible engine parts without engine disassembly, which is a costly operation. Moreover, regular borescope inspections help detecting abnormal conditions early to avoid failures. In case detected abnormalities are above the specified limits then the damages have to be fixed to be able to continue operation. However, if the damage dimensions are below the limits then degradation level is followed up in the next scheduled inspection. Identifying possible damages at early inspection phase enables accurate planning and scheduling of maintenance actions, required resources, etc, therefore shorter park times of an aircraft. On the other hand, identifying and fixing damages at an early phase decreases the cumulative maintenance cost of an engine over engine lifetime (Sayadpishe and Rezaei, 2018). All these return as cost saving for an airline operator.

Borescope device used for inspection must have sufficient illumination, enough resolution, focus and zoom functionalities to enable detection of even small damages and then enabling close-up inspection of them.

Aircraft manufacturer AMMs provide detailed procedures for inspection of each engine module including preparation steps, required tools/equipment, inspection steps to be followed, the damage types to be checked per component as well as acceptable damage limits per type. Inspection operators have to follow these instructions during inspection.

Extreme operating conditions, such as high centrifugal loads, high temperatures, high pressure and vibration, in addition external factors like impurity of the air (volcanic ash, sand, humidity etc) or foreign damage strike (birds, pebble etc) are most usual reasons of aircraft engine material damages.

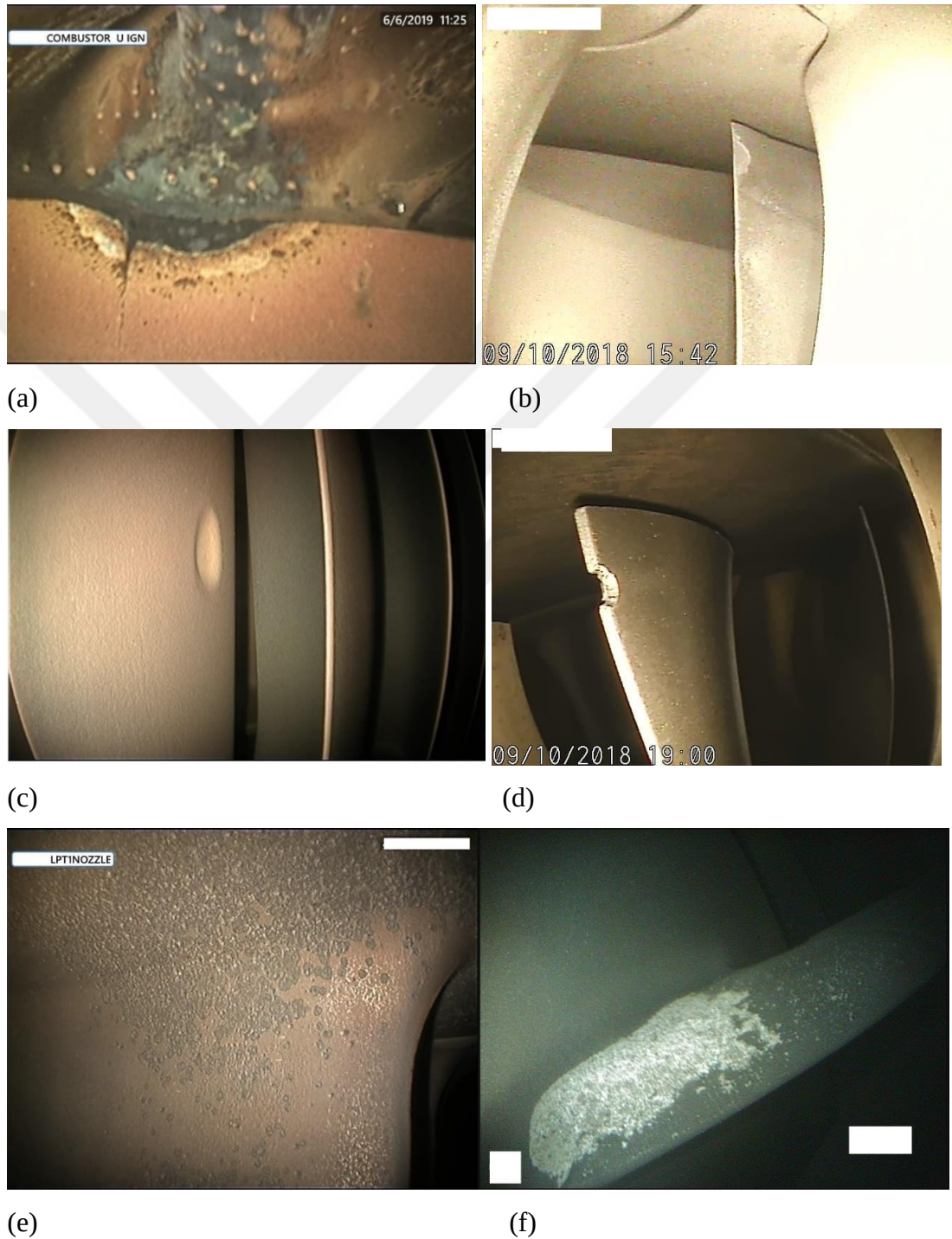


Figure 1.5. Sample borescope images with a) Burn and Crack b) Coating loss c) Dent d) Nick e) Corrosion f) Erosion damages.

Definitions of common damage types subset (Federal Aviation Administration (FAA)/Aviation Supplies & Academics (ASA), 2012) is provided in the glossary section of this thesis. Moreover, Figure 1.5 provides sample images of these common damage types.

1.3 Literature Review

In the literature both ‘damage’ and ‘defect’ terms are used for representing engine flaws in borescope inspection. In this section, the term used in the referred study will be used as it is across the relevant paragraphs.

Vembar et al. (2005) report about developing virtual training tools similar to the borescope in order to provide technicians a training similar to the on-the-job training which shall increase the training quality and human experience level.

The patent (Karpen et al., 2004) aims at providing a method and system for automatically detecting defects in remote video inspection applications. Proposed system performs analysis on images from a remote video inspection apparatus (borescope) to provide an operator with an indication of areas that are potentially out of specification and pattern recognition algorithms are applied to sequential images to search for anomalies or defects in the engine. Moreover, defects such as dents and small cracks are detected using stereo measurement techniques. For detected anomalies, known triangulation algorithms are used to provide depth and distance information to the processing system. Though, what is described in this patent is not an implementation but a system proposal.

Aust and Pons in (Aust and Pons, 2019) propose taxonomy for both engine blade defects and the root-causes, in addition it shows possible relations between two using an ontology. Authors come up with the blade defect categories Surface Damage, Wear, Material Separation and Material Deformation by analyzing the literature and maintenance manuals in addition to their own experience gathered by interviewing domain experts. For the root causes they propose Impact, Environmental causes, Operational causes, Poor maintenance, Poor manufacturing and Fatigue categories. This study provides a single comprehensive list of blade defects and standard blade defect terminology which do not really exist in the industry. The study focuses only

on blade and vane defects which are the most rejected parts in engine maintenance. Although it addresses an important gap in the aviation industry, proposed categorizations and taxonomy are not widely adopted in the industry yet. This was one of the reasons for not utilizing this terminology in our study, but it is considered as a possible future work.

Svensén et al. (2018) report one of the earliest studies on using deep neural networks for borescope image analysis. In their paper, they report results of applying deep neural networks on borescope images obtained from GE commercial engines fleet with the purpose of i) separating borescope images from images created with regular digital cameras ii) classifying images containing various engine parts. For separating borescope images from digital photographs, they use a shallow convolutional neural network and report average accuracy of >95% on the validation data with the observation that the predictions are more accurate on borescope images than on digital photographs. Authors also state that retraining fully connected layer of an VGG16 deep CNN instance for the same purpose did not give better results. However, VGG16 deep CNN is used for engine part recognition which gave 77% average accuracy (89% for the Mixer, 81% for Combustor, 82% for Fuel Nozzle, 67% for HPT Blade and 66% for Other engine hot parts) on validation data set. The model was trained with dropout rate (probability) of 0.5 in order to reduce the risk of overfitting. Engine part recognition supports engine hot parts only, which is justified as hot parts being inspected more frequently than any other engine sections and therefore constituting the largest amount of training data. Authors describe this study as initial steps to enable borescope imaging to become an integral part of the engine's digital thread, for automated measurement of key component condition. Especially, the big number of images used in this study (7098 images to be labeled though 2376 images were excluded) show the team does not have any issue with accessing a big borescope images data set which is quite useful when deep learning is to be used. This study can be seen as early steps of an automated damage detection from borescope images using deep learning, however authors do not explicitly highlight this particular objective in their future work.

Shen et al. (2019) present study using deep learning framework for automatic damage detection in aircraft engine borescope inspection. The framework supports crack and

burn damage types and it uses object segmentation to visualize the damage areas as polygons on input images. They use Fully Convolutional Network (FCN) deep learning architecture adopted from VGG-16 with a transposed convolution operation for up-sampling on top in order to recover damage shapes. The model is implemented using Keras library (Chollet, 2018). Having 1153 training and 290 validation images labeled manually they use channel mean subtraction as image preprocessing method. In addition, they use horizontal image flip, vertical image flip, and compound horizontal and vertical flip data augmentation methods in order to increase the amount of data. Moreover, transfer learning is applied by fine tuning the network on top of the weights trained on ImageNet data set (Deng et al., 2009). The authors claim to get the scores of 0.9803, 0.9726, 0.6789, and 0.6150 for pixel accuracy (PA), mean accuracy (mA), mean intersect over union (mIU) and frequency weighted intersection over union (fwIU) metrics, respectively.

This study is quite similar to our study in multiple aspects however, our study supports crack, burn, nick and dent damage types compared to burn and crack in this study. In addition, we do not support object segmentation, but mark detected damages with a bounding box instead of polygons. The reasons for this design decision are i) some damages (like burn) do not have very clear boundaries therefore, manual labeling of boundaries with polygons can be very subjective ii) some images may overwhelmingly contain a damage area. Our objective is not to accurately draw the damage boundaries but focus on ensuring damages are not missed, instead.

Ren et al (2020) use ensemble of deep learning models for detecting and classifying surface defects on aircraft surfaces. The framework is trained and evaluated on aircraft propeller blade bores borescope inspection images. Stacked ensembles with three CNNs each – called base learners - are trained to classify defect and defect free (binary) aircraft surfaces for MRO operations. Logistic regression algorithm is used as the meta-learner for the final prediction, and it is trained from the output of the base learners. Three ensembles investigated in this study are provided in Table 1.1 (taken from the paper).

Table 1.2. Component base learners of the three stacked ensembles.

Ensemble	Base Learners
Ensemble A	3× ResNet18
Ensemble B	ResNet18, Resnet34, Resnet50
Ensemble C	ResNet18, GoogLeNet, VGG11_bn

The data set consists of 300 defect (crack, scratches and gouges), and 300 defect free images with 224 x 224 pixels size. These images are split training, validation, and testing sets with the ratio of 4:1:1. The base learners are trained on ImageNet data, then fine-tuned with borescope images. Random horizontal flipping, image rotation, brightness and contrast adjustment, and finally Gaussian noise addition data augmentation methods are used to increase the data set, however except random horizontal flipping all other methods are reported to have no improvement on the model performance.

Authors report that ResNet18 and ResNet50 are the best performers in terms of mean accuracy, recall, and F1 score as individual base learners. Moreover, all three ensembles are reported to have > 99% mean accuracy, precision, recall and F2 score which outperforms individual base learners. But mean performance metrics of the three ensembles were not found to be significantly different. This is explained by weighing ResNet18 higher than other less performant base learners by the logistic regression algorithm.

The limitations of this study are that it supports binary classification of a single class only as well as ensembles are computationally expensive.

Wong et al. (2021) present an automatic workflow based on deep learning for detecting damages on rotor blades using borescope videos. The proposed system takes borescope videos as input, using instance segmentation marks (masks) chasing, rotor blades and if available, the damages on them. By applying a special tracking algorithm, it assigns IDs to rotor blades across video frames. This paves the way for the framework to collect statistics about the types and extent of damage found on rotor blades. The framework can also plot a time series graph which shows the amount of damage on each blade relative to the blade projected area while the blade emerges into and then leaves the camera view. The framework supports Casing, Compressor-rotor, Surface-damage (for scratches, but can also refer to compressor fouling, turbine sulfidation

etc.), Material-separation (for nick, tear and cracks) and Material-deformation (for bends) labels, first two for the engine components and latter two for the damage classes. Damage classes are devised from the taxonomy provided in (Aust and Pons, 2019).

Authors use adopted Mask-RCNN architecture (He et al., 2017) along with ResNet50-FPN (Lin, 2017) for the feature extraction. So called PointRend module (Kirillov et al., 2020) was attached on top of Mask R-CNN head to improve the segmentation accuracy of the mask head. They use 89 images for training and 15 images for testing. The weights are initialized from a pre-trained model with the COCO data set. The number of images is quite small compared to instance segmentation studies using deep learning, though the authors justify the positive results with similarity of the frames in a single video feed. Moreover, there is no information on the damage types (scratch, crack, nick etc.) statistics in the training and test data sets, also no information in the results therefore, it is not possible to devise which damage types are supported by the framework with which accuracy levels.

To avoid the framework to miss small features like surface scratches authors employ post processing phase on top of Mask RCNN detections including the steps to resize and crop, apply high-pass filter, mask erosion and finally applying a threshold and enhance operations on pixels with low intensities. The study reports around 0.90 maximum mAP for IOU=0.5 and around 0.84 maximum matched IoU metrics.

The framework and the workflow are offering an elegant automated solution for engine borescope inspection, however the current scope supports blade damages only and has strict assumptions on the input data. Moreover, it is trained and tested with a small number of HPC images only. Extending the framework to support other engine parts as well as other damage types (burn etc.) is expected to require big number of train/test images and will pose challenge on the metrics mAP and average IoU metrics they use. Compared to our study it provides more advanced features like blade tracking and instance segmentation. However, our study supports all engine modules, in addition we support more complicated damage classes like burn.

From Material Science domain Fu et al. (2019) present a deep-learning based approach for steel surface defects classification to facilitate quality control of steel strip

production. They use pre-trained SqueezeNet architecture (Iandola, 2016) and due to the pretrained weights (on images of common objects) it is reported to require only hundreds of defect specific training samples to fine-tune the model and achieve a good detection accuracy. To emphasize the fine-tuning of low-level features in the pre-trained model they set higher learning rates for shallower layers which characterize steel surface defects better. On the other hand, to generate scale-dependent high-level features for accurate classification of surface defects of various sizes they incorporate a so called multi-receptive field (MRF) module into the model. The model supports crazing, inclusion, patches, pitted surface, rolled-in scale, and scratches defect types. For training and test data they use publicly available NEU steel surface defects benchmark dataset (URL-9) which includes 1800 200×200 pixel grayscale images (300 samples for each surface defect category). In addition, in order to increase robustness of the defect classifier they construct a diversity-enhanced testing dataset of steel surface defects which contains severe non-uniform illumination, camera noise, and motion blur. For their proposed model, they report 100% accuracy on NEU test data set and 97.5% accuracy on the enhanced dataset. It is stated that the lightweight classifier can process more than 100 image patches (200×200) per second on a single NVIDIA TITAN X GPU with 12G RAM which makes it usable for real time steel production tasks.

Organization of the thesis is as follows:

The first chapter gives an introduction to the problem and a high-level description of the proposed solution. In addition, it provides introductory information for what turbofan engine is, how it works as well as lists the main manufacturers and engine types. Then, it provides a brief overview of borescope inspection as well as common engine damage types. Finally, in the literature review it provides brief description of related or similar studies performed in the academic world.

Second chapter gives an overview and the differences of the neural networks, convolutional neural networks, and deep convolutional neural networks.

The third chapter describes the materials and methods used in this study. In this context, data collection, processing, labeling steps as well as the utilized deep learning

model and training data and parameters are described. In addition, overfitting methods employed and the utilized hardware/software environments are described.

Chapter four presents the results of different scenarios experimented.

Finally chapter five discusses the results and provides the future work.



2. DEEP CONVOLUTIONAL NEURAL NETWORKS

Neural networks research is an interdisciplinary field that combines biology, mathematics, neuroscience, computer science and statistics. It has been popular since 1950s and 1960s followed by a partly frozen period in funds (though some research papers were still published (URL-10)) until mid 1985s, then received a substantially increasing attention in 1990s. However, the high expectations from application of neural networks did not materialize at that time. This was mainly due to limited amount of available data as well as insufficient computing power capacity. With the emergence of the deep learning architectures (a.k.a. deep neural networks) in the last 15 years thanks to the advances in computing power (like GPUs) as well as available software along with the possibility to get big data sets and labeling them (e.g. Crowdsourcing) made this field very popular again across several domains.

2.1 Neural Networks

Inspired from neurobiology neural networks are assumed to mimic human/animal brain in learning and inference processes. Neuron is the basic unit of a human brain and a human brain is estimated to have around 100 billion neurons (Ebersole and Pedley, 2003). A formal definition of neural networks provided in (Gurney, 2018) is as follows: “A neural network is an interconnected assembly of simple processing elements, units or nodes, whose functionality is loosely based on the animal neuron. The processing ability of the network is stored in the interunit connection strengths, or weights, obtained by a process of adaptation to, or learning from, a set of training patterns”.

The basic computational unit of a neural network is often called a node which is the artificial equivalent of biological neurons (See Figure 2.1). A node usually receives multiple inputs from the other nodes, each having an associated number called weight. In the human brain, connections between neurons are facilitated by electrochemical junctions called synapses which are modeled by the weights in artificial neural networks. The weight value highlights the importance of a connection between two nodes.

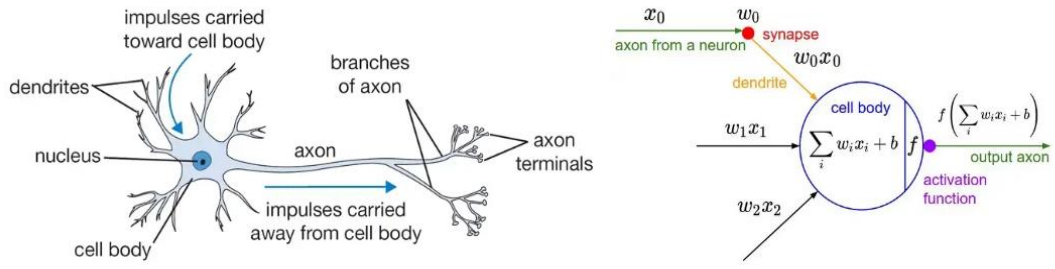


Figure 2.1. Biological neuron (left) and artificial neuron model (right). Taken from (Fumo, 2017).

Each node performs a calculation operation and produces an output. For the calculation, the node multiplies each input value with its weight and then adds a bias value. Finally, all these values are summed up and passed to a thresholding function called activation function. Activation function adds non-linearity to the neural network. Adding non-linearity is important because real world problems are usually nonlinear problems and activation functions enable neural networks to learn such nonlinear patterns. Well known activation functions include unit step function, family of sigmoid function, tanh function, rectified linear unit (ReLU) function (see Figure 2.2) and softmax function.

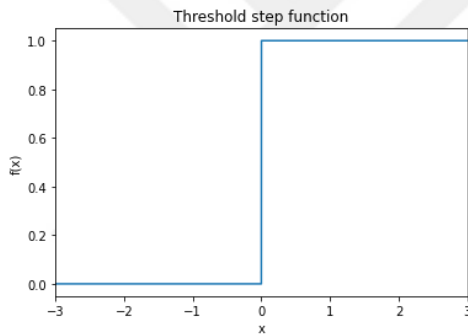
Step function, one of the simplest activation function, returns ‘0’ if the sum is below a threshold, and returns ‘1’ if it is otherwise. It is suitable for binary classification problems, i.e. grouping an input model into one of two groups.

Sigmoid function used to be a prevalent activation function which has an S shape. It takes any real number and maps it into $[0, 1]$ range. Very large negative numbers are mapped to (or near) 0 and very large positive numbers are mapped to (or near) 1. In this sense, sigmoid function generated output is not zero centered. It is suitable for models where probability needs to be predicted as output as well as for binary classifications.

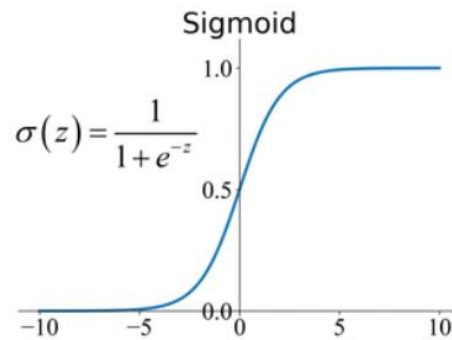
Tanh is the shifted version of sigmoid function that generates output in the $[-1, 1]$ range. It maps large negative numbers to -1 and large positive numbers to 1. However, it almost always performs better than sigmoid function since the mean of the activation is near 0, in other words the output is zero-centered.

The problem with both sigmoid and tanh activation functions is that if the input value is very large or very small then the derivative (slope of the function) is very close to 0 which could slow down the optimizer (gradient descent).

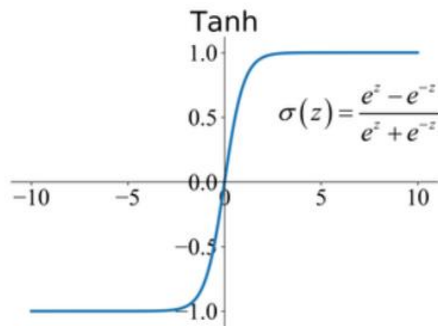
Rectified Linear Unit (ReLU) (URL-11), defined as $f(x) = \max(x, 0)$, became the most successful and widely-used activation function in deep learning. ReLU function outputs 0 if the input value is negative, otherwise it outputs directly the input value. Deep networks with ReLUs are more easily optimized than networks with sigmoid or tanh units, because gradients are able to flow when the input to the ReLU function is positive (Ramachandran, et al., 2017). In addition, ReLU is computationally effective compared to sigmoid or tanh activation functions, therefore it accelerates the training speed of deep neural networks. This is due to the constant value which does not require additional computation.



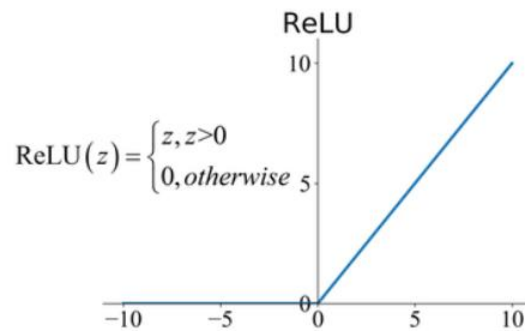
(a) Step function



(b) Sigmoid function



(c) Tanh function



(d) ReLU function

Figure 2.2. Common activation functions (Image in (a) taken from URL-12, images in (b)(c)(d) taken from (Feng et al 2019)).

Softmax is a unique activation function, also called softmax regression, which is used only at the last layer and only for classification tasks between multiple mutually exclusive classes. The function takes an input vector and generates normalized probability distribution of output classes between $[0, 1]$ range. Since the classes are mutually exclusive, the sum of the generated probabilities equals to 1. Typically, the generated class probabilities are interpreted as the confidence score of each class.

In neural networks, nodes are typically organized in layers. The first layer is typically the input layer where no computation is done, instead they just pass the value to the next layer. Then one or more hidden layers with various number of nodes follow. Finally, output layer with one or more nodes follows. Each node in the network is connected to some or all nodes of the next layer (the latter one is called Fully Connected Neural Networks - FCNN) and each connection has a weight. Here the idea is that these weight parameters are adjustable/learnable. Figure 2.3 depicts a simple neural network.

Having fed the input values usually in form of a vector, computation is done one layer at a time up to the output layer. This is called forward propagation. The final value(s) is/are the output of the neural network.

By giving the input data set along with the expected output (called supervised learning) the network calculates a deviation between the generated output vs expected output via an error function and this error is propagated back to adjust the weight parameters in the direction that minimizes the total error over the data set. This is known as the backpropagation which serves the learning process.

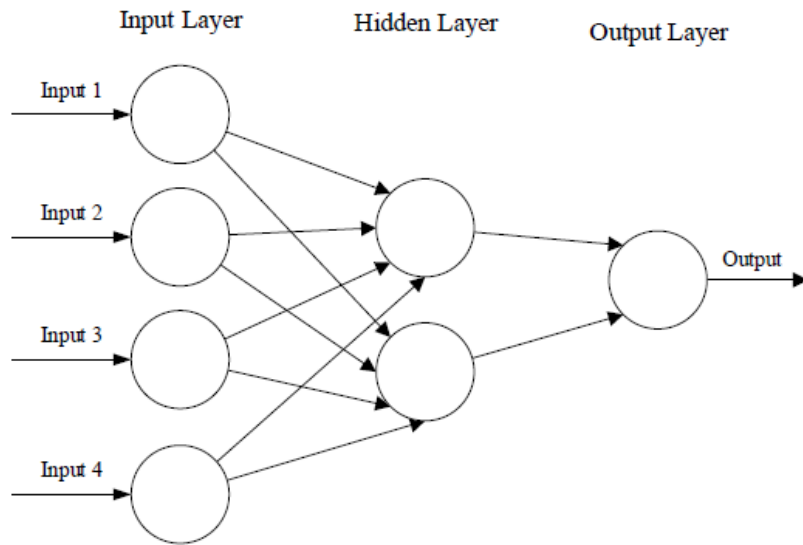


Figure 2.3. Simple neural network with input, one hidden and an output layer. Arrows between nodes represent the weighted connections and arrow directions show the forward propagation direction. Taken from (O'Shea and Nash, 2015).

Neural networks are typically used for regression as well as binary or multi-class classification type of problems.

2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) give computers vision capability without the need to program specific algorithm per problem type as in the old computer vision tasks. They differ from the traditional neural networks in the sense that they have convolution layers that extract key features from images. These key features or structures tell the network what is unique about specific object/class so that correct classification is made (like wings for a bird versus ears for a cat). This makes CNNs a perfect fit for image recognition, object detection and object classification tasks. Extraction of key features/structures in images relies on the fact called local connectivity which suggests single pixels do not get their meanings from their RGB values alone, but adjacent pixels together may become meaningful. Based on this principal CNN convolution layers process patches of pixels together.

CNNs are comprised of three type of layers. Convolutional layers, pooling/down sampling layers and fully connected layers. Figure 2.4 illustrates a sample CNN architecture for an object classification task.

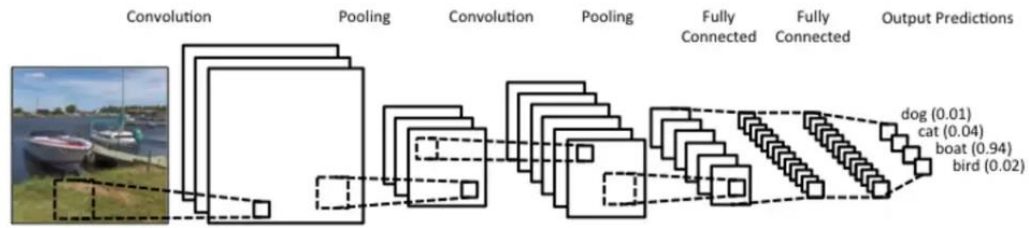


Figure 2.4. A sample CNN for an image classification task. Taken from (Fumo, 2017).

- i) Input layer: holds the raw pixel values of an image. It has image width x height x 3 (in case of RGB channels) number of nodes
- ii) Convolutional layer: each neuron in the convolutional layer can see only a so-called patch (region) in the input image. This layer computes the output of the dot product of the patch and the applied filter (window of weights) for each neuron. Then it applies a non-linear activation function, usually computationally attractive ReLU. This results in feature maps.
- iii) Pooling layer: applies down sampling on the feature maps by applying an aggregation function like max or average, however most CNNs use max pooling (O'Shea and Nash, 2015). This decreases the size of the feature map; therefore, it reduces the computation cost. This also distinguishes CNNs from artificial neural networks. In CNNs there might be multiple pairs of consecutive convolution + pooling layers.
- iv) Fully connected layer: each neuron is fully connected with the previous layer neurons as well as with the output nodes. Unlike previous hidden layers it has no spatial structure. It produces class score(s) for the classification task based on activations.

2.3 Deep Convolutional Neural Networks

Deep neural networks are neural networks with multiple hidden layers, in the same way deep convolutional neural networks (DCNN) are CNNs with multiple hidden layers. To give an idea about the number of layers, the famous Oxford Vision Geometry Group (VGG) deep neural network (Simonyan and Zisserman, 2015) which was the 1st-runner of ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Russakovsky et al., 2015) in 2014 has 16 to 19 layers. The winner of the competition

in 2014 GoogleNet (Inception v1) (Szegedy et al., 2015) has 22 layers. On the other hand, ResNet (He et al., 2016) the winner of the same competition in 2015 has 152 layers.

Training deep CNNs require big number of images (from thousands upwards depending on model complexity) as well as they are quite compute intensive. Although CNNs have been around since long time, training deep CNNs became popular only during the last 10-15 years, especially after AlexNet (8 layers deep CNN) (Krizhevsky, 2017) in ILSVRC-2012 competition outperformed all shallow neural networks. Emergence of large labeled data sets like ImageNet database (Lin et al., 2014), success in competitions like ILSVRC and advances in computing hardware (like GPUs, TPUs) made deep CNNs very attractive. Therefore, they found quite widespread interest across all domains like medicine, self driving cars, material science etc. research among others. Also not to forget is that the release of frameworks and software libraries like TensorFlow (Abadi et al., 2016), PyTorch (Paszke et al., 2019), Google's Tensorflow Object Detection API (URL-4), Keras (Chollet, 2018) that simplifies building, training and evaluation of deep convolutional neural networks in more rapid and much easier way contributed deep CNN research boom. In addition, the idea of transfer learning paved the way to train deep convolutional neural networks using less data (images) thanks to reusing weights trained by a different data set and finetuning typically the fully connected layer with domain specific images.

Deep CNNs are typically quite complex and have too many (VGG-16 layers has approximately 134 million parameters, ResNet-152 has about 58 million parameters in 2D both for 1000 object classes (Leong et al., 2020)) parameters to train so there should be sufficient training data to be able to adjust all the weights to achieve good accuracy. When the data set is not sufficient then the model overfits. In other words, it will not generalize well. To overcome this problem dropout method is introduced by Srivastava et al.(2014). Dropout randomly and temporarily removes nodes along with all its incoming and outgoing connections during the training in order to simplify the model, therefore, to reduce the number of weights to be adjusted.

3. MATERIALS and METHODS

Figure 3.1 summarizes the research and deployment phases of the proposed automated damage detection framework from borescope images. Each phase will be described in the subsequent sections.

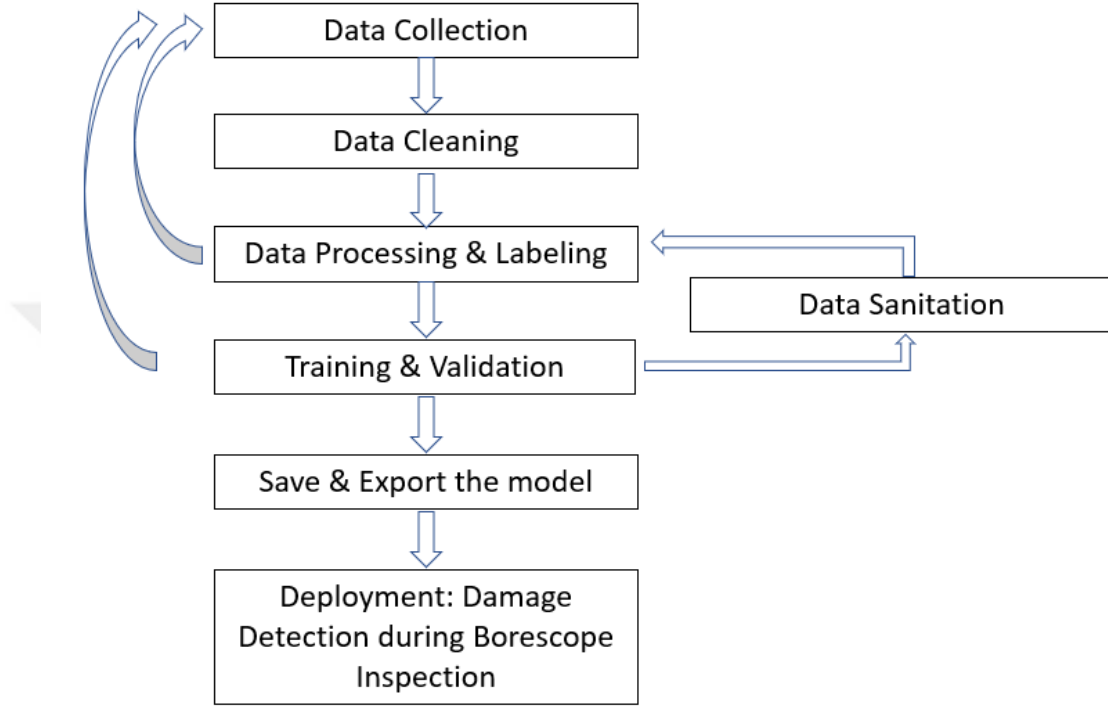


Figure 3.1. Overview of proposed automated damage detection framework development and deployment process.

The process is not like a waterfall model, there were several iterations in data collection, cleaning, processing and labeling as data from different sources arrived at different timeframes. In addition, after first validation rounds insufficient data consequences forced to go back to the data collection step, in addition, detected input data and labeling issues (like missed damages while labeling) were addressed by going back to the Data Processing and Labeling phase.

3.1 Data Collection and Cleaning

Accurate and sufficient data is the primary requirement for success of every deep learning study. In our study, data collection and access to a big number of images/videos possibilities were limited since there is no public aircraft engine borescope inspection data set available. Limited number of videos and images are

available in the web provided by private companies that offer aircraft engine inspection services.

Therefore, we contacted a number of private airlines in Türkiye as well as abroad to request borescope inspection data in form of videos, images or borescope inspection reports. Normally, the first two include unlabeled raw data but inspection reports include damage images along with the damage information in text form which can simplify the labeling process. In some cases, reports include images with marked damages using special image processing software.

In 3 batches we received data from in total three airlines in and outside Türkiye. In addition, some images were downloaded from the web. Received data included videos with and without damage, images and borescope inspection reports containing damage images. The videos were analyzed by an Aviation Engineer with long years of experience in the field and various damage images were extracted from there. In total, we collected around 4.340 images including images we extracted from borescope inspection reports as well as from the videos and images downloaded from the web. With the help of the same Aviation Engineer all these images were analyzed for eligibility, and then eligible ones were labeled for supervised learning process (labeling will be explained in the next section).

In the obtained images there are images of following engine types in various numbers:

- CFM56-3B used in Boeing 737-300, Boeing 737-300, 737-400, 737-500
- CFM56-5B used in Airbus 320
- CFM56-7B used in Boeing 737/800,
- V2500-A5 used in Airbus 320.
- CF6-80C used in Boeing 747, Airbus A330
- RB211 used in Boeing 747, 757, 767, 777
- PW4077 used in Boeing 777

Images from LPC, HPC, CC, HPT, LPT engine parts are available in the data set in various numbers. However, this does not mean for each engine type there are images for all listed engine modules. There is no general rule for availability of images per engine type (for example V2500 does not have any LPC images but for the others).

We went through the images to do eligibility check and filtered out images that:

- are duplicate
- have markings done using image processing software (see Figure 3.2) overlapping the damage area or circling the damage area
- have ignorably small damage (even unnoticeable by expert eye)

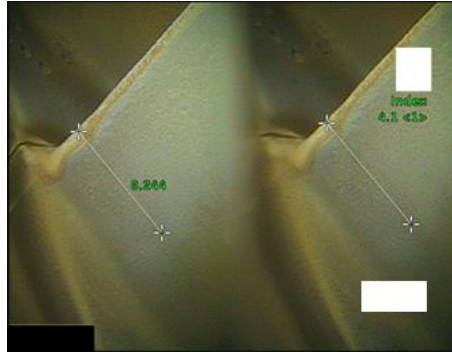


Figure 3.2. Sample borescope image with two cracks marked using a software to indicate the lengths of the cracks. However, the drawings hide the cracks and they can also mislead the learning process about the crack structure.

At the end of the cleaning phase in total 1050 RGB images with damage remained, most of them in JPG format but around 20 of them in BMP format (with image size > 1.3 MB). In addition, we reserved 108 RGB images with no damage for possible usage like binary classification tasks.

3.2 Data Processing and Labeling

On the collected image set following activities are performed before labeling them.

- Sensitive information like engine serial number, company logo etc. were removed from the images for anonymization purpose (See Figure 3.3 for a sample anonymized image).
- In the original file names white spaces were removed and special characters like “(“, “)”, “#” available in- some file names were replaced by underscore “_” in order to avoid these characters jeopardize functionality of any script or cause any file gets ignored.
- BMP files with big size (>900 KB) were converted to JPG format and reduced the resolution so that each image becomes with around 300KB in size. This is to reduce

additional processing during training/validation as well as achieving homogeneous file format.

- Original file names were renamed to have <DataSource>_<EngineType>_<EnginePart>_<OrigName>.jpg/.JPG pattern in order to provide research team visibility on data and allow statistics collection based on file name on various dimensions. In order to provide traceability original image name (after file name sanitation described above) was kept in the file name.

Data source is a generated 3 letters internal code of from where the images are obtained. Engine type is one of the entries from the engine list provided in section 3.1. In case engine type is unknown it gets the value “UNKNOWN”. Engine part can have one of the values LPC, HPC, CC, HPT, LPT, UNKNOWN.



Figure 3.3. Engine serial numbers are removed from the top left corner of both images for data anonymization; however, in the left image engine module (HPT) and blade numbers were kept on purpose for traceability purpose but also as these are not sensitive information.

Using open source labelImg (Tzutalin, 2015) software (Windows version) images were labeled in PascalVoc format (xml) (Everingham and Winn, 2011) as well as YOLO (txt) (Redmon et al., 2016) format with the Aviation Engineer support. Here, no semantic segmentation for damage borders has been done. Labeling was done using bounding box around the damage area only (See Figure 3.4). This is because detecting exact borders of the damages is not in the objective of this study. Instead, we focus on minimizing undetected damages on the risk of increased false positives.

Note that exact borders of the bounding box labels are subject to human bias. This is especially the case for burn type of damage where the damage does not have natural borders. However, this bias is considered to be ignorable by softening the detection overlap threshold which will be discussed in the results section.

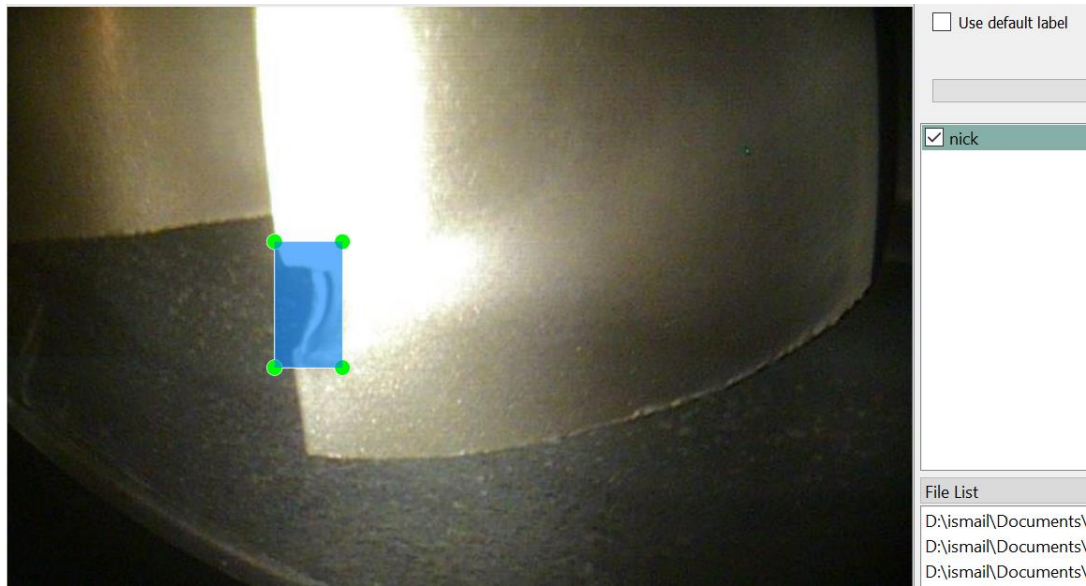
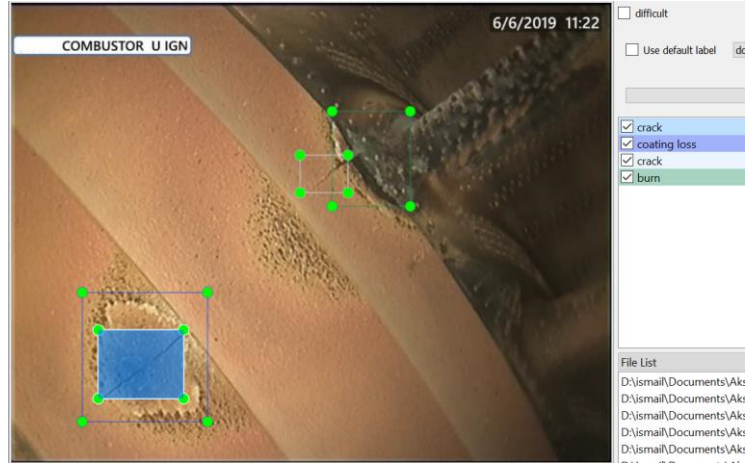


Figure 3.4. Sample borescope inspection image labeled with “nick” label and a bounding box around the damage using labelImg software.

Most of the images had more than one damage of the same or different type in the same image. In addition, several images had overlapping damages. In this case each damage is labeled separately despite the overlap.

Labeling strategy for damages with big area proportional to the image size was a debate initially. One can label the damage with a single, however big bounding box versus one can split the damage area into multiple bounding boxes (see Figure 3.6). Initially, splitting into multiple bounding boxes strategy was followed. However, validation results showed the model tends to detect bigger bounding boxes for the same image (also due to non-maximum suppression (NMS) configuration). Therefore, the labeling strategy was changed to use one bounding box per damage whenever possible. On the other hand, this strategy cannot be applied to damages which exist in the majority of the image (like erosion, corrosion etc.) and one bounding box would include more than one damage area.

(a)



(b)

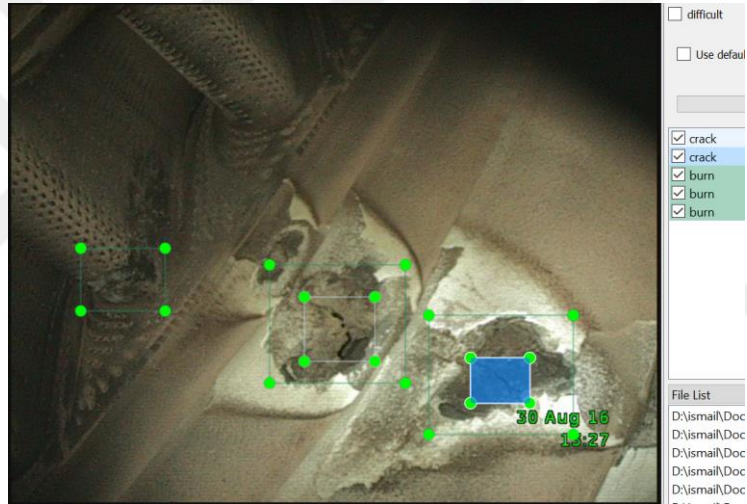
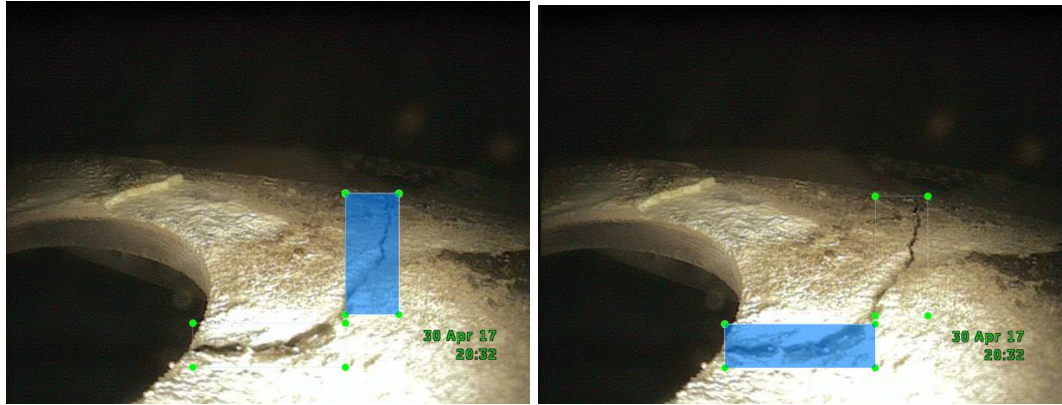


Figure 3.5. Sample images with multiple damages and overlapping damage areas. Each damage is labeled separately.

The challenges we experienced with some BSI images are noisy background due to oxidation or deformation due to heat especially in combustion chamber and low illumination. This could have positive impact in terms of making the model more resilient in case there were few thousands of such images, however having only few (tens to hundred) such images makes the generalization harder.

(a)



(b)

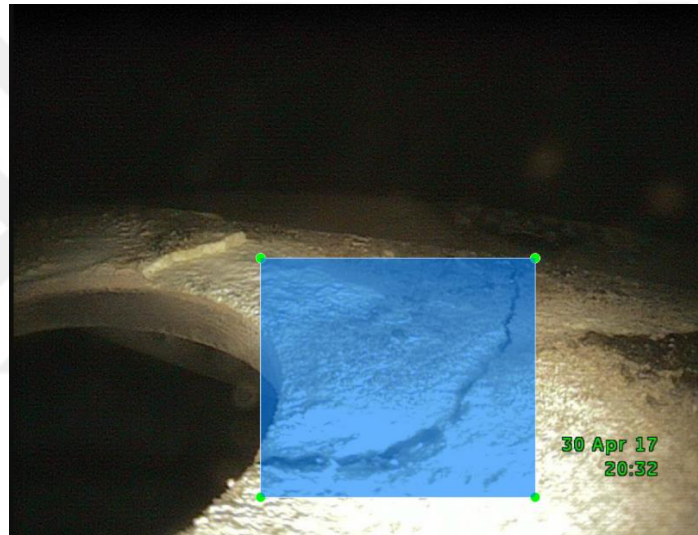


Figure 3.6. A big crack is labeled with two bounding boxes in the upper images (a) vs one bounding box in the below image (b).

After the labeling process is completed, we ended up having 11 damage classes among 1050 image labels. The statistics per damage class is given in Table 3.1. Damage class having less than 200 images were decided to be out scoped with the exception of nick. These are break, coating loss, corrosion, distortion, erosion, missing piece and tear. Note that during labeling some rare damages like “edge burnback” or “surface burn” were categorized under “burn” label to avoid further fragmentation in the data set of alike damages which will eventually cause drop of some images. Nick was kept in the

Table 3.1. Number of images that include each damage class and total number of labels per damage class in the data set. Highlighted rows are kept in the research scope.

Damage Class	Number of Images	Number of Labels
break	3	3
burn	237	455
coating loss	122	216
corrosion	2	3
crack	255	421
dent	354	498
distortion	25	31
erosion	21	32
missing piece	59	67
nick	160	247
tear	11	13
Total distinct images	1050	
Distinct images in the scope	924	

scope due to i) its similarity to dent therefore, the possibility of class merge between two classes ii) relatively high average precision (AP) values nick class had during initial training/validation rounds. This will be discussed in the next section. With this, burn, crack, dent and nick classes remain in the scope and images not having any of these classes were also removed from the training/validation scope. This out scoping led to 924 distinct images left in the data set. As presented in Table 3.2, 446 of these images have more than one damage label (either same or different classes) and out of those, 76 images have more than one damage type. 501 of the images have 752 x 576 resolution, 137 have 720 x 576, 12 have 640 x 480, 14 have 1024 x 768, another 14 have 1920 x 1080 and others have various resolutions. The smallest image has 134 x 202 and largest image has 1920 x 1080 resolution. This level of variation is mainly due to data collection from various sources and various materials.

Table 3.2. Image/Label Statistics.

Images with	Image count
single damage	476
multiple damages	446
one damage class	846
2 damage classes	76

Images were not resized at this phase since we utilized Object Detection API image resizing configuration as image pre-processing step to align all images before the training process. This is a built-in feature in the API itself.

3.3 Deep Learning Model

As (Soviany and Ionescu, 2018) reported two-stage detectors, such as Faster R-CNN or Mask R-CNN, reach the highest accuracy rates, but are typically slower. On the other hand, single-stage detectors, such as YOLO (You Only Look Once) and SSD (Single Shot MultiBox Detector) reach lower accuracy rates, but are much faster than two-stage object detectors.

Suggested framework is using Faster R-CNN network which is an improved version of Fast R-CNN (Girshick, 2015) (which in turn is improved version of Region-based Convolutional Neural Network (R-CNN) (Girshick, 2014)). The main improvement in Faster R-CNN is the so called Region Proposal Network (RPN) that from the image generates region proposals to look at for object detection. This RPN shares the convolutional layers with the object detection network (specifically the feature extractor CNN) which makes it faster than the Fast R-CNN architecture.

For feature extraction Inception V2 CNN is used. This combination is selected as a good trade-off between speed and accuracy (see Figure 3.7). Speed is also a (secondary) parameter here as damage detection from the borescope videos is in the future work list of this research.

COCO-trained models			
Model name	Speed (ms)	COCO mAP ^[1]	Outputs
ssd_mobilenet_v1_coco	30	21	Boxes
ssd_mobilenet_v1_0.75_depth_coco ☆	26	18	Boxes
ssd_mobilenet_v1_quantized_coco ☆	29	18	Boxes
ssd_mobilenet_v1_0.75_depth_quantized_coco ☆	29	16	Boxes
ssd_mobilenet_v1_ppn_coco ☆	26	20	Boxes
ssd_mobilenet_v1_fpn_coco ☆	56	32	Boxes
ssd_resnet_50_fpn_coco ☆	76	35	Boxes
ssd_mobilenet_v2_coco	31	22	Boxes
ssd_mobilenet_v2_quantized_coco	29	22	Boxes
ssdlite_mobilenet_v2_coco	27	22	Boxes
ssd_inception_v2_coco	42	24	Boxes
faster_rcnn_inception_v2_coco	58	28	Boxes
faster_rcnn_resnet50_coco	89	30	Boxes
faster_rcnn_resnet50_lowproposals_coco	64		Boxes
rfcn_resnet101_coco	92	30	Boxes
faster_rcnn_resnet101_coco	106	32	Boxes

Figure 3.7. Tensorflow 1 Object Detection API detection speed and mAP comparisinal results table for various models trained on COCO set. Taken from (URL-13).

The model architecture is depicted in Figure 3.8.

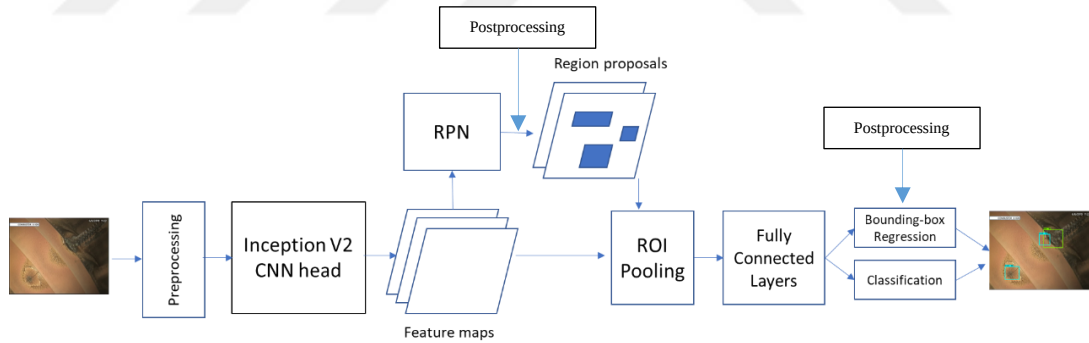


Figure 3.8. Faster R-CNN based damage detection model architecture.

The model is created using Tensorflow Object Detection API. For the first stage region proposal network truncated normal initializer and for the second stage box predictor network for object classification variance scaling initializer is used. These are recommendations of TFOD API group based on their experimentation with COCO data set that gave best results (URL-14). COCO is one of the most popular large-scale object detection, segmentation, and captioning dataset published by Microsoft. The dataset has 330K “images of complex everyday scenes containing common objects in their natural context” (Lin et al., 2014). Over 250K images contain annotations in 80

object categories we encounter during the daily life (like person, car, chair, motorcycle, bus, bottle etc).

Second stage box predictor has two fully connected layers and output layer for object classification as well as for bounding boxes. For the object classification softmax activation function and for the training, log loss function are employed. Similarly, bounding box layer uses Smooth L1 loss function.

3.4 Training

For supervised learning, image data set was split into train and test data sets in semi-random way. For the split, images are sorted by data source, engine type, engine part and although within each group the selection was mostly random, we tried to put between 10-15% (where possible) of each group into the validation set and others were put into the training set. At the end, 829 distinct train and 95 test (validation) images were obtained. The distribution of the train and validation set images and number of labels per damage class are given in Table 3.3.

Table 3.3. Number of images and number of labels per training and validation set.

	Training Set		Validation Set	
Damage class	# of labels	# of images	# of labels	# of images
Burn	410	209	45	28
Crack	378	227	43	28
Dent	451	319	47	35
Nick	225	145	22	15
Total distinct images		829		95

There were several training/validation rounds. Impact of image size, number of training steps, stride size, transfer learning (training from pre-trained checkpoint), dropout and regularization methods for overfitting handling, image preprocessing and data augmentation, damage class count, confidence score threshold are all experimented in different training/validation rounds. The results are observed using Tensorboard GUI which is Tensorflow's visualization toolkit (URL-15).

Tensorflow Object Detection API provides configuration of hyperparameters through so called pipeline config file. Pipeline config file for the training/validation with the best results is given in the Appendix A-1.

Images were resized before training using Tensorflow Object Detection API's built-in feature in order to have homogeneous image size. Here, desired size of the smaller image dimension in pixels was set to 640 pixels and desired size of the larger image dimension in pixels was set to 1024 which were the defaults. This means the smaller dimension will be reduced to 640 and the larger dimension will keep the original aspect ratio.

Hyperparameters:

- Default stride value: 16, however, it was experimented also with 8 (in the later case 8 was used to test if the model would detect smaller damages, however the overall accuracy was lower)
- First stage (RPN) box predictor truncated normal initializer with 0.01 standard deviation
- Second stage box predictor: variance_scaling_initializer (weight: 1.0)
- Optimizer: momentum_optimizer
 - Learning rate: 0.0003
- Fine tune checkpoint: (trained on MS COCO dataset) faster_rcnn_inception_v2_coco_2018_01_28/model.ckpt
- Data augmentation options:
 - random_horizontal_flip
 - random_rotation90
- Confidence level threshold: 0.5
- IoU threshold: 0.25

3.5 Overfitting Handling

3.5.1 Data augmentation

As suggested in several resources including (Wang et al., 2020; Wang and Perez, 2017) data augmentation technique is an efficient way to increase the data set by deriving new images from the train/validation data set. Typical data augmentation techniques

include but not limited to flipping, rotation, zoom in/out, shift, distortion, adjusting brightness, adjusting contrast, extracting part of the image, shading with a hue etc. There are two possible ways of data augmentation:

- i) Generate new images from the data set using a script and add these new images into the training set (Wang and Perez (2017) suggest to put the augmented images in the training set only)
- ii) Use Tensorflow Object Detection API's data augmentation built-in feature which supports all methods listed in (URL-16). Note that TFOD API feature mostly applies random selections (like random horizontal flip may or may not flip a particular image) which may cause two consecutive training rounds of the same model to have deviations in the data sets. Therefore, even validation metrics may have deviations. This makes impact assessment of a parameter on the validation accuracy open to an error factor.

In our study we decided to use TFOD API built in feature and experimented random horizontal flip, random rotation (90 degrees), random crop image, random adjust brightness and random adjust contrast methods only. Random horizontal flip, random vertical flip and random rotation (90 degrees) were found to improve overfitting and increase the accuracy values (to be explained in Section 4).

3.5.2 Dropout

As (Srivastava et al., 2014) describe dropout is an effective method to reduce overfitting which randomly drops units (along with their connections) from the neural network during training. This prevents units from co-adapting too much, and at the same time decreases the number of parameters to train, thus simplifying the network. In our experiments we used dropout at second stage box predictor with 0.5 keep probability. Dropout improved overfitting as it will be shown in section 4.

3.5.3 L2 regularization

Regularization is a method of controlling model complexity to reduce overfitting. It does this by penalizing higher terms in the model. When a regularization term is added, then the model tries to minimize both loss and complexity of the model (Yildirim, 2020).

L2 regularization adds squared sum of the weights to the loss function and removes a small percentage of weights at each iteration so that the weights decay towards zero, but they never become zero.

We utilized L2 regularization at both first stage box predictor and second stage box predictor with 0.01 regularization factor which improved overfitting as it will be shown in section 4.

3.6 Evaluation Metrics

There are various metrics and notations used by the various object detection challenges as well as the scientific community to evaluate object detection accuracy of deep learning models. However, average precision (AP) and its derivations are the most common metrics adopted to assess the detections (Padilla et al., 2020). In this study, PASCAL VOC 2010 detection metrics average precision and mean average precision (mAP) are used for evaluating the quality of the object detection. These metrics are computed by Tensorflow Object Detection API according to the protocol of the PASCAL VOC Challenge 2010-2012 (Everingham and Winn, 2011). AP is a metric for a single class accuracy whereas mAP is the accuracy metric across all classes. The notation used in this study is like $AP@0.25IOU/damage_class$ and $mAP@0.25IOU$ which indicates AP for a given damage class detections for Intersection over Union (IOU) threshold value of 0.25.

IOU is a metric that shows how good a detection bounding box overlaps with the ground truth (label) bounding box. It is calculated as area of the overlap between the detection bounding box and the ground truth bounding box, divided by the area of the union of them (see Figure 3.9). If the overlap is equal or above a given threshold (in above case 0.25) then the detection is considered as being correct (true positive (TP) case), otherwise it is considered as an incorrect detection (false positive (FP) case). FP metric covers also incorrect detection of a nonexistent object. False negative (FN) metric covers the cases where an existing ground truth object is undetected. Note that, in object detection context, false positive (FP) case is irrelevant because there can be infinite number of such cases even in a single image.

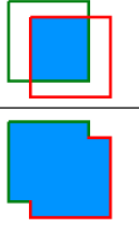
$$IOU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\text{area of overlap}}{\text{area of union}}$$


Figure 3.9. Intersection Over Union (IOU). Picture taken from (Padilla et al., 2020)). Red boxes represent the ground truth and green boxes represent the detection made.

Note that according to the PASCAL VOC 2010 rules multiple detections of the same object in an image are considered false detections. For example if 4 detections of a single object are made then only one is counted as correct detection and the other 3 are counted as false detections.

Two objectives of an object detector are to detect all ground truth objects (FN=0) while detecting existing objects only (FP=0). The success rate of the first objective is called precision (P) and the success rate of the latter one is called recall (R). P and R are calculated as:

$$P = \frac{TP}{TP + FP} = \frac{TP}{\text{all detections}} \quad (3.1)$$

$$R = \frac{TP}{TP + FN} = \frac{TP}{\text{all ground truths}} \quad (3.2)$$

P and R are calculated per damage class for all detections made (with an IOU value above a certain threshold) across all images. Average precision per class is calculated as the area under the P x R curve. Since the P x R curve has a zig-zag behaviour PASCAL VOC 2010 challenge guidelines instruct to use all points interpolation technique which applies interpolation across all detections with taking the maximum precision whose recall value is greater than or equal to next recall value (see Figure 3.10.).

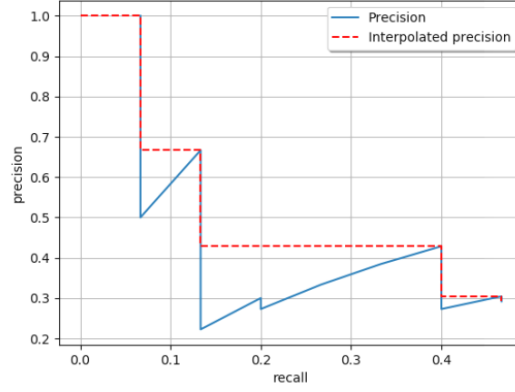


Figure 3.10. Sample Precision x Recall curve with all points interpolation. Picture taken from (Padilla et al., 2020)).

Having calculated AP per class, mAP is calculated as the mean of APs across all classes.

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (3.3)$$

where N is the number of classes.

3.7 Hardware and Software Environment

For development and training/validation TUBITAK TRUBA High Performance and Grid Computing Center infrastructure has been used. (URL-17). The development was done on the so-called interface server (Levrek) and for training/validation the tasks have been delivered to the GPU cluster queues via SLURM workload manager (URL-18). All the images have been transferred to the disk space on TRUBA account using OpenVPN and WinSCP file transfer software. For training/validation, CUDA compute nodes are used which have Compute 3.5+ capability required by Tensorflow (see Table 3.4). Most of the training/validation tasks were run on 2 GPUs and 20 cores.

All compute nodes run Linux CentOS 7.3 operating system. Following software list has been installed and used on the environment:

- Anaconda 5.1.0
- Python 3.7.7
- Tensorflow (GPU version) 1.13.1
- Tensorboard 1.13.1

- Cudnn 7.6.5
- Cudatoolkit 10.0.130
- Tensorflow 1 Object Detection API
- Faster R-CNN with Inception v2 feature extractor model (2018_01_28)

Table 3.4. TRUBA compute nodes used for development (Levrek) and training/validation (Barbun-cuda and Akya-cuda) along with their specifications.

Quantity	CPU/GPU	Processor Model	SPECfp_rate_base2006	Theoretic Gflops	Memory	Description
108	2 CPU x 8 cores (32 thread)	Intel Xeon i5-2690	487	371 Gflops (double)	256 GB 1600MHz ECC	Levrek Interface Server
24	2 CPUs x 20 cores + 2 x Nvidia Tesla P100 GPU	Xeon Scalable 6148 2.40GHz	1400	2048Gflops +9400Gflops	384 GB + 2x16 GB HBM2	Barbun-cuda cluster
24	2 CPUs x 20 cores +4 x Nvidia V100 GPU	Xeon Scalable 6148 2.40GHz	1400	2048Gflops + 4x7800Gflops	384 GB + 4x16 GB HBM	Akya-cuda cluster

4. RESULTS

Training the model with the data set for 4 damage classes (burn, crack, dent, nick) from scratch (without transfer learning) resulted in low average precision values for all damage classes (see Figure 4.1). This is due to the insufficient number of images in the training set to be able to adjust all the weights of the model up to a sufficient level. In addition, the model overfit from the same reason which is recognizable through decreasing training loss (Figure 4.1c) whereas increasing validation loss (Figure 4.1b) charts.

For overfitting handling, methods described in section 3.5 were applied to the model.

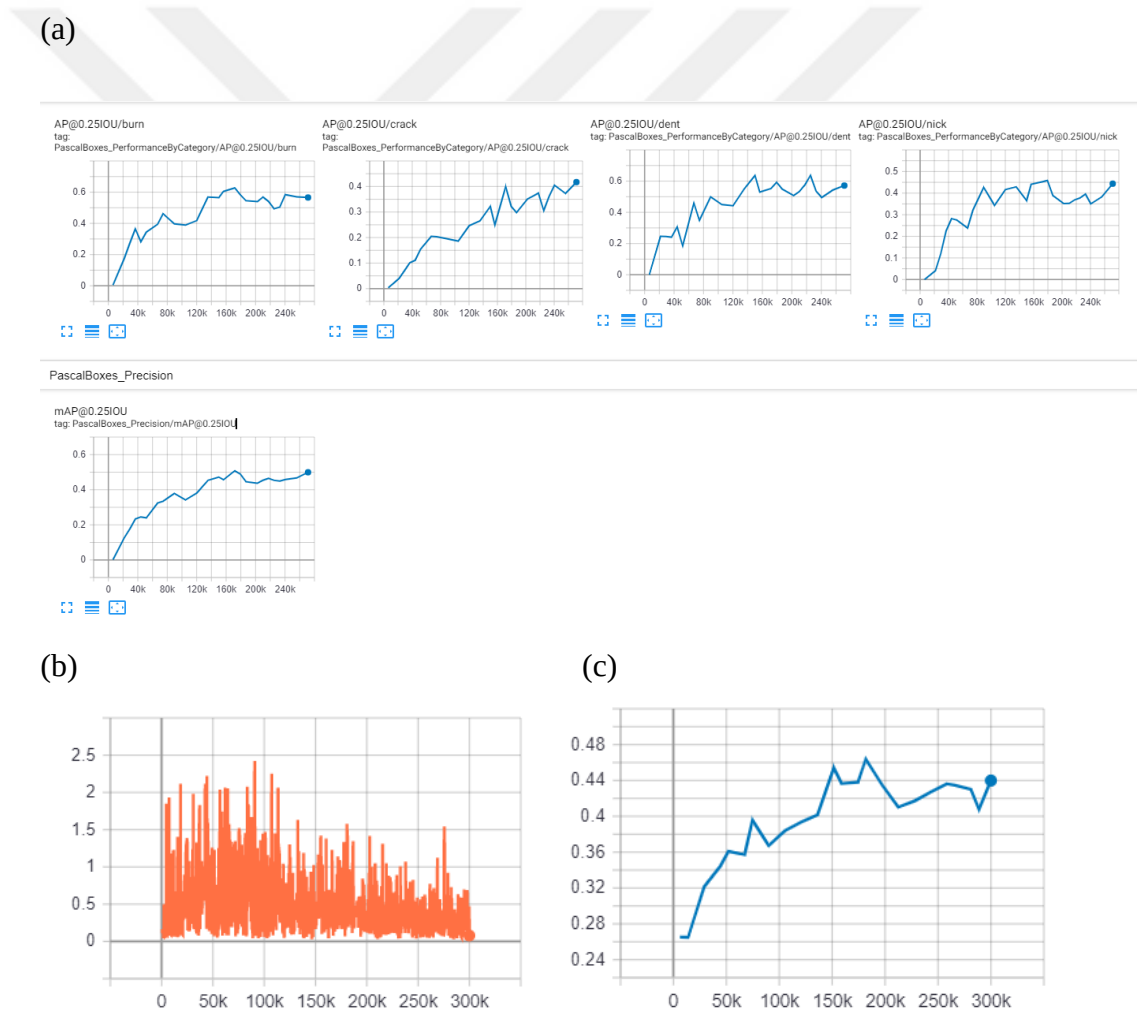
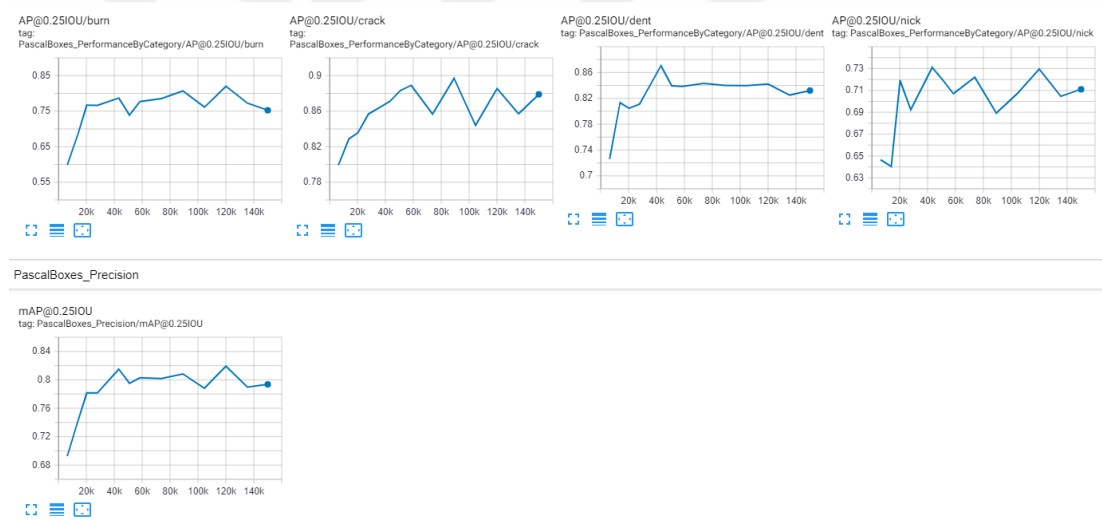


Figure 4.1. Without transfer learning (a) average precision (AP) of burn, crack, dent, nick damage classes and mean average precision (mAP) for IoU=0.25 (b) training loss (c) validation loss.

Training and validation of 4 damage classes for 150K steps by initializing the weights from faster_rcnn_inception_v2_coco_2018_01_28 checkpoint trained on COCO data set, random vertical flip and random rotation (90 degrees) data augmentation methods, dropout with 0.5 drop probability and L2 regularization with 0.01 weight gave maximum average precision (AP at 0.25 IoU) of 80% for burn, 89% for crack, 85% for dent and 74% for nick. mAP@0.25IoU reached maximum 81%. (see Figure 4.2). This is a significant increase compared to the training without transfer learning (where mAP@0.25IoU was 51%).

It was observed that nick AP was still unexpectedly low. Analyzing the detections made vs ground truth labels, it was seen that several dent and nick labels were mixed (misclassified) during detection (see Figure 4.3) which is a source for false positives, therefore lower AP values. This is also not a big surprise since even in the field



(b)

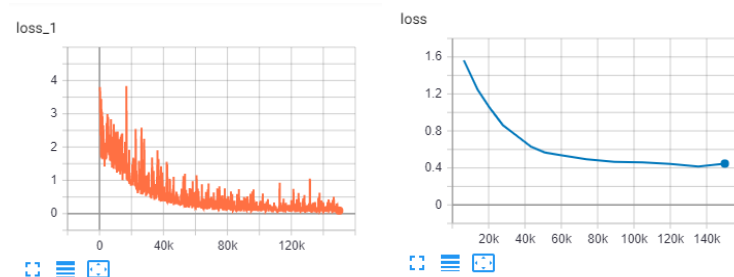


Figure 4.2. (a) AP and mAP of 4 damage classes crack, burn, nick dent at IoU=0.25 and confidence score=0.50 (b) Training loss at the left and validation (total) loss at the right.

classifying a damage as nick or dent can be controversial between inspectors. It is because in some cases the damages look very similar and in other cases it is not possible to infer from the image whether there is a material loss or not (which is the case for nick). In such cases, more images of the same damage with different proximity and different angles are required to be able to judge the correct damage type. In our case, it is not possible at this phase to enrich the data set with all such images. In addition, after nick and dent damages go through blending process², most of the time it is difficult to recognize the original damage from the image with bore blended damage. This means, either all images with bore blended damage have to be categorized into one damage type only (dent or nick) or labeling must be done with a likely one. In either case this becomes a source of possible false negatives. Therefore, we decided to merge two classes into one nick_or_dent class that should decrease false positives.

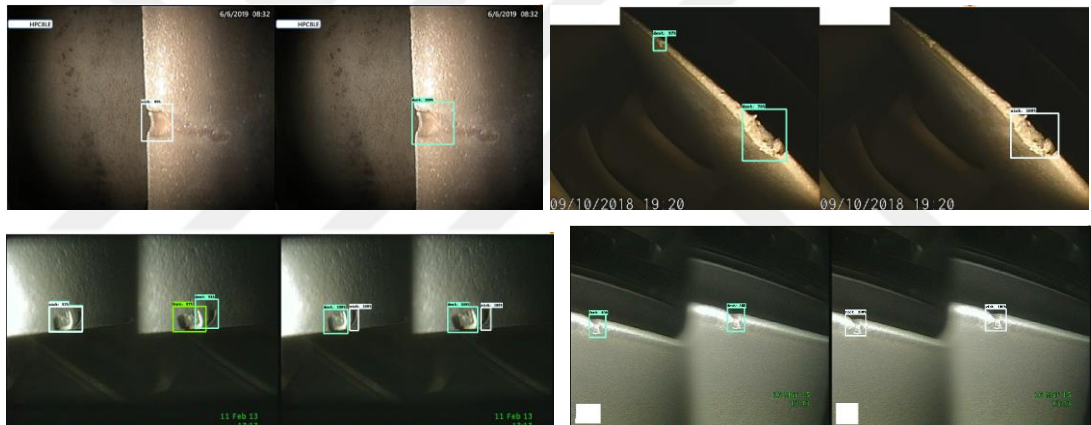


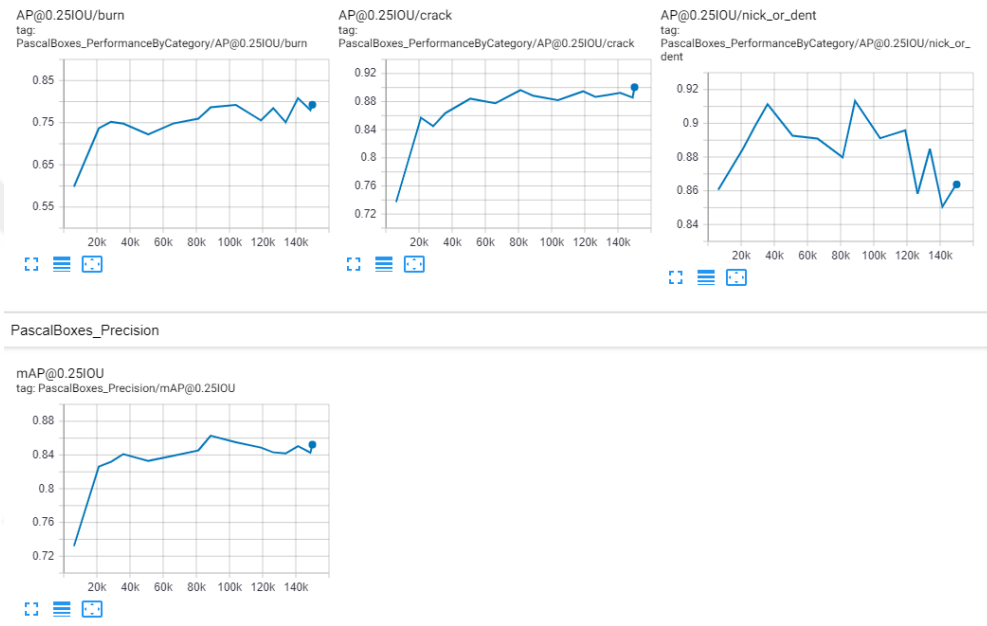
Figure 4.3. 4 image groups with nick/dent misclassifications. In each image group right image has the ground truth label and left image has the detection label.

After the merge of the two classes, training the model for 3 damage classes (burn, crack, nick_or_dent) by starting the weights from pre-trained checkpoint with COCO data set, random horizontal flip, random rotation (90 degrees) data augmentation methods, dropout with 0.5 probability, L2 regularization on both first stage box predictor (RPN) and second stage box classifier and finally 0.5 confidence score threshold, the maximum mAP value reached 86% in 88.69K steps (see Figure 4.4).

² Turbine blade blending is the process of removing and polishing damaged areas from effected components (from <https://www.advancedturbinesupport.com/blending-services/>, accessed on December 9, 2022)

This is a 5% increase compared to the training with 4 classes. “nick_or_dent” AP reached 91%, which is expected, due to less misclassifications but also due to more training images for the combined class. On the other hand, there is a slight decrease (2%) in burn maximum AP which could be either due to the data imbalance occurred after merging nick and dent classes or due to a deviation in the input data set as a result of random data augmentation behavior explained in the section 3.5.1.

(a)



(b)

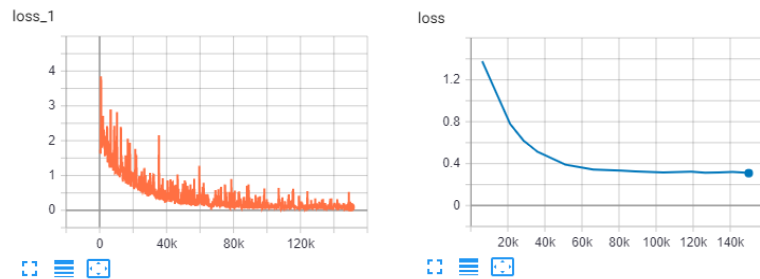
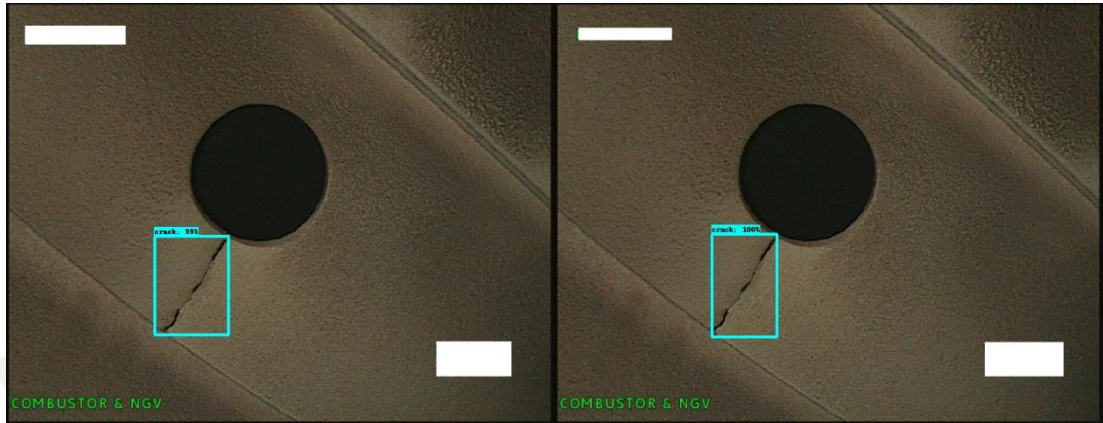


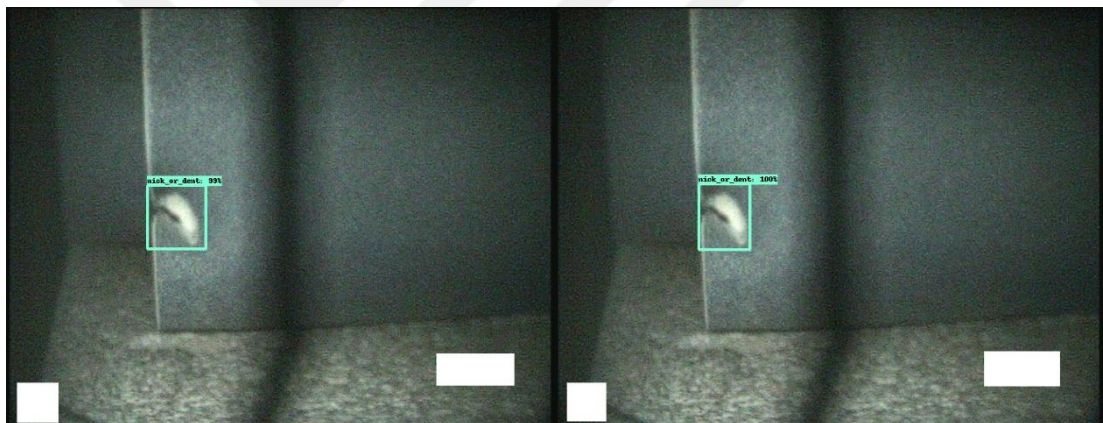
Figure 4.4. (a) AP and mAP of 3 damage classes crack, burn, nick_or_dent at IoU=0.25 and confidence score=0.50 (b) Training loss at the left and validation (total) loss at the right.

Analyzing the damage detection results, one could clearly see the success is quite high when there is single type of damage in the image. Even in case of multiple damages

of the same type the framework performed quite well in detecting the damages (see Figure 4.5 and Figure 4.6 for samples per damage type). In the case of images with different damage types, there are mixed results. Whereas some quite complicated nested damages were detected successfully (see Figure 4.7), in other images not all



(a) Crack detection sample: Right image the ground truth, left image the detection

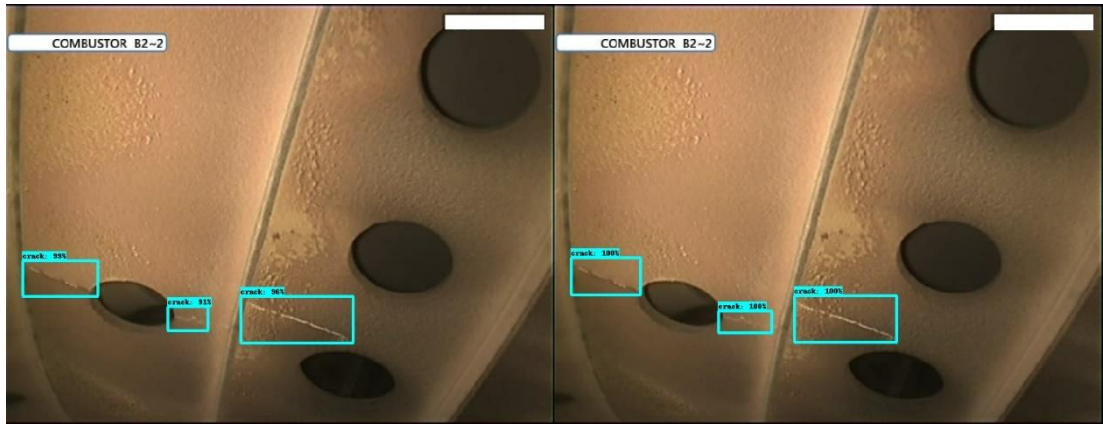


(b) Nick detection sample: Right image the ground truth, left image the detection

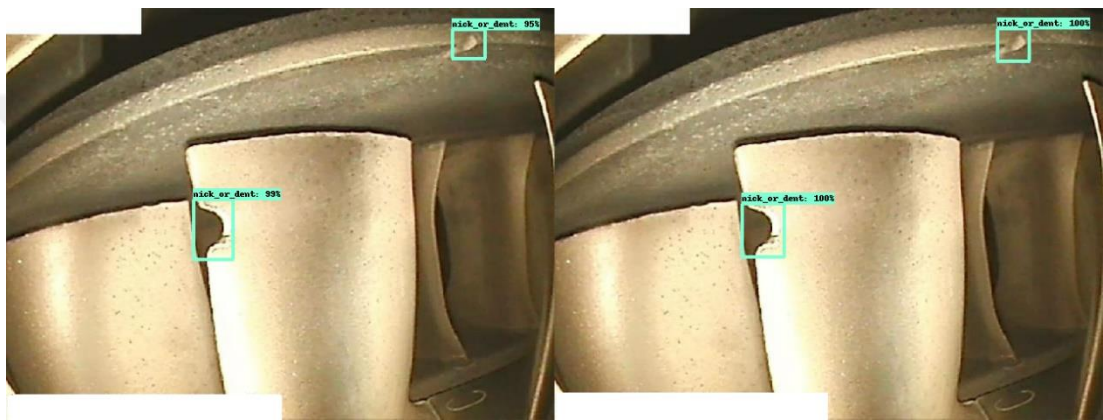


(c) Burn detection sample. Right image the ground truth, left image the detection

Figure 4.5. Detection samples of single damage per damage type.



(a) Multiple crack detection sample. Right image the ground truth, left image the detection

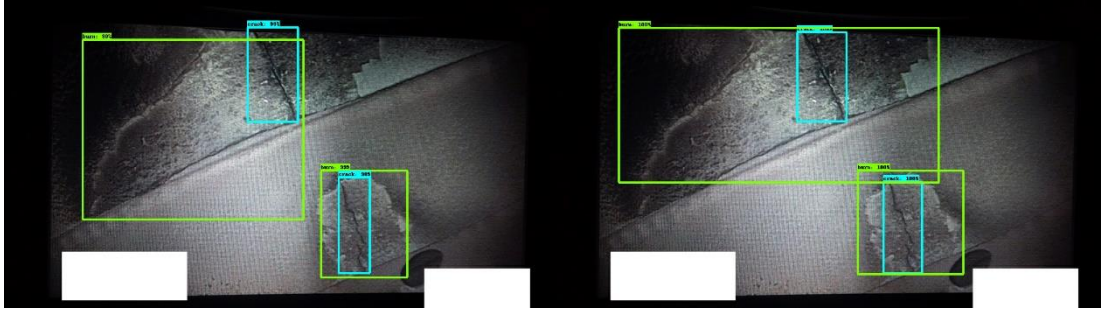


(b) Multiple nick_or_dent detection sample. Right image the ground truth, left image the detection

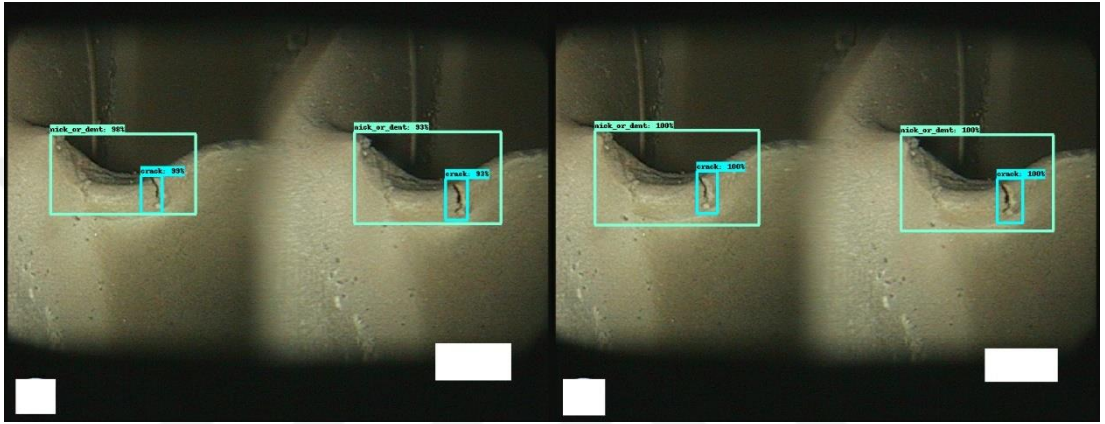


(c) Multiple burn detection sample. Right image the ground truth, left image the detection

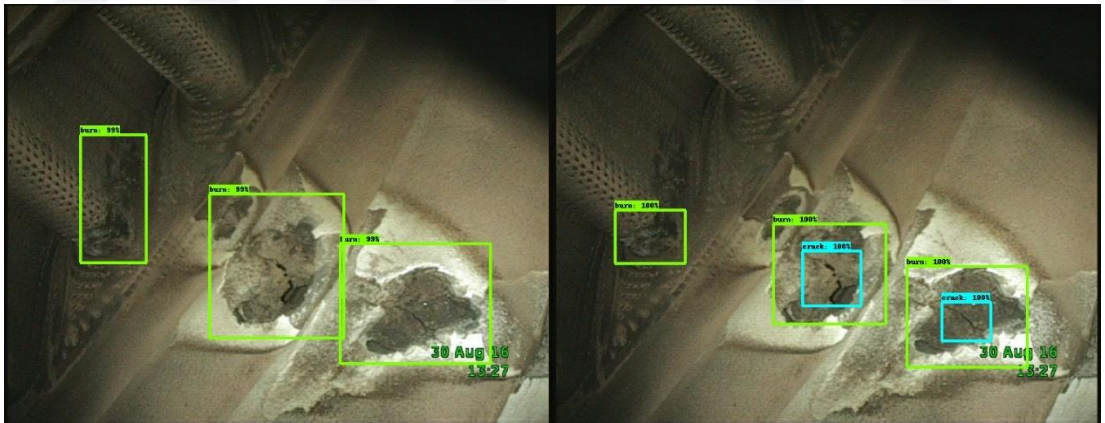
Figure 4.6. Multiple detection samples of the same damage type.



(a) Complex image successful detection sample. Right image the ground truth, left image the detection



(b) Complex image successful detection sample. Right image the ground truth, left image the detection



(c) Complex image partial successful detection sample. Burns were detected successfully whereas crack damages are undetected. Right image the ground truth, left image the detection

Figure 4.7. Detection samples for multiple damages of different types.

damages were detected (false negatives). This is interpreted as insufficient data to train the pattern (as the model complexity increases when number of classes increases) is available. This conclusion is made due to the fact that when trained/validated for a

single damage class the validation results of individual damage classes are quite high (for crack maximum validation AP is 95.99% when trained individually).

The impact of reducing the confidence score threshold was experimented by reducing the value from 0.5 to 0.25 so that more detections are expected to come into the play. Crack AP value had a slight increase whereas burn AP had a slight decrease and in opposite to one may expect the mAP had a slight decrease to 85%.

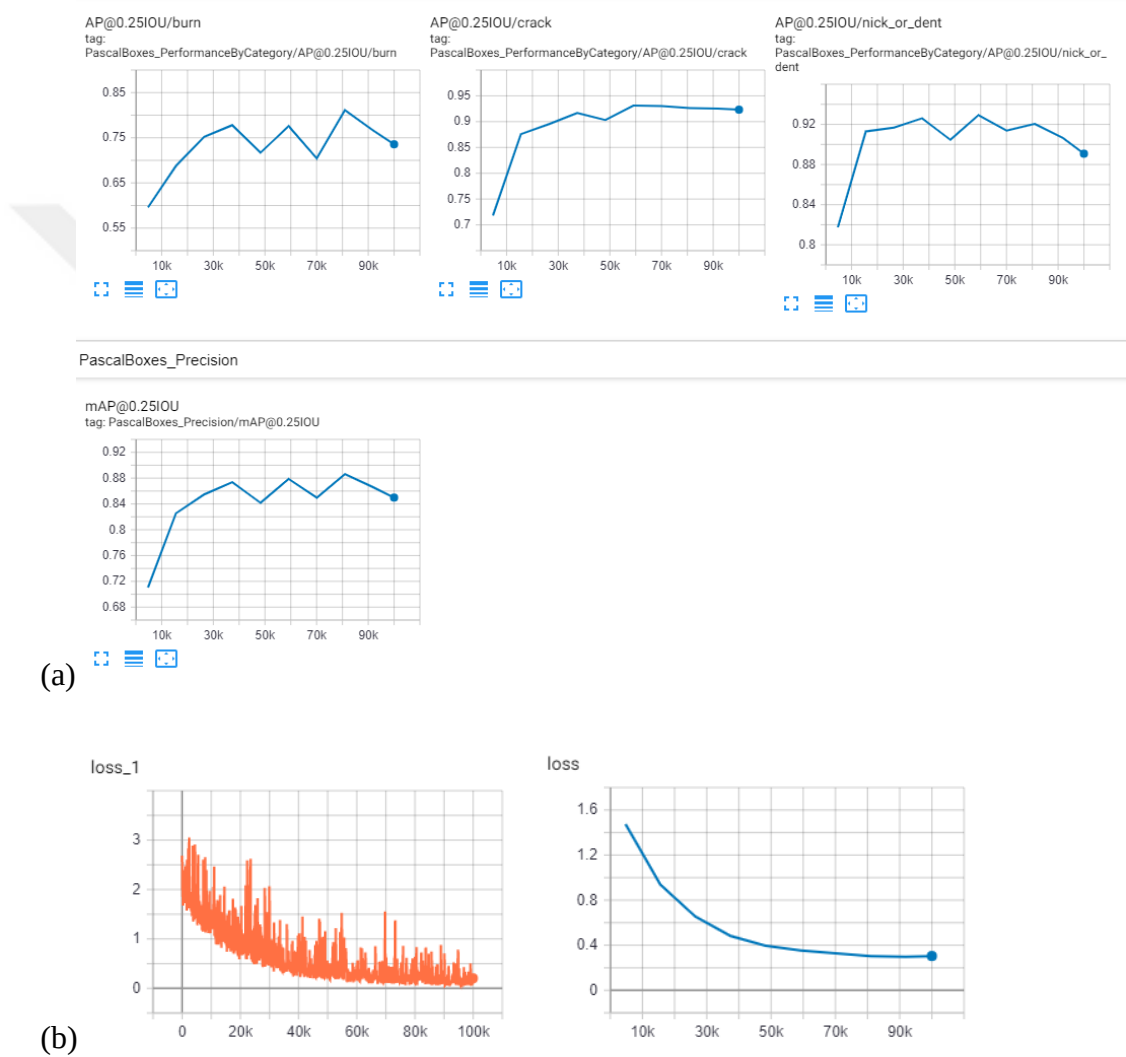


Figure 4.8. (a) AP and mAP of 3 damage classes crack, burn, nick_or_dent at IoU=0.25 and confidence score=0.50 when random vertical flip added (b) Training loss at the left and validation (total) loss at the right.

As another accuracy improvement round by increasing the data set, random vertical flip data augmentation method was added on top of random rotation (90 degrees) and random horizontal flip while keeping the same configuration that led to 86% mAP. This led to 88.6% maximum mAP in 80.95K steps (see Figure 4.8.). All three damage

classes showed an increase at the same time. This is about 2.5% increase in the maximum mAP compared to the settings without random vertical flip which shows increasing the data set can lead to a better performance.

Additional experimentations were made to figure out the individual impact of image preprocessing methods channel mean subtraction (CMS) (URL-19) and contrast limited adaptive histogram equalization (CLACHE) (Pizer et al., 1987) as well as some data augmentation methods on the accuracy of the model.

The summary is provided in Figure 4.9.

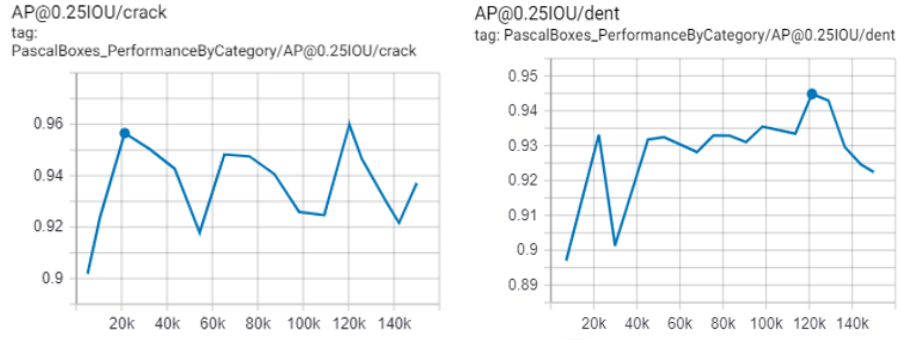
Preprocessing/Augmentation method	Crack AP@0.25IOU	Burn AP@0.25IOU	Nick_or_Dent AP@0.25IOU	mAP@0.25IOU
Channel mean subtraction	↔	↑	↓	↔
Histogram equalization (grayscale)	↓	↓	* ↑	↓
Random adjust brightness	↑	↓	↑	* ↔
Random adjust contrast	↑	↓	* ↑	* ↔
Random rotation (90 degrees)	↑	↑	* ↓	* ↑
Random horizontal flip	↑	↑	↔	↑

Figure 4.9. Impact of image preprocessing and data augmentation methods on accuracy individual AP and mean AP values. (*) indicates a slight change ($\leq 1.5\%$).

Channel mean subtraction caused an increase in the burn AP only whereas the mAP value did not change. However, the same image preprocessing method increased the mAP value when the model was trained with no transfer learning. This indicates that the method contributes when the mAP is low and there is room for improvement. However, here the model already reached its generalization upper limit for crack and nick_or_dent for the given data set. Burn AP is still low and it is likely that CMS contributes making feature extraction for burn easier. CLACHE method even decreased the mAP value, whereas only for nick_or_dent there was slight increase. Out of data augmentation methods only random rotation and random horizontal flip made significant increase in the mAP values which is similar to the findings of (Wang, 2020).

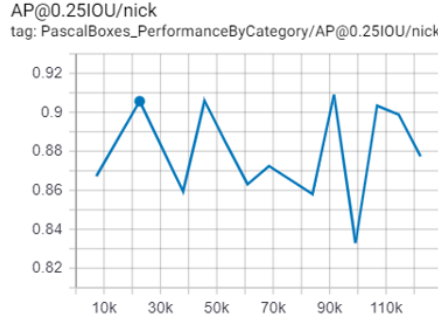
Finally, when the model is trained/validated for individual damage classes, as shown in Figure 4.10. the model gives better results than combined training/validation values. This is an expected result since increasing the number of classes at the same time

increases the complexity, therefore the number of parameters to train. In addition, data amount per class as well as order become parameters impacting the multi-class model

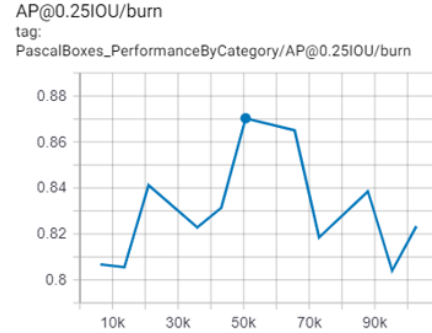


(a) AP for crack only training

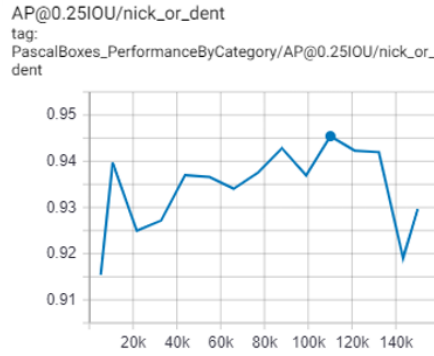
(b) AP for dent only training/validation



(c) AP for nick only



(d) AP for burn only



(e) AP for nick_or_dent only

Figure 4.10. Average precision when the model is trained/validated per class individually.

detection performance. When trained/validated individually, crack AP reaches 95.99%, dent AP reaches 94.48%, nick AP reaches 90.57%, and burn AP reaches 87.03%. Finally, merged nick_or_dent class AP gives 94.54% AP.

5. CONCLUSION and FUTURE WORK

The proposed framework implements Faster R-CNN architecture which is one of the mainstream architectures in the image recognition domain. It supports crack, burn and nick/dent damage types across all turbofan engine modules which is a differentiator from similar studies. It achieved 88.61% mAP@0.25IOU despite i) comparably small data set ii) having been trained and validated with real borescope images from the field in the range of moderate to high level complexity. We believe keeping the complex images in the scope makes our framework closer to the field reality (see Appendix C-1).

Table 5.1 provides summary of the comparison with the most similar studies. Among the similar studies from the same domain, no other study has been encountered that employs Faster R-CNN architecture with Inception v2. In terms of accuracy, Ren et al (2020) report quite higher accuracy than our study however, it is limited to classification (no localization) of a single damage type in an image and supports aircraft propeller bores only whereas our study supports detection of multi class damage types in a single image and the scope covers all engine modules. In addition, ensembles are computationally expensive. Wong et al. (2021) report slightly higher accuracy in addition to its instance segmentation support (vs bounding boxes), however it is limited to the rotor blades only, does not support burn damage type and more importantly reported results rely on quite similar images within the data set (a single video with very similar blades). Shen et al.'s study provides significantly higher accuracy than our study in addition to its object segmentation support. However, nick/dent support is not available and more information is needed to be able to compare performance over all engine modules.

The target audience of the implemented framework are maintenance repair organizations providing aircraft engine borescope inspection service to the airlines as well as the airlines themselves. The framework is suggested as a support tool for borescope inspection, it does not aim – also not realistic - to replace human inspectors. In this way, it could help to minimize subjectivity and the risk of missing a damage during the inspection process which could cause a snowball effect. In addition, it is expected to reduce the inspection duration and enable better maintenance planning.

This would lead to shorter parking time of an aircraft which leads to cost saving for an airline.

Table 5.1. Comparison with similar studies.

Author(s)	Scope	Type	Architecture	Data set	Accuracy
Svensén et al. (2018)	Engine part recognition (Mixer, Combustor, Fuel Nozzle, HPT Blade and Other engine hot parts)	classification	VGG16	4722 images	77%
Shen et al. (2019)	Crack, burn detection with semantic segmentation –no info about supported engine parts!	detection	Fully Convolutional Network (FCN) adopted from VGG16	1443 images	>97%
Ren et al (2020)	Crack, corrosion, scratch, and gouge classification on aircraft propeller blade bores only	classification	Ensemble of NNs (ResNet, GoogleNet, VGG)	600 images	>99%
Wong et al. (2021)	Surface-damage, Material-separation and Material-deformation detection on rotor blades	detection	Mask R-CNN with ResNet50-FPN	89 training and 15 testing images from a single video feed	90%
Our study	Crack, burn, nick/dent detection across all engine modules (parts)	detection	Faster RCNN with Inception v2	924 images	%88 mAP

The framework can run on a standard laptop for making inference. It is currently being tested with images only, however it theoretically supports detection from videos as well. Testing detection from borescope videos is one of the future work items. In addition, support for further damage types is possible and is recommended if required data set can be obtained. In case damage type support set can be extended then using the taxonomy suggested in (Aust and Pons, 2019) would make sense.

Finally, it would be interesting to investigate ways (like (Okajima and Sadamasa, 2019)) to combine the implemented deep learning model with rule-based system having a domain knowledge base which may offer new perspective to influence

especially the damage classification path and through this offer a better classification accuracy.



REFERENCES

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems, 12th USENIX Symposium on Operating Systems Design and Implementation, Savannah, USA, Proceedings, 16, 265–283.
- Aust, J. and Pons, D., 2019. Taxonomy of gas turbine blade defects. *Aerospace*, 6, 5, 58.
- Burgueño Salas, E., 2022. Commercial aircraft engine manufacturers: global market share 2020. URL Address: <https://www.statista.com/statistics/1099835/global-aircraft-engine-manufacturer-market-share/> Date of Access: December 22, 2022.
- Chollet, F., 2018. Keras: The python deep learning library. URL Address: <https://keras.io/> Date of Access: September 1, 2022.
- Deng, J., Dong, W., Socher, R., Li, L.J., Li, K. and Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database, IEEE conference on computer vision and pattern recognition, Miami, USA, Proceedings, 248-255.
- Drury, C. G. and Watson J., 2001. Human Factors Good Practices In Borescope Inspection. URL Address: https://www.researchgate.net/profile/Colin-Drury/publication/251789106_HUMAN_FACTORS_GOOD_PRACTICES_IN_BORESCOPE_INSPECTION/links/02e7e529fb3048c3bb000000/HUMAN-FACTORS-GOOD-PRACTICES-IN-BORESCOPE-INSPECTION.pdf Date of Access: December 22, 2022.
- Ebersole, J.S. and Pedley, T.A., 2003. Current practice of clinical electroencephalography, 3rd edition, Lippincott Williams & Wilkins, Philadelphia, USA.
- Everingham, M. and Winn, J., 2011. The pascal visual object classes challenge 2011 (voc2011) development kit, Pattern Analysis, Statistical Modelling and Computational Learning. URL Address: http://host.robots.ox.ac.uk/pascal/VOC/voc2011/devkit_doc.pdf Date of Access: December 2, 2022.
- Federal Aviation Administration (FAA)/Aviation Supplies & Academics (ASA), 2018. Aviation Maintenance Technician Handbook-Powerplant, Vol 2. URL Address: https://www.faa.gov/sites/faa.gov/files/regulations_policies/handbooks_manuals/aviation/FAA-H-8083-32-AMT-Powerplant-Vol-2.pdf Date of Access: December 22, 2022.
- Feng, J., He, X., Teng, Q., Ren, C., Chen, H. and Li, Y., 2019. Reconstruction of porous media from extremely limited information using conditional generative adversarial networks. *Physical Review E*, 100, 3, 033308.

- Fu, G., Sun, P., Zhu, W., Yang, J., Cao, Y., Yang, M.Y. and Cao, Y., 2019. A deep-learning-based approach for fast and robust steel surface defects classification. *Optics and Lasers in Engineering*, 121, 397-405.
- Fumo D., 2017. A Gentle Introduction To Neural Networks Series — Part 1, URL Address: <https://towardsdatascience.com/a-gentle-introduction-to-neural-networks-series-part-1-2b90b87795bc> Date of Access: December 7, 2022.
- Girshick, R., 2015. Fast R-CNN, IEEE international conference on computer vision (ICCV), Santiago, Chile, Proceedings, 1440-1448.
- Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation, IEEE conference on computer vision and pattern recognition, Columbus, USA, Proceedings, 580-587.
- Gurney, K., 2018. An introduction to neural networks (Book), URL Address: http://www.macs.hw.ac.uk/~yjc32/project/ref-NN/Gurney_et_al.pdf Date of Access: December 7, 2022.
- He, K., Gkioxari, G., Dollár, P. and Girshick, R., 2017. Mask R-CNN, IEEE international conference on computer vision, Venice, Italy, Proceedings, 2961-2969.
- He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition, IEEE conference on computer vision and pattern recognition, Las Vegas, USA, Proceedings, 770-778.
- Henrich, I., 2021. How does a turbofan engine work?, URL Address: <https://aeroreport.de/en/good-to-know/how-does-a-turbofan-engine-work> Date of Access: December 22, 2022.
- Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J. and Keutzer, K., 2016. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size. arXiv preprint arXiv:1602.07360.
- Karpen, T., Bendall, C. and Salvati, J., 2004. Method and system for automatically detecting defects in remote video inspection applications, U.S. Patent No US20040183900A1. Washington, DC: U.S. Patent and Trademark Office.
- Kirillov, A., Wu, Y., He, K. and Girshick, R., 2020. Pointrend: Image segmentation as rendering. IEEE/CVF conference on computer vision and pattern recognition, Proceedings, 9799-9808.
- Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2017. Imagenet classification with deep convolutional neural networks, *Communications of the ACM*, 60, 6, 84-90.
- Leong, M.C., Prasad, D.K., Lee, Y.T. and Lin, F., 2020. Semi-CNN architecture for effective spatio-temporal learning in action recognition, *Applied Sciences*, 10,

- Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B. and Belongie, S., 2017. Feature pyramid networks for object detection, IEEE conference on computer vision and pattern recognition, Hawaii, USA, Proceedings, 2117-2125.
- Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C.L., 2014. Microsoft COCO: Common objects in context, 13th European conference on computer vision, Zurich, Switzerland, Proceedings, 13, 740-755.
- Morrison, M., 2015. The power list: top 10 delivered commercial turbofans. URL Address: <https://www.flightglobal.com/the-power-list-top-10-delivered-commercial-turbofans/116576.article> Date of Access: December 22, 2022.
- Nickles, G., Him, H., Koenig, S., Gramopadhye, A. and Melloy, B., 2019. A Descriptive Model of Aircraft Inspection Activities. URL Address: https://www.faa.gov/about/initiatives/maintenance_hf/library/documents/media/human_factors_maintenance/a_descriptive_model_of_aircraft_inspection_activities.pdf Date of Access: September 1, 2022.
- Okajima, Y. and Sadamasa, K., 2019. Deep neural networks constrained by decision rules, AAAI Conference on Artificial Intelligence, Hawaii, USA, Proceedings, 33, 1, 2496-2505.
- O'Shea, K. and Nash, R., 2015. An introduction to convolutional neural networks. arXiv preprint arXiv:1511.08458.
- Padilla, R., Netto, S.L. and Da Silva, E.A., 2020. A survey on performance metrics for object-detection algorithms, International conference on systems, signals and image processing (IWSSIP), 237-242.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L. and Desmaison, A., 2019. Pytorch: An imperative style, high-performance deep learning library, 33rd international conference on neural information processing systems, 8026–8037.
- Pizer, S.M., Amburn, E.P., Austin, J.D., Cromartie, R., Geselowitz, A., Greer, T., ter Haar Romeny, B., Zimmerman, J.B. and Zuiderveld, K., 1987. Adaptive histogram equalization and its variations, Computer vision, graphics, and image processing, 39, 3, 355-368.
- Ramachandran, P., Zoph, B. and Le, Q.V., 2017. Searching for activation functions, Sixth International Conference on Learning Representations, Vancouver, Canada, Proceedings, arXiv preprint arXiv:1710.05941.
- Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection, IEEE conference on computer vision and pattern recognition, Las Vegas, USA, Proceedings, 779-788.

- Ren, I., Zahiri, F., Sutton, G., Kurfess, T. and Saldana, C., 2020. A Deep Ensemble Classifier for Surface Defect Detection in Aircraft Visual Inspection, *Smart and Sustainable Manufacturing Systems*, 4, 1.
- Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. and Berg, A.C., 2015. Imagenet large scale visual recognition challenge, *International journal of computer vision*, 115, 3, 211-252.
- Sayadpishe, S., Rezaei, A., 2018. Strategy & investigating of different kinds of critical defects recognition in aircraft engine, 5th Iranian International NDT Conference, Tehran, Iran, Proceedings, URL Address: <https://www.ndt.net/article/IranNDT2018/papers/1070-IRNDT-Mr-rezaei-IRNDT2018.pdf> Date of Access: December 22, 2022.
- Shen, Z., Wan, X., Ye, F., Guan, X. and Liu, S., 2019. Deep learning based framework for automatic damage detection in aircraft engine borescope inspection, *International Conference on Computing, Networking and Communications*, Honolulu, USA, Proceedings, 1005-1010.
- Simonyan, K. and Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition, 3rd International Conference on Learning Representations, San Diego, USA, Proceedings, 1-14.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting, *The journal of machine learning research*, 15, 1, 1929-1958.
- Svensen, M., Hardwick, D. and Powrie, H., 2018, July. Deep neural networks analysis of borescope images, *European Conference of the PHM Society*, Utrecht, The Netherlands, Proceedings, 3-6.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A., 2015. Going deeper with convolutions, *IEEE conference on computer vision and pattern recognition*, Boston, USA, Proceedings, 1-9.
- Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z., 2016. Rethinking the inception architecture for computer vision, *IEEE conference on computer vision and pattern recognition*, Las Vegas, USA, Proceedings, 2818-2826.
- Tzutalin, D., 2015. LabelImg GitHub repository. URL Address: <https://github.com/heartexlabs/labelImg> Date of Access: December 9, 2022.
- Vembar, D., Sadasivan, S., Duchowski, A.T., Stringfellow, P. and Gramopadhye, A., 2005. Design of a virtual borescope: a presence study, 11th International

Conference on Human-Computer Interaction, Las Vegas, USA, Proceedings, 9, 22-27.

Wang, J. and Perez, L., 2017. The effectiveness of data augmentation in image classification using deep learning, Convolutional Neural Networks Visual Recognition, 11, 1-8.

Wang, Z., Yang, J., Jiang, H. and Fan, X., 2020. CNN training with twenty samples for crack detection via data augmentation. Sensors, 20, 17, 4849.

Wong, C.Y., Seshadri, P. and Parks, G.T., 2021. Automatic borescope damage assessments for gas turbine blades via deep learning, AIAA Scitech 2021 Forum. URL Address: <https://doi.org/10.2514/6.2021-1488> Date of Access: December 22, 2022.

Yildirim, S., 2020. L1 and L2 Regularization — Explained. URL Address: <https://towardsdatascience.com/l1-and-l2-regularization-explained-874c3b03f668> Date of Access: September 1, 2022.

URL-1, <<https://www.statista.com/statistics/262971/aircraft-fleets-by-region-worldwide/>> , Date of Access: November 19, 2022.

URL-2, <<https://aitproducts.com/iflex-fiberscope.html>>, Date of Access: November 19, 2022.

URL-3, <<https://aitproducts.com/products/video-borescopes/iris5-videoscope-video-borescope.html>>, Date of Access: November 19, 2022.

URL-4, <<https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/tensorflow-1.14/>>, Date of Access: September 1, 2022.

URL-5, <<https://skybrary.aero/articles/turbofan-engine>>, Date of Access: December 22, 2022.

URL-6, <<https://en.wikipedia.org/wiki/Turbofan>>, Date of Access: December 22, 2022.

URL-7, <<https://www.aerotime.aero/articles/32417-who-are-the-world-s-largest-aircraft-engine-manufacturers>>, Date of Access: December 22, 2022

URL-8, CC BY-SA 3.0, <<https://en.wikipedia.org/w/index.php?curid=21007866>>, Date of Access: December 22, 2022.

URL-9, <<https://www.kaggle.com/datasets/kaustubhdikshit/neu-surface-defect-database>>, Date of Access: November 19, 2022.

URL-10, <<https://lucidar.me/en/neural-networks/how-popular-are-neural-networks-over-the-years/>>, Date of Access: November 19, 2022.

URL-11, <<https://deepai.org/machine-learning-glossary-and-terms/relu>>, Date of Access: December 07, 2022.

URL-12, <<https://deepai.org/machine-learning-glossary-and-terms/activation-function>>, Date of Access: January 18, 2023.

URL-13, <https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf1_detection_zoo.md>, Date of Access: November 19, 2022.

URL-14, <https://github.com/tensorflow/models/blob/master/research/object_detection/samples/configs/faster_rcnn_inception_v2_coco.config>, Date of Access: November 19, 2022.

URL-15, <<https://www.tensorflow.org/tensorboard>>, Date of Access: December 9, 2022.

URL-16, <https://github.com/tensorflow/models/blob/master/research/object_detection/protos/preprocessor.proto>, Date of Access: November 19, 2022

URL-17, <<https://www.truba.gov.tr/index.php/en/main-page/>>, Date of Access: June 29, 2020

URL-18, <<https://slurm.schedmd.com/overview.html>>, Date of Access: December 01, 2022

URL-19, <<https://cs231n.github.io/neural-networks-2/#datapre>>, Date of Access: November 19, 2022

APPENDIX LIST

APPENDIX A-1

APPENDIX A-2

APPENDIX-B

APPENDIX C-1



APPENDIX A-1: TFOF API pipeline config file faster_rcnn_inception.config

Faster R-CNN with Inception v2, configuration for MSCOCO Dataset.
Users should configure the fine_tune_checkpoint field in the train config as
well as the label_map_path and input_path fields in the train_input_reader and
eval_input_reader. Search for "PATH_TO_BE_CONFIGURED" to find the fields
that should be configured.

```
model {  
  faster_rcnn {  
    num_classes: 3  
    image_resizer {  
      keep_aspect_ratio_resizer {  
        min_dimension: 600  
        max_dimension: 1024  
      }  
    }  
    feature_extractor {  
      type: 'faster_rcnn_inception_v2'  
      first_stage_features_stride: 16  
    }  
    first_stage_anchor_generator {  
      grid_anchor_generator {  
        scales: [0.25, 0.5, 1.0, 2.0]  
        aspect_ratios: [0.5, 1.0, 2.0]  
        height_stride: 16  
        width_stride: 16  
      }  
    }  
    first_stage_box_predictor_conv_hyperparams {  
      op: CONV
```

```

regularizer {
  l2_regularizer {
    weight: 0.01
  }
}

initializer {
  truncated_normal_initializer {
    stddev: 0.01
  }
}

first_stage_nms_score_threshold: 0.0
first_stage_nms_iou_threshold: 0.6
first_stage_max_proposals: 300
first_stage_localization_loss_weight: 2.0
first_stage_objectness_loss_weight: 1.0
initial_crop_size: 14
maxpool_kernel_size: 2
maxpool_stride: 2
second_stage_box_predictor {
  mask_rcnn_box_predictor {
    use_dropout: true
    dropout_keep_probability: 0.5
    fc_hyperparams {
      op: FC
      regularizer {
        l2_regularizer {
          weight: 0.01
        }
      }
    }
  }
}

```

```

    }
    initializer {
      variance_scaling_initializer {
        factor: 1.0
        uniform: true
        mode: FAN_AVG
      }
    }
  }
}
second_stage_post_processing {
  batch_non_max_suppression {
    score_threshold: 0.0
    iou_threshold: 0.5
    max_detections_per_class: 100
    max_total_detections: 300
  }
  score_converter: SOFTMAX
}
second_stage_localization_loss_weight: 2.0
second_stage_classification_loss_weight: 1.0
}
}
train_config: {
  batch_size: 1
  optimizer {
    momentum_optimizer: {
      learning_rate: {

```

```

manual_step_learning_rate {
  initial_learning_rate: 0.0003
  schedule {
    step: 900000
    learning_rate: .00003
  }
  schedule {
    step: 1200000
    learning_rate: .000003
  }
}

momentum_optimizer_value: 0.9
}

use_moving_average: false
}

gradient_clipping_by_norm: 10.0

fine_tune_checkpoint:
"/truba/home/iuzun/ts_bsi_resnet152_p_gpu1/models/research/object_detection/faster_rcnn_inception_v2_coco_2018_01_28/model.ckpt"

from_detection_checkpoint: true

load_all_detection_checkpoint_vars: true

num_steps: 150000

data_augmentation_options {
  random_horizontal_flip {
  }
}

data_augmentation_options {
  random_rotation90 {
  }
}

```

```

    }
}
train_input_reader: {
  tf_record_input_reader {
    input_path:
"/truba/home/iuzun/ts_bsi_resnet152_p_gpu1/models/research/object_detection/train
.record"
  }
  label_map_path:
"/truba/home/iuzun/ts_bsi_resnet152_p_gpu1/models/research/object_detection/train
ing/labelmap.pbtxt"
}
eval_config: {
  metrics_set: "pascal_voc_detection_metrics"
  num_examples: 94
  min_score_threshold: 0.50
  num_visualizations: 94
  max_evals: 94
}
eval_input_reader: {
  tf_record_input_reader {
    input_path:
"/truba/home/iuzun/ts_bsi_resnet152_p_gpu1/models/research/object_detection/test.r
ecord"
  }
  label_map_path:
"/truba/home/iuzun/ts_bsi_resnet152_p_gpu1/models/research/object_detection/train
ing/labelmap.pbtxt"
  shuffle: false
  num_readers: 1
}

```

APPENDIX A-2 : TFOD API labelmap.pbtxt

```
item {  
  id: 1  
  name: 'burn'  
}
```

```
item {  
  id: 2  
  name: 'crack'  
}
```

```
item {  
  id: 3  
  name: 'nick_or_dent'  
}
```

APPENDIX B : SLURM script file

```
#!/bin/bash

#SBATCH -p akya-cuda
#SBATCH -A iuzun
#SBATCH -J bsi_od_proc
#SBATCH -N 1 # hesabin dagitilacagi node adedi
#SBATCH -n 20 # is icin toplamda kullanılacak cekirdek adeti
#SBATCH --gres=gpu:2
#SBATCH --time=20:00:00
#SBATCH --mail-type=ALL
#SBATCH --mail-user=uzunismail2016@gmail.com
#SBATCH --workdir=/truba_scratch/iuzun

now=$(date +"%T")
echo "Start time : $now"

source activate tensorflow_gpu1
module load centos7.3/lib/cuda/10.0

cd /truba/home/iuzun/ts_bsi_resnet152_p_gpu1/models/research/object_detection

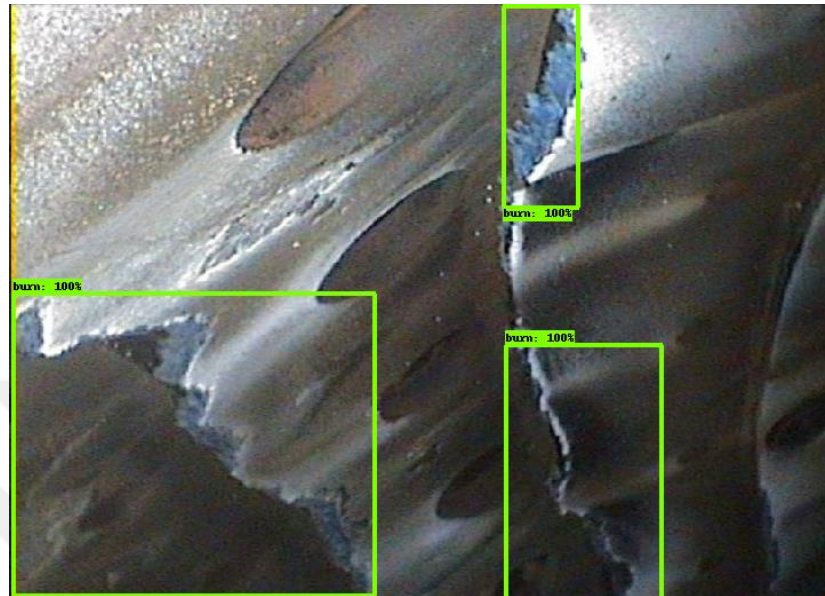
python      model_main.py      --logtostderr      --model_dir=training/      --
pipeline_config_path=training/faster_rcnn_inception.config

now=$(date +"%T")
echo "End time : $now"

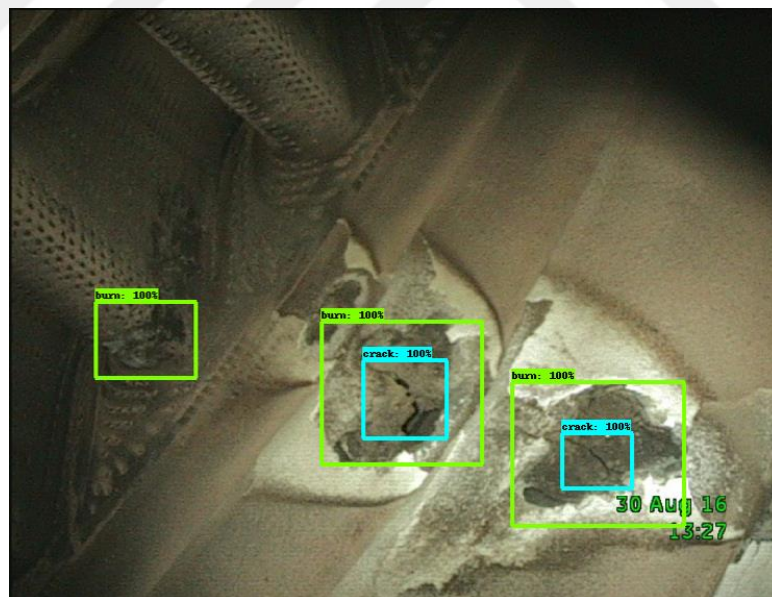
sstat --j $SLURM_JOB_ID # isin o ana kadar kullanmis oldugu sistem kaynaklari
echo "SLURM_NODELIST $SLURM_NODELIST"
echo "NUMBER OF CORES $SLURM_NTASKS"
exit
```

APPENDIX C-1 : Complex image and label samples

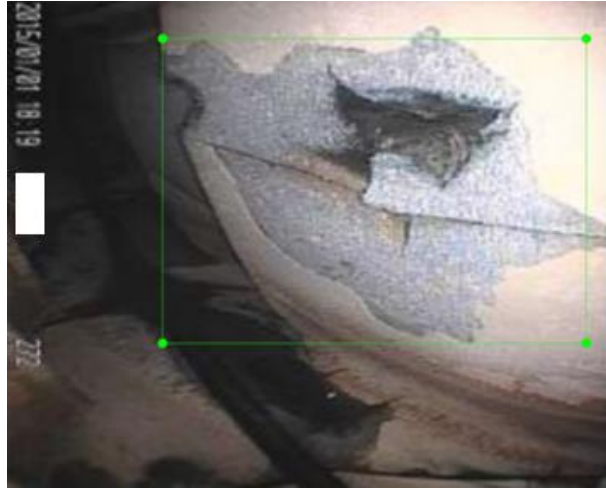
Note that the bounding boxes are the damage labels not detections.



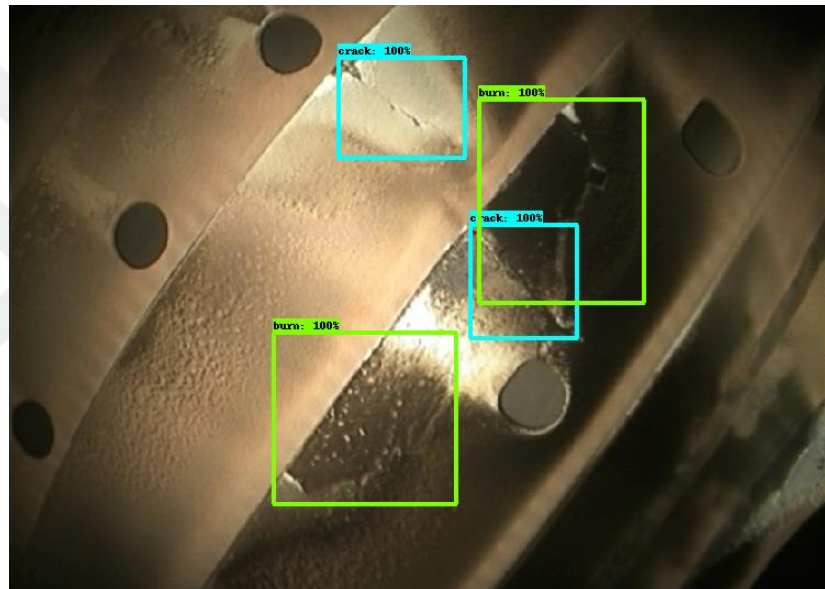
(a) Multiple burns on edges in the same image



(b) Multiple burn and nested crack damage labels in the same image.



(c) Deformed background



(d) Multiple burn and cracks with overlapping areas in the same image



(e) Multiple burn and cracks with overlapping areas as well as huge burn with no natural boundaries



(f) Quite deformed background

CURRICULUM VITAE

Name and Surname : Ismail Uzun

EDUCATION

B.Sc. : Hacettepe University, Computer Science and Engineering, 1995-2000

M.Sc. (1) : Bilkent University, Electrical and Electronics Engineering, 2001-2004

M.Sc. (2) : Duisburg-Essen University, Germany, Computer Engineering, 2003-2005

Ph.D. : Aksaray University, 2016-2023

PROFESSIONAL EXPERIENCE

1. Nokia Networks Germany, Solution Architect / Domain Technical Manager, 2014-2021
2. Nokia Solutions and Networks, Lisbon, Solution Architect, 2010-2014
3. Nokia Networks Germany, System Design Engineer, 2006-2010
4. Nomiko e.V. in Germany, Part-time Software Developer, 2005-2006
5. CyberSoft, Software Engineer, 2001-2004
6. Bilkent University Communications and Spectrum Management Research Center, Software Engineer, 2000-2001

REWARDS

Full tuition fee scholarship for the master study at Bilkent University