Introducing AI in Garment Fault Detection Using YOLOv5 to Reduce Bottleneck

by

Jasia Sanjana
22141069
Abdullah Al Muhit
18201024
Asma Zia
18101540

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
BRAC University
August 2023

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing a degree at Brac University.

2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.

3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.

4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Jasia

Jasia Sanjana

22141069

Muhit

Abdullah Al Muhit

18201024

Asma Zia

Asma Zia

18101540

Approval

The thesis/project titled "Introducing AI in Garment Fault Detection Using YOLOv5 to Reduce Bottleneck" submitted by

1. Jasia Sanjana (22141069)
2. Abdullah Al Muhit (18201024)
3. Asma Zia (18101540)

Of Fall, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 23, 2023.

Examining Committee:

Supervisor:
(Member)

Md. Khalilur Rhaman

Dr. Md. Khalilur Rhaman

Professor
Department of Computer Science and Engineering
BRAC University

Co-Supervisor:
(Member)

[illegible]

Jannatun Noor Mukta

Lecturer
Department of Computer Science and Engineering
Brac University

Ethics Statement

It is hereby proclaimed that all the comparisons made and conclusions drawn in this thesis are based on the findings obtained from the group's individual research on the topic. All relevant resources used have been acknowledged through proper citation. Appropriate steps have been taken to ensure the transparency of the analysis. This thesis has previously not been submitted in any form to any other institute.

Abstract

In order to reduce manpower and bottleneck in the inspection system of industrial garments, we explored the application of the YOLOv5 model using our very own dataset of defective clothing pieces. The economy of Bangladesh heavily depends on the garment industry. However, in this day and age of advanced technology, it is getting harder to have efficient manpower in the garment industry. Motivated to solve this problem, we decided to devise ways to explore AI implementations, particularly in the Bangladeshi garment industry system. Since there is no existing and efficient defective garment dataset specifically for our desired research work so we created our own dataset. This dataset has a total of 2,525 images and 7 different classes. By thoroughly analyzing our data from pre-processing to its performance after the application in the YOLOv5 model, we have tried to create a useful dataset. The models have achieved a good mean average precision across all 7 classes. Our research has only scratched a small surface of an area of interest where advanced AI and machine learning technologies can bring a lot more advancement.

Keywords: Primary Dataset, data analysis, computer vision, object detection algorithm, machine learning, YOLOv5.

Dedication)

Dedicated to the garment workers of Bangladesh, who work hard to ensure economical prosperity through Ready-made Garment sector.

Acknowledgement

We are thankful to Allah for our completion of thesis. We are also greatful to our supervisior Dr. Md. Khalilur Rhaman sir, Co-supervisor Ms. Jannaturn Noor Mukta ma'am, and Research assistant Sayantan Arko for their support and advice throughout our work.
And finally we express our heartfelt gratitude to our parents without whom we would not be able to come this far in life.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Our country Bangladesh holds third place in exporting garments products in the world [44]. Evidently, there are around five thousand garment factories in our country [14]. Around 20 million people in our country earn their livelihood by working in these garments and textile sectors **3**. As a result, our economy is heavily dependent on the manufacturing of textile and garments industries. Even with the setback of a few crises and COVID-19, this sector has had a great impact on our economy. A developing country like ours has been able to capture this sector vastly because of low labor costs. However, with the development of urban and rural areas has drawn a different scenario over recent years. Not only have technologies and equipment been developed, but also the work environments and opportunities have also been changed along with it. Consequently, working people are diverted toward a new set of circumstances and grab opportunities with it. Additionally, it is leading to a noticeable change in the garment and textile industries as it has led to a sharp decrease in the number of garment workers. In this circumstance, implementing AI can take care of this downside and create a passage through a new era of advancement. In the garment and textile industry, there are many procedures to follow from producing the products to ensuring quality. The products we get to come to our hand by being processed variously after it is manufactured. From raw material to making the end-most product requires gradual steps like laying, marking, cutting, stitching, checking, finishing, pressing and packaging, etc [29]. There might be several types of defects in each step of producing products. We will focus on common defect detected in workmanship that includes stitching defects, seaming defects, assembling defects, washing defects, pressing defects, printing defects, etc [31]. Effective garment scrutiny to detect defects work in a few steps. The first step is to inspect the product. Secondly, the defect is to be identified. Thirdly, while recording the data of error, it should be reported to the in-charge personnel. Next, the in charge will find out the responsible worker and repair the defects [32]. Thus, without ensuring the quality of the garment products, a certain entity might be abolished as it will not be able to make it in the long run. The demand for clothing has risen dramatically due to factors such as population growth, changing consumer attitudes toward fast fashion, and rising incomes in many countries. This demand is being met by developing countries, which have become major manufacturing centers for international fashion brands. Despite the availability of automation technology, many garment manufacturers in these countries continue to rely on manual labor, due to cheap labor costs and high initial investments in automation. However, the need for high-quality clothing and increased competition is pushing many manufacturers to adopt automation technology [10]. In our research we will try to inspect the defects by using AI techniques' sub-sectors such as deep learning and You Only Look Once (YOLOv5) which will help us to detect objects that are divided into grids. This algorithm is named that way as it needs to go through the media only once.

1.1 Research Statement

As mentioned above, work in a garment follows orderly procedures such as product design, fabric selection, and inspection, patternmaking, grading, marking, spreading, cutting, bundling, sewing, pressing or folding, finishing and detailing, dyeing and washing, quality control, etc [29]. In these operations, whether that is taken care of by machines or humans, some products may come out as faulty. Some of these defects are amendable, others are irrecoverable. While inspecting, the quality inspector must find those recoverable defects and send them for repair. Additionally, defects can be classified into three categories: critical, major, and minor defects. To start with, critical defects are errors that have breached compulsory regulations and put a threat to safety concerns. As this triggers safety concerns of the end users, these are considered undeniable errors which cause the cancellation of the entire batch of orders. For example, broken buttons or zippers, the presence of any sharp object, missing caution levels, insecure trims, drawstrings in kids' wear, etc. On the other hand, major defects are the ones that do not raise safety concerns but decrease the brand value and life cycle of the products. For instance, damaged garments, holes, open seams, loose buttons, main label spots, color defects, etc. Finally, minor defects are workmanship defects that are beyond the requirements that were to be fulfilled by the distributor according to the buyer. Such as washable dirt on garments, misprinting on labels, small threading or unstitching problem, etc. [12]. In our research, we will use Artificial Intelligence AI approaches to detect the defects in any of the steps. Although, many researchers have already conducted experiments and analyses in a few fields of these topics. Among them, fabric defect detection is one of the most common research subjects. Therefore, our main focus is on less explored concepts like the implementation of AI in the process of workmanship and the ways of handling defects. To elaborate, the defects detected in workmanship are pattern-making defects, spreading/laying and cutting defects, stitching defects, seaming defects, assembling defects, finishing and washing defects, printing defects, pressing defects, packing defects, etc. Firstly, pattern making is creating templates of patterns that will help garments to sew [7]. Some of the pattern-making defects are errors in measurements, misplaced parts, unaligned patterns, errors in pattern detection, see-through marking, etc. Secondly, spreading is unrolling the rolls of garments on a large table to cut them into small pieces of garments [40]. Some of the spreading and cutting defects are: drill marks, wear out edges, indelicate and inaccurate knife cuts, the unaligned and incorrect direction of the bundle, etc. Thirdly, stitching is a core element of sewing. Stitching defects include loose stitch, crooked stitch, torn stitch, wobbly buttons, uneven buttons, and button-hole, missed bar track, etc. Then, seaming is the way of attaching multiple clothes together using threads [22]. Some seaming defects are open seam, insecure seam, twisted seam, cracked seam, folded seam, unaligned and unmatched seam, uneven seam length, etc. Additionally, assembling is the process of joining parts together [42]. Instances of assembly defects are uneven or misshapen collar points or ends, incorrect or uneven button placket, twisted or uneven sleeve, and leg length, uneven or incorrect cuff and pockets, etc. Moreover, there are a few finishing and washing defects such as uncut threads, wash stains, fading or damage after washing, wash product mark, visible pen or chalk mark, etc. There may be some defects that are visible after printing which are: missing, misplaced, faulty or inaccurate design, mismatched coloring, etc. Again, pressing defects are burnt or wrinkled garments, visible water spots, etc. Lastly, packing-related defects occur while packaging the product for shipping. Such as inaccurate folding and labeling, incorrect quantity or size, etc. [31][12]

1.2 Research Objective

1.2.1 Garment Defects

Garment defects in industries are one of the main reasons for quality failure. Common garment defects that are observed and identified in the garment industry are: seam puckering, broken or open seam, seam slippage, broken stitch, skipped or drop stitch, loose or uncut thread, distorted knitting, and needle thread breakage. The required apparatuses are a garment inspection machine, cutting-sewing machine, inspection table, light source, paper, pencil, hardboard., calculator, etc. Reducing the number of defects in the production of trims and accessories is important because without reducing defects, it is not possible to increase the productivity and efficiency of the production process. The presence of defects in garments can lead to a significant loss in value for the company, as the products must be sold at a lower price. To prevent this, it is crucial for manufacturers to minimize fabric defects at every stage of the production process. One way to do this is by using an automated defect detection and identification system, which can improve product quality, enhance productivity and ultimately reduce costs associated with off-quality products. The detection of defects and remedying them immediately can lead to efficient garment manufacturing in the industry. all stages of the production process, from cutting and spreading to finishing, play a role in the occurrence of fabric defects. [13]

1.2.2 Bottleneck

Bangladesh has a huge manual labor force in the garment manufacturing industry. As a result, bottlenecks may exist in a system which can negatively impact production efficiency and capacity. A bottleneck is a point in the production process where the flow of work is slowed down or stopped due to a lack of resources or capacity. The line balancing technique can be an answer to this problem. Line balancing is a technique used to optimize the production process by ensuring that the workload is evenly distributed among the different workstations and that the flow of work is smooth and efficient. By using line balancing techniques, it is possible to reduce bottlenecks and improve the efficiency of the production process. This can be achieved by identifying and addressing the root causes of bottlenecks, such as uneven distribution of workload, lack of resources, and inadequate capacity. The implementation of line balancing techniques can lead to a reduction in cycle time, which is the time it takes to complete one cycle of the production process. This reduction in cycle time can increase production capacity, allowing the industry to produce more garments in a shorter period of time. Additionally, by using line-balancing techniques, the industry can reduce the costs associated with off-quality products and improve customer satisfaction. Overall, the use of line-balancing techniques can be an effective way to improve the performance of garment manufacturing industries in Bangladesh. It can help to reduce bottlenecks, improve production efficiency, increase production capacity, and ultimately benefit the overall success of the industry. [8] In garment production, the sewing stage is one of the most important and critical stages. However, due to the many different operations involved and the varying skill levels of workers, problematic areas, also known as bottlenecks, can occur in the assembly line. These bottlenecks can lead to congestion in the production system, resulting in lower output. By identifying and addressing bottlenecks in the sewing line, in order to increase productivity. One way to do this is through the use of techniques such as line balancing and lean application, which aim to identify and eliminate non-value-added processes from the production process. By using these techniques, improvement can be seen in efficiency and productivity by reducing cycle time, maximizing space utilization, and reducing inventory and transportation. By implementing these techniques, significant improvements in productivity and line efficiency can be achieved. [19]. Again in garment manufacturing processes and operations, sewing is a very time-consuming and pivotal task. Most of the bottleneck is generated in this particular sector. Reorganizing the workload with the efficiency of time among the workers can increase output. So, identifying and addressing the bottleneck in the sewing line results in an increase in production rate and efficiency [59]. As we know that bottleneck means more input than output which causes downstream in a work environment. As soon as it is eliminated, the environment can produce more while being efficient in the system. While identifying the tailback, ensuring the prevention of future bottlenecks is also important [30]. Wrong worker orientation or workflow, machine disturbance or less supply, carelessness of workers, non-sequential stock of products, improper line balancing, manufacturer defect, insufficient appliances management, etc can raise bottleneck problems. [38] and [18]. There are several ways to reduce bottlenecks such as early detection of faults, improving workstation layout, increasing human resource and work hours, sharing capacity, utilizing time-saving tricks, etc [2].

1.3 Research Motivation

Bangladesh is a developing country which is yet to explore many advanced technologies in the field of the garment and textile sector. Thereby, most of the work that is required to function in a garment factory is done manually by human laborers. Under the supervision of the superintendent, they are assigned to certain sectors with certain tasks. For example, a group of people might be working in the pattern-making department, another group of people might be working in the spreading department, other groups of people are working undercutting and sewing, etc. Nevertheless, humans are prone to make mistakes. As certain tasks are divided among certain groups the mistakes are very much detectable if a particular group of people makes a mistake but it is a very lengthy and monotonous process for the quality assurance people to detect shortcomings in these sub-groups. However, machines for inspection are very expensive for these garments in our country. Most of the garment factories have very calculated and rigid budgets which are the reasons for the insufficiency of these tools in these sectors. Moreover, these machines are not locally produced in our country. As a result, these need to be shipped from overseas which is the reason for the excessive cost of these machines. This is where AI implementation comes in handy for inspection. By using AI approaches here we can help to save workload for quality inspection personnel. The YOLOv5 algorithm will only take one iteration to inspect the object's fault and detect the bottleneck that is caused in the particular garment manufacturing sector. As soon as the tailback is found which has been occurring continuously and hampering the quality and batch production, the right steps to terminate that action can be taken. In this way, the technology can be incorporated to subside the error in the system. Additionally, we can help ease the pressure of human resources in the sector and utilize it differently. Moreover, we can save both time and money by using artificial intelligence here. Lastly, it will help to provide accuracy and lessen the defects by detecting the subtle details of the error that the human eye might miss.

Chapter 2

Literature Review

Throughout our research, we encountered several studies on the defect detection process, including fabric defect detection, button defect detection, zipper defect detection, etc. These studies have provided valuable insights, allowing us to compare and understand different approaches, and ultimately shape the direction of our research goals. Many of the studies we reviewed have utilized different methods. Our work focuses on detecting faults in ready-made garments, and the aforementioned studies have enlightened us and provided guidance to narrow down our method and achieve our detection goal. It is crucial to be aware of any potential bottlenecks in the process in the ready-made garment sector.

2.1 Defect Detection Techniques

Garment defects in industries are one of the main reasons for quality failure. Common garment defects that are observed and identified in the garment industry are: seam puckering, broken or open seam, seam slippage, broken stitch, skipped or drop stitch, loose or uncut thread, distorted knitting, and needle thread breakage. The required apparatuses are a garment inspection machine, cutting-sewing machine, inspection table, light source, paper, pencil, hard board., calculator, etc. Reducing the number of defects in the production of trims and accessories is important because without reducing defects, it is not possible to increase the productivity and efficiency of the production process. The presence of defects in garments can lead to a significant loss in value for the company, as the products must be sold at a lower price. To prevent this, it is crucial for manufacturers to minimize fabric defects at every stage of the production process. One way to do this is by using an automated defect detection and identification system, which can improve product quality, enhance productivity and ultimately reduce costs associated with off-quality products. The detection of defects and remedying them immediately can lead to efficient garment manufacturing in the industry. all stages of the production process, from cutting and spreading to finishing, play a role in the occurrence of fabric defects. There has been a significant amount of work conducted in the field of fabric defect detection, resulting in a wide range of proposed solutions. In a reviewed journal, Silvestre-Blanes, J., Albero-Albero, T., Miralles, I., Pérez-Llorens, R., & Moreno, J. (2019) presented how a machine vision system can identify defects in a public fabric database. The goal of this study was to create a publicly available, annotated benchmark of a large collection of images with and without defects, in order to facilitate direct comparisons of detection and classification methods. Additionally, various methods were reviewed, and one was applied to the set of images, with the results reported in the research. [15].

In another article, Moreno, J.J., Aguila, A., Partida, E., Martinez, C.L., Morales, O., & Tejeida, R. (2017), the authors discussed the use of artificial vision, also known as computer vision, to detect errors in garment manufacturing. They primarily focused on errors that are difficult to detect by human workers, such as small defects or variations in the fabric or stitching. By using computer vision, they aimed to increase the speed of quality checks by detecting errors at an early stage of manufacturing, before they become more significant and costly to correct. The authors of the research used several computer vision techniques to detect errors in garment manufacturing. These include image processing, pattern recognition, and machine learning. They also incorporated a camera-based system to capture images of the garments during the manufacturing process, which were then analyzed by computer vision algorithms to identify any errors. The authors found that the use of computer vision techniques improved the efficiency and accuracy of detecting errors in garment manufacturing. Additionally, the system was able to detect errors that are difficult to detect by human workers, thus providing an additional layer of quality control. The study demonstrates the potential of computer vision in improving efficiency and accuracy by detecting errors at an early stage of manufacturing. [6].

Ersoz, Taner, Zahoor, Hamza, & Ersoz, Filiz. (2021) used a data mining approach to detect faults in fabric and production. With the exponential growth of the ready-made garment sector, maintaining quality is crucial to avoid economic consequences from wasted materials. To achieve this, the authors proposed identifying manufacturing problems at an early stage using data mining techniques. The authors also employed algorithms such as decision trees, naive Bayes, random forest, and gradient-boosted trees. Additionally, they took into consideration factors such as product variety, quantity, and categories of faults in their research. [27].

Boresta, M., Colombo, T., & De Santis, A. (2022) discussed the implementation of a convolutional neural network (CNN) in the detection of faults in ready-made clothing products. They successfully used automated technology that operates in two parts: first, detection is done through multi-threshold inspection based on certain features, followed by a deep learning classifier that provides the final outcome. The authors of the study report that the use of CNNs in this context allows for robust fault detection, and the multi-threshold inspection further improves the accuracy of the system. Additionally, the use of deep learning classifiers allows the system to adapt to different types of faults and clothing products, making the system more versatile. Overall, the study shows the effectiveness of using CNNs for the automated detection of faults in ready-made clothing products, which can have significant implications for the textile and clothing industry, potentially reducing costs and improving efficiency. [52] .

Kim, H., Jung, W., Park, Y., Lee, J., & Ahn, S. (2022) have used image processing and convolutional neural network (CNN) feature maps to detect sewing problems. In their proposed model, they used visual media to detect broken stitches in sewing defect detection from the primitive layers of a pre-trained VGG-16 model while extracting a feature map. Their image-based research used photos such as sewing images, normal images with fabricated faults, and images from different angles. They were able to detect the broken stitches with 92.3% precision and better time complexity with real-time computation. They also rescaled images and deployed a lightweight deep learning library which has proven to be achievable and convenient to use in the garment sector. The study by the authors shows that the use of image processing and CNN feature maps can be an effective method for detecting sewing problems in the garment industry. By using visual media to detect broken stitches, the system can detect defects that are difficult to detect by human workers, such as small defects or misalignment of patterns. The use of a pre-trained VGG-16 model and extracting feature maps from the primitive layers also enables the system to detect defects with high accuracy. The use of real-time computation and a lightweight deep-learning library also makes it convenient and easy to use in the garment industry. Overall, the study demonstrates the potential of using image processing and CNN feature maps for detecting sewing problems in the textile and clothing industry, which can improve the efficiency and quality of garment manufacturing. [54].

Wong, W., Yuen, C., Fan, D., Chan, L., & Fung, E. (2009) performed research on stitching defect detection and classification using a Back Propagation Neural Network (BPNN) and thresholding method on pyramid wavelet transform. The images were segmented using thresholding and filtering methods, and nine insubstantial attributed binary images were collected and sent to the BPNN for classification. According to the performance, the five classes could detect stitching defects efficiently while following the approaches. The study focused on the detection and classification of stitching defects in the textile and clothing industry. The authors used a combination of image processing techniques such as thresholding and filtering, and a backpropagation neural network (BPNN) to classify stitching defects. The use of pyramid wavelet transform helped to reduce the computation time. Their results show that this approach is able to detect and classify stitching defects with high accuracy, which can help to improve the efficiency and quality of garment manufacturing. Overall, this study highlights the potential of using image processing techniques and neural networks for the detection and classification of stitching defects in the textile and clothing industry. The use of BPNN and thresholding method on pyramid wavelet transform can enable the system to detect and classify defects with high accuracy, which can ultimately help to improve the efficiency and quality of garment manufacturing. [1].

2.2 Detection Using YOLOv5

Chang, Y.-H., & Zhang, Y.-Y. (2022) discussed the use of a lightweight YOLOv5 algorithm to recognize clothing patterns by implementing deep learning in their reviewed journal. With the advancement of technology, computational power, speed, and learning complexity are often issues with bigger projects. However, a one-stage detector like YOLOv5 integrated with Google Colab can come in handy and save resources. The sources of the datasets used are fashion clothes such as DeepFashion, DeepFashion2, and web crawling. The authors compared deep learning object detection algorithms, with YOLOv1, YOLOv2, YOLOv3, YOLOv4, YOLOv5, SSD being one-stage algorithms, and R-CNN, Fast R-CNN, and Faster R-CNN being two-stage algorithms. The results showed that YOLOv5 could detect objects faster and more accurately than other algorithms. YOLOv5 can be used for object detection while Python via Google Colab can be used for model training and testing at the same time. The secondary image dataset was collected from several sources and was divided into five groups by the authors: plaid, plain, block, horizontal, and vertical. However, only the tops of the clothes were taken into account for this research. The authors used mosaic augmentation by cut-mix method to create new training data using the existing data. Then, using the bounding box and anchor box, the objects were typically detected. In the YOLOv5 architecture, feature extraction and fusion were done in the backbone and the neck, and objects were predicted in the head. Overall, the authors have stated that YOLOv5 has been successful in detecting given objects in the images far better than other algorithms. [53].

Wang, X., & Estil, Z. T. (2021) proposed a system for detecting defects in buttons using the YOLO algorithm. The quality of buttons, being an essential component of clothing, plays a critical role in the sales of clothing. The implementation of this system can greatly improve the efficiency and accuracy of identifying defects in buttons, thus improving the quality of the final product. The use of a button defect detection system can significantly decrease the need for manual labor and lower the error rate in detection. The authors have suggested a system to enhance the accuracy of button detection. The system was created to tackle issues and difficulties encountered in manual detection methods, and to increase the efficiency and accuracy of button detection. The outcome of this research is a detection system. The proposed system starts with taking button images and marking the defect within the image. Then after training the model, the model would be able to detect the defects and provide the output. The whole system will use the YOLO algorithm. The system however is divided into different modules which are image acquisition, image processing, system call, and software management module. At first, in image acquisition with hardware appliances, the sample button pictures are taken and the defective ones are labeled. Secondly, in image processing, the sample button files are trained with the YOLO model. Thirdly, a system call establishes a connection between hardware platforms and then builds the connection further by adjusting attributes that are needed in the platform. Lastly, software management designs the user interface and generates the detection result within the software and its interface. The system is integrated based on OpenCV machine vision technology. The system is able to detect multiple defects as well as real-time detection. This study introduces a unique aspect in that it combines the training phase of the model into the system, resulting in a comprehensive system for both training and detection. The system chronologically integrates the hardware platform, then the YOLO algorithm, and the software environment and interface. According to the authors, by utilizing the YOLO target detection algorithm for button defect detection, the system is able to achieve outstanding performance in identifying defects. [58].

Jin, R. & Niu,Q. (2021) implemented their research with the help of automation in fabric detection with improved YOLOv5. In their research, they stated that the traditional method of manual detection is characterized by its inefficiency, prolonged time consumption, physical demands, and high costs. Also for the insufficiency of fabric defects in images a student-teacher architecture had been introduced in which the teacher network could help to detect defects and the student network could help to detect them in real-time as well. In their research, they talked about the CNN model which is widely used in Computer Vision. The authors also mentioned other object detection methods such as RCNN, fast-RCNN, faster-RCNN, and SDD including YOLO where they stated that in terms of concurrency, YOLO proved their supremacy as others have high computational costs. The improved YOLOv5 they mentioned, specifically addresses the detection of fabric defects and can be utilized in a fabric defect detection system. Then they talked about the pipeline of the YOLOv5 and spatial and channel attention mechanisms. Their proposed methods were divided into two stages: training and testing. In the training phase, the augmented image data were sent to the teacher network for fabric defect detection. Then, the teacher network sent the filtered data to the student network. In the testing phase, the student network detects particular fabric defects in real time. The teacher network's structure takes image data as input. Then it is sent to the backbone of the YOLOv5 algorithm which is processed through PANet. Then the image data go through attention enhancement which divides into the normal/abnormal or fabric defect classification. Then the merges back into the fusion module which generates the output probability. Then, the output image data is generated with a label. The student network structure takes image data as well and sends it to Backbone where BottleNeckCSP takes place. Then it is sent to do enhancement with spatial and channel attributes. Again, the processes are repeated where the image data go through attention enhancement which divides into the normal/abnormal or fabric defect classification. Then the merges back into the fusion module which generates the output probability. Then, the output image data is generated with a detection. They talked about loss function calculation as well and collected databases from public sources and trained-tested those. After that, they compared their results. Finally, they came to the conclusion on how fabric defect detection is more convenient using the YOLOv5 algorithm. [34].

2.3 Faulty Fabric Detection Techniques

The review article written by Rasheed, A., Zafar, B., Rasheed, A., Ali, N., Sajid, M., Dar, S. H., Habib, U., Shehryar, T., & Mahmood, M. T. (2020) presented a comprehensive overview of the various approaches to fabric defect detection that have been developed using computer vision and image processing methods. These methods include techniques that rely on the histogram, color, segmentation, and texture of the fabric. In the textile industry, identifying defects in the fabric can be a challenging task as the quality and cost of a textile product are closely linked to the accuracy and efficiency of the defect detection process. The use of computer vision techniques can be beneficial in overcoming human limitations in identifying defects. The authors conducted an in-depth research study to compile a comprehensive list of approaches based on different techniques such as histogram, color, image segmentation, frequency domain, texture, sparse operation, image morphology operation, etc. The research was designed to provide a detailed analysis of the different techniques used for fabric defect detection and how they can be used to improve the accuracy and efficiency of the process. Overall, the article provides valuable insights for professionals and researchers in the field of textile technology and computer vision. [20].

Li, C., Li, J., Li, Y., He, L., Fu, X., & Chen, J. (2021) in their review article said that fabric defect detection is a crucial task in the textile industry as it directly affects the quality and cost of textile products. The traditional manual inspection methods are time-consuming and prone to human error. With the advancements in technology, various methods have been proposed for automatic fabric defect detection using techniques such as image processing, pattern recognition, and machine learning. These methods have been shown to be more efficient and accurate than manual inspection methods. The researchers have further discussed the Traditional and Learning-based models to detect defects. They further listed some Traditional models such as Statistical, Spectral, Structural, and Model-based methods. The Learning-based models are classical Machine learning and Deep learning methods. The proposed Statistical methods for fabric detection are Entropy filtering and minimum error thresholding, Sylvester matrix-based similarity method, Based on two fundamental structural properties, Regularity and local orientation (anisotropy). The spectral methods for fabric detection are Multiscale wavelet features and fuzzy C-means clustering, Dual-tree complex wavelet transforms (DTCWT), The method based on information entropy and frequency domain saliency, L0 gradient minimization method and two-dimensional fractional Fourier transform (2D-FRFT) for obtaining the saliency map of the quaternion image, Gabor preprocessed golden image subtraction, Fuzzy back propagation neural network (FBPNN) with Gabor features, Using learning-based local textural distributions in the contourlet domain. The Structural methods are Wavelet preprocessed golden image subtraction (WGIS), Isotropic Lattice Segmentation (ILS), etc. A model-based method for fabric detection support vector data description (SVDD) and Elo rating (ER) are also discussed there. On the other hand, for the dictionary learning approach proposed methods are Low-rank representation (LRR), Low-rank representation, Gabor filter, and tensor low-rank recovery, Low-rank decomposition with gradient information, Multi-scale convolutional neural network and low-rank decomposition model, Weighted double-low-rank decomposition method (WDLRD) to treat the matrix singular values differently by assigning different weights, Low-rank decomposition of multichannel feature matrices, Sparse and dense mixed low-rank decomposition and A randomized low-rank and sparse matrix decomposition model named GoDec. For traditional machine learning the proposed methods are Multiview stereo vision (MVS) and bag-of-features (BOF), K-nearest neighbor (KNN) algorithm, Artificial neural networks (ANN), Snake active contour and support vector machines, L0 gradient minimization (LGM) and the fuzzy c-means (FCM) method to detect various fabric defects with diverse textures. Then, the deep learning proposed method for detecting objects is divided into two categories which are the One-stage detection algorithm and the Two-stage detection algorithm. One-stage detectors are YOLO, SSD (Single Shot Detector), YOLO v2/v3/v4, RefineDet, and RetinaNet. Two-stage detectors are Faster RCNN (Region-Based Convolutional Neural Network), Mask RCNN, Cascade RCNN, FPN (Feature Pyramid Network), and R-FCN. Lastly, the deep learning proposed or tested methods for detecting fabrics is Mobile-Unet (One-stage), YOLOV2 (One-stage), Stacked convolutional autoencoders (One-stage), Adaptive method based on DenseNet-SSD (One-stage), A multilayer perceptron with a Levenberg–Marquardt (LM) algorithm (One-stage), Multiscale convolutional denoising autoencoder network model (One-stage), Improved RefineDet (One-stage), Segmentation network and decision network (Two-stage). In the conclusion, the authors put their observation where they stated that often models or approaches are combined to achieve better outcomes.[36].

The research article of Ashraf, R., Ijaz, Y., Asif, M., Haider, K. Z., Mahmood, T., & Owais, M. (2022) explored the use of a Convolutional Neural Network (CNN) to classify images of woven fabric as either faulty or non-faulty. The authors proposed an automated approach for identifying malfunctioned fabric within a batch of good and bad fabric, using the GoogleNet image processing approach. The proposed method was tested using the TILDA dataset, which is a dataset of textile textures, and it demonstrated an accuracy of 94.46% in classifying defective and non-defective fabrics. The results of the study indicated that the proposed method outperforms other established techniques such as Bayesian, BPNN, and SVM in identifying defective fabrics. Overall, the study provides a promising solution to the problem of identifying defective fabrics in the textile industry, using the powerful image processing capabilities of CNNs. [51].

2.4 Defected Button Detection Techniques

Han, Y., Wu, Y., Cao, D., & Yun, P. (2017) proposed a machine vision approach for identifying faulty buttons using a weighted least-squares model. They aimed to reduce false alarms caused by spatial skewness and improve the accuracy of the model. To do this, they used a sample image of a defect-free button and an additional image, which they referred to as an "extra image." The authors claimed that their proposed model is able to achieve accurate results for a variety of different types of buttons. Overall, the study is focused on the use of computer vision techniques for identifying defects in mechanical and electronic components such as buttons, by analyzing images of these components and identifying any defects or anomalies. [5].

In their article, Wu, S., Wu, Y., Cao, D. & Zheng, C. (2019) advanced their model with a Siamese network with imbalanced samples to detect fast button surface defects. They stated that imbalance occurs in surface defect detection as it gets very crucial as complex surface texture, different kinds of defects, and scarcity of defect samples. To address these problems the authors presented a similarity function based on the Siamese network which is implemented in a practical machine vision integrated structure. The similarity function is a real number-based function that computes the resemblance of two objects. This Siamese network is proposed by the authors to mark the faults in cloth material. The network is particularly useful for the calculation of loss function which is very convenient to do preprogrammed autonomous feature extraction and find similar objects. The proposed method for detecting defects in buttons uses a machine-learning process to minimize a specific loss function. This loss function is designed to drive the intra-class distance among positives (i.e. non-defective buttons) to be smaller and the inter-class distance (i.e. between non-defective and defective buttons) to be larger in the feature space. As a result, after training, defect-free samples are clustered together while defect samples are mapped to outliers. This allows for high accuracy in detecting defects, even in cases where the dataset is imbalanced. The method has been evaluated on button datasets with multiple kinds of defects including dents, cracks, stains, holes, and unevenness. Results show that the proposed method has a 98% detection precision, and a 95% detection precision when dealing with imbalanced datasets, indicating its advantage over conventional methods. Additionally, the proposed loss function outperforms other recently published loss functions in the fields of face recognition and ReID. The method has been optimized with different strategies and it reaches a 6 fps detection speed on an embedded DSP platform, indicating its potential for use in online detection during production. [16].

In the reviewed research paper the authors Liu, L., Cao, D., Wu, S., Wu, Y., and Wei, T. (2018) described a method for detecting defects in button surfaces using convolutional neural networks (CNNs). The authors aimed to develop a fast and accurate method that can detect multiple types of defects including dents, cracks, stains, holes, and unevenness. The method proposed in the research consists of three main steps: image acquisition, image preprocessing, and CNN-based detection. In the image acquisition step, button images are captured with hardware appliances. In the image preprocessing step, the images are preprocessed to improve the quality of the images and to make them suitable for CNN-based detection. The authors pointed out that the traditional methods for button defect detection are mainly based on human visual inspection which is time-consuming and labor-intensive. While the recent methods are mainly based on machine vision, which is more efficient, accurate, and reliable. However, the main drawback of these methods is that they are computationally expensive and not suitable for real-time application. The authors claim that the proposed method solves these problems by using CNNs, which are computationally efficient and suitable for real-time applications. Overall, the authors proposed a fast and accurate method for detecting defects in button surfaces using CNNs. The proposed method is able to detect multiple types of defects and it is suitable for real-time applications. The method achieved good results in detection precision and it is optimized to run on an embedded DSP platform. [9].

Li, X., Xu, L., Liu, X., and Sun, S. (2021) developed their model specifically for finding defects in metal buttons with improved speed, precision, and validity which will be done via a detector turning into a classifier. They used Cascading Extreme Learning Machine (ELM) and Sparse Representation Classification (SRC). The approach is referred to as ELM-SRC. The input button images were initially preprocessed by removing reflections, extracting edges, and reducing their dimensionality. Initially, to preprocess the metal button images, the authors employed three methods: an exemplar-based algorithm, the Laplace transform algorithm, and Principal Component Analysis (PCA). Extreme Learning Machine (ELM) was used to quickly and accurately determine the probability of defective buttons, using parameters from a cross-validation method and the Sparse Representation-based Classification (SRC) technique was employed to reclassify button images with high levels of noise which is not only improved detection accuracy but also ensured a fast detecting speed. According to the authors, ELM-SRC adopted different approaches such as determining the optimal regularization parameters by utilizing a five-fold cross-validation technique. Additionally, they claimed During the training phase of Extreme Learning Machine (ELM), the number of hidden nodes was determined by conducting experiments and selecting the value that produced the highest f1-score, using various parameters. Lastly, they stated that in the training phase of Extreme Learning Machine (ELM), the number of input parameters was chosen by conducting experiments on the dimensionality reduction of metal button images and selecting the optimal value. The image processing was done by collecting images, enhancing data, eliminating reflection, extracting edges, and reducing dimensionality. The data were split into training and testing and sent for preprocessing. The preprocessed test data were directly sent to the ELM-SRC model. The preprocessed training data constructed a sparse dictionary using positive and negative samples. The preprocessed train data get trained with the ELM model. The model checks if the error classification is greater than the threshold value or not. If it is greater the model generates output, if not the data is sent to a sparse dictionary which later generates the output. According to their findings, the higher the error classification value, the more accurate results are obtained. After being successful with their research, lastly, the authors drew their plan to use the network for detecting defects in shell and plastic buttons to improve its detection performance. [37].

Karunarathne, S. & Ganganath, R. (2021) presented a model to detect button and measurement errors using image processing methods. In the apparel industry, the process of apparel production involves multiple steps, from material procurement to final product manufacturing. In order to maintain efficiency and effectiveness throughout the entire sewing process, dressmakers must pay attention to the proper measurement of dimensions and accuracy. However, it is challenging for humans to maintain these measurements without incorporating technology. As accurate measurements are crucial for apparel production, humans may not always measure lengths using the correct measurement or mixup lengths sometimes. To remove this issue, this research introduced a system to monitor the measuring lengths and detect measurement errors. As a result, the model contributed to the quality and productivity of men's plain-collared shirts by controlling imperfections and measurement errors. Furthermore, the system enabled the path to identify measurement errors accurately and defined the correct measurements, and finalized the product status as a quality product. In summary, this research aims to develop a system based on image processing methods that detected measurement errors in the apparel industry to improve the quality and productivity of men's plain-collared shirts while minimizing defects. [35].

2.5 Defected Zipper Detection Techniques

In a reviewed journal by Fang, H., Xia, M., Liu, H., Chang, Y., Wang, L., & Liu, X. (2021) came up with a multi-scale convolutional network to detect automatic zipper tape defects. Zipper defects often occur during the manufacturing of the apparel. Inspection of zippers using traditional methods involves the use of skilled inspectors, is time-consuming, and is not very efficient or accurate. The authors suggested a model detector framework using fully convolutional networks in a two-stage cascade approach, where the first stage is a rough estimate, and the second stage refines that estimate to a more precise outcome. Their specific application, the detection of zipper tape defects, presents the challenge of multi-scale characteristics. Most existing deep learning methods excel at detecting large-scale defects with distinct features but struggle to detect small-scale defects due to their less striking features and their tendency to be located in a large background area. To address this issue, they proposed a multi-scale detection architecture that is highly efficient in detecting small-scale defects by first identifying large local context regions that contain them. The architecture integrates a new detection branch that fuses features from shallow layers into high-level layers to enhance the detection performance of context regions. Subsequently, the small-scale defects are precisely detected from the local context regions identified in the first stage, as they are now considered large-scale objects that are more easily detected. So basically in short, the authors presented a new approach to addressing the problem of detecting defects in zipper tapes through a two-stage cascade framework. This approach first detects large-scale defects and regions that contain small-scale defects in the first stage. In the second stage, it then detects small-scale defects from these regions, acting as a refining process. By doing this, small-scale defects are treated as large-scale objects, which makes them more identifiable. Overall, the proposed two-stage coarse-to-fine cascade framework aims to solve the problem of zipper tape defect detection by detecting large-scale defects first and then using that information to identify and refine the detection of small-scale defects. Then, they proposed a new, enhanced multi-scale network architecture for the first stage of their approach, which aimed to improve the detection accuracy of small-scale defects which was achieved by adding a new detection branch to the high layer of the multi-scale detection network. This branch integrated features from the shallow layer, which contains more information about edges and contours, into the high layer. This fusion of features improved the accuracy of identifying the local context regions that contain small-scale defects, making it more accurate in detecting small-scale defects. The pitched new improved multi-scale network architecture, by adding a new detection branch that integrated shallow layer features into the high layer, aimed to boost the detection accuracy of the local context region containing small-scale defects, which led to more accurate detection of small-scale defects. In order to further improve the detection accuracy of medium-scale defects in the first stage, we integrate a spatial pyramid convolution (SPC) module into the medium-layer detection branch of our network architecture. The SPC module is achieved by combining spatial pyramid feature maps with different fields of view (receptive fields) and it helps to detect medium-scale defects more accurately. In summary, the integration of the SPC module obtained by concatenating the spatial pyramid feature maps with different receptive fields into the medium-layer detection branch of our network architecture aims to enhance the detection accuracy of medium-scale defects. Lastly, Their extensive comparative experiments reveal that the proposed method offers high detection accuracy while maintaining high detection efficiency compared to the state-of-the-art methods and has good robustness in complex cases. [28].

Zhang, X., Wang, Q., Liu, J., Liu, Z., & Gong, J. (2020) showed another computer vision integrated method to classify zipper detect defects. Currently, manual methods are commonly used to evaluate the appearance quality of zipper products, which is both inefficient and unreliable. The authors proposed a reliable and accurate method to address this issue. In industrial production, zippers come in various types, shapes, and colors, and lighting and other factors can greatly impact the detection results. To overcome these challenges, this paper suggested an adaptive region-growing algorithm and template-matching algorithm. In this research, the region growing algorithm was used to remove the background and obtain the complete inner region of the zipper, then the deflection angle of the zipper was corrected using the smallest external rectangle of the zipper, and finally, the zipper type was determined using a template matching algorithm. For defect detection, firstly, the complete zipper is extracted using a morphological method, then the defect location is identified by searching the area of the connected domain, and finally, the defect is determined by comparing gray values. The experimental results demonstrate that this method has high accuracy and efficiency in zipper classification and defect detection using computer vision. [21]

Chapter 3

Dataset

In this section, we will explain how we have collected and processed our data further. After studying thoroughly the defect detection methods used in other research works, we looked into the existing datasets available for our research problem. While we could find existing and efficient fabric and clothing datasets on popular data collection resources, we were not able to find a defective dataset of clothing parts (e.g. buttons, buttonholes, zippers, threads, etc). Our desired dataset has to contain defects that can be specifically detected during the final inspection process and can be mended so that the piece of the garment does not get rejected. Due to the lack of quality data on the internet and other dataset repositories, we decided to study and make our own dataset for the work. Such a dataset that is created from a first-hand resource is called a primary dataset.

3.1 Dataset Resource

To begin with, we required enough clothing articles like shirts and pants which we collected from the recycled clothes gathered from all team members' homes and filtered the clothes that can be used for our dataset. After collecting these clothes, we had to filter the clothes that will be of use to us. We have used the following tools to process our data - a phone camera of 64 mp, clothes we have collected, and a small hammer, needles, and scissors to replicate defective clothing. Roboflow [56] for annotation and data augmentation. By replicating existing clothing defects and using the information we achieved from [13] we started creating our own clothing defects e.g. loose buttons, broken buttons, torn seams, and broken zippers.

3.2 Dataset Collection

According to our studies and personal experience of owning defective clothes, we find these to be some prevalent defects - loose buttons, broken buttons, different colors or sizes of buttons that do not fit onto their designated buttonholes, torn or open seam, puckered seam, broken zipper, separated or defective zipper teeth. Our dataset has been influenced by these said defects.
For the object detection model, YOLOV5, it is difficult to predict the amount of data that can help achieve high accuracy as it differs for different objects. So we started with a dataset of 5050 which consists of 20 classes. This dataset had only

1 background image example which was not sufficient at all. In order for the image quality to be good we focused on lighting, and angle too much. However, it didn't help much since the dataset lacked variety. Consequently, it failed to achieve good accuracy in our first attempt to make a dataset. However, it helped us to learn what went wrong, and what has to be done. We have explained this further in 5.1.1.

3.3 Dataset Preprocessing

For the data to be ready to be deployed into a model, we have to ensure its integrity and efficiency. That's where data pre-processing comes in. We have described the steps we have taken to pre-process our data which has played a very important role in our research work.

3.3.1 Data Cleaning

Data cleaning is undoubtedly one of the most important parts of data preprocessing. Essentially, data cleaning means cleaning up any data that will hamper the integrity of the dataset. Blurry, duplicate, inconsistent, or incorrect data will decrease the overall performance of the model no matter how many parameters are manipulated to learn more. We got rid of blurry and duplicated data while we were annotating. After annotation, we further discarded a few more data where there was too much distracting background information that could decrease the model's performance.

3.3.2 Defect Creation

To replicate the defects we used some tools. Using a small hammer we created broken buttons, by using a needle and thread we loosely stitched buttons so that they look identical to real loose buttons, by using scissors we created small holes in seams, we twisted the zippers with pull-outs until they broke off and thus created broken zippers.

3.3.3 Annotation

As mentioned before, we have used roboflow [56] to preprocess our data since it has a good statistical representation of the dataset. In our first attempt to make a dataset, we only used bounding boxes. However, after examining the data again we noticed that annotating zippers with bounding boxes was not efficient. Therefore, in our final dataset, we annotated most of the zippers in polygon shapes so that it can learn the shapes of the zippers more accurately. fig 3.1 and fig 3.2.

3.3.4 Augmentation

Next, we encounter the issue of the lack of data quantity given our limited resources. Data augmentation can help increase the number of data. After careful consideration, we augmented all the images to have 90-degree rotation clockwise, anti-clockwise, and upside-down. This augmentation helped us increase the number of data by 3 times and it helps to detect the object irrespective of camera orientations. After augmenting all our datasets, the button dataset had 1,191 training

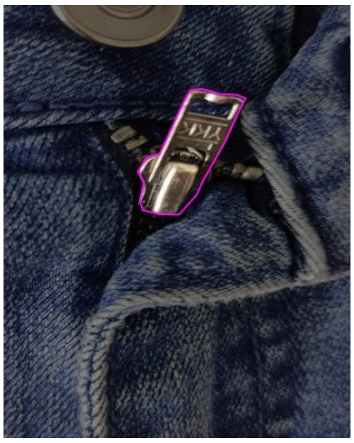
Figure 3.1: Polygon annotation of a zipper

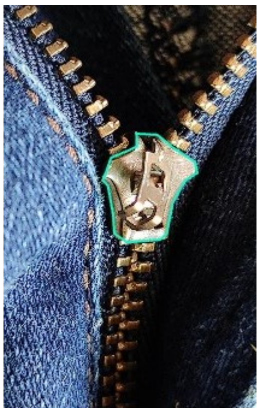
Figure 3.2: Polygon annotation of a broken zipper

images, the button dataset had 1,108 training images and the button dataset had 1,075 training images.

3.4 Dataset Statistics

Finally, among all classes of data in different datasets, we divided the dataset into 80:20 ratios for training and validation. We tested the model using randomly picked data from our entire dataset in order to check its performance. In our first attempt to create a dataset, we had exactly 5,050 images, and 7,061 annotations across 20 classes. In our 2nd attempt, we further divided the entire dataset into 3 categories while training, unlike last time so that we can preserve the data integrity despite our relatively smaller dataset. We prepared 3 separate datasets for each category - button, seam, and zipper. The dataset of buttons has 663 images for 3 different classes which are good buttons, broken buttons, and loose buttons. For the dataset of the seam, we were able to collect 601 images to create 2 detection classes which are good seam and defected seam. Lastly, for the dataset of zippers, we have 587 images for 2 detection classes - good zipper and defected zipper. Among these different categories, we had roughly 500 annotations per class and 5 to 10 percent background images to reduce false positives. We further applied augmentation to the button dataset in order to increase the quantity of the dataset. After running our model with the augmented dataset we came to the conclusion that it did not cause any loss of accuracy. Finally, we have 2,525 proper images in our dataset.

Chapter 4

Research Methodology

Human vision works like when we see something with our eyes, it gives signals to the brain and our brain helps us to detect the entity by its size, color, and corresponding coordinates. Similarly, machines detect an entity with the help of an intermediate device and AI techniques such as machine learning algorithms and deep learning methods. As a computer can sense, think, act and adapt, it can detect entities with its vision and learning capabilities.

4.1 Used Architecture

YOLO (You Only Look Once) is known as the state of art for object detection. It is trained on several weights on MS COCO (Microsoft Common Objects in Context) dataset which basically detects objects within a single glance. COCO is an open-source large-scale object detection, segmentation, and captioning dataset. COCO has 330K images (¿200K labeled) data which even come with a bounding box and annotation [25]. YOLO is also known for working on the single-stage detector principle. A single-stage detector means that it just takes one propagation or iteration of a particular image to detect all the objects that are present in that one snap. It outperformed all of the previous object detection methods in terms of speed and accuracy. It also excels in terms of learning capability. This open-source detector keeps on improving its versions which are very efficient for current object detection and image segmentation. It can detect objects in both photos and video streams. For videos, it goes by each frame to frame and detects objects with given parameters. The YOLO algorithm integrates Convolutional Neural Network (CNN) to detect the objects. This is one of the specialties of the YOLO algorithm. It takes at most one iteration past a neural network to locate any object of a given visual media. In another sentence, a single algorithm run can locate any media with YOLO. Generally, the neural network understands numbers only. For single object detection, it simply converts the image into a single vector. However, for n number of object detection, it follows a grid system. To add more, It basically functions by dividing the whole image into multiple grid systems and each cell of the grid is responsible for detecting objects within itself. The objects are detected while being surrounded by a bounding box. Typically each bounding box has a center point. If multiple objects are included in a bounding box it determines the elements of the object by using the center point of that particular bounding box. The bounding boxes are coordinate-specific. As mentioned a neural network only understands the numbers, the image data are generally converted into vectors where each element of the vector represents a different property of the object. Normally the YOLO algorithm takes images with bounding boxes and trains the images while converting them into vector dimensions. Lastly, it predicts the objects within one forward pass and detects the object. To elaborate, the YOLO framework takes the entire image in a single instance and predicts the bounding box coordinates and class probabilities for the boxes. At first, it divides the input image into equal dimensions (M* M) which are referred to as grids. Each grid predicts B bounding boxes. A bounding box is a layout that features an object in any particular image. A particular image can have multiple objects, hence multiple bounding boxes. If in an image B=1 then the grid has one bounding box. If the center of an object falls into a grid cell, the grid cell is responsible for detecting the object. The bounding box which has a value over the confidence threshold value should be returned. All other bounding boxes have a confidence threshold below when an object's center replies on a certain grid, that grid is responsible for identifying the object. [33] Encoding a single bounding box is required to highlight the properties of the bounding box, hence the objects in the given visual media. The properties are usually y = pc, bx, by, bw, bh, Cn. Here, pc = probability of the bounding box containing an object, bx = x coordinate of bounding box's center, by = y coordinate of bounding box's center, bw =width of bounding box, bh =height of the bounding box, Cn= probability of the cell containing an object that belongs to a class given the bounding box contains an object (n denotes that an image can contain multiple classes. In this case if C1= 1 then, C2 or all other classes must be 0). For encoding multiple bounding boxes if the cell's dimension is (M* M), CNN will predict y for each cell. Here if B (bounding box) = 2, the properties will be y = pc1, bx1, by1, bw1, bh1,pc2, bx2, by2, bw2, bh2 Cn.In this case, if C1= 1(which denotes class 1) then, C2 = 2 (which denotes class 2). The size of the output would be M*M*(5B+C). Here B is the bounding box value and C is Class Value. Intersection over union (IOU) is used to evaluate the performance of object detection when the occurrence of two overlapping bounding boxes takes place. It is considered the evaluation metric for evaluation object detection. During the training, the images, and IOU between the ground truth (original hand-labeled bounding boxes), and the predicted bounding boxes are calculated. If there is a complete overlap the value is considered as 1, no overlap is considered as 0 and the partial overlap has a value from (0-1). We can get the value by,

$$\mathrm{loU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

The closer the value of IOU is to 1, the more accuracy we get in terms of detecting the object in the image. On the other hand, if the value of IOU is close to 0, it means that the object is not where it is predicted. When an overlap occurs discard that non-maximum suppression method is generally used to achieve the unique bounding box which perfectly detects only the object. Non-maximum suppression method refers to a single unit chosen from the multiple overlapping attributes [39]. In another word, IOU ensures that the output of the bounding boxes is placed perfectly in the given image. It also removes the redundancy of the bounding boxes which do not satisfy the requirements. Anchor box is a technique used in YOLO (You Only Look Once) to provide a fixed set of bounding box shapes that the model can predict. In YOLO, anchor boxes are used to detect objects of various shapes and sizes in an image. Anchor boxes are defined in terms of width and height, and are typically chosen to cover a range of aspect ratios and scales. During training, the model learns to adjust the anchor boxes to better fit the objects in the image. In YOLOv5, the anchor boxes are learned automatically in real-time from data during training, this means that the algorithm adapts to the distribution of object sizes in the training data, instead of using predefined anchor boxes. The anchor boxes are used in combination with the objectness score and class scores to generate the final bounding boxes for the detected objects. The anchor boxes help the model to be more robust to different object scales, shapes, and aspect ratios. [45] [24] In YOLOv5, a new feature called "striding" allows the model to make predictions on different scales, enabling the model to detect small and large objects in the same image. The loss function is the feature that is responsible for measuring the difference between the predicted bounding boxes and the true bounding boxes. YOLOv5 uses a combination of the mean squared error (MSE) and the cross-entropy loss to train the model. The latest technological advancement has unleashed a newer path to object detection. As a result in past years, YOLO has taken new turns with newer versions within the past seven to eight years. The variants of the YOLO are: YOLOv1, YOLOv2 or YOLO9000, YOLOv3, YOLOv4, YOLOv5, YOLOv6, YOLOv7, YOLOv8, PP-YOLO (PaddlePaddle YOLO), YOLOP (YOLO for Panoptic Driving Perception), YOLOX(Exceeding YOLO series), YOLOF(You Only Look One-level Feature), YOLOS (You Only Look at One Sequence), YOLOR(You Only Learn One Representation), YOLACT(You only look at the coefficients) [23] to [47]. There are some subversions such as YOLO-Lite which is similar to YOLO9000 but not entirely similar. PP-YOLOv1 and PP-YOLOv2 are subversions of PP-YOLO. For YOLOv5 there are different versions of YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x. Among all these, YOLOv8 is the latest release of the year 2023. [3] to [57] At first, outperforming all the previous object detection algorithms YOLOv1 was introduced by Redmon and Farhadi on 8 June 2015. The speed of the detection was 45 frames per second (fps). It is a single convolutional neural network (CNN) that can simultaneously detect multiple objects in an image. The original YOLO algorithm was published in the paper "You Only Look Once: Unified, Real-Time Object Detection" which was presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [3] After only 1 year on 25 December 2016, Redmon and Farhadi introduced a new architecture and improved training techniques, resulting in a significant increase in accuracy. In they improved the speed at the rate of 67 frames per second (fps) and launched YOLO9000 or YOLOv2. [4] Less than 6 months later on 8 April 2018, the authors Redmon and Farhadi again introduced another version of the YOLO as YOLOv3 and declared that it had better accuracy than the previous version and added support for 3D object detection. [11] After that the conflict between personal value and military application and privacy concerns, Mr. Farhadi backed off and did not get involved in working on the models further. On 23 April 2020, Bochkovskiy, Wang, and Liao developed YOLOv4 with the most favorable accuracy and speed. They implemented their model with new techniques such as DarkNet, Mosaic data augmentation, DropBlock regularization, and Cross mini-Batch Normalization and further added more libraries to operate it smoothly. [17]

However, within 50 days on 18 May 2020, Glenn Jocher introduced YOLOv5 to the world which created many controversies. For the first time, he implemented YOLO in PyTorch which is under active development. Not only was the python and torch library integrated all together, but it also could eradicate the darknet framework's limitation and auto-anchoring, training custom data at ease, and mosaic augmentation by default such things were improvised and introduced within a single detector model. This new model surpassed every previous model and had a breakthrough in terms of improved accuracy and speed. [50] to [48] The YOLOv6 was released in June 2022 where the modification of the architecture was noticed and it is claimed to be standardized for industrial applications with better accuracy. [55] The YOLOv7 was released in July 2022 and is known to surpass the other models in terms of speed and accuracy but it is still under improvement. [57] Lastly, YOLOv8 is released on 10 January 2023 and is known for its extensibility which is designed as a framework that supports all previous versions of YOLO. It is even possible to switch between the versions. [49] After being thorough with our research we have chosen the YOLOv5 algorithm as our model for object detection as it has brought a motion in terms of better accuracy and great speed. As we know YOLOv5 was introduced on 18 May 2020 which is an open-source network that is implemented in Pytorch and can be trained on a single GPU or multiple GPUs. It requires a relatively powerful GPU to achieve high processing speeds. It also supports quantization and pruning that allows faster inference and running on less powerful devices, like mobile phones. It is trained on the COCO dataset for 300 epochs. In the COCO benchmark, YOLOv5 has been reported to have an mAP of around 43% at a processing speed of about 200 frames per second (fps) on a single GPU. Some pre-trained models depending on the weights of YOLOv5 are YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5xl. The weights are denoted as n for nano, s for small, m for medium, l for large, and xl for extra large. The weights are basically for supporting multiple sizes of image inputs. The weights are efficient for different resolutions of images. These versions are designed to improve the accuracy and speed of the original YOLOv5 mode. These are differentiated by their model size, backbone architecture, and other training techniques used. However, if the weight is larger it will work slower. On the contrary, the larger the weight, the more accurately it works with a more powerful backbone. As we mentioned, YOLOv5 uses weight standardization, Mosaic data augmentation, CutMix, DropBlock regularization, and CIoU loss function to improve the accuracy and performance of the model by augmenting the training data. Additionally, YOLOv5 uses anchor boxes with different aspect ratios and it allows it to detect objects with different shapes. Previous versions of YOLO, such as YOLOv4 and YOLOv3, use anchor boxes as well. In YOLOv3, anchor boxes are predefined in terms of width and height and are used to generate the final bounding boxes for the detected objects. During training, the model learns to adjust the anchor boxes to better fit the objects in the image. YOLOv4 also uses anchor boxes, in the same way as the previous version, YOLOv3, they are predefined and used to generate the final bounding boxes for the detected objects. However, YOLOv4 introduced some improvements over YOLOv3, such as using anchor scale and aspect ratio adaptation, which allows the model to generate more accurate bounding boxes for objects of various shapes and sizes. In summary, anchor boxes are a common technique used in all versions of YOLO to provide a fixed set of bounding box shapes that the model can predict, YOLOv5 is the only version that learns the anchor boxes from the data during training, which allows the model to adapt to the distribution of object sizes in the training data. On top of that, the weight of the dataset can be categorized into two portions: pre-trained weights and custom weights. Pre-trained weights are the ready-made dataset where weights have been trained on a large dataset and are available for usage. The pre-trained weights can be used as a starting point for fine-tuning the model on a specific dataset. On the other hand, custom weights are the dataset that is specially made or personalized data that is made with its own specification. They can be used to improve the performance of the model on a specific dataset. Custom weights can be generated by fine-tuning a pre-trained model on a specific dataset. As the name suggests, You Only Look Once (YOLO) takes a glance at the object and detects the object. It is a single-stage object detector, meaning that it does not use a separate region proposal network to generate object proposals before predicting bounding boxes and class probabilities. To serve the purpose of detecting garment faults we have trained our dataset on YOLOv5s which is capable of detecting objects at the rate of up to around 400 frames per second (fps) on a single GPU depending on the specific implementation and hardware. Our chosen model is YOLOv5s which works with a smaller number of parameters that make it faster and lightweight. This reduction in the number of parameters is achieved by using smaller convolutional filters and fewer feature maps. It can achieve an mAP of around 30 percent on common benchmarks such as COCO. It can be trained and fine-tuned on various datasets, such as COCO, VOC, and custom datasets, etc. The striding feature here allows the model to make predictions on different scales, this enables the model to detect small and large objects in the same image. In this model, there's an improvement in anchor scale and aspect ratio adaptation that allows the model to generate more accurate bounding boxes for objects of various shapes and sizes. It also uses data augmentation techniques during training that help to improve the generalization of the model and reduce overfitting. Besides our model has been chosen for other benefits as:

- It has high accuracy in object detection tasks which can be beneficial for detecting defects in garments, as high accuracy can help reduce the number of false positives and false negatives.

- It is an open-source project which means it can be easily customized to suit specific use cases. This can be beneficial for a defect detection system, as it can be fine-tuned to detect specific types of defects using custom datasets in garments.

- It is designed to be scalable that can be run on different devices, from mobile phones to powerful GPUs. It can be beneficial for a defect detection system in an industrial setting, where it can be run on a variety of devices depending on the available computational resources.

- It is able to detect objects of different sizes. It can be beneficial for a defect detection system, as defects in garments can come in various sizes.

Here in fig 4.2 CSP refers to Cross Stage Partial Network which is the backbone of the YOLOv5 model architecture. The CSP darknet is a CNN that uses CSPDarknet53 for object detection and it splits and merges back the feature map to increase more gradient flow. [41]. The SPP stands for Spatial Pyramid Pooling which is used on CNN's last layer to avoid fixed size limitations on the network. [26]. In the backbone layer for feature extraction, the data are input into CSPDarknet. Then,

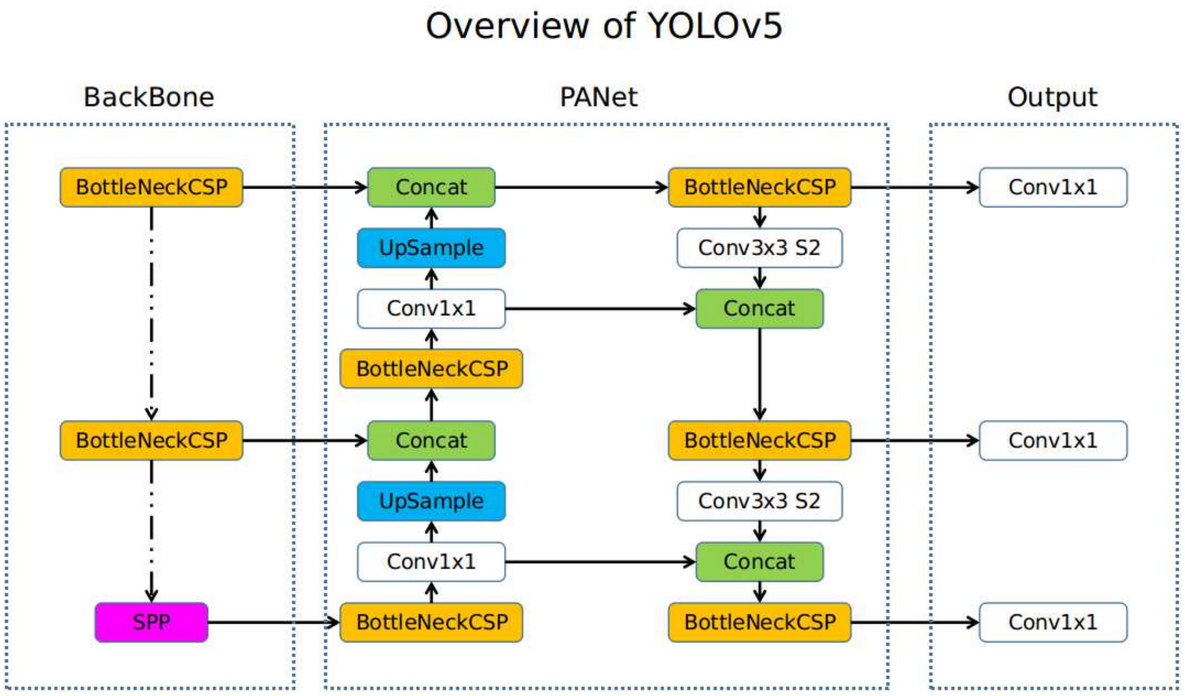

Figure 4.1: YOLOv5 Architecture

in the neck, the PANet is used to get feature pyramids. Concat stands for concatenating or adding functions. Upsampling is used to expand or filter data. When the CSPDarknet sends the input data to the PANet structure it usually does feature fusion. Lastly, there comes the head which is known as the output or YOLO layer where the final detection or outcome is acquired. Conv means convolutional layers which contain a set of filters and parameters. The output layer provides the object detection outcome such as class, location, size, etc. So basically In YOLOv5 architecture, feature extraction and fusion are done in the backbone and the neck and objects are predicted in the head. In short, firstly the backbone layer is responsible for extracting features from the input image. YOLOv5 uses a residual network as its backbone. Secondly, the neck layer is used to further process the features extracted by the backbone network. In YOLOv5, the neck network includes several convolutional and upsampling layers. Finally, the head layer takes the features from the neck network and produces the final object detections. The head network includes several convolutional layers, as well as a series of anchor boxes that are used to generate the bounding boxes for the detections.

4.2 Implementation of Architecture

Our proposed model detects said defects with the help of YOLOv5. The process of inspection through this method is that first, it needs to take the necessary sample data. Then the data is processed and classified. The classification helps the model to predict and localize the defected and non-defected according to their class. The following flowchart gives a visual representation of our methodology -

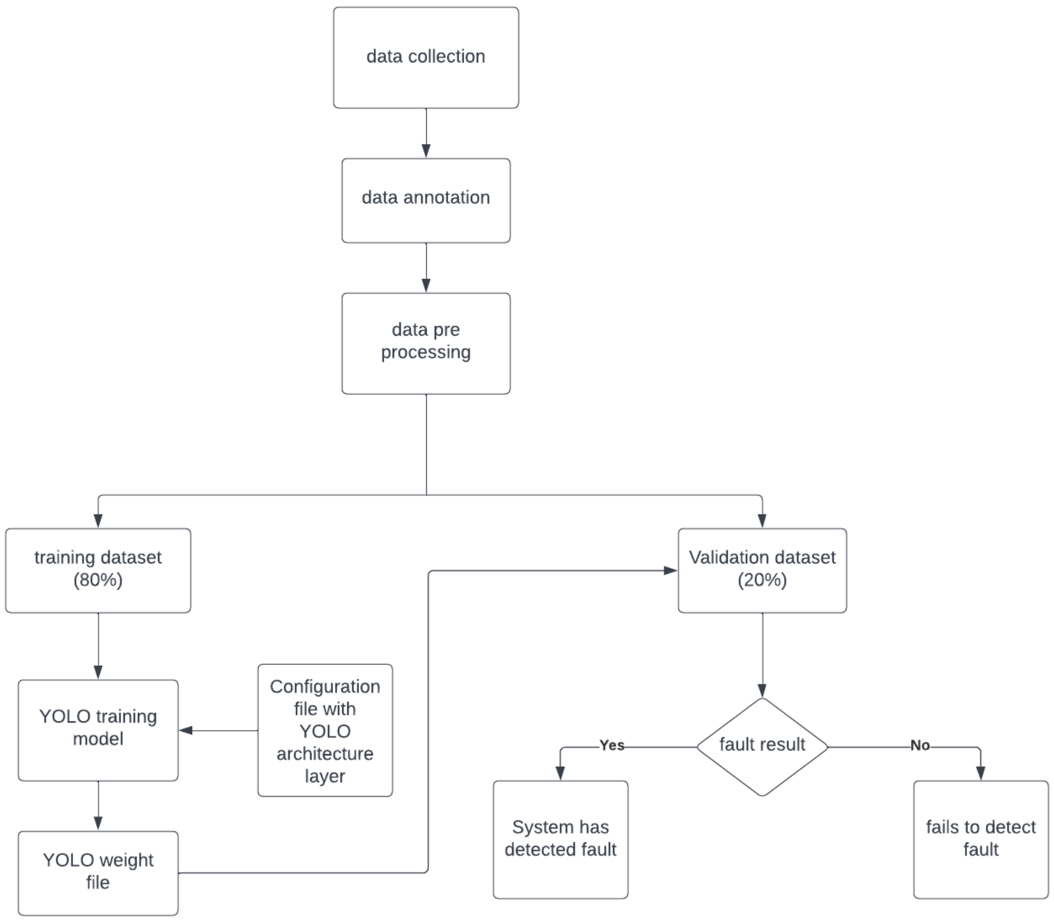

Figure 4.2: The flowchart of the proposed detection model for our custom dataset

4.3 Evaluation method

In this section, we will be using 4 metrics to evaluate our YOLOv5 models which are precision, recall, mean average precision, and confusion matrix. In the following figures from 4.3 to 4.14 we demonstrate the curves of the highest results we could achieve by increasing the epochs of our model. An epoch generally means a period of time when an event takes place. In our model's case, one epoch describes how the validation set of images passes "forward and backward", quoted from [43], through the algorithm exactly once. Other hyperparameters for our model like batch size were at 16 and we increased the epoch after we analyze the result for every 100 epochs to get a better result. All of these metrics are based on the 4 concepts - true positive (TP), true negative (TN), false positive (FP), and false negative (FN). When a system predicts a ground truth correctly, it is either true positive or true negative. On the other hand, when the system predicts that the ground truth is wrong, it's called a false positive or false negative. A confusion matrix is simply a matrix representation of the correctness of the predictions given the true values of the validation dataset. Precision is the value that represents the correct predictions among all predictions we made. The formula to find precision is -

$$Precision, (P) = \frac{TP}{(TP + FP)} \tag{4.1}$$

Recall on the other hand is a metric that represents the sensitivity by representing the correct predictions among all instances. The formula for the recall is -

$$Recall, (R) = \frac{TP}{(TP + FN)} \tag{4.2}$$

Mean average precision or mAP [46], is the sum the of average precision of our model which is a benchmark metric to evaluate an object detection model's robustness. To use our model in the final inspection system of the garment industry, we need to give priority to the recall of our system. Because our system needs to be able to detect more defects regardless of the accuracy of the prediction. However, risking a very high value of recall at expense of precision may cause a bottleneck in our own system. So in order to thoroughly analyze both the precision and recall of our system, we further evaluate the f1 score and confusion score of our models.

4.3.1 Button Dataset Performance Evaluation

We first run our model at 100 epochs to create a baseline and judge how much to increase the epochs to achieve better performance. This dataset is a multi-class dataset with the 3 following classes - good button, broken button, and loose button. With the good button class achieving 87% mAP and the loose button class at 89% and the broken button class at 92% mAP, the overall mAP is 89%. We run another 100 epochs and at 200 epochs, the precision and recall are shown in fig 4.3 For all classes, the mAP has increased. From fig 4.4 The confusion matrix shows that the model classified 96% of broken buttons correctly, and 4% were classified incorrectly. 94% of buttons were correctly classified as loose buttons but 6% were misclassified. With 87% correct classification good button has the most background false positives.

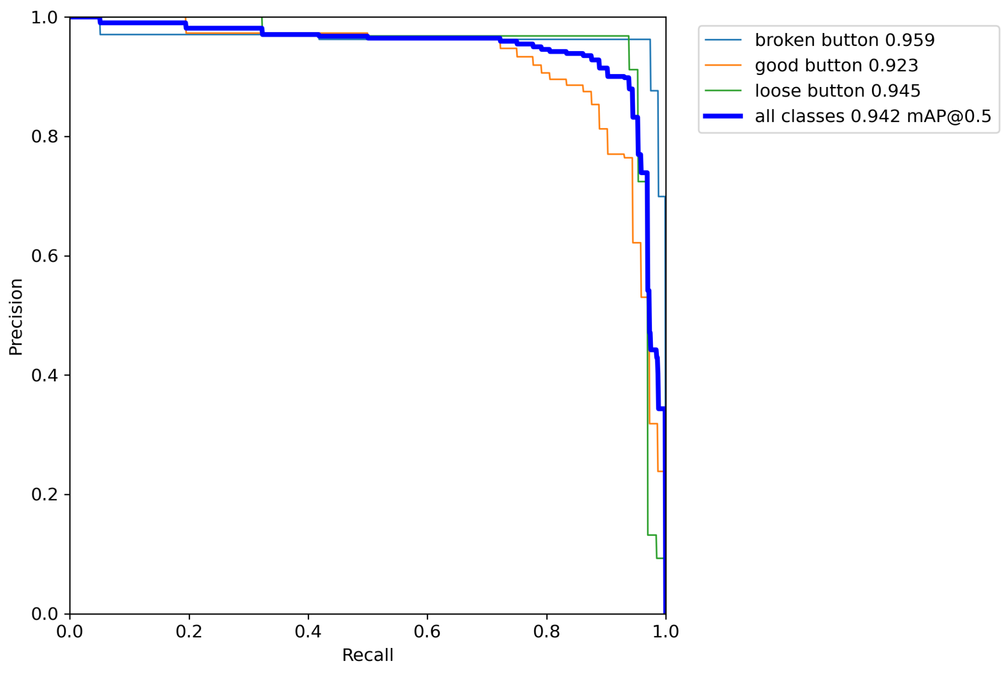

Figure 4.3: PR curve of button dataset at 200 epochs.

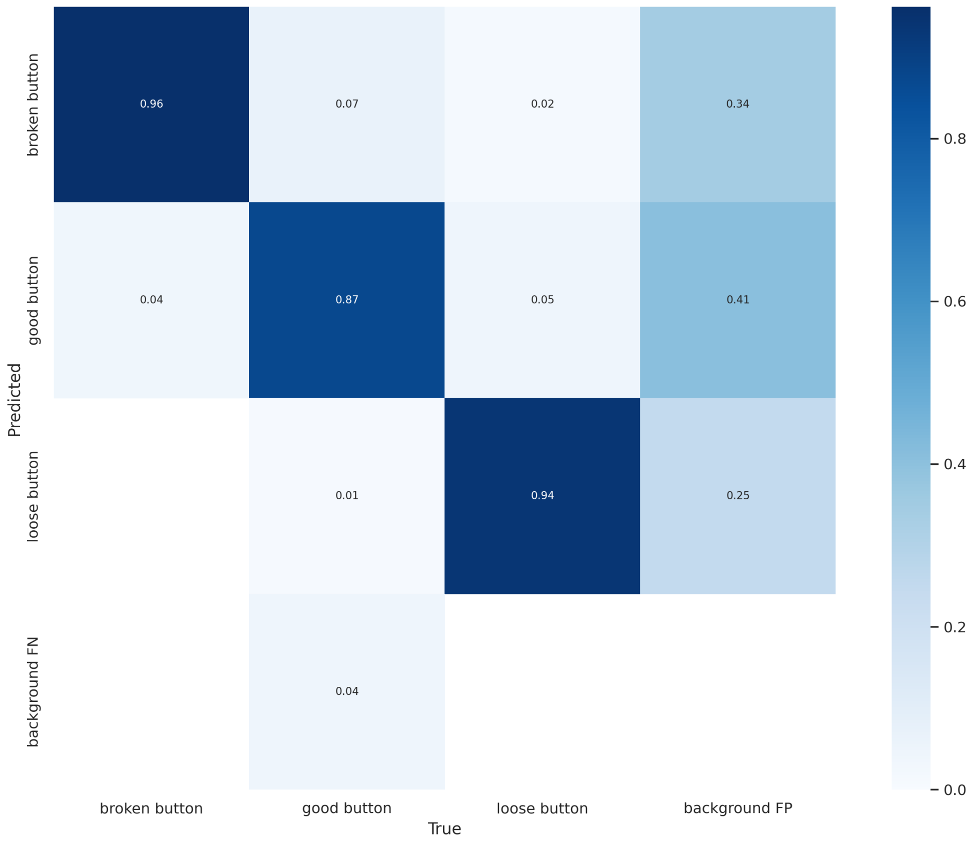

Figure 4.4: Confusion Matrix of button dataset at 200 epochs.

To increase our data quantity, we applied the augmentation described in fig 3.3.4. The new augmented button dataset has 1,337 images. In fig 4.5 we can see that the overall class mAP for the augmented dataset has increased by 2.7% and is at 96.9% mAP. we compared these 2 models at the same epochs, which is 200 epochs, for a fair comparison. From fig 4.6 The confusion matrix shows that the model classified 99% of broken buttons correctly and only 1% were classified incorrectly. As for the loose buttons, 95% of the buttons were correctly classified and 5% were misclassified. With 94% correct classification background false positives for the good button, the class has reduced misclassification to 6%.

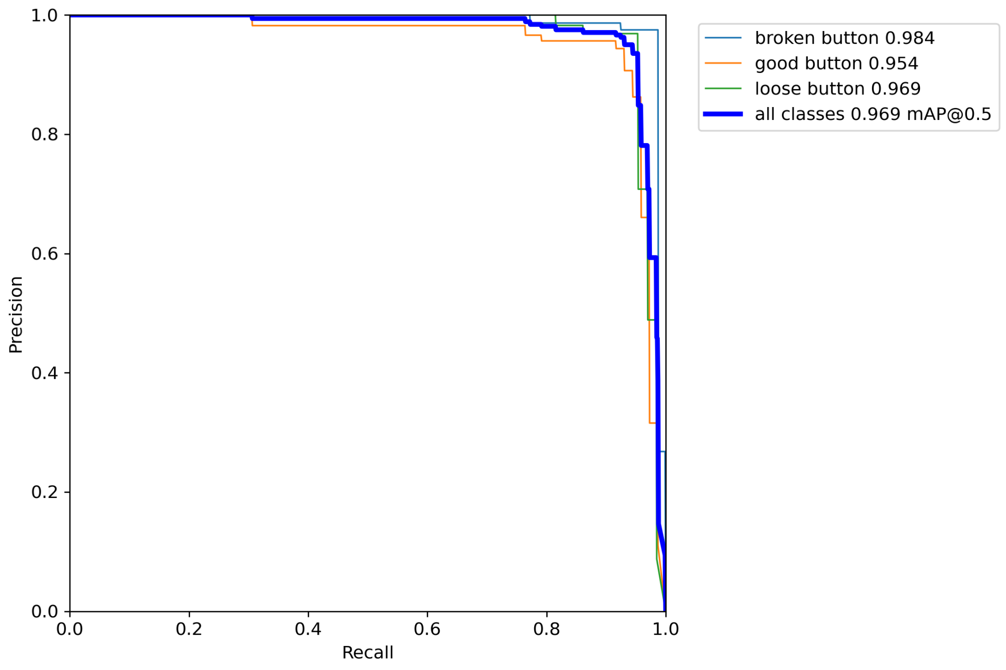

Figure 4.5: PR curve of augmented button dataset at 200 epochs.

4.3.2 Seam Dataset Performance Evaluation

Next, to evaluate our model on our seam dataset we first run our model at 100 epochs and later increase the epochs up to 300 to achieve better performance. This dataset is a binary class with the 2 following classes - good seam, defected seam where both the classes are fed with 492 and 508 data respectively. The overall mAP for the dataset is 46.6% where the mAP for the good seam is only 14.3% and for the defected seam, it's 79%. The precision and recall for both classes are 46.5% and 43.1%. Due to the low overall class mean average precision, we run 300 epochs next instead of 100 epochs.
This time, the good seam class achieves a 46.5% mAP and the defected seam class 91.9% mAP, the overall mAP is 69.2% as seen in fig 4.7. Even though all the mean average precisions have increased, due to the imbalanced mAP ratio for the 2 classes, the overall mAP has decreased significantly. From fig 4.8, The confusion matrix shows that the model has been able to correctly classify 90% of good seams while classifying 1% incorrectly. In the case of the good seam, 53% were classified correctly making a 47% misclassification.

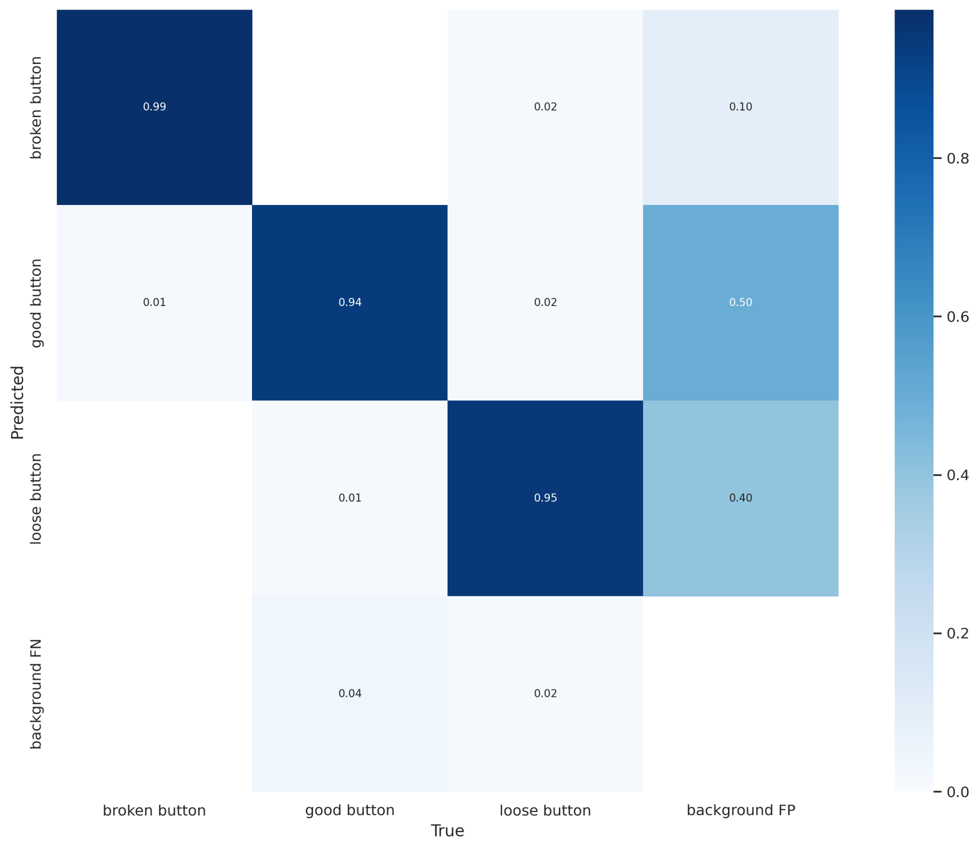

Figure 4.6: Confusion Matrix of augmented button dataset at 200 epochs.

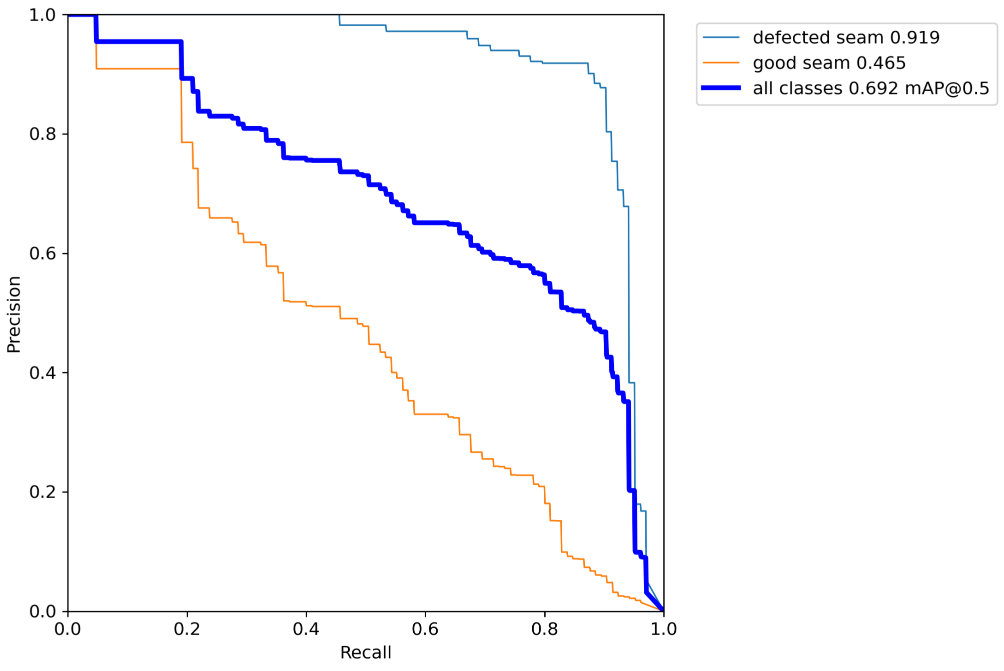

Figure 4.7: PR curve of seam dataset at 300 epochs.

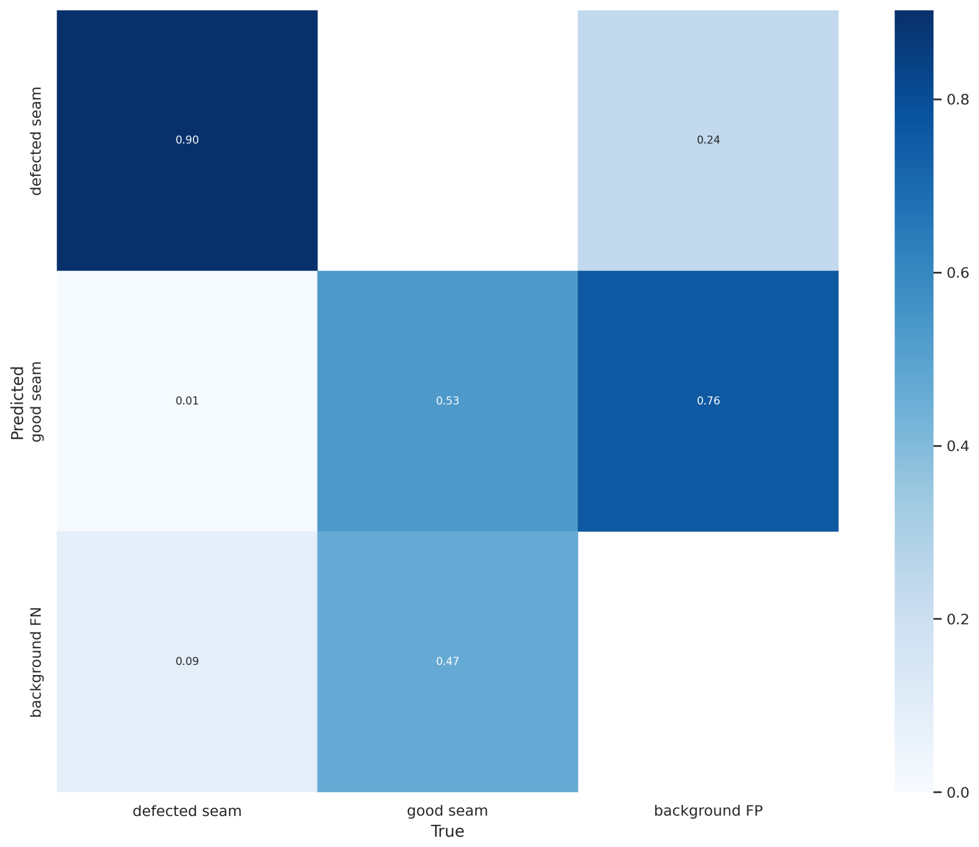

Figure 4.8: Confusion Matrix of seam dataset at 300 epochs.

Just like the button dataset, to enlarge our dataset, we applied the 90-degree rotation augmentation technique. The new seam dataset has 1,237 images. Again, for a fair comparison, we compared these 2 models at 300 epochs. fig 4.9, the overall class mAP for the augmented dataset is 76.8% mAP increasing it by 6.9%. As for the individual classes, the defected seam class mAP has decreased by 5% mAP but good seam mAP has gone up by 15.8%. From fig 4.10, the confusion matrix shows that this time, the model could classify 91% defected seams correctly and 1% were classified as background false positives. For the good seams class, 6% of seams were correctly classified and 4% were misclassified. The good seems class has learned to classify 7% more data correctly than before if we compare with fig 4.8

4.3.3 Zipper Dataset Performance Evaluation

Starting with 100 epochs we evaluate our model on the zipper dataset and later increase the epochs to 200 and achieve better performance. This dataset too is a binary class with the classes - good zipper and broken zipper where both the classes are fed with 287 and 250 data respectively. For the first 100 epochs, the overall mAP for the dataset is 0.763 where the mAP for the good zipper is 85% and for the broken zipper, it's 67.7%. The precision and recall for both classes are 65.1% and 71.6%. While the mAPs are mostly below 8%, it's now very low. So, we move on to 200 epochs. This time, the good zipper class achieves a 96.5% mAP and has therefore increased by 11.5%. Meanwhile, the broken zipper class has increased by 29% and has now 97.6% mAP. Along with these 2 classes achieving very high

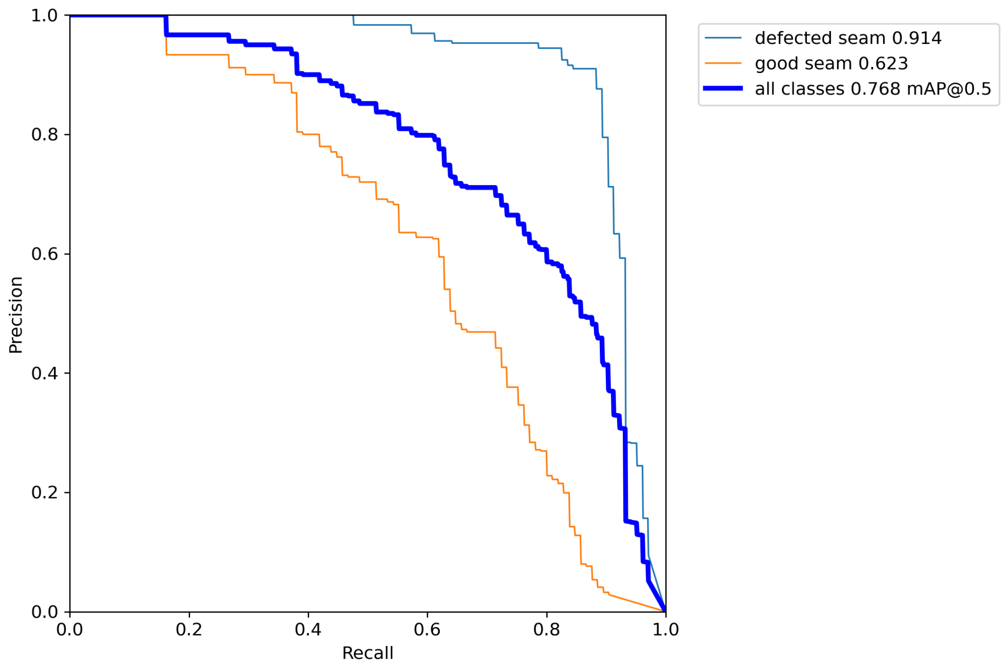

Figure 4.9: PR curve of augmented seam dataset at 300 epochs.

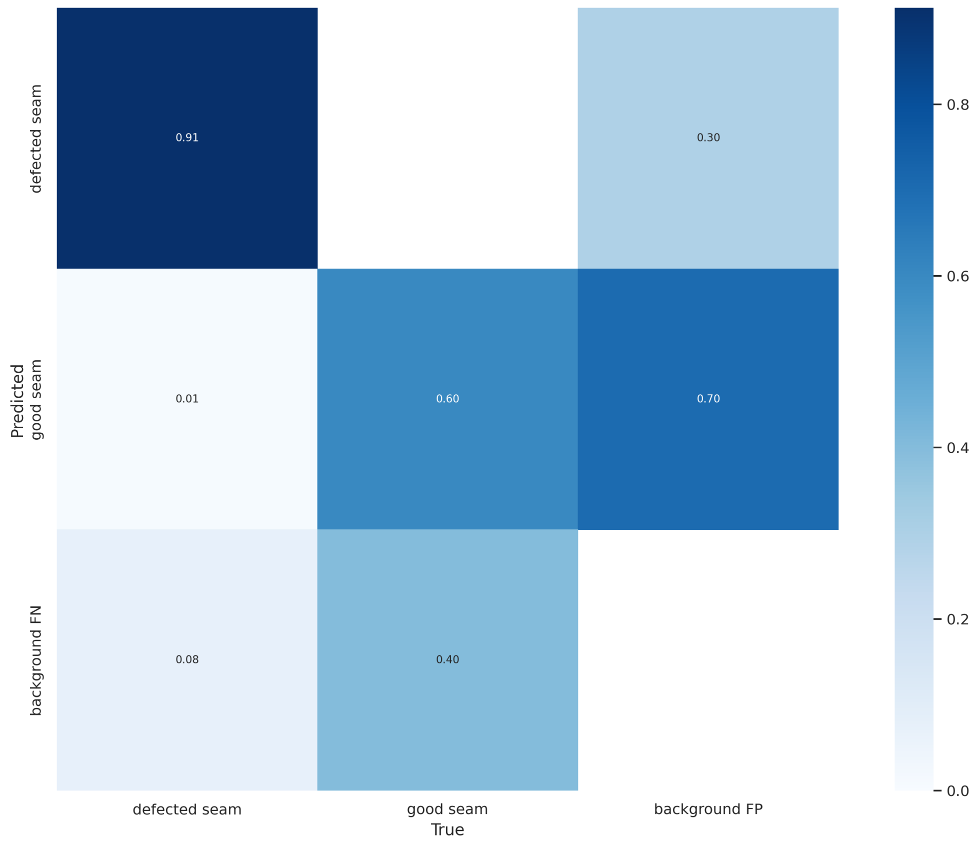

Figure 4.10: confusion matrix of augmented seam dataset at 300 epochs.

mAPs, the overall mAP has also increased to 97.1% as seen in fig 4.11 From fig 4.12, the confusion matrix shows that the model for this particular set of datasets has been able to correctly classify 98% of broken zippers while classifying only 2% incorrectly. In the case of good zippers, 89% were classified correctly making an 11% misclassification. Next, we opt for augmentation for this set of datasets too in order to increase the data quantity. However, at 200 epochs, the augmented zipper dataset starts to underfit. In fig 4.13, we can see that the overall class mAP for the augmented dataset has decreased by 20.9% and is at 76.2% mAP. We compared these 2 models at the same epochs, which is 200 epochs, for a fair comparison. From fig 4.14, the confusion matrix shows that the model classified 57% of broken zippers correctly and incorrectly classified 43%. This model misclassifies 41% more data than fig 4.12 for the broken zipper class. Similarly, for the good zipper class, the augmented dataset misclassifies 12% more data. we analyze this issue further in fig 5.1.2.

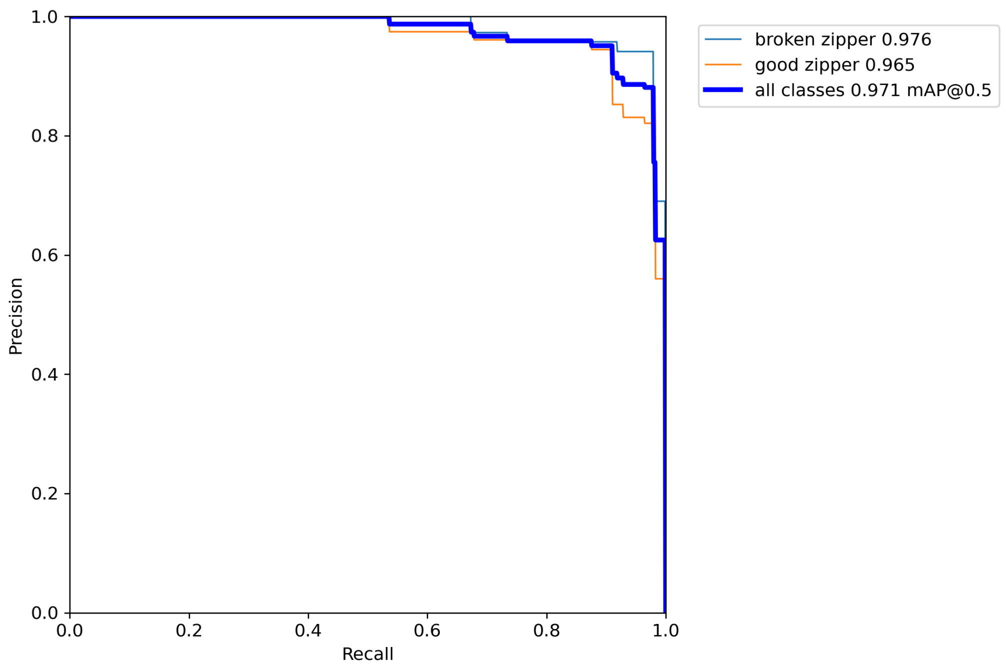

Figure 4.11: PR curve of zipper dataset at 200 epochs.

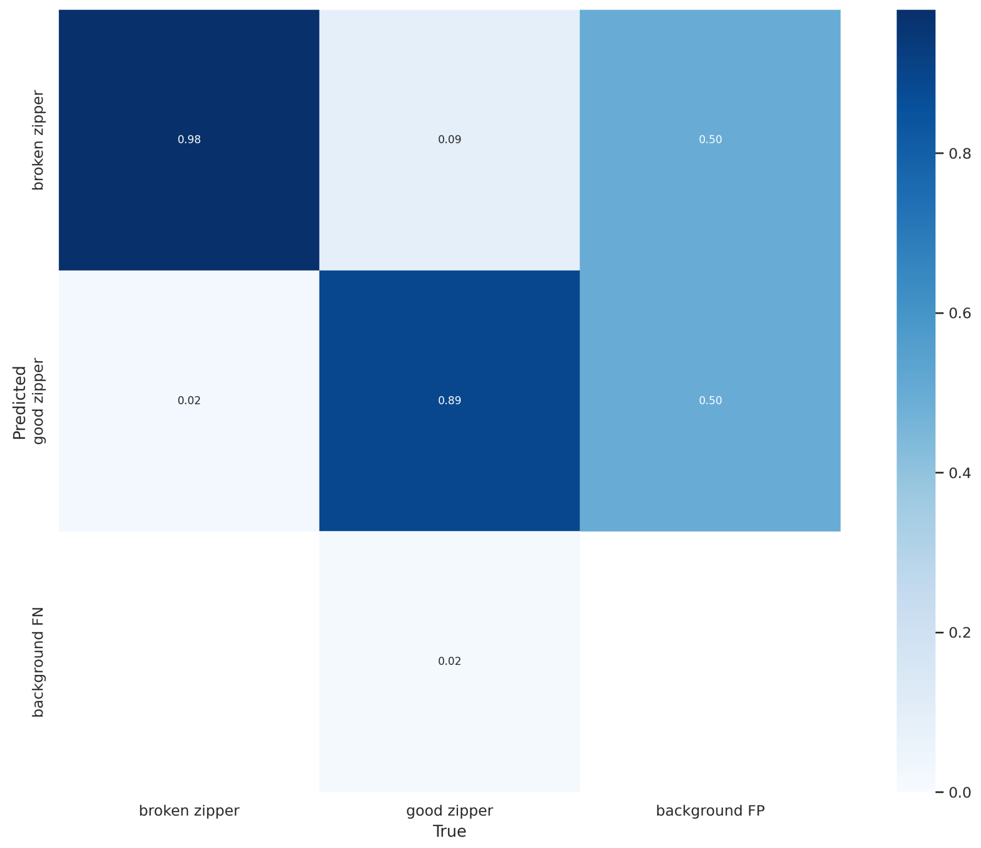

Figure 4.12: Confusion Matrix of zipper dataset at 200 epochs.

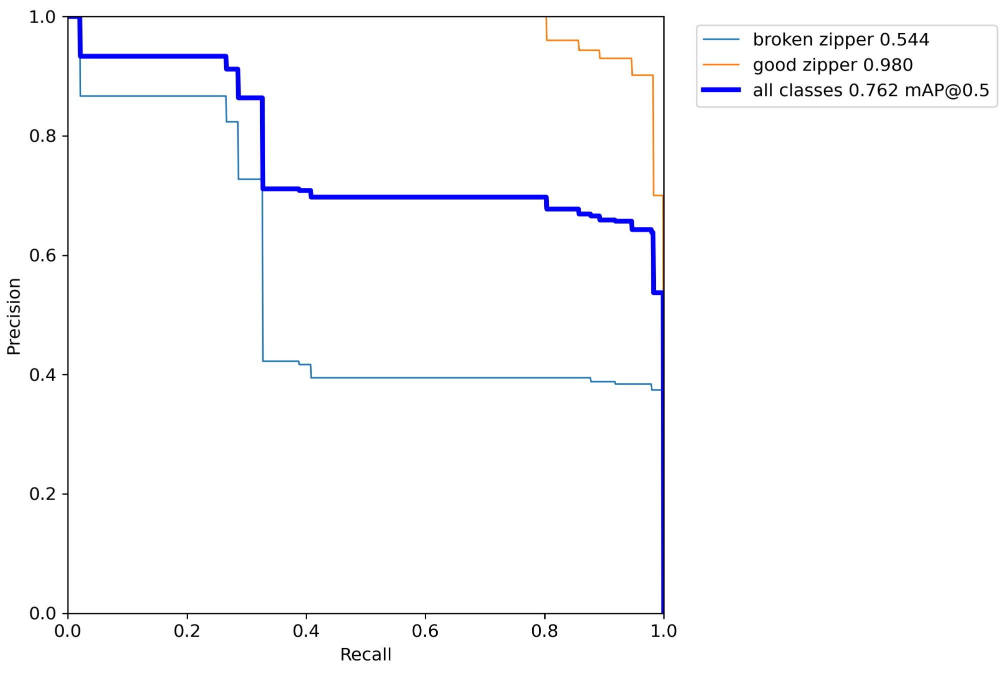

Figure 4.13: PR curve of augmented zipper dataset at 200 epochs.

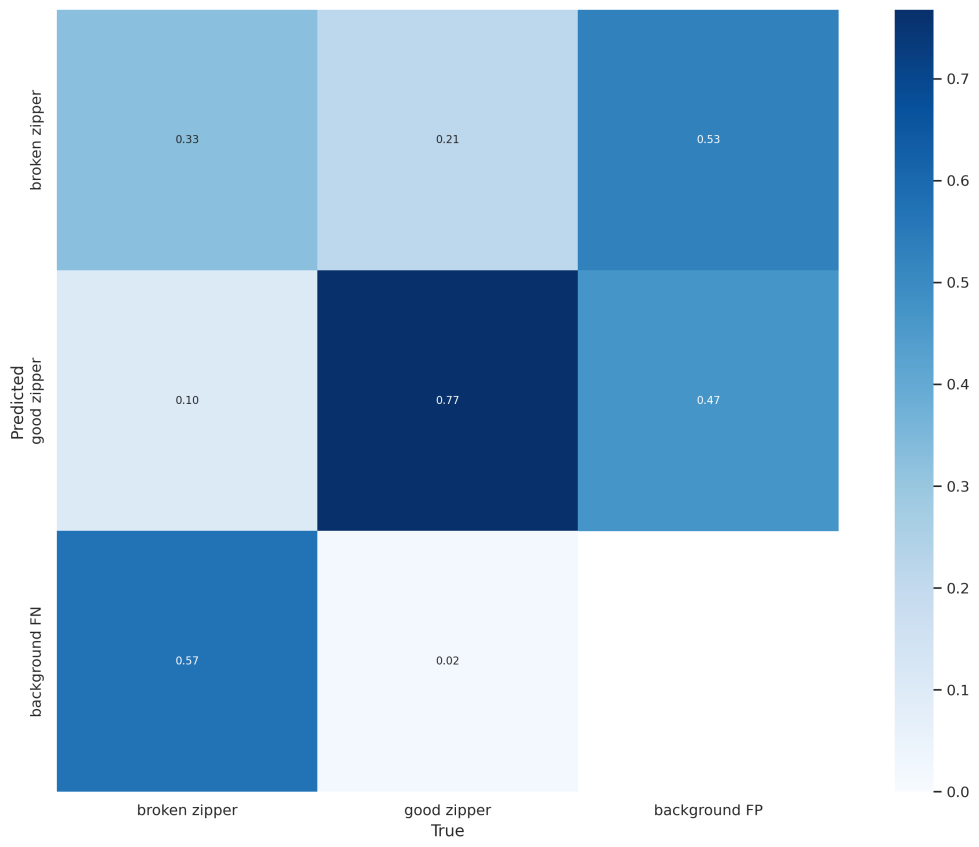

Figure 4.14: Confusion Matrix of augmented zipper dataset at 200 epochs.

Chapter 5

Result Analysis

5.1 Visual Analysis

5.1.1 Experiment-1

In our first collected dataset, we had exactly 5,050 data, and 7,061 annotations across 20 classes. This set of data resulted poorly when it was run in the YOLOV5 model. It had an overall precision of - 15.1%, recall - 20.7%, and mAP - 10.9%. After studying all the annotated data again, we found several issues that could be contributing to the low accuracy of our model. The first issue we noticed is too much variety of data. Different colors and patterns to different shapes of data were present in our dataset. While having variety in the dataset can be a good quality of our dataset, we need enough training data as a prerequisite to support this statement. Another mistake was that the classes were very heavily imbalanced. For example, "good button" is an over-represented class with 850 annotations, and "loose button" is an under-represented class with only 171 annotations. Class balance is a necessity for the model to be able to classify the dataset accordingly without giving one class precedence. Due to the lack of training data, this model is incapable of detecting unseen data. so even when the model was fed more training images for certain classes, those classes were not able to detect with a confidence threshold of at least 50%. we attach some examples of low confidence thresholds in fig 5.1, fig 5.2, fig 5.3. We also decided to get rid of some classes such as missing buttons, missing zippers, etc. These classes were designed to look for when an object is missing. However, it needs a preprocessing module so that the model can calculate and look for the buttons only in areas it can be missing. This class surged the model's chances to increase false positives. The test result showed that our simple approach to detecting missing objects misled the model. fig 5.4
Among these data, there was no background example. Ideally, practitioners such as [43], recommend adding up to 10 percent background images so that the model can learn when an image does not have the desired object to detect. In this case, we should have included more null examples, preferably up to 500 background images for 5,050 images of the dataset. While going through the annotation we noticed that some of the annotation bounding boxes were done on partially visible data. This caused the model to miss classify when the object is not broken but partially visible and when the object is actually broken, which leads the model to fail to recognize the difference between a broken button and a proper button. This confusion defeats the

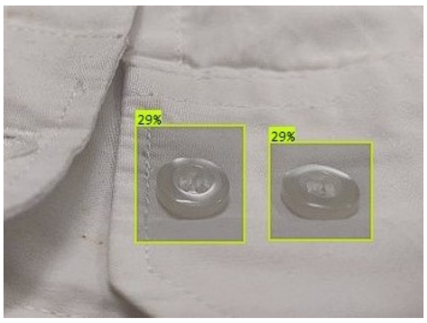

Figure 5.1: ”good button” class test image result

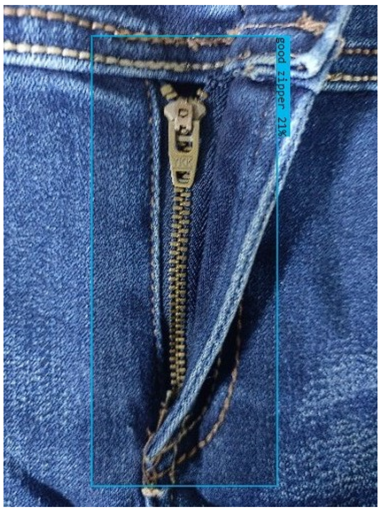

Figure 5.2: ”good zipper” class test image result

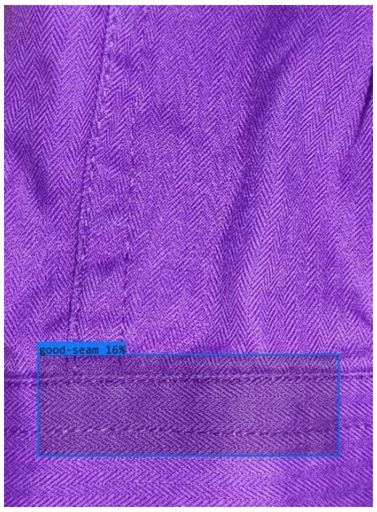

Figure 5.3: ”good seam” class test image result

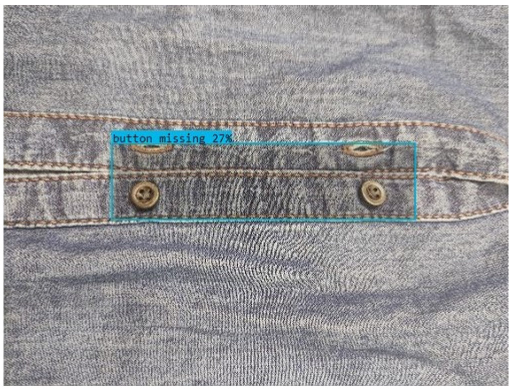

Figure 5.4: ”missing button” class test image result

purpose of creating the model, which is to identify the defects of garments. Similarly, for zippers, labeling them simply using a bounding box brought down the accuracy of classes relevant to zippers. While running test images we noticed that although with a very low confidence threshold, some zipper teeth were being detected as good zippers. Certainly, better annotation will create better results from our model.

5.1.2 Experiment-2

Our button dataset has performed well after its first 100 epochs of training in the model and it performs even better after 300 epochs of training. as we can see in fig 5.5 the model has been able to detect all 3 buttons in the image accurately and has a confidence level of 85%, 77% and 88%. However, after applying augmentation to increase the quantity of this dataset we observed a surge in its pr curve fig 4.3, which has increased by 2.7% mean average precision. we notice how confidence thresholds for the same test image increase after augmenting the button dataset in fig 5.6. for this model, the image has 85%, 88%, and 85% confidence for the buttons starting from left to right. we can conclude, our model has learned more sufficiently. Similarly, for fig 5.7, the ”broken button” of the image has been detected accurately with 88% confidence. in fig 5.8, the loose button has a confidence of 93%. Finally, We have a total of 1,337 images in our button dataset. Thus we conclude that the augmentation helps us to achieve better training results in YOLOv5. Due to the success of this experiment, we have applied augmentation to the dataset of seams and zippers.
In the case of the seam dataset, we compare the test results with our metric’s performances. After the first 100 epochs, the defected seam class had a better performance than the good seam class. An example is the fig 5.9 where the defected part of the seams are accurately detected although with a low confidence threshold of 57%, 47% for defected seam, and only 12% for the good seam. However, the good seam has a slightly inaccurate result where the model has detected part of the ”defective seam” as a ”good seam”. Along with the precision and recall we can see the confidence of the model going up after 300 epochs in fig 5.10 where the confidence is 59% and fig 5.11 where the confidence is 83%. We augment the dataset to check its performance after it’s fed more training data and we do notice an increase in the mean average precisions as discussed before in fig **??**. Similarly, we see the good seam has a better confidence level in fig 5.12. However, the ”good

seam" class is still underperforming. So, we trace back to our original dataset in order to check why the result for good seams is repeatedly lesser than the rest of the classes in the entire dataset. We found out that in our particular dataset of seams, the color of the garment and the color of the thread used to sew the seam are graphically hard to identify. However, in the case of a defected seam class, the defects are graphically easy to identify and thus we got better results for this class. In this case, even if we increase the epochs to make the model learn what a "good seam" looks like, the "defected seam" class will start to overfit due to it already achieving high mean average precision.
Essentially, the zipper dataset had a near-average result after the first 100 epochs which did very well after 200 epochs. With better precision and recall we achieve a better confidence level of our model. for example in fig 5.13 and fig 5.14 we achieve 81% and 71% confidence respectively. However, despite the good result when we run the augmented dataset for the zipper, we see a downfall in its mAPs if we compare fig 4.11 and fig 4.13. While after 100 epochs the overall class had achieved 76.3% mAP, the augmented dataset achieved 66.7% mAP. We see a similar result in the test results. fig 5.13 has a higher confidence threshold than 5.15 when we test the models using the same test image. This phenomenon where the data starts to underperform severely and can not generate output in the light of the input anymore is called data underfitting, [60]. So we leave the original dataset for the zipper as it is. Finally, including the augmented images, we conclude that we have the following dataset - 1,337 images of buttons, 587 images of zippers, and 1,237 images of seams.

5.2 Graphical analysis

To achieve better accuracy we increased the epoch numbers up to 300 epochs depending on how the datasets perform. However, increasing epochs increased the time taken for the code to run as well. we show a graphical analysis of how with more training data and more epochs, the model takes more time in fig 5.17 and a tabular form of the performance metrics in fig 5.1.

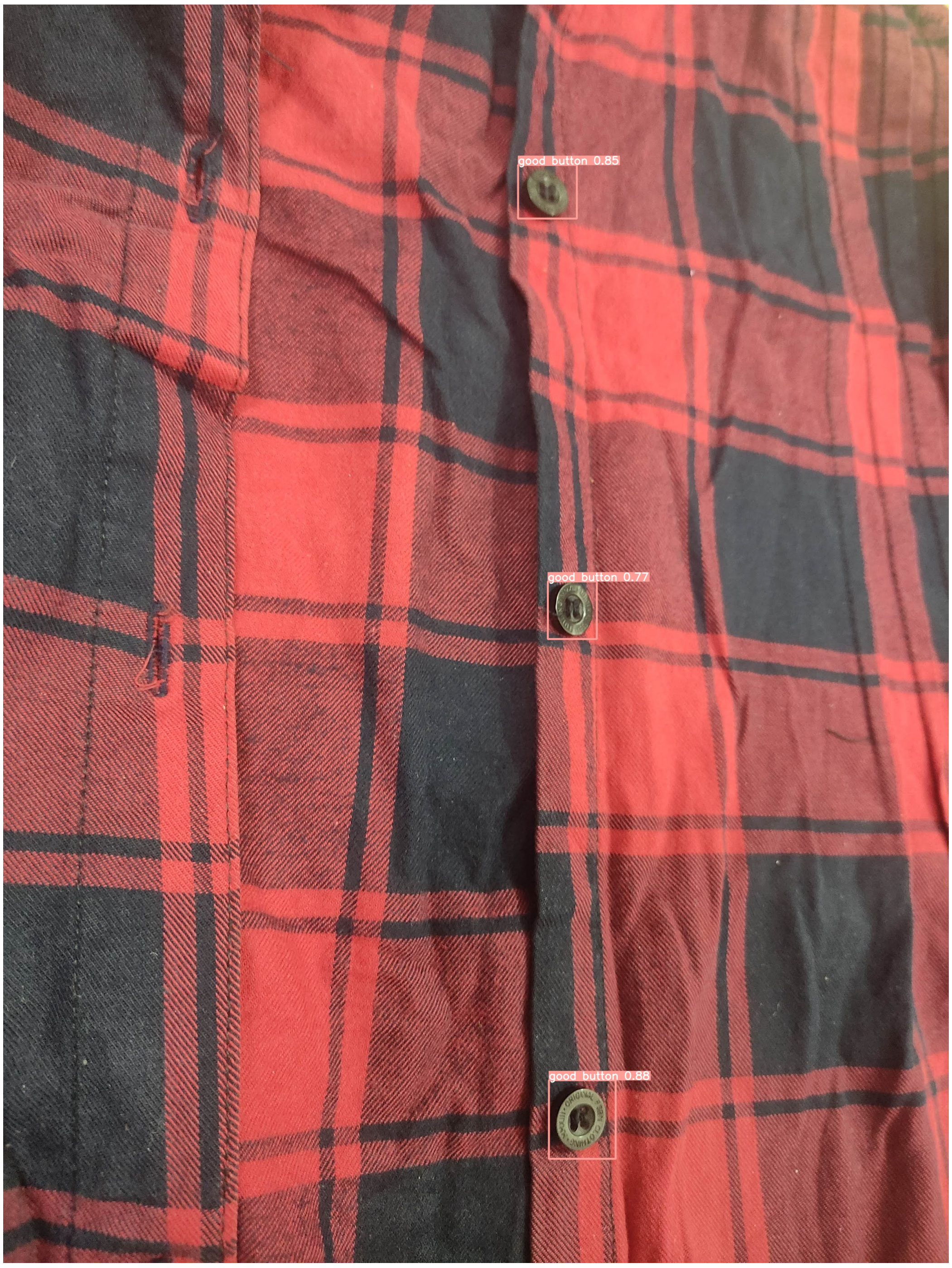

Figure 5.5: ”good button” class test image result after 200 epochs

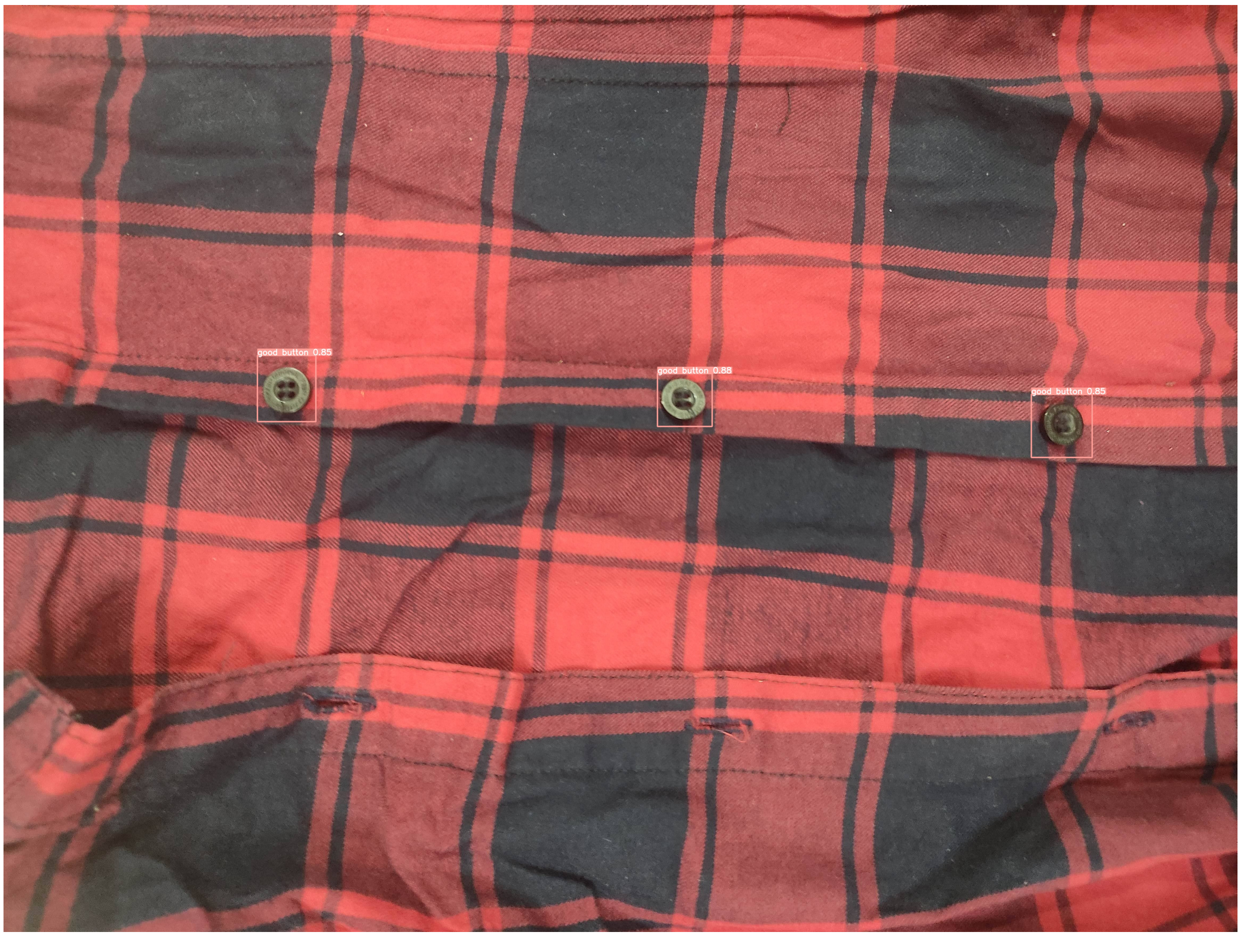

Figure 5.6: augmented "good button" class test image result after 200 epochs

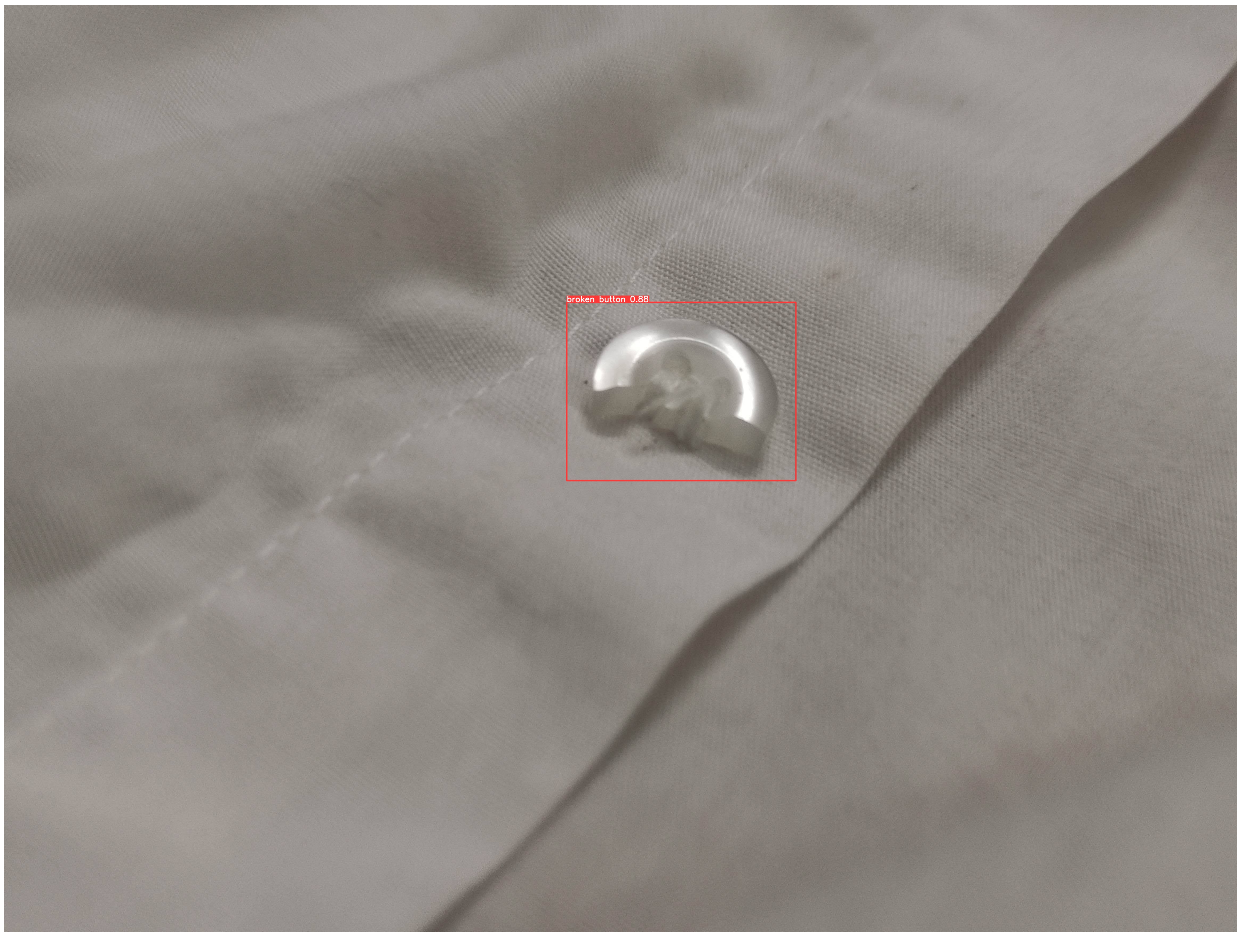

Figure 5.7: ”broken button” class test image result after 200 epochs

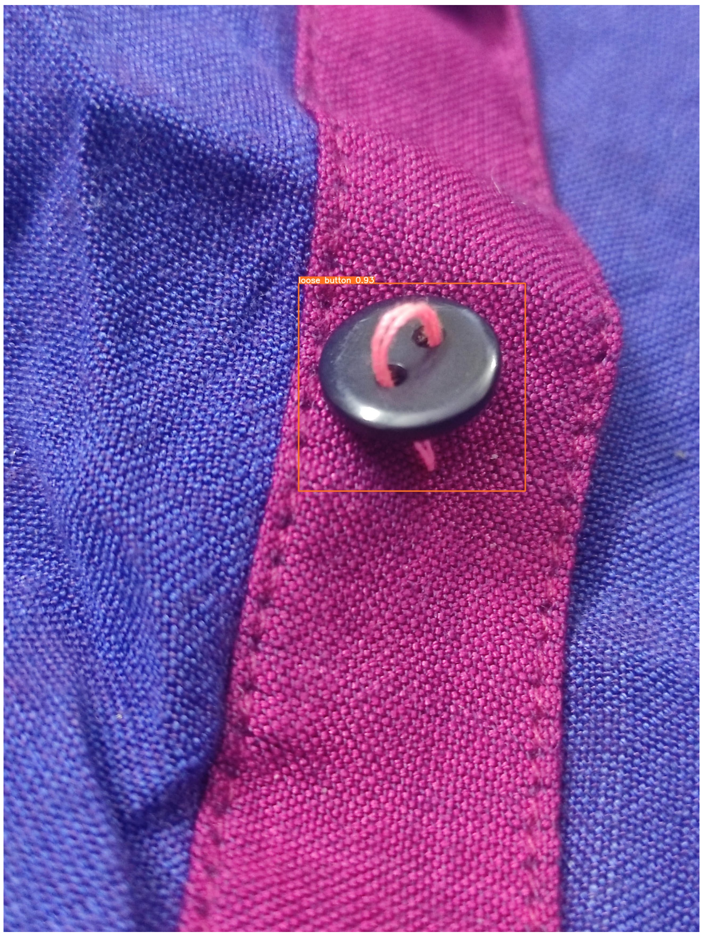

Figure 5.8: "loose button" class test image result after 200 epochs

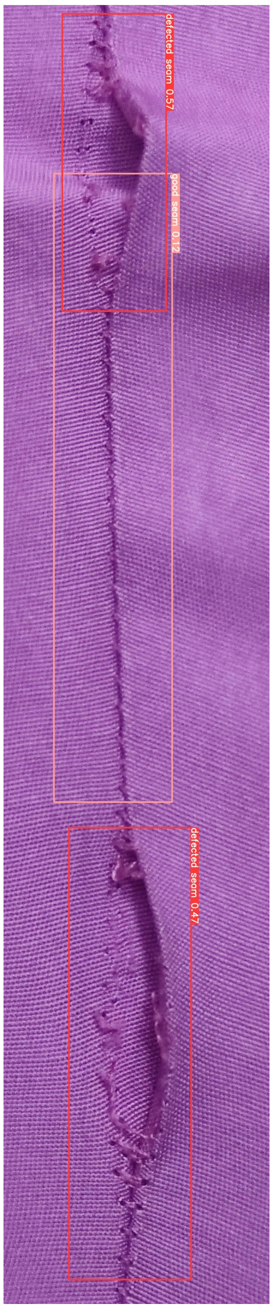

Figure 5.9: ”good seam” class test image result after 100 epochs

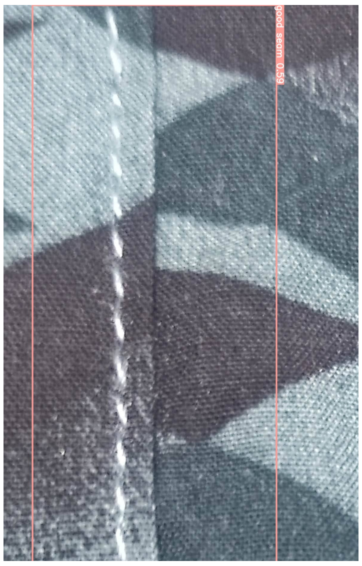

Figure 5.10: "good seam" class test image result after 300 epochs

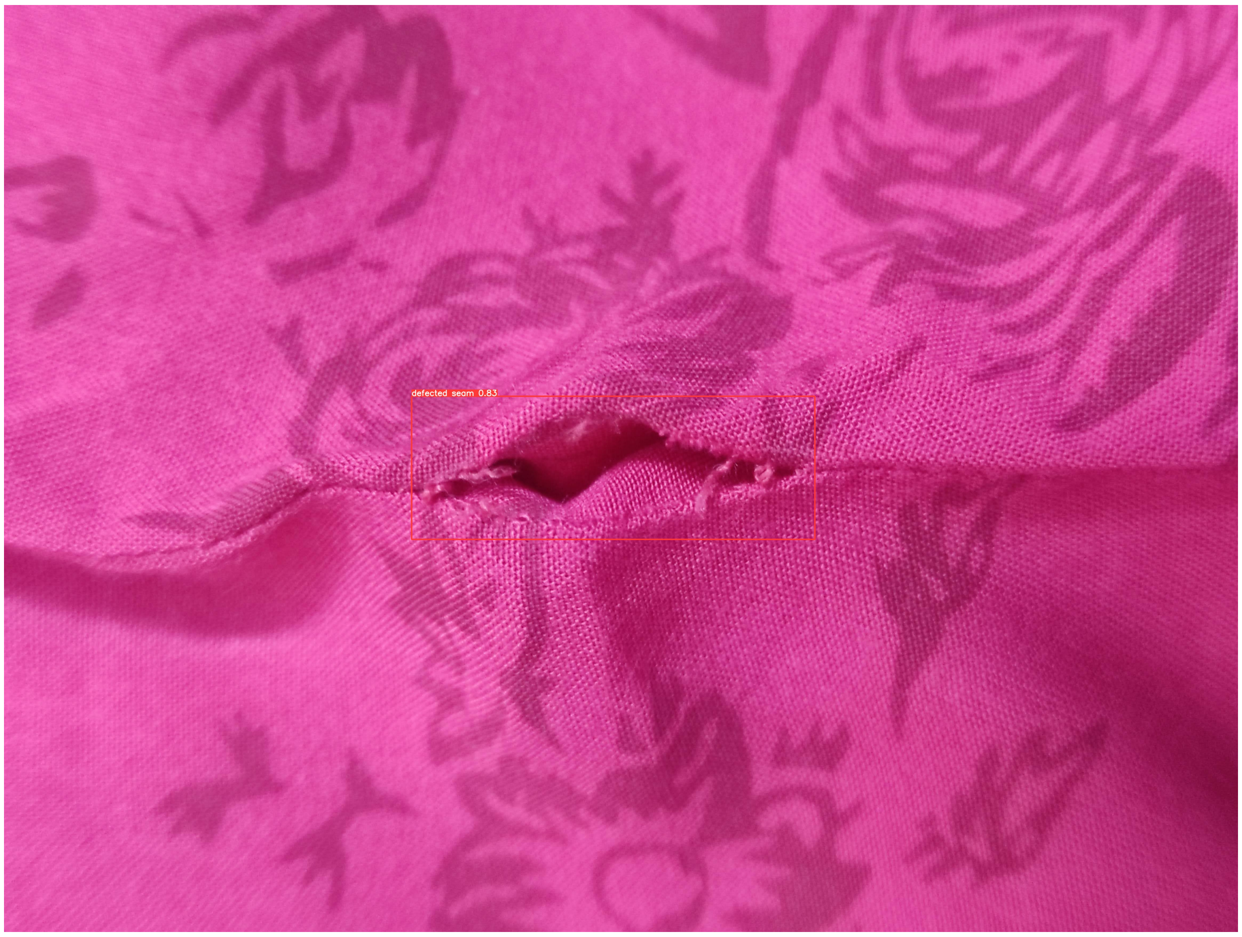

Figure 5.11: "defected seam" class test image result after 300 epochs

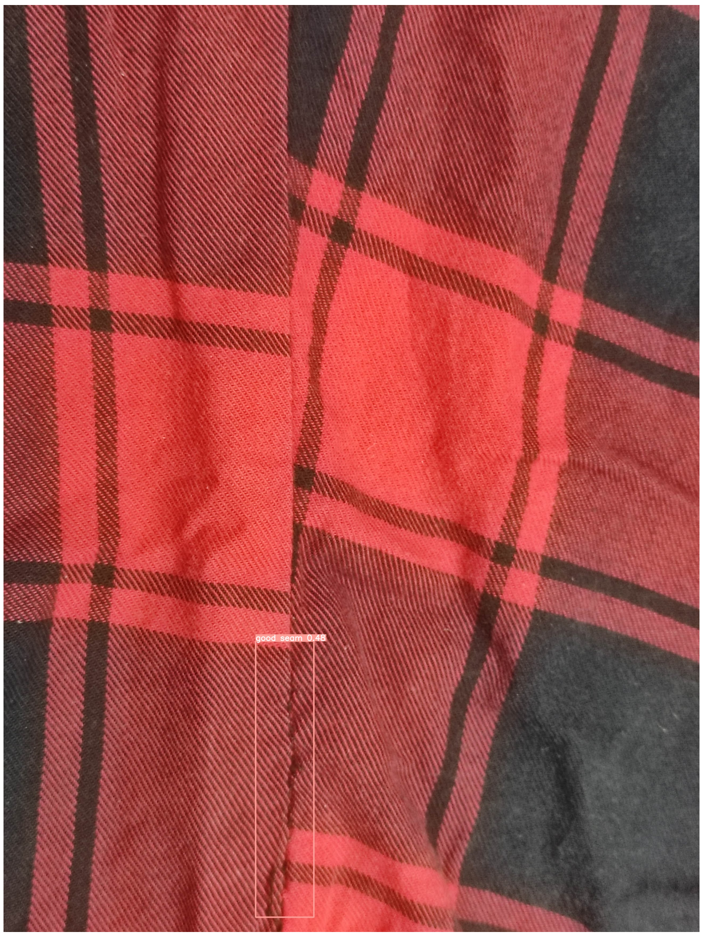

Figure 5.12: augmented "good seam" class test image result after 300 epochs

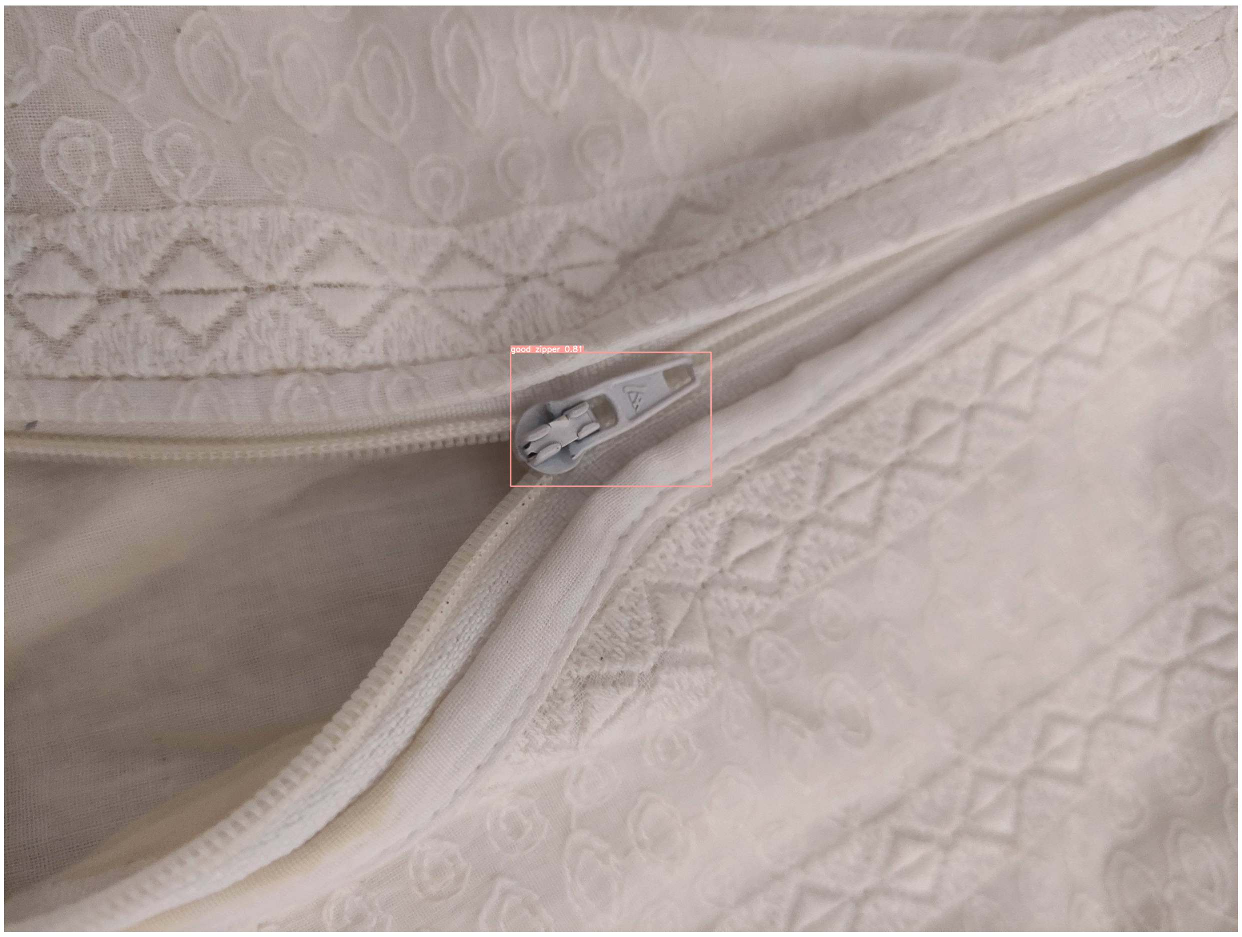

Figure 5.13: "good zipper" class test image result after 200 epochs

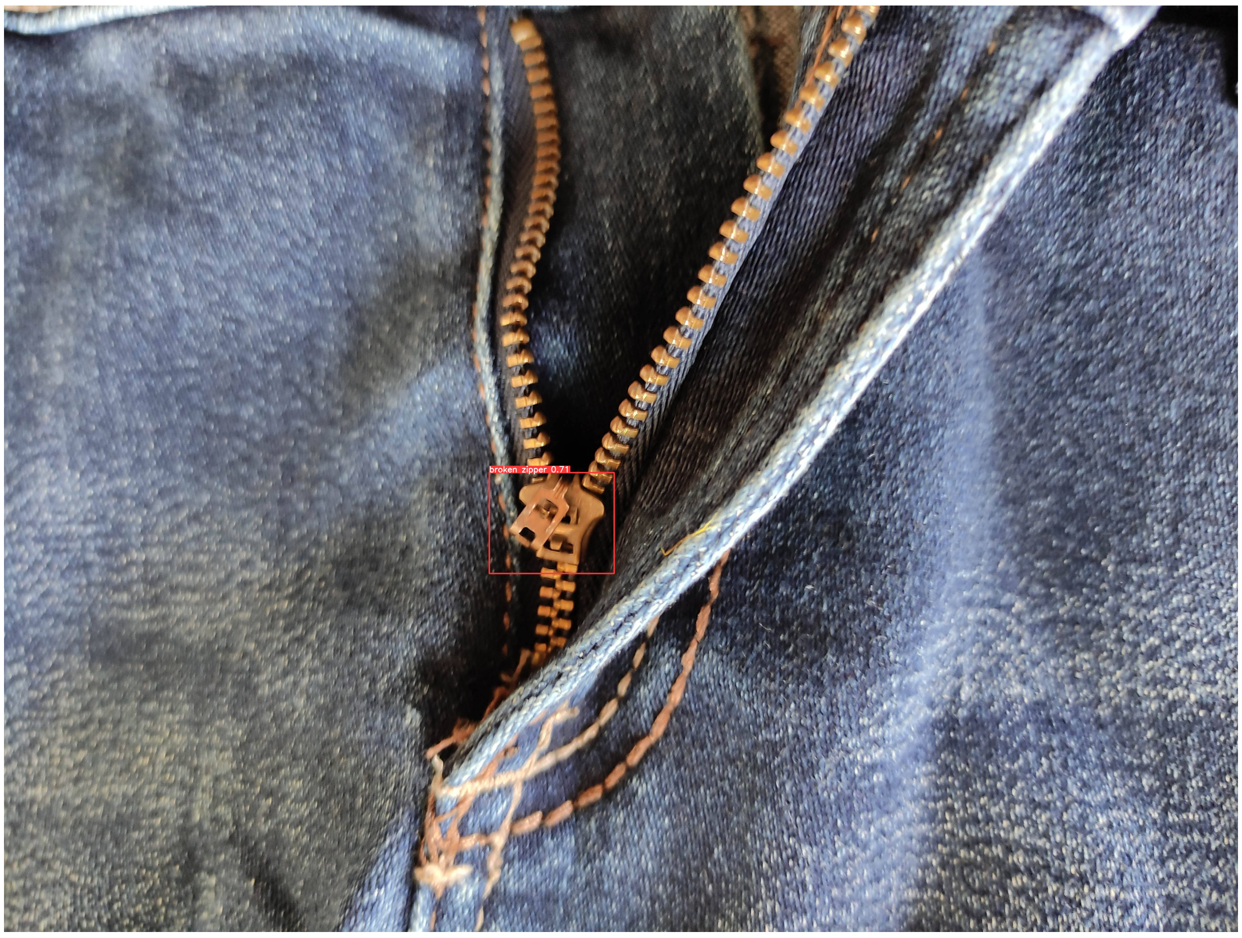

Figure 5.14: "defected zipper" class test image result after 200 epochs

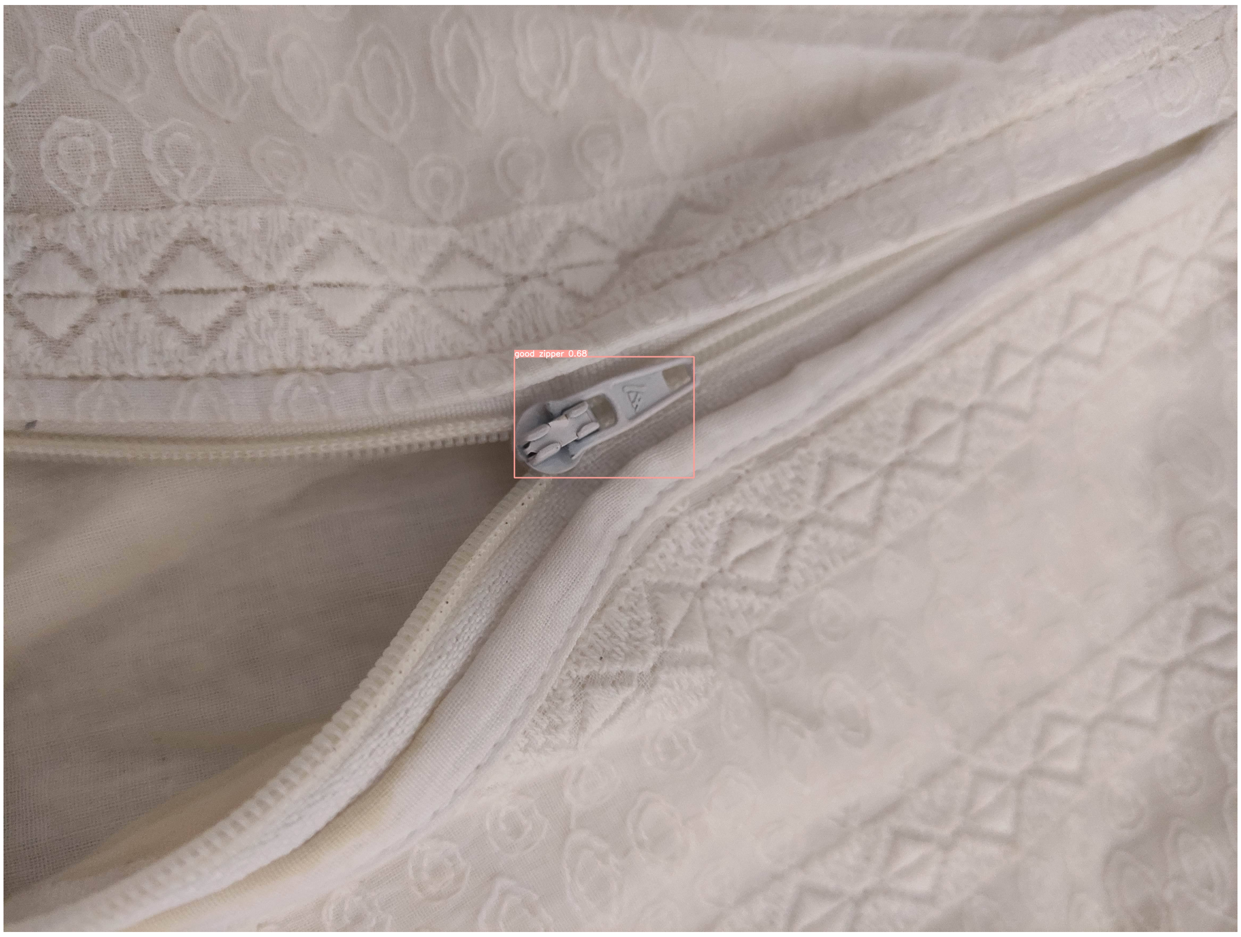

Figure 5.15: augmented "good zipper" class test image result after 200 epochs

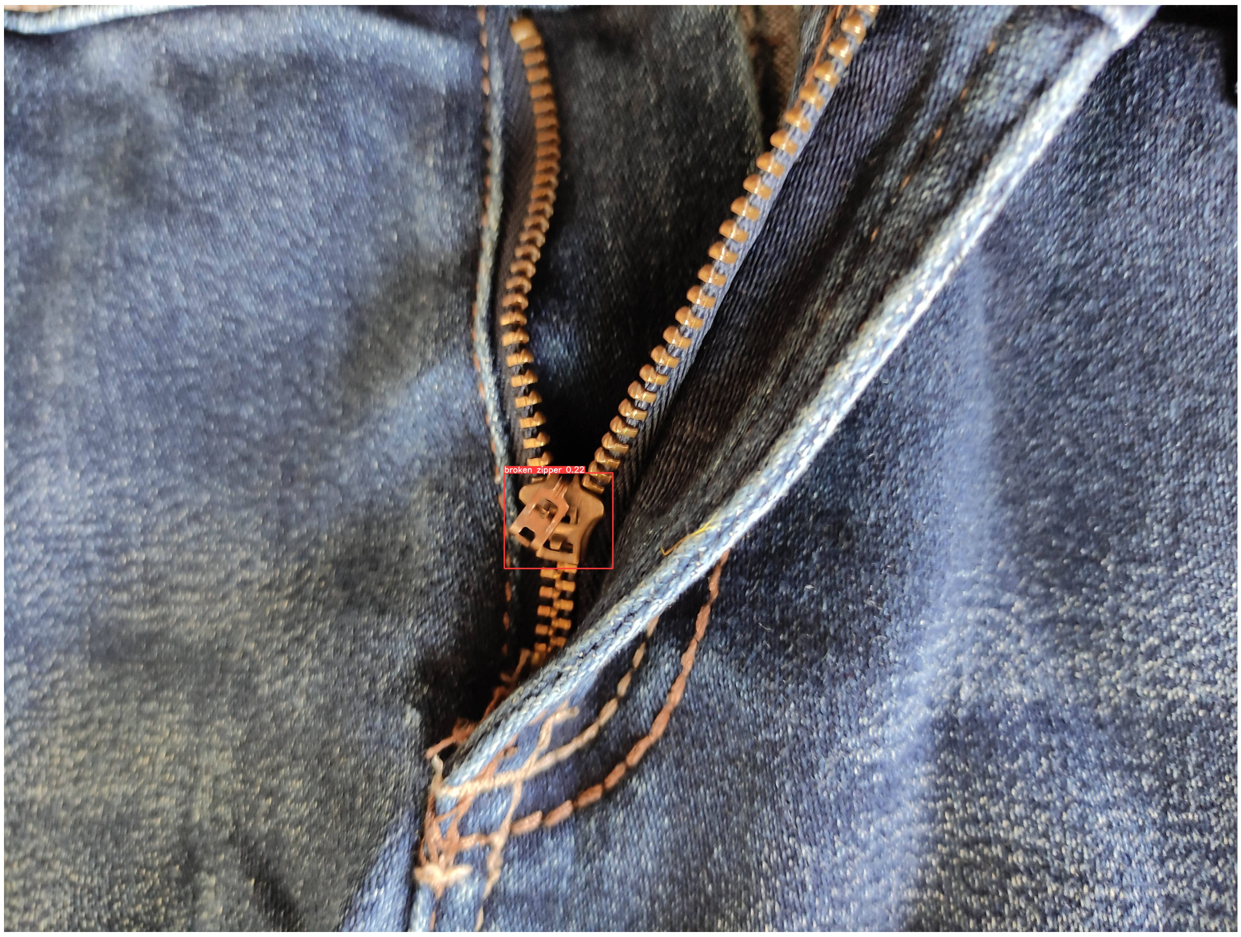

Figure 5.16: augmented "defected zipper" class test image result after 200 epochs

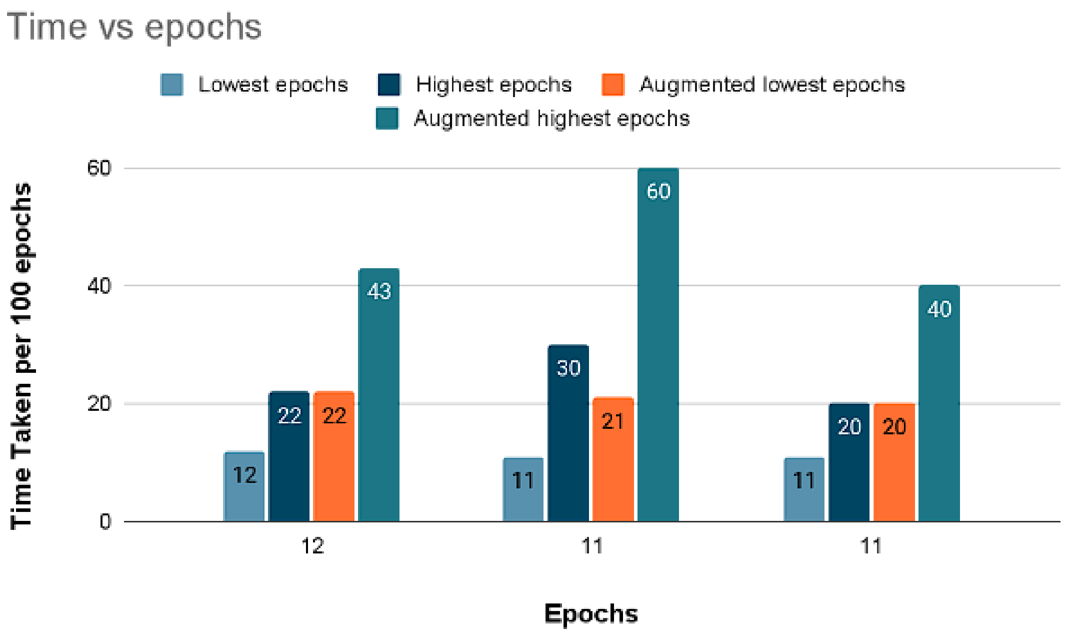

Figure 5.17: comparing each epoch for the datasets given the time taken for each epoch

	100 epochs	200 epochs
Button Dataset	Precision: 91.9% Recall: 87.4% mAP.5: 95.1% Time: 11 min 46-sec	Precision: 94.8% Recall: 92.1% mAP.5: 96.3% Time: 22 min 14-sec
Augmented Button Dataset	Precision: 91.9% Recall: 94% mAP.5: 95.8% Time: 22 min 19-sec	Precision: 96.2% Recall: 95% mAP.5: 96.9% Time: 43 min 8-sec
Zipper Dataset	Precision: 65.1% Recall: 71.6% mAP.5: 76.3% Time: 10 min 33-sec	Precision: 92.5% Recall: 92.7% mAP.5: 97.1% Time: 20 min 1-sec
Augmented Zipper Dataset	Precision: 42.5% Recall: 91.6% mAP.5: 66.7% Time: 20 min 10-sec	Precision: 50% Recall: 94.4% mAP.5: 76.2% Time: 39 min 38-sec
	100 epochs	300 epochs
Seam Dataset	Precision: 46.5% Recall: 43.1% mAP.5: 46.6% Time: 10 min 35-sec	Precision: 66.9% Recall: 68.1% mAP.5: 69.2% Time: 29 min 41-sec
Augmented Seam Dataset	Precision: 70.1% Recall: 66.6% mAP.5: 68.5% Time: 21 min 15-sec	Precision: 78.5% Recall: 71.3% mAP.5: 76.8% Time: 59 min 52-sec

Table 5.1: Overall class information of all datasets.

Chapter 6

Limitations and further work

While trying to achieve higher accuracy, we had to sacrifice making a faster model. However, in the fast-paced world of the garment industry, it is crucial that we give an accurate result along with a fast result. Since faster inaccurate results can cause a lot more damage than slower accurate results.

Using preprocessing modules, we can use these datasets to find more critical defects that need more computation such as - we can look for missing objects among multiple objects, detect objects of different colors, or sizes and identify such defects. We can also make a hybrid model based on the YOLO series that will give us faster and more accurate results. Our dataset can be of contribution in the said cases.

Chapter 7

Conclusion

Our multiple attempts to make a dataset have taught us a lot. The YOLOv5 model has achieved good accuracy in our final set of data.
The implementation of Artificial Intelligence in the garment sector to detect defects in garments can broaden the horizon of our new economical and technological era. Though previously there was some work done on fabric defect detection, our research topic can be useful to find the bottleneck of the garment industries' inspection method swifter by object detection methodology. While finding the flaws the inspection sector can find out the mistake of a certain department and handle it within a short span of time. Another obstacle had arisen in the garment industry when the pandemic struck Bangladesh. Along with other educational, commercial, and governmental institutions, industrial institutions like garment and textile factories were shut down at that moment, as there were so many people who had to work in a congested, dense, and enclosed space. Our economy had to bear this loss greatly at that moment. So the garments and other industries that had help from artificial intelligence could keep the industry running with fewer people being physically present.
To conclude, an AI-based model can be a solution for detecting defects and handling bottlenecks in garments and textile industries while being a time savior with complication and error.

Bibliography

[1] W. K. Wong, C. Yuen, D. Fan, L. Chan, and E. Fung, "Stitching defect detection and classification using wavelet transform and bp neural network," *Expert Systems with Applications*, vol. 36, no. 2, pp. 3845–3856, 2009.

[2] *7 ways to reduce wip from bottleneck in garment industry*, 2012. [Online]. Available: https://www.onlineclothingstudy.com/2012/01/7-ways-to-reduce-wip-from-bottleneck.html.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, *You only look once: Unified, real-time object detection*, 2015. DOI: 10.48550/ARXIV.1506.02640. [Online]. Available: https://arxiv.org/abs/1506.02640.

[4] J. Redmon and A. Farhadi, *Yolo9000: Better, faster, stronger*, 2016. DOI: 10.48550/ARXIV.1612.08242. [Online]. Available: https://arxiv.org/abs/1612.08242.

[5] Y. Han, Y. Wu, D. Cao, and P. Yun, "Defect detection on button surfaces with the weighted least-squares model," *Frontiers of Optoelectronics*, vol. 10, no. 2, pp. 151–159, 2017.

[6] J. Moreno, A. Aguila, E. Partida, C. Martinez, O. Morales, and R. Tejeida, "System of error detection in the manufacture of garments using artificial vision," in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, vol. 272, 2017, p. 012 014.

[7] D. Datta and P. Seal, "Various approaches in pattern making for garment sector," *Journal of Textile Engineering & Fashion Technology*, vol. 4, no. 1, pp. 29–34, 2018.

[8] M. T. Haque, M. R. Hossain, and M. S. Hasan, "Bottleneck problem reduction of a garment manufacturing industry in bangladesh by using line balancing technique," *Int J Res Adv Eng Tech*, vol. 4, no. 2, pp. 28–32, 2018.

[9] L. Liu, D. Cao, S. Wu, Y. Wu, and T. Wei, "A fast button surface defects detection method based on convolutional neural network," in *2017 International Conference on Optical Instruments and Technology: Optoelectronic Measurement Technology and Systems*, SPIE, vol. 10621, 2018, pp. 45–53.

[10] R. Nayak and R. Padhye, "Introduction to automation in garment manufacturing," in *Automation in garment manufacturing*, Elsevier, 2018, pp. 1–27.

[11] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[12] *148 garment defects found in readymade garments*, 2019. [Online]. Available: https://www.onlineclothingstudy.com/2019/04/148-garment-defects-found-in-readymade.html.

[13] M. Ahmed, T. Islam, and M. Ali, "Study on different types of defects and their causes and remedies in garments industry," *Journal of Textile Engineering & Fashion Technology*, vol. 5, no. 6, pp. 300–304, 2019.

[14] *Number of garment factories in bangladesh from 2009 to 2019*, 2019. [Online]. Available: https://www.statista.com/statistics/987697/bangladesh-number-garment-factories/.

[15] J. Silvestre-Blanes, T. Albero-Albero, I. Miralles, R. Pérez-Llorens, and J. Moreno, "A public fabric database for defect detection methods and results," *Autex Research Journal*, vol. 19, no. 4, pp. 363–374, 2019.

[16] S. Wu, Y. Wu, D. Cao, and C. Zheng, "A fast button surface defect detection method based on siamese network with imbalanced samples," *Multimedia Tools and Applications*, vol. 78, no. 24, pp. 34 627–34 648, 2019.

[17] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[18] *Details about bottleneck in garment industry*, 2020. [Online]. Available: https://www.onlineapparelstudy.com/2020/01/Details-about-Bottleneck.html.

[19] M. Hosen, "Reduction of garments bottleneck processing time on the sewing line of the garments industries," *International Journal of Textile and Fashion Technology*, vol. 5, pp. 1–10, May 2020. DOI: 10.33552/JTSFT.2020.05.000611.

[20] A. Rasheed, B. Zafar, A. Rasheed, *et al.*, "Fabric defect detection using computer vision techniques: A comprehensive review," *Mathematical Problems in Engineering*, vol. 2020, 2020.

[21] X. Zhang, Q. Wang, J. Liu, Z. Liu, and J. Gong, "Zipper classification and defect detection based on computer vision," in *2020 39th Chinese Control Conference (CCC)*, IEEE, 2020, pp. 6521–6526.

[22] *7 different types of seams and how to use them in garments*, 2021. [Online]. Available: https://www.masterclass.com/articles/7-different-types-of-seams-and-how-to-use-them-in-garments.

[23] *A guide to yolo models*, 2021. [Online]. Available: https://blog.roboflow.com/guide-to-yolo-models/.

[24] *Anchor boxes for object detection*, 2021. [Online]. Available: https://www.mathworks.com/help/vision/ug/anchor-boxes-for-object-detection.html.

[25] *Coco: Common objects in context*, 2021. [Online]. Available: https://cocodataset.org/#home.

[26] *Cspdarknet53: A convolutional neural network architecture for real-time object detection*, 2021. [Online]. Available: https://paperswithcode.com/method/cspdarknet53.

[27] T. ERSÖZ, H. ZAHOOR, and F. ERSÖZ, "Fabric and production defect detection in the apparel industry using data mining algorithms," *International Journal of 3D Printing Technologies and Digital Industry*, vol. 5, no. 3, pp. 742–757, 2021.

[28] H. Fang, M. Xia, H. Liu, Y. Chang, L. Wang, and X. Liu, "Automatic zipper tape defect detection using two-stage multi-scale convolutional networks," *Neurocomputing*, vol. 422, pp. 34–50, 2021.

[29] *Garment production process*, 2021. [Online]. Available: https://www.textileschool.com/193/garment-production-process/.

[30] *Getting rid of bottleneck operations in garments industry*, 2021. [Online]. Available: https://www.textiletoday.com.bd/getting-rid-of-bottleneck-operations-in-garments-industry/.

[31] *Inspection defect checklist for garment workmanship and appearance*, 2021. [Online]. Available: http://garmentstech.com/inspection-defect-checklist-for-garment-workmanship-and-appearance/.

[32] *Inspection steps in garments inspection*, 2021. [Online]. Available: https://textilestudycenter.com/inspection-steps-in-garments-inspection/.

[33] *Introduction to yolo algorithm for object detection*, 2021. [Online]. Available: https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/.

[34] R. Jin and Q. Niu, "Automatic fabric defect detection based on an improved yolov5," *Mathematical Problems in Engineering*, vol. 2021, 2021.

[35] S. Karunarathne and R. Ganganath, "A sovereign button detection and measure the alignment using image processing," in *2021 IEEE 16th International Conference on Industrial and Information Systems (ICIIS)*, IEEE, 2021, pp. 359–364.

[36] C. Li, J. Li, Y. Li, L. He, X. Fu, and J. Chen, "Fabric defect detection in textile manufacturing: A survey of the state of the art," *Security and Communication Networks*, vol. 2021, 2021.

[37] X. Li, L. Xu, X. Liu, and S. Sun, "Method of metal button defect detection based on extreme learning machine and sparse representation," *AATCC Journal of Research*, vol. 8, no. 1_suppl, pp. 62–68, 2021.

[38] *Line balancing and bottleneck in garment production line*, 2021. [Online]. Available: https://textilelearner.net/line-balancing-and-bottleneck-in-garment-production-line/.

[39] *Non-maximum suppression*, 2021. [Online]. Available: https://paperswithcode.com/method/non-maximum-suppression.

[40] *Spreading and layering the fabrics*, 2021. [Online]. Available: https://www.textileschool.com/336/spreading-layering-the-fabrics/.

[41] *The architecture of the yolov5 model which consists of three parts: I) backbone, ii) neck, and iii) head*, 2021. [Online]. Available: https://www.researchgate.net/figure/The-architecture-of-the-YOLOv5-model-which-consists-of-three-parts-i-Backbone_fig1_360834230.

[42] *The assembly operation in garment construction*, 2021. [Online]. Available: https://www.mcalpin-ind.com/article.cfm?ArticleNumber=33.

[43] *The practical guide for object detection with yolov5 algorithm*, 2021. [Online]. Available: https://towardsdatascience.com/the-practical-guide-for-object-detection-with-yolov5-algorithm-74c04aac4843.

[44] *Top garment manufacturing countries in world 2021*, 2021. [Online]. Available: https://newyorkspaces.com/top-garment-manufacturing-countries-in-world-2021/.

[45] *Training yolo: Select anchor boxes like this*, 2021. [Online]. Available: https://towardsdatascience.com/training-yolo-select-anchor-boxes-like-this-3226cb8d7f0b.

[46] *Understanding mean average precision*, 2021. [Online]. Available: https://www.v7labs.com/blog/mean-average-precision.

[47] *Yolact: Real-time instance segmentation*, 2021. [Online]. Available: https://medium.datadriveninvestor.com/yolact-real-time-instance-segmentation-5cbe6fc2ba36.

[48] *Yolov5*, 2021. [Online]. Available: https://github.com/ultralytics/yolov5.

[49] *Yolov8*, 2021. [Online]. Available: https://docs.ultralytics.com/#ultralytics-yolov8.

[50] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, *Tph-yolov5: Improved yolov5 based on transformer prediction head for object detection on drone-captured scenarios*, 2021. DOI: 10.48550/ARXIV.2108.11539. [Online]. Available: https://arxiv.org/abs/2108.11539.

[51] R. Ashraf, Y. Ijaz, M. Asif, K. Z. Haider, T. Mahmood, and M. Owais, "Classification of woven fabric faulty images using convolution neural network," *Mathematical Problems in Engineering*, vol. 2022, 2022.

[52] M. Boresta, T. Colombo, and A. De Santis, "Convolutional neural networks and multi-threshold analysis for contamination detection in the apparel industry," *arXiv preprint arXiv:2207.12720*, 2022.

[53] Y.-H. Chang and Y.-Y. Zhang, "Deep learning for clothing style recognition using yolov5," *Micromachines*, vol. 13, no. 10, p. 1678, 2022.

[54] H. Kim, W.-K. Jung, Y.-C. Park, J.-W. Lee, and S.-H. Ahn, "Broken stitch detection method for sewing operation using cnn feature map and image-processing techniques," *Expert Systems with Applications*, vol. 188, p. 116 014, 2022.

[55] C. Li, L. Li, H. Jiang, *et al.*, "Yolov6: A single-stage object detection framework for industrial applications," *arXiv preprint arXiv:2209.02976*, 2022.

[56] *Roboflow (version 1.0) [software]*, 2022. [Online]. Available: https://roboflow.com.

[57] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv preprint arXiv:2207.02696*, 2022.

[58] X. Wang and Z. T. Estil, "Button defect detection system using yolo algorithm," *Journal of Smart Cities*, vol. 7, no. 1, 2022.

[59] M. Islam, L. F. Sultana, L. M. S. I. Chowdhury, J. H. Mridha, and L. T. R. Sarker, "Implementation of work sharing technique to improve line efficiency in sewing section: A case study,"

[60] *What is underfitting?* [Online]. Available: https://www.ibm.com/topics/underfitting..