19/20

C059

INVESTIGATION OF AN INTELLIGENCE SOLUTION FOR SURGICAL GAUZE DETECTION IN OPERATION THEATRE

INVESTIGATION OF AN INTELLIGENCE SOLUTION FOR SURGICAL GAUZE DETECTION IN OPERATION THEATRE

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Ooi Jing Xuan

SCHOOL OF MECHANICAL AND AEROSPACE ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY

Year 2019/2020

INVESTIGATION OF AN INTELLIGENCE SOLUTION FOR SURGICAL GAUZE DETECTION IN OPERATION THEATRE

SUBMITTED

BY

OOI JING XUAN

SCHOOL OF MECHANICAL AND AEROSPACE ENGINEERING

A final year project report
presented to
Nanyang Technological University
in partial fulfilment of the
requirements for the
Degree of Bachelor of Engineering (Mechanical Engineering)
Nanyang Technological University

Year 2019/2020

Abstract

Unintended retained surgical gauze in a patient's body cavity is a significant medical error that has the potential to cause fatal harm to the patient. This error has been an existing problem for hospitals worldwide for a long time, which causes massive consequences for both the patient as well as the hospital. Currently, there are methods available to mitigate the problem which focuses on the usage of RFID (Radio-frequency identification) tags on the surgical gauze to determine the location of the gauze post-surgery. However, these methods are expensive and inefficient for hospitals. As such, this project demonstrates the application of neural networks in real-time detection of the surgical gauze to tally the number of surgical gauze that are being dispensed and returned. This would aid in minimizing the human error in accounting for surgical gauze during the whole process of the surgery. Hence, this application would increase the efficiency of accounting for all surgical gauze before the closure of patient and reversal of anaesthesia.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my project supervisor Prof. Cai Yiyu and Singapore General Hospital (SGH) surgeon, Dr. Luke Tay, for the opportunity to work on this project as well as their invaluable guidance over the course of this project. I would also like to thank Nicole Goh and Aaron Tham, both of whom are nurses from SGH, for tirelessly providing numerous images of the surgical gauze for the training of the neural network as well as comprehensive information required regarding the process of the operation.

Secondly, I would like to thank a friend, Mr Ang Peng Seng, for his guidance and expertise in machine learning. I would also like to thank my friends and family, for all the care and support that they have given me throughout the duration of this project.

Finally, I would like to thank NTU School of Mechanical and Aerospace Engineering for making this invaluable learning opportunity a possibility.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Unintended retention of surgical gauze or gauze in a patient's body cavity is known as gossypiboma [1]. Gossypibomas are a significant but avoidable surgical error that has the potential to cause significant harm to the patient. This error has resulted in professional and medico-legal consequences including major lawsuits cases, monetary compensations and humiliation for the hospitals involved [2]. Although Gossypibomas have an incidence of 0.3 to 1.0 per 1000 abdominal operations, they have the potential to threaten patients' lives, and usually an extra operation would be required for removal [3, 4].

Despite having standardised protocols to account for the usage of surgical gauze, there has been an estimation of 1500 medical cases out of over 28 million surgeries per year in United States alone that involved the unintended retention of surgical gauze [5]. For these standardized protocols during the surgical procedure, nurses are required to manually tally the surgical gauze in the mist of undertaking their primary duties of assisting the surgeons. As such, although the nurses have declared the "correct" count of surgical gauze post-operation, there are still inevitable cases of count discrepancies.

One study at a major academic medical center in 2000-2004, analyzed that 1062 of 153,263 operations had count discrepancies in surgical gauze [6], which results in an

error rate of 0.7 percent. As part of the standardised procedure, the final count discrepancies protocol were able to identify 77 percent of such cases, whilst preventing 54 percent of retained items. As such, the positive predictive value of a discrepant count leading to a retained foreign body was only 1.6 percent. Other factors that lead to an increase in count discrepancies are late time procedures, change in personnel, number of nursing teams, greater usage of surgical gauze (due to longer surgery duration).

Another study aimed to estimate the rate of intra-operative count discrepancies during a field observation of 148 cases. There were 29 count discrepancies happening in 19 cases out of 2,476 distinct counting episodes. Out of the 29 count discrepancies, 41 percent were attributed to human error while 59 percent were due to misplaced surgical items [7].

Therefore, despite the hospitals' standardised protocols which is a current attempt to reduce the number of surgical gauze count discrepancies, they are still far away from achieving zero retention of surgical gauze in patients. As such, there is still an urgent need to incorporate existing technologies available to reduce the rates of retained surgical gauze.

1.2 Current Methodologies

Currently, there are standardised hospital protocols in place for accounting the number of surgical gauze. Over the course of the surgery, the gauze are not considered as "dispensed" until requested by the surgeon. When a request is made, the scrub nurse will then place the appropriate number of gauze needed by the surgeon and note it down. Retrieval of the gauze post surgery or after usage is then done in a methodical manner where the nurse places all gauze used in a grid formation on a white piece of paper or green towel for ease of counting. The final gauze count is generally done just before the surgeon closes up the patient. Throughout the surgery, the gauze are opened in sets of 10. This would minimize the amount of time required to perform a final tally on all used gauze and expedite the completion of the surgery, especially since the

number of gauze utilised could easily exceed a hundred.

Aside from the standardised protocols in place for hospitals, there are new methods that have attempted to incorporate existing technologies to reduce the number of retained gauze.

One of the methods employed is the usage of a handheld wand-scanning device that is capable of detecting RFID tags sewn on surgical gauze. The device uses proximity sensing to determine the location of any of the surgical gauze in the patient's body, reducing time and inconvenience of manually searching in the case of count discrepancy. An initial clinical evaluation conducted in Stanford University Medical Center for 8 patients undergoing abdominal or pelvic procedures, showed an accuracy of 100 percent for detection. However, the study concluded that there still remains a possibility of human error if the handheld scanning on the patient is incorrectly performed [8].

Furthermore, the estimated cost of the handheld wand-scanning device is USD $ 400.00 [9]. Each RFID-sewn surgical gauze cost USD $ 0.50, as compared to the cost of a normal surgical gauze which is as little as USD $ 0.01 to USD $ 0.10 per piece. As such, this RFID system would significantly increase the costs of each operation for the hospital.

Therefore, there is a need for a system that helps to track the number of surgical gauze as well as display the number of surgical gauze in real-time for effective communication. With real-time display of the number of surgical gauze dispensed and returned, early detection of any unaccounted surgical gauze would allow the medical team to immediately search for the surgical gauze rather than during the wound closure and reversal of anaesthesia. This would also improve the information flow within the medical team in the case of any change in personnel. Lastly, in the long run, this system would potentially be more cost-effective as compared to the RFID tags sewn on surgical gauze.

1.3 Objectives

The objective of this project is to conduct a preliminary study on the feasibility of using Artificial Intelligence to train a computer vision model. This model will be equipped with the capability to detect and account for the number of surgical gauze in real-time during the surgical procedure.

1.4 Scope of Work

This project will investigate and compare the different techniques of computer vision to detect and keep a count of the number of 10cm x 10cm surgical gauze during a live video feed. There are various possible computer vision techniques available, such as object detection, semantic segmentation, and image segmentation. As such, a comprehensive review will be done to determine the technique best suited for use in this project.

After establishing the model used, image augmentation schemes will also be applied to real images in order to increase the number of viable training samples. The full dataset of images will then be provided to the model for training. Following which, the model will perform predictions based on real video footage with an overlay on top of the video feed showing the number of surgical gauze. Finally, object tracking will be implemented in order to track each gauze to keep count of the total number of gauze.

The performance of the computer vision model will be evaluated based on the precision and accuracy of the number of surgical gauze during prediction. Further optimization of the model will be done after the evaluation of its accuracy.

Chapter 2

Literature Review

2.1 Deep Learning

As one of the many applications of Artificial Intelligence, Machine Learning (ML) equips machines with the ability to automatically learn and improve based on its training through pattern recognition with minimal human involvement [10, 11]. With the rapid transition into automation in the recent years, the trend of integrating ML into various sectors has undeniably transformed the economy [12]. In fact, under the umbrella of ML, deep learning is one of the methods widely used. Inspired by the structure of the human brain, the architecture behind deep learning is based upon artificial neural networks [13].

2.1.1 Convolutional Neural Networks

Out of the numerous architectures of deep learning, Convolutional Neural Networks (CNNs) is undoubtedly one of the most well-known and effective method for deep learning. Since late 1980s, CNNs have already been applied to visual tasks. However, CNNs were not popular until after 2000s when large technological advancements were made in terms of computing power and algorithms [14]. Generally, CNNs are employed to analyze visual imagery in the field of computer vision.

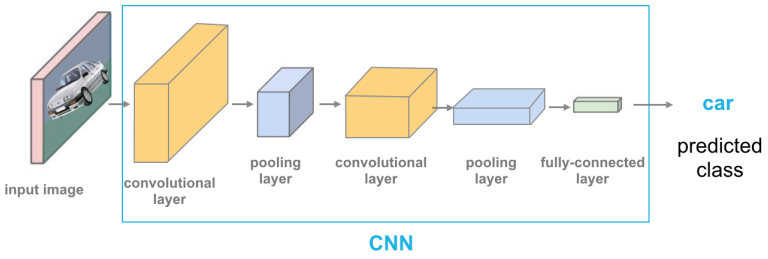


Figure 2.1: Example of Convolutional Neural Network [15]

As seen in Figure 2.1 as an example, they mainly consist of 3 types of hidden layers, namely:

1. The convolutional layer, the core building block of CNNs, contains a set of filters that would be used to create a feature map by detecting and extracting crucial information from input data.

2. The pooling layer, always following the convolutional layer, captures the local maximum or average of different regions of the feature map, resulting in subsampling of input data. As such, this allows semantically similar features to merge, reducing dimensionality while increasing scale and supporting translation invariance [16].

3. The Fully Connected (FC) layer is at the end of the network, after multiple rounds of stacks that alternate between convolutional layer and pooling layer to create a deep network for better extraction of features representations. Taking in the output of the previous layer, the fully connected layer would classify the output by converting the 2D feature maps from the output into 1D feature vector [13].

The values that are used for the layers are derived from the training process that CNNs undergo. One of the popular learning algorithm for training CNNs is called the Backpropagation, which is short for "Backward propagation of errors". Through this supervised learning algorithm, CNNs are able to adjust their parameters, weights, network biases, in order to achieve the desired network output [14, 17]. During the training process, given the training input parameters, the Backpropagation algorithm involves many iterations of the whole step by step process as stated below:

1. Forward Propagation: Based on the user's input training, the network will make a prediction about the input training data set provided as the output of the forward propagation.

2. Backward Propagation: For any wrong guesses, the error will be backpropagated to adjust any network incorrect parameters such as the values of weights and biases in the direction that has lesser error. This can be achieved by using the loss function as well as the 'Gradient Descent' method for calculation of the "gradient" between the prediction and the expected outputs[18]. The 'Gradient Descent' method is commonly used to define the distinction between 2 values during the training process of neural networks [19, 20].

3. Re-calculation of weight value: This step aims to update the values of the weights before the reiteration of this whole process.

The iteration of this step by step process will proceed with the different given input samples until there are sufficient iterations that is specified by the user input parameters at the start of the training. However, the number of iterations is not directly proportional to accuracy in prediction. With a large number of iterations, this might result in overfitting where the trained network model is biased towards the training input data [21]. Essentially, this means that the trained network model generalize poorly to unseen test data where it is memorising the training data rather than analyzing it.

In order to prevent the issue of overfitting, one method is to stop the training early [22]. On top of that, the technique 'Cross validation' is introduced so as to analyze and assess the performance of the trained network. This technique involves the reservation of a part of the input dataset that will not be used in training. This reserved dataset named "Validation dataset" will be used after training to determine the effectiveness of the trained model's performance [23]. As the performance of the validation dataset improves over time while the network is being trained, we select the model that has the best performance as according to evaluation metrics before the occurrence of overfitting [22].

2.1.2 Object Detection

Object detection is a computer vision technique that aims to identify, locate, label objects in a given image, video or webcam feed. This is achieved by predicting an axis aligned bounding box to represent the position and scale of the object. In general, the frameworks of object detection are categorized into two types, region proposal based and regression-based methods. For methods that are the region proposal based, they include Region with CNN features (R-CNN) [24], Fast R-CNN [25], Faster R-CNN [26] and Mask R-CNN [27]. For methods that uses the regression-based framework, they include You Only Look Once (YOLO) [28], Single Shot Multibox Detector (SSD)[29].

The region proposal based framework is a two-step process where it scans the scenario given, and generates region proposals by focusing on the regions of interest. By inserting CNN into this framework, CNN is firstly used to extract high-level feature representations from the generation of multiple region proposals per image. Following which, these region proposals would be assigned a score to be used for the adjustment of bounding box regression. Finally, it would be filtered to produce the prediction of the final bounding boxes. As such, R-CNN is able to predict object bounding boxes with significantly higher quality after using CNN for extraction of high-level features [30].

Whereas the regression based framework is a one-step process where it directly maps the bounding box coordinates and class probabilities to the image pixels. As such, this one-step process reduces the amount of time required for training. Comparing these two frameworks, R-CNN produces better results. However, R-CNN require a longer duration for training and detection resulting in it being more costly.

To solve this problem, the method of using Region Proposal Network (RPNs) was introduced by Ren *et al.* in Faster R-CNN [26]. RPNs introduces a change in algorithm to the region proposal step in R-CNN that reduces the cost of proposal computation to nearly zero. It is implemented by sharing the computation of convolutional feature maps of the image with the CNN module. Hence, this change drastically increases the

efficiency for training as well as detection [30].

2.1.3 Image Segmentation

Combined with object detection, image segmentation is another technique of computer vision. This process is used to output a pixel-wise mask for each detected object in the image by grouping pixels based on their attributes. Under image segmentation, there are 2 types of processes namely, semantic segmentation and instance segmentation.

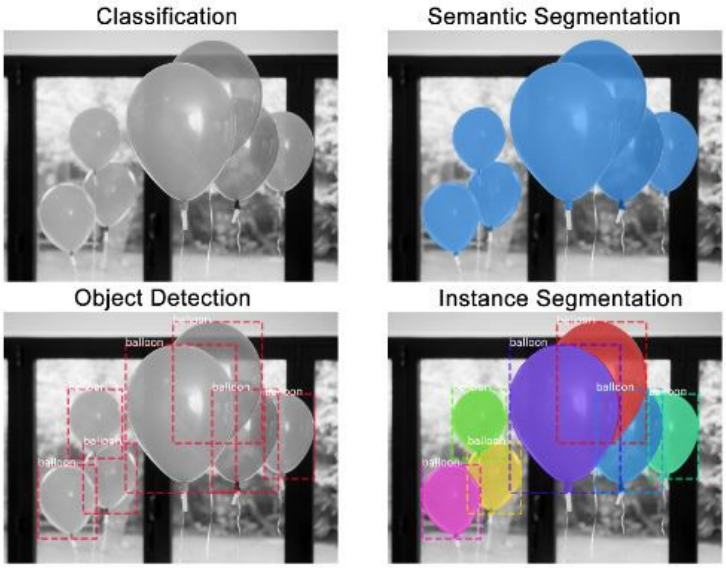


Figure 2.2: Computer Vision Techniques [31]

Semantic segmentation is a basic level of image segmentation where pixels are grouped under a class label. As seen in Figure 2.2, the balloons are grouped as one class label shown by the color. However, instance segmentation is capable of differentiating individual instance of a class label. As seen in Figure 2.2, despite being grouped under the same class label, each balloon is considered as a separate entity. As compared to semantic segmentation, instance segmentation allows for the accounting of the number of instances of a class label, which is a crucial feature in this project.

2.2 Image Data Augmentation

In order to produce a skilful model that is able to accurately detect objects, a large amount of data is required. Training a neural network on a small dataset would commonly result in overfitting where the trained network model is biased towards the training input data, which is a poor generalization of unseen data. Furthermore, as every new image would require annotation of the objects, it is labour intensive and time consuming to obtain the large amount of data required. One of the methods to overcome this issue is the usage of image data augmentation. This method aims to artificially create a large variety of data through augmentation based off existing training data. This is achieved by image manipulation through different transformations such as shifts and flips.

Image data augmentation has been proven effective for improving performance of the neural network as well as its ability to generalize [32, 33]. Furthermore, research has also shown that traditional transformations (rotating, flipping or cropping) are much more successful data augmentations as compared to the more modern transformations (adding noise or shearing of the image). This could be attributed to simple traditional transformations containing clearer defining features of the object that are more similar to actual real training images [33].

2.3 Object Tracking

As one of the most important components in computer vision applications, object tracking has been a challenging topic that involves active ongoing research for a long time due to the increasing demand to automate video analysis [34].

Object tracking refers to tracking an object given its location in its first frame [35]. In most object tracking algorithms, there are 3 main steps. The first step aims to detect the moving object of interest, which can be automated by using CNNs. This is achieved by making use of bounding boxes that the CNN produces that is made up of connecting the coordinates of 4 edges around the object detected. Secondly, to track the object of

interest from frame to frame of the video. Finally, to analyse the object tracks in order to recognize their behavior [36].

With these 3 steps, there are numerous applications of object tracking in tasks such as any motion-based recognition tasks. However, object tracking has remained as a challenging avenue of research due to various difficulties such as sudden object motion or any change in appearance in terms of the scene or the object itself, camera motion and the list goes on [36]. Despite the difficulties faced, there are a few state-of-the-art (SOTA) models in place such as Simple Online and Real time Tracking (SORT), Fast Online Object Tracking and Segmentation which is also known as SiamMask, and so on.

Chapter 3

Methodology

3.1 Masked R-CNN Instance Segmentation

The object detection framework used in this case is one of the most state-of-the-art (SOTA) computer vision frameworks for instance segmentation - Mask R-CNN [27]. As seen in Figure 3.1 below, Mask R-CNN is ranked 1st for having the highest mask Average Precision (AP) under the category of instance segmentation. In this project, the implementation of Mask R-CNN by Matterport's *maskrcnn* repository [37] will be used.

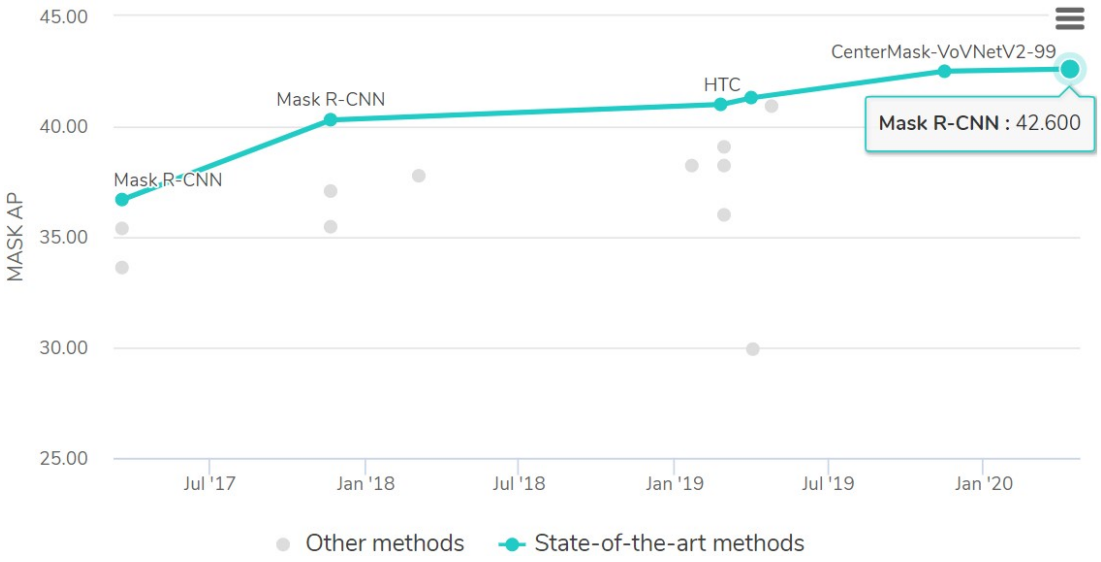


Figure 3.1: State-of-the-art (SOTA) frameworks for Instance Segmentation [38]

This implementation of Mask R-CNN is done on Python 3, Keras and TensorFlow, while the repository provides suitable training tools as well as models for evaluation by using different metrics.

To understand more about Mask R-CNN, the network architecture of Mask R-CNN can be split into 2 parts namely, the backbone architecture and the network head. The network head comprises of object detection as well as the segmentation parts. In the case of Mask R-CNN, the network head uses almost the same architecture as Faster R-CNN, with an extension that predicts a mask for each object detected. Whereas, backbone architecture is mainly used for feature extraction, which Mask R-CNN utilises the Feature Pyramid Network and ResNet-101 as its backbone architecture.

For this project, accuracy will be prioritised over training, detection and computation time as the aim is to produce high accuracy of the number of surgical gauze. As such, Mask R-CNN is preferred over the other methods such as YOLO which is faster but less accurate.

3.1.1 Understanding Feature Pyramid Network (FPN)

Traditionally for object detection, there are several ways to achieve detection of a specified object, such as having the feature extractors use a pyramid of the same image at multiple scales as seen on the left in Figure 3.3, or simply creating a pyramid of feature maps as seen on the right in Figure 3.2.

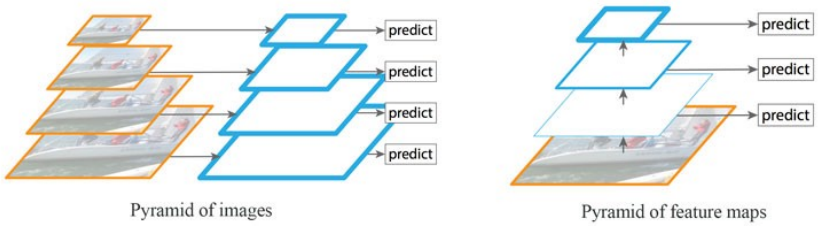


Figure 3.2: Previous Common Feature Pyramids [39]

However, FPN replaces this traditional method, and generates multiple feature map layers instead [39]. FPN is essentially an extractor for features that is designed with the concept of pyramids as seen in Figure 3.3.

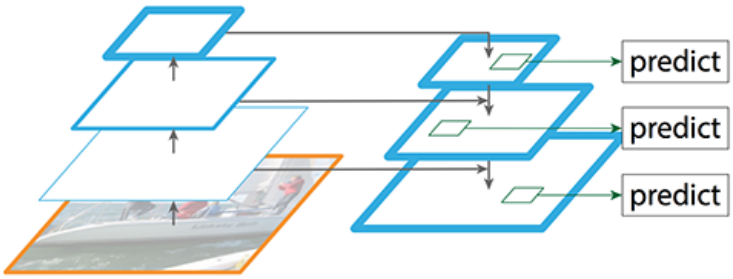


Figure 3.3: Feature Pyramids Network (FPN) [39]

The crucial difference in FPN is that FPN comprises of a bottom-up as well as a top-down pathway. As compared to traditional methods in CNNs, their feature extraction only focuses on the bottom-up pathway. As a result, it only considers the top layers with high semantic value while the bottom layers with higher resolution are ignored for object detection [39]. As such, consisting of both types of pathways, FPN is able to construct layers with high resolution and high semantic value. Although this method reduces precision, a solution was quickly found. The solution was to add a lateral connection between the feature maps (on the left in Figure 3.3) and the corresponding reconstructed layers (on the right in Figure 3.3). As such, FPN is able to produce better multiple feature map layers that contains information of higher quality as seen in Figure 3.4 where FPN based Faster R-CNN has the highest AP.

Faster R-CNN	proposals	feature	head	lateral?	top-down?	AP@0.5	AP	AP_s	AP_m	AP_l
(*) baseline from He *et al.* [16]†	RPN, C_4	C_4	conv5			47.3	26.3	-	-	-
(a) baseline on conv4	RPN, C_4	C_4	conv5			53.1	31.6	13.2	35.6	**47.1**
(b) baseline on conv5	RPN, C_5	C_5	2*fc*			51.7	28.0	9.6	31.9	43.1
(c) **FPN**	RPN, $\{P_k\}$	$\{P_k\}$	2*fc*	✓	✓	**56.9**	**33.9**	**17.8**	**37.7**	45.8

Figure 3.4: Faster R-CNN with different feature extractors methods [39]

3.1.2 Understanding ResNet101

ResNet-101 is a deep CNN that consist of 101 layers that was released by Microsoft Research Asia. ResNet was introduced to solve the issue of degradation of the network. As the networks converge with increasing depth, there is saturation of accuracy then it degrades rapidly. In Figure 3.5, it can be seen that ResNet-101 is second to having the lowest rates of error for object detection. Furthermore, ResNet-101 has been pre-trained based on over a million images and 1000 object

classes from the ImageNet Large Scale Visual Recognition Challenge 2012 classification dataset. As such, ResNet-101 contains biases and weights that are already associated with the ImageNet dataset. By training on this pre-trained model, it provides a richer feature representations and learned features that can be transferred to our object detection tasks

method	top-1 err.	top-5 err.
VGG [41] (ILSVRC'14)	-	8.43†
GoogLeNet [44] (ILSVRC'14)	-	7.89
VGG [41] (v5)	24.4	7.1
PReLU-net [13]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

Figure 3.5: Error rates on ImageNet Validation - ResNet ImageNet Results 2015[40]

3.2 Preparation of Dataset

In preparation of the dataset, there are real training images and augmented images to be used. In this project, only Ray-Tec® gauze that are commonly used will be considered in training the neural network for consistency. These Ray-Tec® gauze have a black line that is radio-opaque, meant for scenarios where if any Ray-Tec® gauze are unintentionally left in the patient, an X-ray would be able to detect the missing gauze. Furthermore, this black line would also aid in training of the neural network as it would be considered as a feature representation of the gauze as seen in Figure 3.6 below.

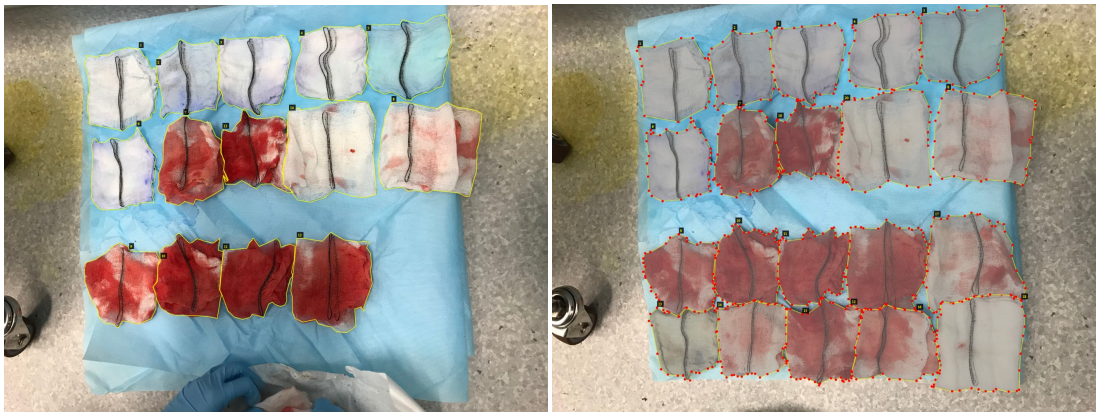
Figure 3.6: Sample image of a set of Ray-Tec® gauze [41]

3.2.1 Annotation

As the dataset is based off fresh photographs, manual annotation for each and every image is required as there are multiple surgical gauze in each image. For this project, the VGG Image Annotator (VIA) introduced by Abhishek Dutta from the University of Oxford will be used [42, 43]. This is a simple, standalone and online manual annotation software for images and videos which does not require any additional installation of software or setup. This software is an open source online repository project that is based only on HTML, Javascript and CSS. After annotation, a Json file containing the coordinates of the points annotated on all the input images, will be exported.

In order to prevent the occurrence of overfitting as mentioned earlier [21], the images will be split into 3 different datasets. These datasets are named - (1) train, (2) validation, and (3) test [44]. The train dataset will contain 60% of the entire dataset, while the validation and test dataset will each contain 20%.

For better accuracy, each gauze in the image will be annotated based on its own unique individual shape. As seen in Figure 3.7, in a single image, each gauze would have yellow lines representing the outline drawn by connecting the red dots. With detailed annotation being done on each gauze per image, the annotation process is very long and time-consuming.



(a) Annotation with basic outlines

(b) Annotation with all gauze' coordinates shown

Figure 3.7: Sample images of VGG Image Annotator (VIA) [42, 43]

In order to reduce amount of time required to annotate each image, a script has been written in order to generate annotations for images. In other words, this script takes in the base annotation coordinates of a base image and predicts the annotations for a set of images that was augmented with the base image. This is achievable as the prediction of the new coordinates can be computed as long as the type of transformation such as flipping and rotating, did not involve distorting or moving the particular image pixel. For other images where the pixels have been distorted or moved, manual annotation will be required.

3.2.2 Real Training Images

The dataset for this project requires images that are one of a kind and are unable to be found online. As such, photos of various sets of surgical gauze will be directly taken by the nurses from Singapore General Hospital (SGH) whom will be assisting in this project. For the consistency and accuracy of the detection of surgical gauze, the photos taken have to satisfy a set of conditions:

1. The photos must consist of a top-down view of the set of surgical gauze. This would provide a clear bird's eye view of all the gauze for the neural network to be trained, minimizing any error due to angle of the photo.

2. The view of any of the surgical gauze must not be obstructed by any item.

3. The photos must be from different surgeries to provide a large variety of conditions. Different surgeries would require different medical solutions and different bloodstains found per gauze, resulting in gauze that would differ in colors and shape. Furthermore, this would prevent the trained network from over-generalizing surgical gauze from a single surgery, ultimately increasing accuracy.

4. Each surgical gauze must be placed separately from one another. This would aid in instance segmentation where the trained network aims to separate and produce a mask for each instance of a detected surgical gauze.

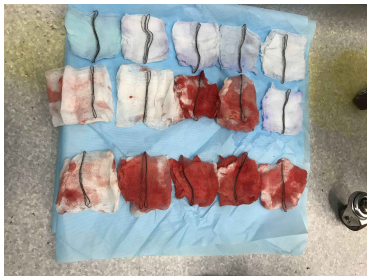
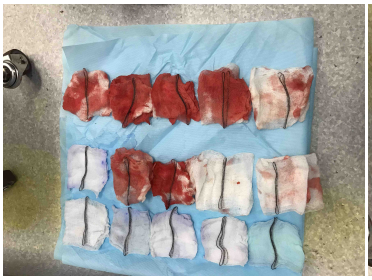
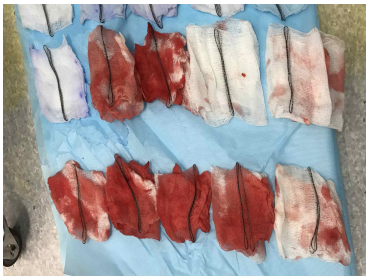
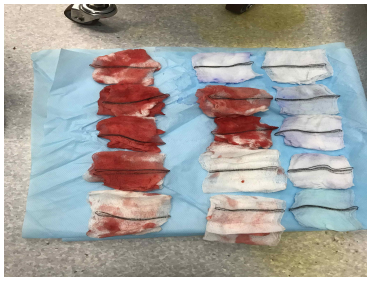
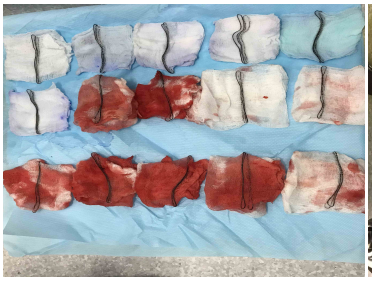
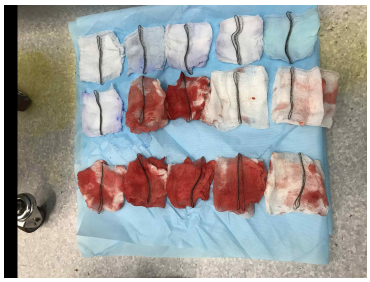
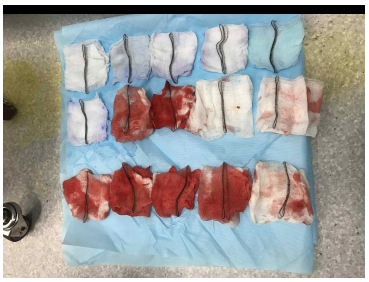
3.2.3 Image Augmentation

In order to train a neural network with high accuracy, a large dataset would normally be required. However, due to a tight timeline for this project, there will be a limited number of photos available. As such, for this project, images will be required to undergo augmentation in order to expand the current dataset. The image augmentation library that will be implemented is Imgaug [45], a well-known and stable python library that focuses on allowing batches of images to undergo over 60 types of image augmentation. For this project, the images will be passing through an augmentation scheme consisting of random transformations as shown in the following section.

Traditional Augmentation

The traditional transformations used in the random augmentation scheme are as shown.

Table 3.1: Sample methods of Traditional Augmentation

Geometric Transformations		
		
Flip Horizontally	Flip Vertically	Perspective Transformation
		
Rotate	Scale X	Scale Y
		
Translate X	Translate Y	

Modern Augmentation

The modern transformations used in the random augmentation scheme are as shown below.

Table 3.2: Sample methods of Modern Augmentation Part I

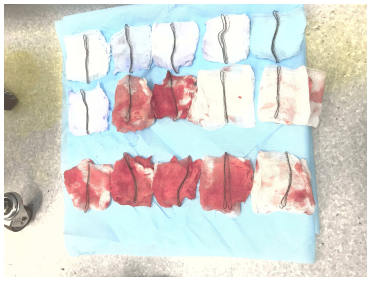
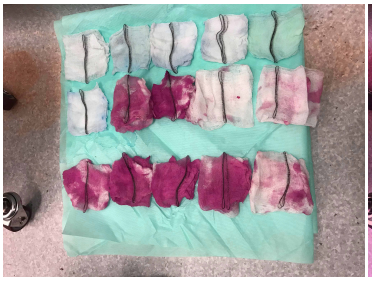
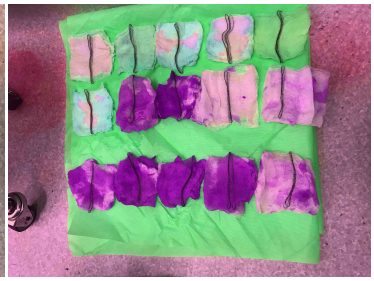
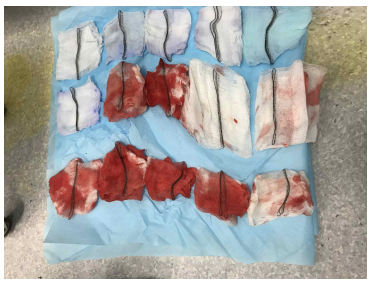
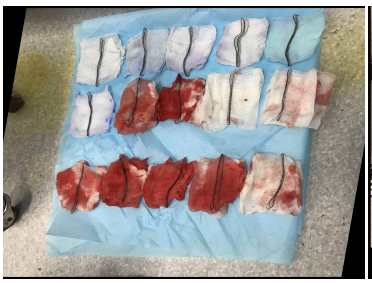
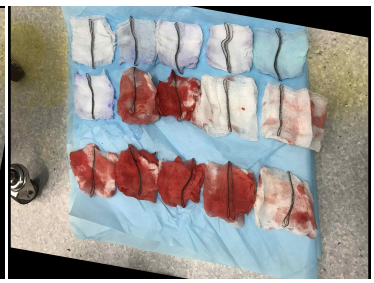
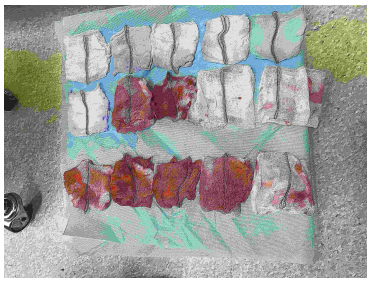
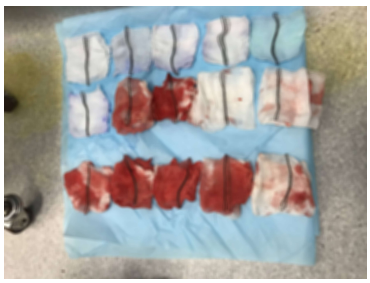
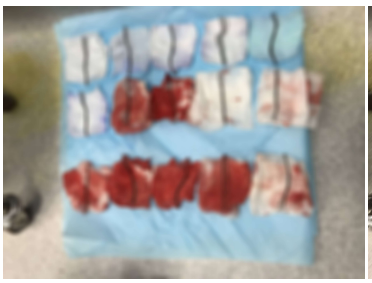
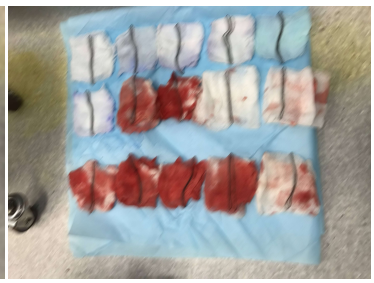
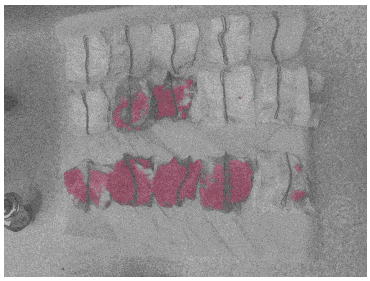
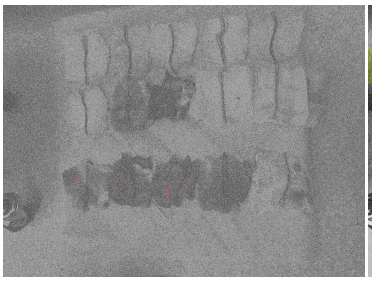
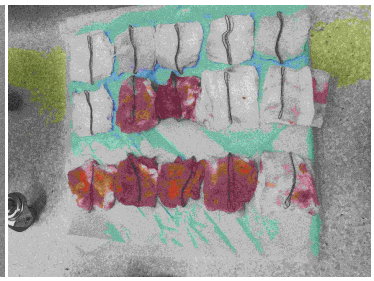
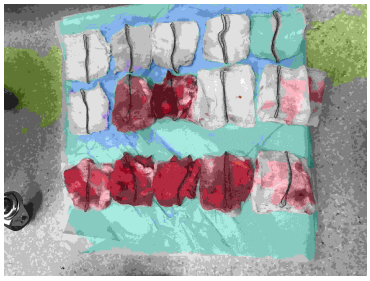
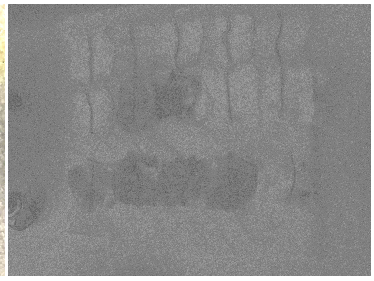
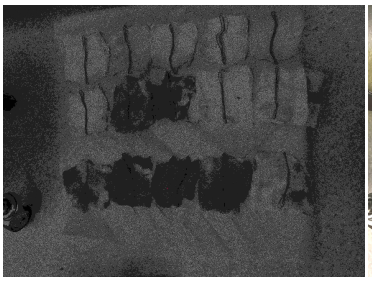
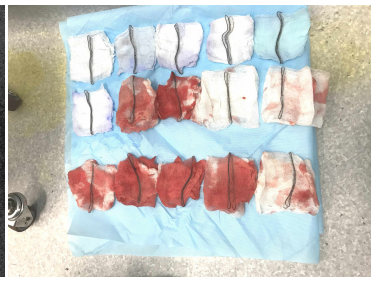
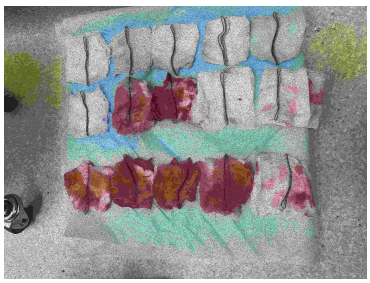
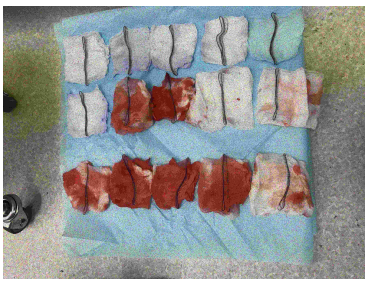
Colour and Contrast Transformations		
		
Brightness	Colorspace	Add hue and saturation
Geometric Transformations		
		
Piecewise Affine	Shear X	Shear Y
Other Transformations		
		
Emboss	Pooling	Superpixel
Blurring Transformations		
		
Average blur	Gaussian blur	Motion blur

Table 3.3: Sample methods of Modern Augmentation Part II

Arithmetic Transformations		
		
Additive Gaussian noise	Additive Laplace noise	Poisson noise
		
Jpeg Compression	Multiply	Salt and pepper noise
		
Dropout channel = False	Dropout channel = True	Add
		
Coarse dropout channel = False	Coarse dropout channel =True	

3.3 Methods of Evaluation of Trained Network

For object detectors such as Mask R-CNN and Faster R-CNN, Average Precision (AP) is a popular metric used to measure and benchmark accuracy. AP are also sometimes known as Mean Average Precision (mAP), these two are usually used interchangeably. However, for consistency, in this project, we will only mention AP. The general mathematical definition for AP is the area under the Precision-Recall (PR) Curve.

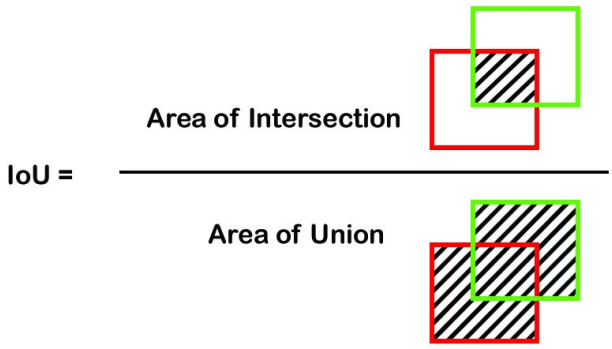
$$\mathrm{AP} = \int_0^1 p(r)dr$$

3.3.1 Intersection Over Union

In order to compute AP, precision, recall and Intersection Over Union (IoU) values are required. Firstly, to calculate IoU, there are a few definitions to be understood as follows:

- True Positives (TP): The outcome with correct identification of gauze.
- False Positives (FP): The outcome with incorrect identification of gauze.
- True Negatives (TN): The outcome where no gauze were correctly identified.
- False Negatives (FN): The outcome where no gauze were incorrectly identified.

IoU is commonly used to determine if the predicted box is TP, FP or FN. Essentially, IoU is the ratio of the area of intersection and union of the actual bounding boxes based on coordinates given in training data and the bounding boxes predicted by the neural network.



Based on commonly used predefined thresholds for IoU of 0.5, this can be used in determining TP, FP, FN as follows:

- If IoU > threshold: 0.5, and object is correctly classified, then it is TP.
- If IoU > threshold: 0.5, but object is wrongly classified, it will be a FN.
- If IoU < threshold: 0.5, it is a FP

3.3.2 Average Precision, Precision and Recall

Using the definitions above, precision is a measurement of accuracy of predictions done by the neural network. Precision is usually in terms of the percentage of the correct predictions, in other words, the positive predicted value. As seen below, it is mathematically defined by TP over the total of TP and FP which is the total positive result. Whereas, recall is a measurement of ability of the neural network to identify all relevant cases which is the true positive rate. As seen below, it is mathematically defined by TP over the total of TP and FN.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

As the range of precision and recall values are always within 0 and 1, AP is expected to fall within the same range. Before getting the area under the PR curve, the interpolated precision must be calculated as see in the formula below where for a given recall value (r), the p(r)~ is the measured precision at recall r~.

$$p_{interp}(r) = \max_{\tilde{r} \geq r} p(\tilde{r})$$

Finally, by taking the area under the PR curve and segmenting the recalls evenly into 11 parts, AP can be calculated as seen in the new formula below.

$$\begin{aligned} AP &= \frac{1}{11} \sum_{r \in \{0.0,\ldots,1.0\}} AP_r \\ &= \frac{1}{11} \sum_{r \in \{0.0,\ldots,1.0\}} p_{interp}(r) \end{aligned}$$

3.4 Object Tracking

Given a video feed, for each frame, Mask R-CNN will be able to detect and distinct each object. However, Mask R-CNN is unable to track an object from frame to frame. As such, the Simple Online and Real-time Tracking (SORT) algorithm introduced during the 2016 IEEE International Conference on Image Processing (ICIP), will be used [46]. SORT was ranked as the best open source Multiple Object Tracker (MOT) based on the MOT benchmark at time of publication in 2016. SORT is a minimalistic tracker that aims to create unique identities for multiple objects during initial detections, and tracking these objects across frames in videos while maintaining the identity assignment. As SORT does not include a neural network, its accuracy is mainly attributed to detections done by a trained neural network.

Tracking the object can be achieved as the SORT algorithm estimates multiple object models for each frame. Each object model consists of spatial information in regards to the object's position, scale, as well as the ratio of the bounding box. With the

information available, the object models are able to predict the object's motion for the incoming frame by using the unscented Kalman filter for estimation.

The Kalman filter is an algorithm that is commonly used for linear systems with continuous changes [47]. It is essentially a iterative mathematical process which is also known as a linear quadratic estimation, which is used to estimate unknown variables given information measured or observed over time. The core idea of using the Kalman filter is aimed at using the current available detections along with predictions previously done, to achieve the best guess available while accounting for any possible errors.

Currently, there are only online implementations of SORT on YOLO version models [48], with no such tutorial or open source repositories where SORT is implemented on Mask R-CNN. However, for this project, implementation of SORT on Mask R-CNN will proceed as Mask R-CNN would be more ideal for better accuracy of detecting the surgical gauze.

Chapter 4

Results and Discussion

4.1 Evaluation of Mask R-CNN model

For this project, in order to gauge and get the best performance of the Mask R-CNN model, the model was trained a couple of times separately with changes made to the number of epochs and steps per epoch, given a standard batch size. The dataset consisted of 101 real images, 303 images with traditional augmentation and 2,020 images with modern augmentations. Before explaining the results, a few definitions are required for better understanding:

- Each epoch is defined by the entire dataset undergoing backpropagation through the neural network once. The number of epochs is crucial to fine tuning the neural network for higher accuracy as using a single epoch would lead to underfitting while too many epochs would lead to overfitting.
- Batch size is the total number of training images available per batch.
- Steps per epoch refers to the number of batches required for a single epoch's completion.

4.1.1 Average Precision, Precision, Recall

Comparison of the various metrics namely, AP, precision as well as recall, have been commonly used to for performance evaluation on the experiment to gauge the optimal number of epoch and steps per epoch. As seen in Figure 4.1, training Mask R-CNN with 60 epochs along with 60 steps per epoch revealed the highest AP of 0.98, given a dataset with real images with all augmentations.

Table 4.1: Metrics Evaluation for Experimental results

Mask R-CNN Detector Training			
Real Images + Images with Traditional Augmentation			
	Metrics		
	AP	Precision	Recall
30 Epochs | 30 Steps per Epoch	0.94	0.90	0.57
30 Epochs | 60 Steps per Epoch	0.93	0.92	0.56
60 Epochs | 30 Steps per Epoch	0.96	0.93	0.57
60 Epochs | 60 Steps per Epoch	**0.96**	**0.92**	**0.57**
Real Images + Images with All Augmentation			
	Metrics		
	AP	Precision	Recall
30 Epochs | 30 Steps per Epoch	0.97	0.92	0.57
30 Epochs | 60 Steps per Epoch	0.97	0.90	0.54
60 Epochs | 30 Steps per Epoch	0.97	0.93	0.56
60 Epochs | 60 Steps per Epoch	**0.98**	**0.98**	**0.57**

This shows that the Mask R-CNN trained model has decent performance. Furthermore, it can be noted that the data with the all augmentation performed better as seen in the metrics where the mean AP for all image augmentation is 0.9750 while the mean AP for traditional augmentation is 0.9475. Although the difference is minimal nearly negligible, the image augmentation is still crucial to expand small dataset to avoid underfitting. However, this proves a point that traditional image augmentation is mostly effective as the image augmentation that included the modern methods only increased the mean AP by a 0.0275.

4.1.2 Loss and Validation Loss

In order to find out if overfitting or underfitting has occurred, other metrics such as loss and validation loss must be taken into consideration.

As seen below, loss refers to the total of classification loss, bounding box regression loss and mask loss. Classification loss refers to the confidence of the model to predict the correct class. Whereas, bounding box regression loss refers to the distance between the actual box information and the predicted box information. In other words, the bounding box regression reflects the confidence of the model to correctly locate the object. The mask loss essentially penalize wrong per-pixel binary classifications.

$$Loss = Classification\ Loss + Bounding\ Box\ Regression\ Loss + Mask\ Loss$$

In the Mask R-CNN implementation repository, there are tools provided to calculate loss. In fact, loss can be calculated by computing the loss of these "small" losses:

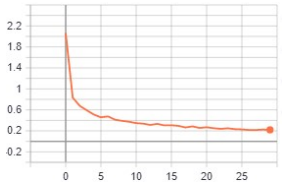
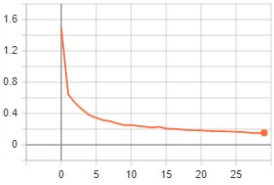
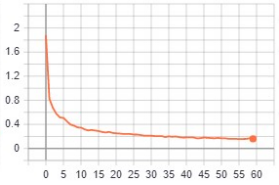
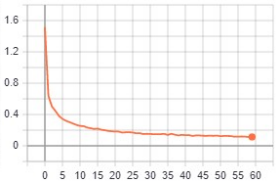
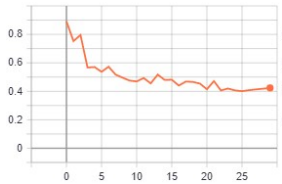
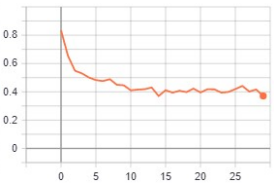
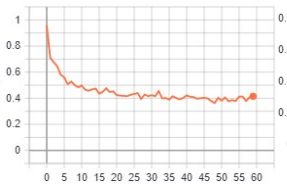
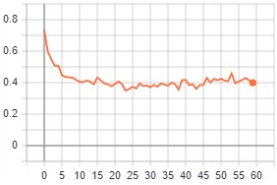
- RPN Classification loss: Determines how well RPN is able to separate the background with objects.
- RPN Bounding Box loss: Determines how well RPN is able to localize objects.
- Mask R-CNN Classification loss: Determines how well Mask R-CNN is able to classify the objects according to their class labels.
- Mask R-CNN Bounding Box loss: Determines how well Mask R-CNN is able to localize objects.
- Mask R-CNN Mask loss: Determines how well Mask R-CNN is able to segment objects.

Each of the 12 graphs below consist of each type of loss that are plotted against each epoch, as they are computed at the end of every epoch. For all 4 of the experiments with different epoch and different number of steps per epoch, it can also be seen that

the trend for all the graphs show that they have reached its minimum. This shows that the losses are at its minimal and that there are no problem of overfitting for any of the 4 experiments.

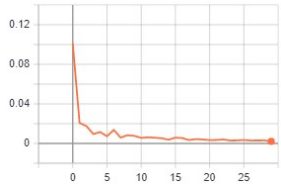
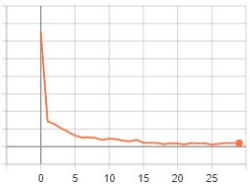
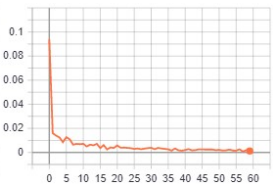
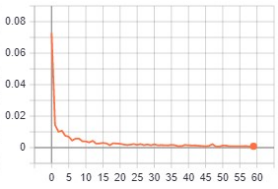
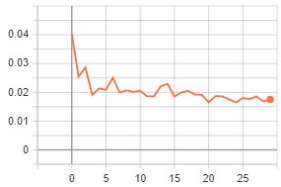
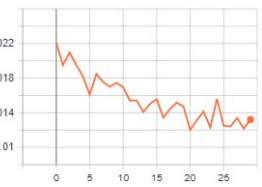
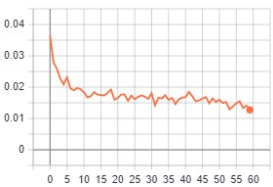
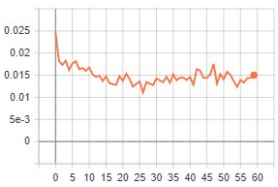
For all 4 cases, the loss are rather low with an approximation of 0.15. However, for validation loss, it can be seen that the value ranges between 0.3 to 0.45 which is a little high but it can definitely be lowered with a larger dataset. As loss and validation loss are made up of the total of the other types of losses, the data can be found below for better visualisation.

Table 4.2: Losses Evaluation Part 1

Losses			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch
Validation Losses			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch

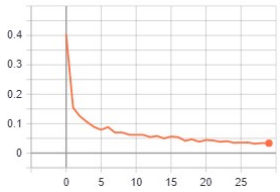
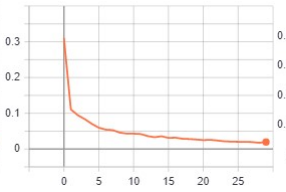
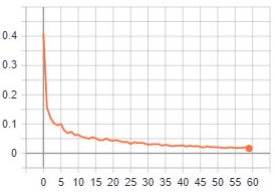
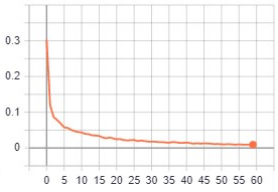
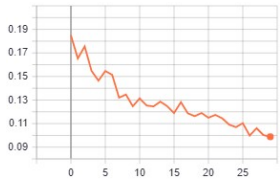
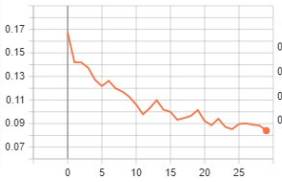
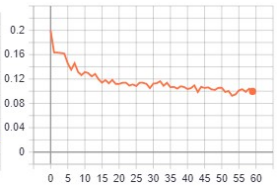
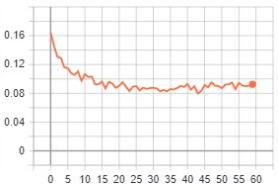
Generally, for RPN classification loss, the normal loss and the validation loss are decently low. As RPN classification loss is nearly zero while the validation RPN classification loss is also quite low below 0.02. This shows that RPN is doing well with differentiating between the background and the object.

Table 4.3: Losses Evaluation Part 2

RPN Classification loss			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch
Validation RPN Classification Losses			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch

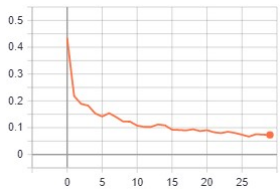
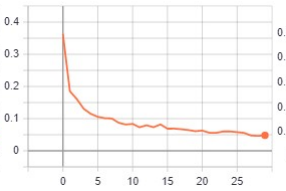
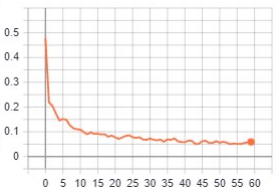
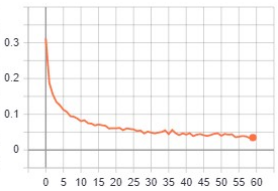
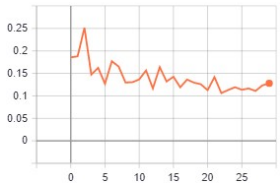
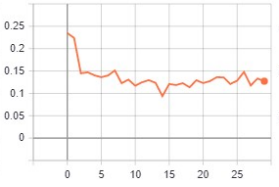
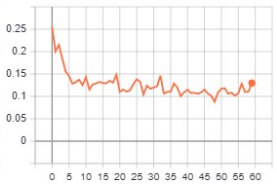
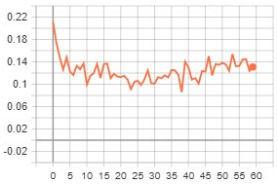
As for RPN bounding box loss, the normal loss and validation loss are also low. As RPN bounding box loss is nearly zero while the validation RPN bounding box is generally low below 0.012. However, it seems like for validation loss, the cases with 30 epochs are doing slightly better with a range between 0.09 and 0.10 but the cases with 60 epochs have a range between 0.11 and 0.12. This shows that RPN managed to localize the objects better with 30 epochs than 60 epochs.

Table 4.4: Losses Evaluation Part 4

RPN Bounding Box loss			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch
Validation RPN Bounding Box Losses			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch

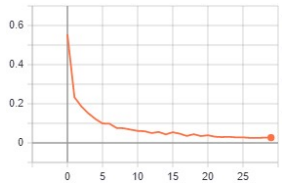
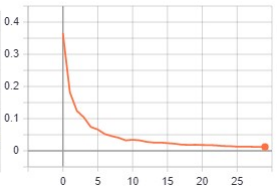
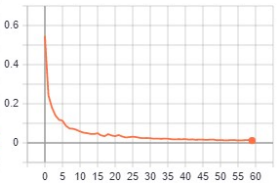
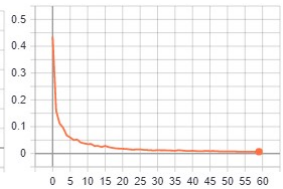
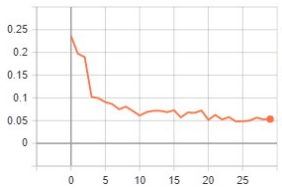
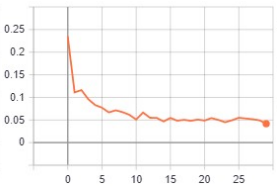
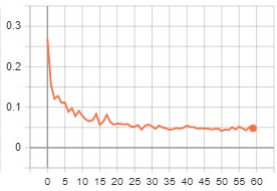
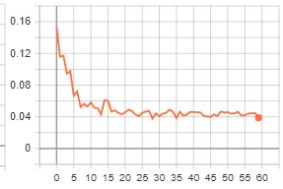
For Mask R-CNN classification loss, it can be seen that the cases with 60 epochs have lower loss and validation loss than cases with 30 epochs. The cases of 60 epochs have an approximate loss of 0.04 but the cases with 30 epochs have a loss that is approximately above 0.05. This shows that Mask R-CNN is better at classifying objects according to the class label as the number of epochs increase.

Table 4.5: Losses Evaluation Part 5

Mask R-CNN Classification loss			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch
Validation Mask R-CNN Classification Losses			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch

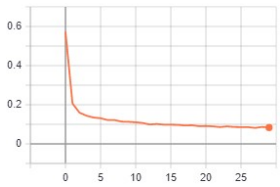
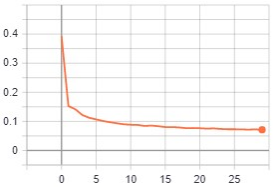
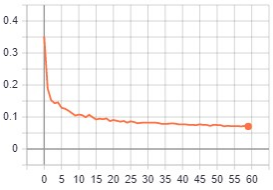
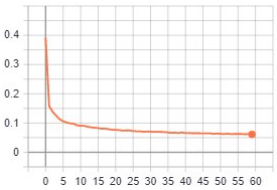
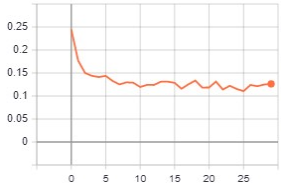
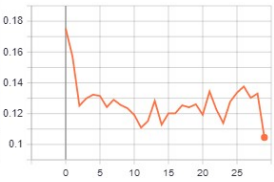
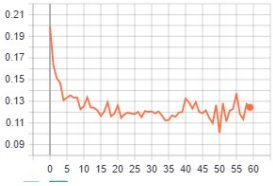
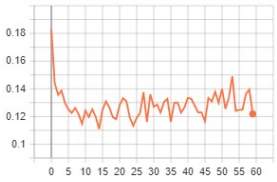
As for Mask R-CNN bounding box loss, the normal loss and validation loss are very low. As Mask R-CNN bounding box loss is nearly zero with the increasing epochs, while the validation Mask R-CNN bounding box loss also have the same trend with increasing epoch as the value tends to 0.04. This shows that Mask R-CNN managed to localize the objects better with 60 epochs than 30 epochs.

Table 4.6: Losses Evaluation Part 6

Mask R-CNN Bounding Box loss			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch
Validation Mask R-CNN Bounding Box Losses			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch

As for Mask R-CNN mask loss, the normal loss and validation loss are very low. Mask R-CNN mask loss is approximately 0.05 with the increasing epochs. However, the validation Mask R-CNN mask loss seems to be doing same among the 4 cases as the value is approximately 0.12 despite the spikes. This shows that overall, Mask R-CNN did very well in terms of segmenting the images.

Table 4.7: Losses Evaluation Part 8

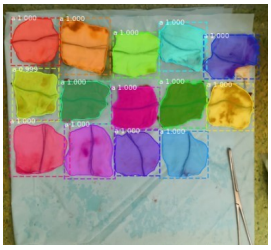
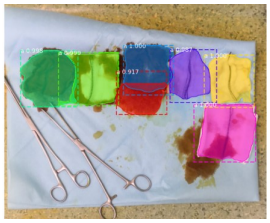
Mask R-CNN Mask loss			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch
Validation Mask R-CNN Mask Losses			
			
30 Epoch 30 steps / Epoch	30 Epoch 60 steps / Epoch	60 Epoch 30 steps / Epoch	60 Epoch 60 steps / Epoch

4.1.3 Detection

Generally, the Mask R-CNN performance in regards to detection is quite good. During detection of the gauze, the confidence level is quite high as seen in the examples below as most of the gauze could be detected.

As seen in Table 4.8, the top row of 4 examples shows the successful good detections using Mask R-CNN trained model. In the second row, the examples depicts the problematic detections which will be further elaborated below.

Table 4.8: Examples of Detection Results

Examples of Good Detections			
			
Example 1(a)	Example 1(b)	Example 1(c)	Example 1(d)
Examples of Problematic Detections			
			
Example 2(a)	Example 2(b)	Example 2(c)	Example 2(d)

However, in Example 2(a), Example 2(b) and Example 2(c) of Table 4.8, it can be seen that currently the model is unable to differentiate among the usual surgical gauze of 10cm x 10cm, the abdominal swab (bottom left, circled in red) and the cotton wool (top left, first five circled in red), a cup used to hold liquid (bottom center, circled in red). Finally in Example 2(d), it can be seen that the patches stained by the gauze previously. This shows that the model is still slightly inaccurate.

4.1.4 Runtime Analysis

For this project, it is required that this implementation of Mask R-CNN in combination of SORT object tracker to be able to run during the surgery. As mentioned earlier, the Mask R-CNN object detector is well-known for its accuracy but not its time required for training the model or detection of the objects given the video feed.

In regards to the amount of time required for training, it will definitely increase as the number of epoch, the number of steps per epoch, and total number of images increase. The training and detection of the Mask R-CNN object detector is run on the Graphics Processing Unit (GPU), GeForce RTX 2060. This GPU is currently ranked as one

of the top 30 GPUs in market [49] with the ability to compute 6,832 operations per second. For better benchmark, the GPU that is ranked first, is able to compute 10,571 operations per second.

For a better benchmark, another GPU was used for training and detection, GeForce GTX 1060. This GPU is ranked 76 with an ability to compute 4,991 operations per second. In the Table 4.9 below, the dataset used is the real images included with images with all augmentations. For training time, we can see that the amount of time required to train the Mask R-CNN model of 60 epoch with 60 steps per epoch takes approximately 4 hours for GeForce RTX 2060 which is a high-end GPU. Whereas, for GeForce GTX 1060, it takes approximately 6.6 hours.

Table 4.9: Runtime Detection Evaluation

Comparison of Runtime between 2 GPUs				
	GeForce GTX 1060		GeForce RTX 2060	
	Train	Detection	Train	Detection
30 Epoch | 30 Steps per Epoch	10,138 sec	943 sec	5,646 sec	518 sec
30 Epoch | 60 Steps per Epoch	12,421 sec	937 sec	7,456 sec	511 sec
60 Epoch | 30 Steps per Epoch	16,575 sec	931 sec	10,185 sec	513 sec
60 Epoch | 60 Steps per Epoch	**23953** sec	**933** sec	**14587** sec	**516** sec

Aside from the training time, as seen in the Table 4.9 above, the detection time based on 60 epoch and 60 steps per epoch for GeForce GTX 1060 is 15.5 while the detection time for GeForce RTX 2060 is approximately 8.6 minutes. The amount of time required for detection is still considered quite high. The detection is done on a video received from SGH with a total number of frames of 857.

In order to tackle the issue of high detection time, a recommendation is to skip 5 to 10 frames so as to speed up detection of the frames. This is because typically for 1 second, the number of frames captured can be either 30 or 60 depending on the camera. As such, by skipping a few frames, this would reduce the unnecessary detection for frames that are nearly alike.

However, it is not recommended to skip too many frames as it would affect the trajectory of the gauze moving away which might affect the unique ID given to it.

This is because the SORT object tracker might treat it as a new object as skipping too many frames will result in loss of information that was supposedly to be fed to the tracker for prediction. As such, more investigation work is needed in order to understand how skipping a number of frames will affect the SORT object tracker accuracy.

4.2 Object Tracking Analysis

The implementation of Mask R-CNM in combination of SORT object tracker is successful. Generally, the SORT object tracker produced decent results of tracking the gauze based on Mask R-CNN as its object detector. When a new gauze enters the view of the camera, the SORT object tracker is able to successfully assign an unique ID for that gauze.

4.2.1 Results from Object Tracking

As seen in the table, it shows a few examples where the gauze are correctly assigned IDs across a couple of frames. These examples are taken from a few test videos provided by SGH. These images are extracted from video analysis that only checks every 10 frames. However, it can be seen below that the increment of IDs given is still unstable.

Table 4.10: Examples of Object Tracking Results

Object Tracking Results


However, there are other occasions where the tracker did not work as it should have due to certain circumstances which will be addressed in the sub-section below.

4.2.2 Problem of Obstruction in line of sight

Firstly, as seen in the table below from left to right, they show process of when a hand obstructs the line of sight between the camera and the surgical gauze. This caused the tracker to create a new unique identity for the same gauze as seen in the Example 3(a) to Example 3(d) where the same gauze had 3 different IDs.

In Example 3(a), the gauze circled in red, had the ID of 3. In Example 3(b), the nurse's hand obstructed the direct view of the surgical gauze from the camera. In Example

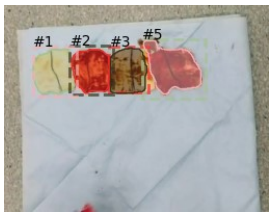
3(c), it resulted in the initial gauze ID to become 12. Finally, in Example 3(d), as the nurse removed his/her hand, the view was obstructed again and the same gauze was assigned another new ID of 18.

Table 4.11: Examples of Problem: Obstruction in line of sight

Set of Images depicting Tracking Issue 1			
			
Example 3(a)	Example 3(b)	Example 3(c)	Example 3(d)

As such, in order to mitigate this problem, the suggestion was made to the SGH nurse. The suggestion was that during the placement of the gauze, the nurse's hand will only enter the view of the camera from the bottom right. This would avoid such a problem from occurring as seen in the table below from left to right. The images shows the process of the identities of the gauze no longer having the problem as mentioned earlier.

Table 4.12: Examples of Solution: Obstruction in line of sight

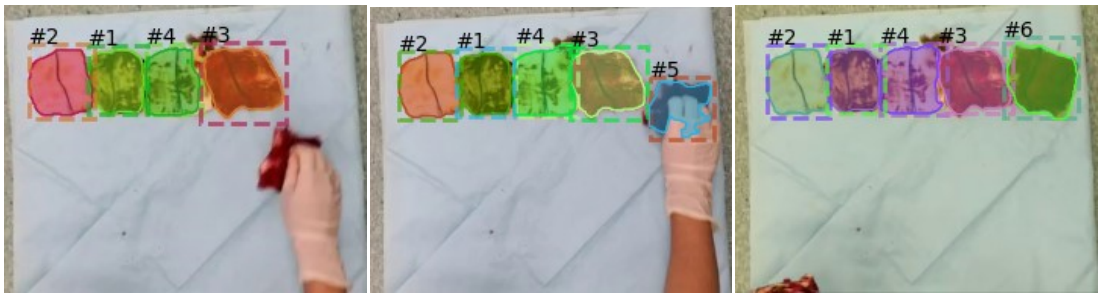
Set of Images depicting Solution for Tracking Issue 1			
			
Example 4(a)	Example 4(b)	Example 4(c)	Example 4(d)

In regards to the main aim of the project which was to tally the total number of surgical gauze, this problem made the object tracker less reliable. As according to the written algorithm, the total number of surgical gauze is determined by the last numbered surgical gauze. This is because mid-way during the surgery, the nurse might remove and pack the gauze into sets of 10, as such, the gauze with a ID will leave the view of the camera. This makes it crucial that each and every gauze has to have an unique ID and that the total number of surgical gauze should be accounted based on the last gauze that entered the view of the camera.

4.2.3 Problem of small dataset

Furthermore, it can be noted that in some frames as seen in the example below, the nurse's gloves are detected as a gauze as well. It could be due to his/her hand grabbing the gauze, along with the white latex gloves with resulted in the object detector identifying it as a gauze. Furthermore, due to the different colors of gauze under different circumstances, the gauze could be tainted with different colors aside from just blood due to the solution used during surgery. In some cases, the black line in the middle are unable to be extracted as a feature due to the dark color of the gauze.

Table 4.13: Example of problem of small dataset

Set of Images depicting Tracking Issue 2


However, this can be solved with a bigger dataset. With an expanded dataset, the object detector will be able to correctly detect the surgical gauze. Alternatively, a suggestion could be for the nurse to wear blue gloves instead of the white ones.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

The main objective of this project was to build a prototype of a machine learning model that is able to detect and track surgical gauze of 10cm x 10cm. In regards to the main objective, a prototype of a Mask R-CNN model with object tracking has been successfully implemented with decent performance. As the performance of SORT is highly dependent on the performance of the object detector, the implementation of Mask R-CNN on SORT was a good choice since the model was able produce good predictions with high precision.

However, it is to be noted that the dataset was considered small as it only consisted a total of 101 real images, 303 images with traditional augmentations, 2,020 images with modern augmentations.

Due to the Coronavirus (COVID-19) situation, the collection of the dataset was affected due to busier schedule at SGH. As such, for the model to produce high AP despite the small dataset, Mask R-CNN was a good choice to go with. Whereas for the investigation in regards to the number of epochs and number of steps per epochs, it has shown that the case of 60 epoch and 60 steps per epoch produced the highest accuracy.

5.2 Contributions

During literature review, aside from reading up on theoretical papers to gain an in-depth understanding of Deep Learning, numerous tests were done. These tests were ran on the samples provided by Matterport's *maskrcnn* repository [37], so as to provide a better visual understanding of the code for Mask R-CNN.

All real images of the surgical gauze were taken by SGH nurses during their work. These real images were first manually annotated by using the VGG annotator before any image augmentation. For this project, two scripts were written to further ease the annotation process of the augmented images:

1. A script was written to augmented image traditionally by rotating 90 degrees 3 times, and automatically generate annotations coordinates for these new photos so as to reduce the amount of time spent on image annotation. The purpose of this script is to use a folder of sample images as well the Json file holding the generated annotation data to write another set of annotation coordinates based on the augmentation as mentioned. Specifically, the creation of these new coordinates are made from the information in the original Json file for the un-augmented images.

2. Another script was written to augment images by using Imgaug as mentioned earlier. This script takes in a folder of images for augmentation, then it will save the images. Although this script is similar to the previous, the purpose of this script is to augment images with the modern methods instead of traditional methods. These newly augmented images will require manual annotation.

Furthermore, before officially recording the model during training, algorithm adjustments made to the Mask R-CNN model in order to fit the requirements of the project. The number of epochs and steps per epoch have been experimented with different values to train the model. Initially, Google Colab was used to due to the ability to use better GPUs for faster processing. However, there are limitations:

1. Google Colab only allows 12 hours of usage each time. Once the duration is used, the user will not be able to run the code for a period of time.

2. Google Colab only has 12GB of RAM. As such, for a more intensive training that require higher processing power to reduce the training time, it might overshot the duration of 12 hours in some cases.

3. Google Colab updates their software from time to time. In the duration of this project, the update affected the whole code as it updated a few important library that this software is based on, such as Keras, Tensorflow. As a result, this caused a huge delay in the project as there were alot of fixes required. In this case, in order to use this Mask R-CNN repository, certain versions of Keras and Tensorflow are required as these libraries are only compatible with the code as well as each other for those versions.

Finally, for object tracking, a script was written for the implementation of SORT on top of Mask R-CNN for object tracking. This script uses the trained Mask R-CNN model to do detection for each frame scanned with the python CV2 library. Following that, the script uses SORT object tracker to track the object from frame to frame.

5.3 Recommendations for Future Work

For future work in order for this prototype to be a proper working software, there are several recommendations to be considered.

5.3.1 Images

In terms of images, it is crucial to expand the main dataset for real images of the surgical gauze from numerous operations in order to increase the AP of the model.

Aside from expansion of dataset, exploration of other augmentation such as rotating the image by a few degrees is possible as well. Although this would require more time

for annotation, it is possible to write a script to automate and generate the annotations since the coordinates can be calculated.

5.3.2 Model

In order to avoid the incorrect detection cases such as stains and hands, dataset collection for the other items used in the surgery should be considered. By adding more class labels along with the images that consist the objects of interest such as abdominal swap, cotton wool, hand, surgery tools, this would enhance the detection process. This is because the model will be able to identify the different objects rather than considering them as background. Although this means more time is required for training these new datasets to recognise the other objects, this would definitely reduce a large margin of error in detection. Furthermore, images for the dataset of these objects can be found online as compared to the surgical gauze.

Finally, other famous implementations of Mask R-CNN can be considered, such as Mask R-CNN by Facebook Research *maskrcnn-benchmark* repository [50]. An investigation of other types of object detectors such as YOLO [28] or SSD [29] could be done to compare the AP and whether the trade-off of accuracy for time is worth it. These object detectors are well-known for their detection speed. Although these object detectors have lower AP and accuracy, the amount of time required for detection by the current Mask R-CNN could be heavily reduced if these object detectors are used.

References

[1] Biswas, R. S., Ganguly, S., Saha, M. L., Saha, S., Mukherjee, S. & Ayaz, A. (2012). Gossypiboma and surgeon-current medicolegal aspect–a review. *Indian Journal of Surgery*, *74*(4), 318–322.

[2] Gümüs, M., Gümüs, H., Kapan, M., Önder, A., Tekbas, G. & Baç, B. (2012). A serious medicolegal problem after surgery: Gossypiboma. *The American journal of forensic medicine and pathology*, *33*(1), 54–57.

[3] Zejnullahu, V. A., Bicaj, B. X., Zejnullahu, V. A. & Hamza, A. R. (2017). Retained surgical foreign bodies after surgery. *Open access Macedonian journal of medical sciences*, *5*(1), 97.

[4] Stawicki, S., Evans, D., Cipolla, J., Seamon, M., Lukaszczyk, J., Prosciak, M., Torigian, D., Doraiswamy, V., Yazzie, N., Gunter Jr, O. Et al. (2009). Retained surgical foreign bodies: A comprehensive review of risks and preventive strategies. *Scandinavian Journal of Surgery*, *98*(1), 8–17.

[5] Gawande, A. A., Studdert, D. M., Orav, E. J., Brennan, T. A. & Zinner, M. J. (2003). Risk factors for retained instruments and sponges after surgery. *New England Journal of Medicine*, *348*(3), 229–235.

[6] Egorova, N. N., Moskowitz, A., Gelijns, A., Weinberg, A., Curty, J., Rabin-Fastman, B., Kaplan, H., Cooper, M., Fowler, D., Emond, J. C. Et al. (2008). Managing the prevention of retained surgical instruments: What is the value of counting? *Annals of surgery*, *247*(1), 13–18.

[7] Hariharan, D. & Lobo, D. (2013). Retained surgical sponges, needles and instruments. *The Annals of the Royal College of Surgeons of England*, *95*(2), 87–92.

[8] Macario, A., Morris, D. & Morris, S. (2006). Initial clinical evaluation of a handheld device for detecting retained surgical gauze sponges using radiofrequency identification technology. *Archives of Surgery*, *141*(7), 659–662.

[9] Wiederkehr, J. C., Gama, R. R., Wiederkehr, H. A., Stelmasuk, K., Carvalho, C. A. & Wiederkehr, B. A. (2014). Radio-frequency identification of surgical sponges in the abdominal cavity of pigs. *Annals of Medicine and Surgery*, *3*(2), 31–33.

[10] Frank, M. R., Autor, D., Bessen, J. E., Brynjolfsson, E., Cebrian, M., Deming, D. J., Feldman, M., Groh, M., Lobo, J., Moro, E. Et al. (2019). Toward understanding the impact of artificial intelligence on labor. *Proceedings of the National Academy of Sciences*, *116*(14), 6531–6539.

[11] Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

[12] Brynjolfsson, E. & Mitchell, T. (2017). What can machine learning do? workforce implications. *Science*, *358*(6370), 1530–1534.

[13] Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S. & Lew, M. S. (2016). Deep learning for visual understanding: A review. *Neurocomputing*, *187*, 27–48.

[14] Rawat, W. & Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, *29*(9), 2352–2449.

[15] Camacho, C. (n.d.). Convolutional neural networks. https://cezannec.github.io/Convolutional_Neural_Networks/

[16] LeCun, Y., Bengio, Y. & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436–444.

[17] Hecht-Nielsen, R. (1992). Theory of the backpropagation neural network, In *Neural networks for perception*. Elsevier.

[18] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. Et al. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

[19] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent, In *Proceedings of compstat'2010*. Springer.

[20] Zhou, X. (2018). Understanding the convolutional neural networks with gradient descent and backpropagation, In *Journal of physics: Conference series*. IOP Publishing.

[21] Hawkins, D. M. (2004). The problem of overfitting. *Journal of chemical information and computer sciences*, *44*(1), 1–12.

[22] Sarle, W. S. (1996). Stopped training and other remedies for overfitting. *Computing science and statistics*, 352–360.

[23] Santos, M. S., Soares, J. P., Abreu, P. H., Araujo, H. & Santos, J. (2018). Cross-validation for imbalanced datasets: Avoiding overoptimistic and overfitting approaches [research frontier]. *ieee ComputatioNal iNtelligeNCe magaziNe*, *13*(4), 59–76.

[24] Girshick, R., Donahue, J., Darrell, T. & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation, In *Proceedings of the ieee conference on computer vision and pattern recognition*.

[25] Girshick, R. (2015). Fast r-cnn, In *Proceedings of the ieee international conference on computer vision*.

[26] Ren, S., He, K., Girshick, R. & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks, In *Advances in neural information processing systems*.

[27] He, K., Gkioxari, G., Dollár, P. & Girshick, R. (2017). Mask r-cnn, In *Proceedings of the ieee international conference on computer vision*.

[28] Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. (2016). You only look once: Unified, real-time object detection, In *Proceedings of the ieee conference on computer vision and pattern recognition*.

[29] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. & Berg, A. C. (2016). Ssd: Single shot multibox detector, In *European conference on computer vision*. Springer.

[30] Zhao, Z.-Q., Zheng, P., Xu, S.-t. & Wu, X. (2019). Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, *30*(11), 3212–3232.

[31] Abdulla, W. (2018). Splash of color: Instance segmentation with mask r-cnn and tensorflow. Matterport Engineering Techblog. https://engineering.matterport.com/splash-of-color-instance-segmentation-with-mask-r-cnn-and-tensorflow-7c761e238b46

[32] Taylor, L. & Nitschke, G. (2017). Improving deep learning using generic data augmentation. *arXiv preprint arXiv:1708.06020*.

[33] Perez, L. & Wang, J. (2017). The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*.

[34] Wu, Y., Lim, J. & Yang, M.-H. (2013). Online object tracking: A benchmark, In *Proceedings of the ieee conference on computer vision and pattern recognition*.

[35] Babenko, B., Yang, M.-H. & Belongie, S. (2010). Robust object tracking with online multiple instance learning. *IEEE transactions on pattern analysis and machine intelligence*, *33*(8), 1619–1632.

[36] Yilmaz, A., Javed, O. & Shah, M. (2006). Object tracking: A survey. *Acm computing surveys (CSUR)*, *38*(4), 13–es.

[37] Abdulla, W. (2017). Mask r-cnn for object detection and instance segmentation on keras and tensorflow. Github.

[38] Papers with code - coco minival leaderboard: Papers with code. (n.d.). https://paperswithcode.com/sota/instance-segmentation-on-coco-minival

[39] Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B. & Belongie, S. (2017). Feature pyramid networks for object detection, In *Proceedings of the ieee conference on computer vision and pattern recognition*.

[40] He, K., Zhang, X., Ren, S. & Sun, J. (2016). Deep residual learning for image recognition, In *Proceedings of the ieee conference on computer vision and pattern recognition*.

[41] Propax sterile x-ray detectable gauze 32 ply 10cm x 10cm. (n.d.). https://www.eboshealthcare.co.nz/catalogue-products/wound-management/gauze--non-woven-gauze/propax-gauze-x-ray-32-ply/

[42] Dutta, A. & Zisserman, A. (2019). The VIA annotation software for images, audio and video, In *Proceedings of the 27th acm international conference on multimedia*, Nice, France, ACM. https://doi.org/10.1145/3343031.3350535

[43] Dutta, A., Gupta, A. & Zissermann, A. (2016). VGG image annotator (VIA).

[44] Reitermanova, Z. (2010). Data splitting, In *Wds*.

[45] Jung, A. B., Wada, K., Crall, J., Tanaka, S., Graving, J., Reinders, C., Yadav, S., Banerjee, J., Vecsei, G., Kraft, A., Rui, Z., Borovec, J., Vallentin, C., Zhydenko, S., Pfeiffer, K., Cook, B., Fernández, I., De Rainville, F.-M., Weng, C.-H., … Laporte, M. Et al. (2020). imgaug.

[46] Bewley, A., Ge, Z., Ott, L., Ramos, F. & Upcroft, B. (2016). Simple online and realtime tracking, In *2016 ieee international conference on image processing (icip)*. https://doi.org/10.1109/ICIP.2016.7533003

[47] Welch, G., Bishop, G. Et al. (1995). An introduction to the kalman filter.

[48] Bathija, A. & Sharma, G. (n.d.). Visual object detection and tracking using yolo and sort.

[49] Passmark software - video card benchmarks - gpu compute video cards. (n.d.). https://www.videocardbenchmark.net/directCompute.html

[50] Massa, F. & Girshick, R. (2018). maskrcnn-benchmark: Fast, modular reference implementation of Instance Segmentation and Object Detection algorithms in PyTorch.