Automatic Tooth Numbering in Bitewing Radiographs using Deep Learning

Submitted for the fulfilment of

MSc in Data Science Dissertation (COMM002)

September 2023

Department of Computer Science

University of Surrey

Declaration

I confirm that the submitted work is my own work. No element has been previously submitted for assessment, or where it has, it has been correctly referenced. I have clearly identified and fully acknowledged all material that is entitled to be attributed to others (whether published or unpublished) using the referencing system set out in the programme handbook.

I agree that the University may submit my work to means of checking this, such as the plagiarism detection service Turnitin® UK and the Turnitin® Authorship Investigate service. I confirm that I understand that assessed work that has been shown to have been plagiarised will be penalised.

Abstract

Dental radiography is an essential X-ray imaging technique in medical diagnosis. Although all dentists are professionally trained in reviewing dental radiographs, various imaging artifacts introduced during the image capturing process can significantly affect dentists' interpretation and the quality of their diagnosis. Additionally, clinical work efficiency is impacted by the considerable amount of administrative work involved in preparing dental charts and entering records.

Deep learning is one of the most actively pursued research areas in Artificial Intelligence (AI) in recent decades, as it enables the efficient and accurate extraction of information from voluminous images, which cannot be achieved through manual efforts. It has gained increasing interest in dental disease detection and tooth identification/classification in recent years. However, the application of deep learning models for tooth numbering in bitewing images is a relatively unexplored area.

This dissertation aims to implement a new stage-wise approach that can automatically detect teeth in bitewing radiographs and identify their tooth numbers. The work in this dissertation contributes to one of the objectives of the Dental Disease Detection (DDD) project launched in 2020 by the University of Surrey: the development of a dental charting application that utilizes the sample radiographs collected within the DDD project. The dissertation project's work includes: (a) Data collection – Developing a workflow on Zooniverse.org for ground truth label annotations; (b) Data cleansing – Examining and cleaning the collected data; and (c) Data analysis – Employing YOLOv5 with oriented bounding box support as object detection models and introducing a novel stage-wise detection approach to enhance detection performance by incorporating the spatial relationships between pairs of teeth into the decision-making process. The analysis results indicate the importance of separating the detection of left and right quadrants as a distinct step, which contributes to achieving more accurate detection performance in comparison to the direct tooth number identification by YOLOv5.

Acknowledgement

<Redacted>

Table of Contents

Chapter 1 Introduction 9
1.1 Project Initiative 9
1.2 Aims of Dissertation 10
1.3 Structure of Dissertation 10
Chapter 2 Literature Review 12
2.1 Tooth Numbering Notations 12
2.2 Tooth Recognition Techniques in Dental Radiographs 13
2.3 Computerised Solutions to the Tooth Detection Problem 14
2.4 Review of Research Works on Tooth Detection 15
2.5 Convolution Neural Network 17
2.6 Object Detection Network 21
2.7 Research Gap Identified 26
Chapter 3 Research Methodology 29
3.1 Data Collection Process 29
3.2 Data Preparation Process 35
3.3 Tooth Detection and Numbering Model 37
Chapter 4 Presentations and Analysis Results 51
4.1 Data Description 51
4.2 Analysis of Results in Stage 1 54
4.3 Analysis of Results in Stage 2 57
4.4 Analysis of Results in Stage 3 61
4.5 Analysis of Results in Stage 4 67
Chapter 5 Conclusions 79
5.1 Project Evaluation 79
5.2 Project Limitations and Future Improvements 79

List of Figures

Figure 1 FDI Notation in Tooth Numbering. Image source [48]. 12
Figure 2 Comparison of root patterns of molars in upper and lower quadrants. Image source [49]. 13
Figure 3 Bitewing image showing teeth leaning towards the lip side. 13
Figure 4 Bitewing images with different teeth conditions. Left: prostheses; Middle: treated root; Right: implant 16
Figure 5 Comparison of architecture of ANN (left) and CNN (right). In CNN, the input is a 3D matrix (width x height x R,G,B channels), the kernel is also a 4D matrix with size (kernel width x kernel height x R,G,B channels x depth). Image source [23]. 17
Figure 6 Convolution operation on input feature map with one channel (left) and multiple channels (right). Image source [22]. The receptive field is 2x2=4 and the depth is also 1. 19
Figure 7 Padding zeros along edges and slide the kernel by 1 step (1 pixel) to generate the output map. Image source [21]. 19
Figure 8 Max pooling operation to reduce the output map size. Image source [22]. 19
Figure 9 Different features of the input image (centre) are captured in various feature maps after the convolution layer operation. Image source [11]. 20
Figure 10 Typical layer arrangement of CNN as an image object classifier. 20
Figure 11 Components of Object Detection Model. Image source [27]. 22
Figure 12 Overview of processing steps in R-CNN. Image source [30]. 23
Figure 13 Illustration of Faster R-CNN (left) and the anchor boxes for object region proposals (right). Image source [32][33]. 24
Figure 14 Workflow and model architecture of YOLO. Image source [34]. 25
Figure 15 Use of horizontal bounding boxes (left) and oriented bounding boxes (right) in tooth detection on bitewing images. 26
Figure 16 Example of three types of dental radiographs. Image source [1] [46]. 27
Figure 17 Recognizing teeth individually without considering their spatial relationship. 28
Figure 18 Recognizing teeth by considering their spatial relationship with neighbours. 28
Figure 19 Examples of images with artefacts and excluded from analysis. 29
Figure 20 Homepage of the DDD project to access new workflows 30
Figure 21 Screenshot of Practice Tooth Numbering Workflow 31
Figure 22 Screenshot of Tooth Numbering Workflow 31
Figure 23 Screenshots of Topical Field Guides 32
Figure 24 Screenshot of Tooth Numbering Tutorial 33
Figure 25 Illustration of Annotation Criteria 34
Figure 26 Examples of multiple annotations on the same image and its result of reconciliation. All teeth numbers are prefixed with the letter 't' in the image. 36
Figure 27 Frequency of tooth numbers appeared in annotations 37
Figure 28 Stage-wise detection model in this dissertation 38
Figure 29 Illustration of Stage 3 decision making process 48
Figure 30 Illustration of Stage 3 decision making process when Stage 2 output is not optimal 49
Figure 31 Illustration of Stage 3 decision making process when there is a missing tooth. 49
Figure 32 Illustration of Stage 3 decision making process when there are not enough teeth detected in each quadrant 50

Figure 33 Illustration of combination process in Stage 4 to deduce 32 tooth numbers 50
Figure 34 Annotation frequency of tooth numbers 51
Figure 35 Boxplots of distances between adjacent tooth pairs in upper and lower quadrants 52
Figure 36 Images with different zoom levels. 53
Figure 37 Boxplots of IoU between adjacent tooth pairs in upper and lower quadrants 53
Figure 38 Overlapping of bounding boxes of adjacent teeth 54
Figure 39 Change of average training loss in learning rate finder 55
Figure 40 Training loss and validation loss of Stage 1 left/right classifiers 55
Figure 41 Confusion matrix and misclassified examples 56
Figure 42(b) Validation loss of Stage 2 Object Detector using YOLOv5 58
Figure 43 Training and validation loss of Stage 2 Object Detector using YOLOv5n 60
Figure 44 Training and validation loss of Stage 2 Object Detector using YOLOv5m 61
Figure 45 Scatterplots of tooth pairs. *Ui_Uj* and *Li_Lj* refer to the teeth pairs (t_i, t_j) in the upper and lower quadrants respectively. t_i denotes a nonmolar tooth and t_j denotes a molar tooth both in either upper or lower quadrants. 62
Figure 46 Scatterplots of tooth pairs. *Ui_Uj* and *Li_Lj* refer to the teeth pairs (t_i, t_j) in the upper and lower quadrants respectively. t_i and t_j denote molar teeth both in either upper or lower quadrants. 63
Figure 47 Scatterplots of tooth pairs. *Ui_Uj* and *Li_Lj* refer to the teeth pairs (t_i, t_j) in the upper and lower quadrants respectively. t_i and t_j denote nonmolar teeth both in either upper or lower quadrants. 63
Figure 48 Scatterplots of tooth pairs. *Ui_Lj* refer to the teeth pairs (t_i, t_j) in the upper and lower quadrants respectively. Here we examine *Ui=U5* and *Ui=U6*. 65
Figure 49 Scatterplots of tooth pairs. *Li_Uj* refer to the teeth pairs (t_i, t_j) in the lower and upper quadrants respectively. Here we examine *Li=L5* and *Li=L6*. 66
Figure 50 Three approaches to tooth object detection and tooth number classification. 68
Figure 51 Detection results of the three approaches. The single-stage detector is confused by the left / right quadrants. 72
Figure 52 Detection results of the three approaches. The single-stage detector and the three-stage detector mistakes t_{13} with t_{14}, and t_{45} and t_{44}. 73
Figure 53 Detection results of the three approaches. The single-stage detector and the three-stage detector fail to detect some challenging teeth t_{25} and t_{26} as their "objectness" score is not high above the confidence threshold set at 0.6. 74
Figure 54 Intermediate detection results of the three-stage and the four-stage approaches. The confidence threshold in the YOLOv5 is lowered to 0.1 to allow more bounding boxes to be shown. 75
Figure 55 Detection results of the three approaches. The four-stage detection may give wrong prediction when there are no non-molar and polar teeth identified simultaneously in the same quadrant, as shown in the top left quadrant above. 77
Figure 56 Detection results of the three approaches. The four-stage detection cannot make correct predictions when if the detection results in any stage, e.g. stage 2 as shown above, makes wrong decision to identify upper (t_{17}, t_{15}, t_{16}) and lower quadrant teeth (t_{47}, t_{45}, t_{46}). 78

List of Tables

Table 1 Definition of classifiers to classify tooth pairs in the same quadrant. The input feature vector $vij = Wi, Wj, Dij$, where Wi, Wj are the widths of the bounding boxes of t_i tooth and t_j in the same quadrant, Dij is distance between the centres of these bounding boxes. The output is the tooth pair label (t_i, t_j). 41
Table 2 - Definition of classifiers to classify tooth pairs in the opposite quadrant. The input feature vector consists of $vij = Wi, Wj, Dij$, where Wi, Wj are the widths of the bounding boxes of i-th tooth and j-th tooth in the opposite quadrants, Dij is distance between the centres of these bounding boxes. The output is the tooth pair label (t_i, t_j)..... 42
Table 3 Performance metrics of YOLOv5 with three model sizes..... 59
Table 4: $\varphi q, vij$ classifies tooth pairs (t_i, t_j) for both t_i *and* t_j in the same quadrant and $i, j = 3,4,\ldots,8$ 64
Table 5: $\chi q, vij$ classifies tooth pairs (t_i, t_j) for both t_i and t_j in the opposite quadrants 67
Table 6 Performance metrics of tooth detection and tooth number classification with YOLOv5n implementation. 69
Table 7 Performance metrics of tooth detection and tooth number classification with YOLOv5m implementation 70
Table 8 Performance metrics of tooth detection and tooth number classification with YOLOv5x implementation 71

Chapter 1 Introduction

1.1 Project Initiative

Dental radiography is an essential X-ray imaging technique in medical diagnosis. A large number of radiographic images are evaluated by dentists every day to assess variations in tooth anatomy and surrounding oral structures. The diagnosis of common dental diseases, such as periodontal bone loss and dental caries, also heavily relies on radiographs [1,2]. Although all dentists are professionally trained in reviewing dental radiographs, imaging artifacts introduced by poor contrast, loss of focus, noise, and misalignment during the image capturing process can significantly affect dentists' interpretation and the quality of their diagnosis. Additionally, clinical work efficiency is impacted by the considerable amount of administrative work involved in preparing dental charts and entering records. With the rapid advances in Artificial Intelligence (AI) in recent decades, the AI-assisted diagnosis approach is a key development to streamline clinical processes, improve reproducibility through objective assessments, and enhance the work efficiency of dentists and nurses [3].

Deep learning is one of the most actively pursued research areas in AI, as it enables efficient and accurate extraction of information from voluminous images, which are not achievable by manual efforts. It has gained increasing interest in dental disease detection [4,5,6,7,8,9] and tooth identification/classification [10,11,12,13,14] in recent years. Against this backdrop, a Dental Disease Detection (DDD) project, led by Dr. Y.P. Li of the University of Surrey, was launched in 2022 on Zooniverse [15]. The overall DDD project aim is

"... to use AI to bridge the gap in radiographic diagnosis, by building a software prototype enabling computers to recognise normal and abnormal anatomical structures and differentiate between subtle abnormalities present in radiographs. Using state of the art and specialised object detection techniques, we aim create a radiograph viewing and automatic dental charting application with AI assisted disease detection and localisation."

As a result, the research work reported in this dissertation contributes to one of the objectives of the DDD project: the development of a dental charting application that utilizes the sample radiographs collected within the project.

1.2 Aims of Dissertation

This dissertation aims to implement a new approach that can automatically detect teeth in bitewing radiographs and identify their tooth numbers according to the FDI World Dental Federation notation [16]. Since the DDD project can provide over 3,000 radiographs without ground truth labels, a new workflow must be developed in the Zooniverse to engage volunteers in annotating the radiographs. Existing deep learning object detection algorithms are then studied and applied to solve the problem. The impact of model parameters on the performance of tooth detection is also studied to identify the best options. The following is the scope of the project:

a. Data collection – Develop a workflow in Zooniverse for ground truth label annotations.
b. Data cleansing – Examine and cleanse the collected data.
c. Data analysis – Implement machine learning models and object detection algorithms for tooth numbering.

1.3 Structure of Dissertation

This dissertation contains five chapters as follows:

Chapter 1 Introduction

In this chapter, the initiative and aims of this dissertation are explained, along with an elaboration of its scope and structure.

Chapter 2 Literature Review

This chapter reviews the background of tooth numbering and existing research on object detection algorithms, focusing on their applications to tooth identification and numbering. The goal is to understand the current landscape and establish the research directions for this dissertation.

Chapter 3 Research Methodology

The methodology for implementing the deep learning models for tooth identification and numbering is explained in this chapter. It covers the Zooniverse workflow design, data collection and preprocessing, as well as the settings for training and testing the machine learning models.

Chapter 4 Presentations and Analysis of Results

In this chapter, the tooth identification and numbering results are presented and analyzed. Additionally, a discussion on the comparisons of different choices of model parameters and their impact on the model's performance is included.

Chapter 5 Conclusions

This final chapter concludes the research work by evaluating the achieved requirements, difficulties encountered, lessons learned, and suggestions for future improvements.

Chapter 2 Literature Review

2.1 Tooth Numbering Notations

There are a total of 32 teeth in most adults (permanent dentition), with 8 teeth in each of the upper left, upper right, lower left, and lower right quadrants of the mouth. The Federation Dentaire Internationale (FDI) World Dental Federation notation system is a commonly used system for numbering and naming these 32 teeth. In the FDI notation, each tooth number in permanent dentition is designated by 2 digits. The first digit represents the quadrant number, and the second digit represents the tooth number, starting from 1 at the mid-lines and incrementing toward the jaw, as shown in Figure 1. Although there are other tooth numbering notations, such as the Universal Tooth Numbering System and the Palmer Notation [47], the FDI notation is adopted in this dissertation due to its worldwide popularity. Moreover, to avoid confusion with regular numbers, all tooth numbers appearing in this report are prefixed with the letter 't', e.g., t_{11}, t_{12}, t_{13}, etc.

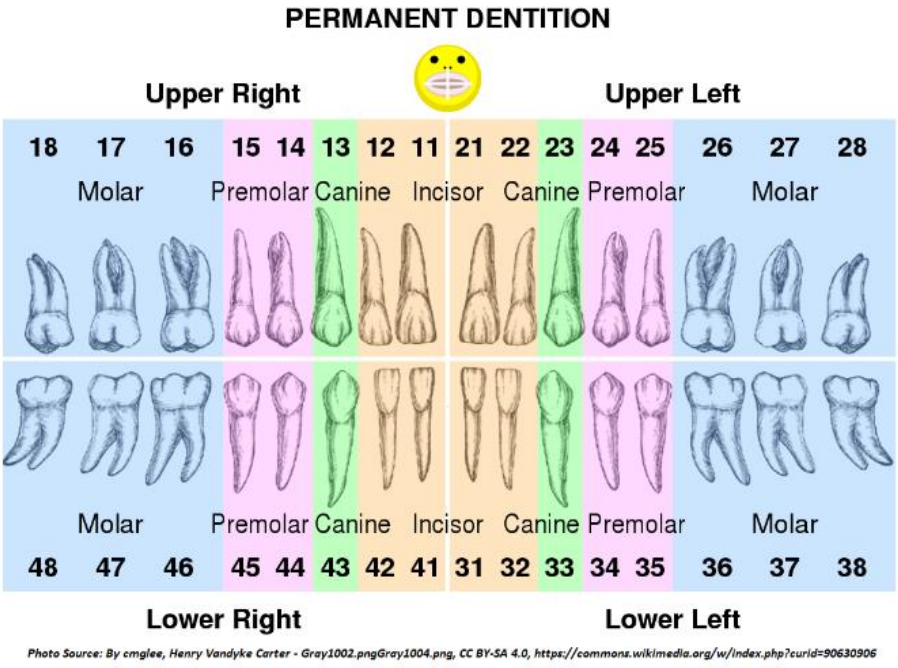


Figure 1 FDI Notation in Tooth Numbering. Image source [48].

2.2 Tooth Recognition Techniques in Dental Radiographs

Without relying solely on tooth anatomy, one of the notable differences among teeth is their relative sizes. It is easy to differentiate molar teeth from non-molar teeth as the former are usually larger and wider. Apart from tooth size, it is also possible to distinguish between upper molars and lower molars by considering their root patterns. As shown in Figure 2, upper molars typically have 3 roots, whereas lower molars typically have 2 roots. Since premolar teeth are noticeably smaller in size than molar teeth, recognizing a 2-root pattern molar allows us to identify it as a tooth in the lower jaw, while those with 3 roots are located in the upper jaw.

Figure 2 Comparison of root patterns of molars in upper and lower quadrants. Image source [49].

After identifying the upper and lower sides using root patterns, the next step is to determine the left/right quadrants. This can be recognized by the directions of the long axes of teeth, which typically lean towards the lip. The example bitewing image shown in Figure 3 is taken from the right quadrant, as the long axes of all teeth are leaning towards the lip.

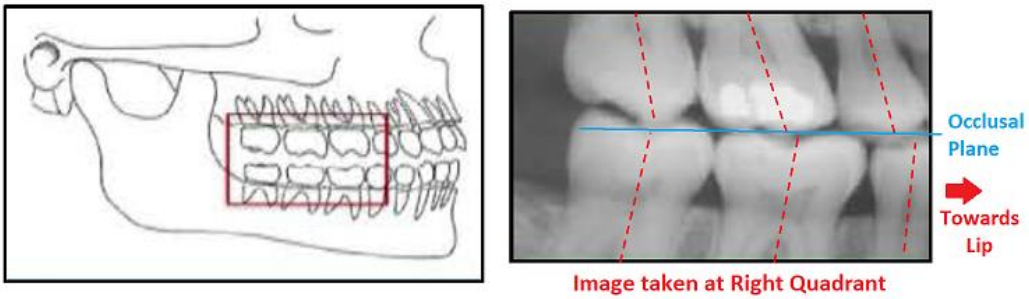


Figure 3 Bitewing image showing teeth leaning towards the lip side.

Based on the above two techniques, we can essentially distinguish the four quadrants of the mouth from a given dental radiograph, such as a bitewing image. To precisely recognize each tooth number, we then need to consider the natural tooth sequence starting from 1 to 8 in FDI notation and the spatial relationships among them.

2.3 Computerised Solutions to the Tooth Detection Problem

The above tooth detection techniques are applicable to both humans and computers. Computerized solutions to tooth detection problems have been explored by applying various object detection algorithms. Recognizing objects of different classes, such as cars, ships, puppies, and humans, is an actively researched area in computer vision. As defined in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [17], the algorithms to recognize objects can be categorized into:

- Image Classification: to recognize one or more classes of objects appearing in the image.
- Single-object localization: to produce a list of object categories present in the image, along with an axis-aligned bounding box indicating the position and scale of one instance of each object category.
- Object detection: to produce a list of object categories present in the image along with an axis-aligned bounding box indicating the position and scale of every instance of each object category.

Before the advent of the deep learning era, object recognition problems were solved by techniques such as Histograms of Oriented Gradients [18] and Selective Search [19]. These methods were based on traditional features such as gradient vectors extracted from the image and usually suffered from complicated feature engineering processes and expensive computational workloads to search and localize objects before performing final classification. In the last decade, the rapid increase in computational power brought by Graphics Processing Units (GPUs) catalyzed the fast adoption of artificial neural networks (ANNs), which are at the heart of deep learning algorithms. The plasticity (ability to learn) of an ANN lies in the trainable parameters (weights) of the network, which can adapt to the nature of the problem through the training

process. As a result, ANN-based solutions dominate the approach to solving object recognition problems.

In this dissertation, previous research on tooth detection using object detection algorithms is reviewed to identify the research gap that this dissertation will address. Next, the theoretical background of object detection using CNN is also reviewed.

2.4 Review of Research Works on Tooth Detection

There have been numerous research works on tooth classification and numbering before the advent of the deep learning era. Most of these approaches involve two stages: first, tooth boundary segmentation and feature extraction, followed by teeth number classification.

For example, in [41], the researchers extracted the teeth features from the Fourier descriptors of the teeth contours and used Bayesian classification to classify the teeth into molars and premolars in a bitewing image. Then, each tooth was numbered by considering the spatial relationship between the teeth. In [42], the researchers first performed image enhancement to boost the bitewing image contrast to extract crown and pulp contours. The crown size and pulp length-to-width ratio were then calculated as features for tooth classification using a support vector machine classifier. In [43], the researchers first performed image enhancement to boost the bitewing image contrast to extract crown and pulp contours. The crown size and pulp length-to-width ratio were then calculated as features for tooth classification using a support vector machine classifier. This traditional approach demands carefully designed image enhancement and feature extraction techniques and assumes the teeth images are intact. However, it is very common to encounter radiographs from patients with different teeth conditions, such as prostheses, treated roots, and implants, as shown in Figure 4.

Figure 4 Bitewing images with different teeth conditions. Left: prostheses; Middle: treated root; Right: implant

The advent of deep learning has revolutionized the solution to this problem. Compared to traditional approaches, deep learning can learn features directly from raw data input without the need for manual feature extraction. Several deep learning-based approaches have been developed.

In [11], due to the limited amount of labeled periapical images, the researchers adopted a cascade network with multiple CNNs responsible for predicting the tooth quadrants in a sequential manner. The first network identifies the teeth vs. background, and the results are fed to the second network to predict upper/lower quadrants, followed by the third network to predict the tooth number. A rule-based model is then used to refine the tooth number classification results, achieving over 95% precision and recall. The same team of researchers proposed another approach based on Faster R-CNN, along with a rule-based module for tooth detection in periapical images, achieving over 90% precision and recall [12]. In [10], the researchers adopted the Faster R-CNN model to first detect the bounding boxes of all teeth in panoramic images. Then, the box-bounded tooth images were cropped to train a CNN image classifier for tooth number classification. They also applied a heuristic function based on the tooth sequence to correct misclassifications and improve accuracy. In [44], a similar approach to [10] was adopted to detect and number teeth in panoramic images. However, the researchers further applied a U-Net model to perform rough segmentation of tooth regions before training the Faster R-CNN. This approach improved the performance in detecting tooth regions that are not vividly obvious in typical panoramic images. Instead of Faster R-CNN, the researchers in [14] adopted the EfficientDet, an object detection model based on the EfficientNet backbone optimized for

resource-constrained environments [45]. This model was used to detect and recognize both tooth numbers and tooth treatment patterns in panoramic images.

2.5 Convolution Neural Network

Since a computer represents a colored digital image file as a 3-channel (Red, Green, and Blue) 2-dimensional (Width and Height) matrix, the Convolutional Neural Network (CNN), which was pioneered by LeCun [20] in the 90s, becomes the go-to solution for many computer vision tasks, including object recognition and detection. Like ANN, a CNN contains a layered architecture with trainable parameters in each layer. However, the heart of a CNN is the operation of a set of kernels, or filters in signal processing theory, applied to the 2-dimensional feature maps that are output from the previous layers. The kernels usually have a width and height smaller than the input image, but with a higher number of channels. The number of channels of a kernel is called depth.

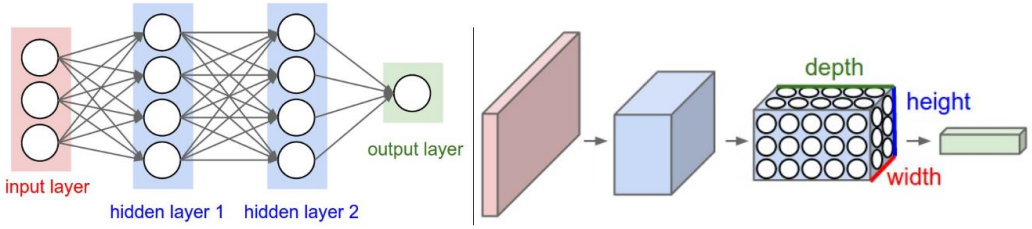


Figure 5 Comparison of architecture of ANN (left) and CNN (right). In CNN, the input is a 3D matrix (width x height x R,G,B channels), the kernel is also a 4D matrix with size (kernel width x kernel height x R,G,B channels x depth). Image source [23].

There are three essential operations to compute the output in a CNN: (i) convolution operation, (ii) stride and padding, and (iii) pooling. Figure 6 illustrates the convolution operation. In convolution, each kernel is multiplied elementwise onto the input feature map and then subjected to a weighted linear combination. The weight parameters of each kernel are learned during the training process [21,22]. After performing a convolution operation on the first matrix element of the input map, i.e., the top-left pixel value of a channel, the kernel then slides to the right for a step called stride and performs the convolution operation again to obtain another output element.

This process creates a receptive field for each output element pixel, which can be interpreted as a linear combination of a number of input element pixels.

The purpose of the padding operation is to augment the size of the input feature map along its edges so that the convolution operation can be performed symmetrically on the edges to preserve the information extracted on the perimeters [22]. The pooling operation is typically implemented in a separate layer called the pooling layer. Since the convolution layer output is usually down-sampled spatially (i.e., the output map size is smaller than the input map size) but up-sampled into multiple channels (from having R,G,B channels to “depth”” channels, where “depth” is the kernel depth dimension), the pooling layer further down-samples each layer spatially. The pooling operation involves a kernel, but instead of performing a convolution operation, the pooling layer typically adopts a linear average or a maximum selection operation. Figure 8 illustrates one such pooling operation by selecting the maximum value of the input map.

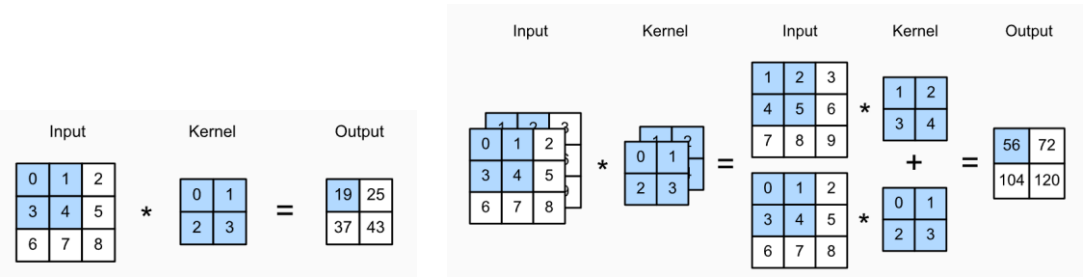


Figure 6 Convolution operation on input feature map with one channel (left) and multiple channels (right). Image source [22]. The receptive field is 2x2=4 and the depth is also 1.

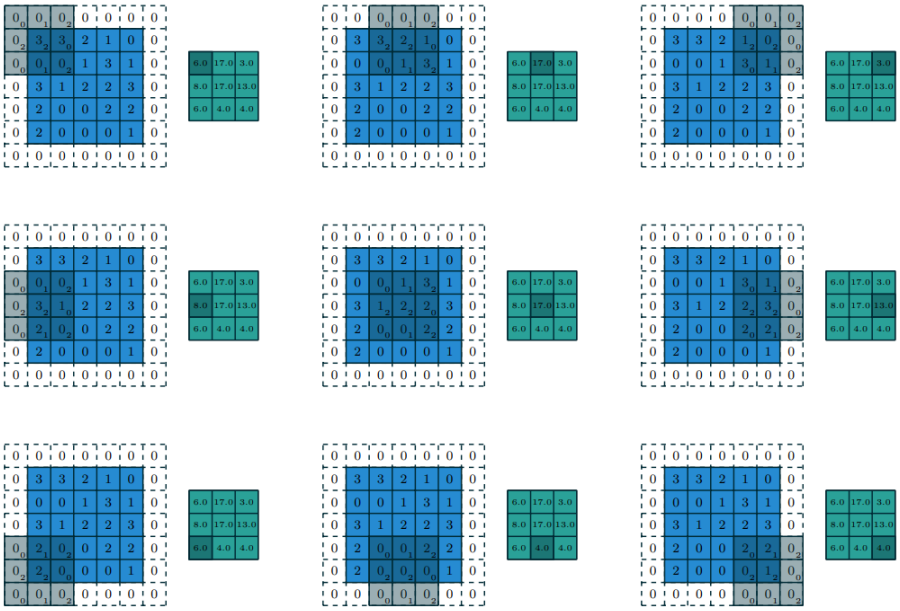


Figure 7 Padding zeros along edges and slide the kernel by 1 step (1 pixel) to generate the output map. Image source [21].

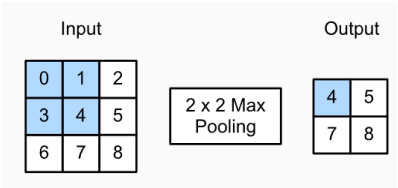


Figure 8 Max pooling operation to reduce the output map size. Image source [22].

The beauty of CNN comes from its relatively lower computational cost in matrix multiplication compared to the traditional 'fully connected' layers in ANN. The output of a convolution layer

carries different features of the input image in the form of feature maps, thanks to the use of a kernel with multiple depth, as shown in Figure 9. Since the kernel parameters are learned during the training process, the feature maps will be tuned to extract relevant features according to the target object characteristics of the classification task. The extracted features of a target object are said to be location-invariant [14], meaning they are independent of the location and orientation of the target in the input image.

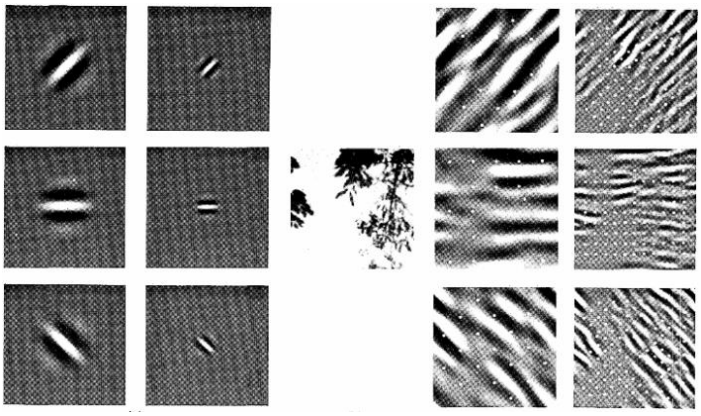


Figure 9 Different features of the input image (centre) are captured in various feature maps after the convolution layer operation. Image source [11].

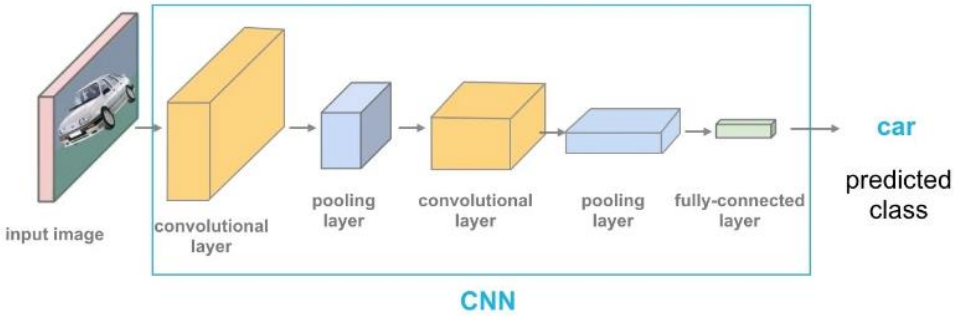


Figure 10 Typical layer arrangement of CNN as an image object classifier.

Figure 10 illustrates the typical architecture of a CNN for an image classification task. The input image goes through the convolution operation and the pooling operation in multiple layers before being fed to the final fully-connected (FC) layer for image classification. The design of CNN is the cornerstone of various computer vision tasks. Over the last decade, numerous deep neural

networks proposed, such as AlexNet[24], VGG16[25] and ResNet[26], have adopted the principles of CNN to effectively extract features from a large number of sample images, resulting in excellent classification accuracy. Besides solving image classification problems, CNN architectures are also utilized as 'feature extractors' in more advanced deep neural networks to tackle challenging object detection problems.

2.6 Object Detection Network

General Architecture

With CNN as the essential component, a sophisticated object detection network generally has many sub-networks, each performing different tasks. In general, these component networks can be classified as Backbone, Neck, and Head [27] [28]. The Backbone is responsible for feature extraction. A typical Backbone will be implemented using well-developed CNN architectures, such as ResNet or VGG16. The Neck allows features with different scales, extracted from the Backbone, to interact and aggregate with each other. Typically, a Neck will be implemented in multi-layered multi-scaled ANN architectures with bi-directional top-down and bottom-up paths, such as Feature Pyramid Network [29]. The Head is responsible for bounding box regression and object classification. Bounding box regression is a process to locate and refine the coordinates of a bounding box, typically a rectangle, around a recognized object in the image. A bounding box is not necessarily for target objects; it can also refer to the background of an image, and this is indicated by the foreground/background score to measure the probability of having an object inside the box. The object classification task is to classify the object inside the bounding box into different labels. Figure 11 illustrates the three components of an object detection network, and different CNN model names taking the role of that component.

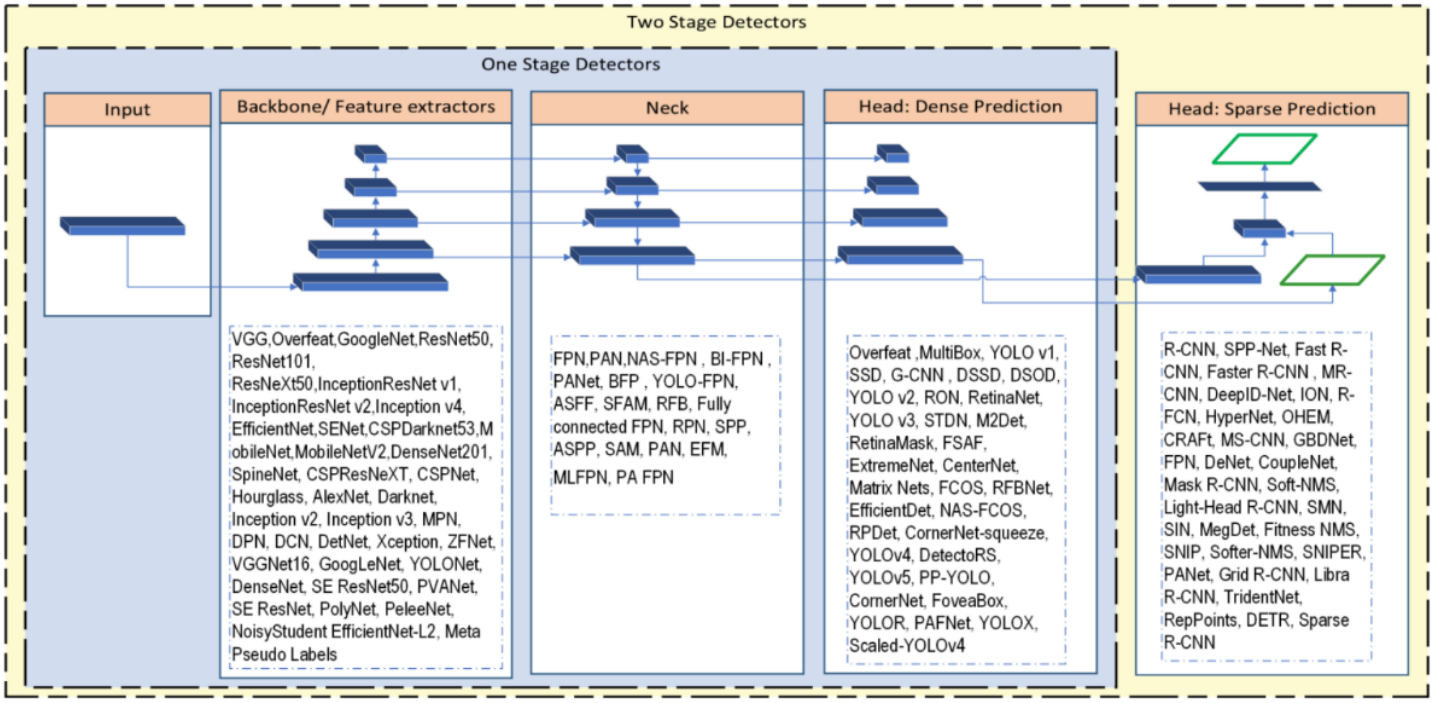


Figure 11 Components of Object Detection Model. Image source [27].

Two-stage Object Detection

Another conceptual categorization of object detection networks is two-stage and one-stage detectors. The first large-scale and successful application of CNN to object localization and detection, “Regions with CNN Features” (R-CNN) [30], is a typical two-stage detector. The working principle of R-CNN is to perform “selective search” [19] of regions potentially containing objects, which are called Region of Interest (RoI). The Head of the model will be a pre-trained CNN such as ResNet or VGG, which will be fine-tuned during the training process to classify objects in each RoI. A regression model is also trained to fine-tune the coordinates of the bounding box. Figure 12 illustrates the overview of the processing steps of R-CNN.

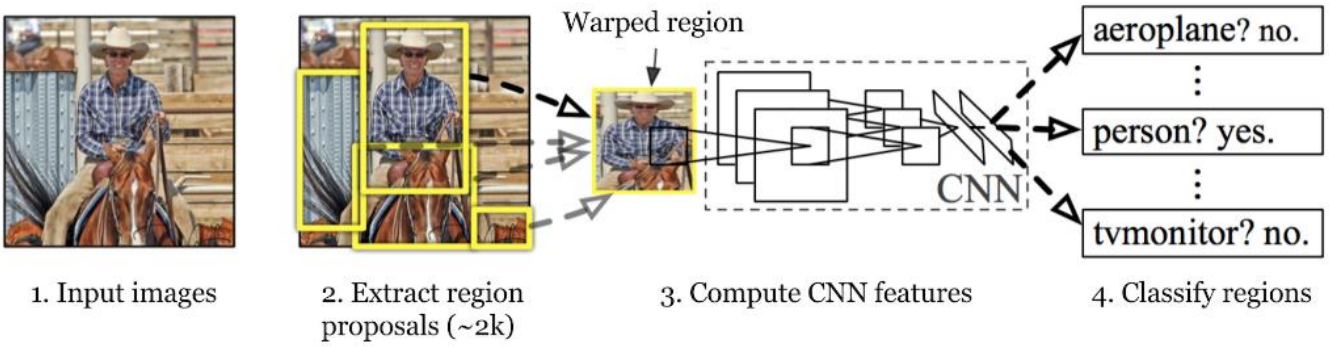


Figure 12 Overview of processing steps in R-CNN. Image source [30].

There are two drawbacks in R-CNN that render it slow during training. The first is that it involves computationally expensive selective search process to identify thousands of RoIs for each image. The second is that it has to train three models: one for feature extraction, one for object classification, and one for bounding box regression, without resource sharing among them. As a result, the original authors proposed a Fast R-CNN version [31] by replacing the selective search with a window extraction and RoI pooling mechanism to generate region proposals. The separately trained models were also joined together for sharing training results to improve the training speed. Two years later, the same research team developed the Faster R-CNN [32] by unifying the region proposal mechanism into the region proposal network (RPN). Inside the RPN, anchor boxes parameterized by their length and width are fitted to generate region proposals, as shown in Figure 13. The research team reported that Faster R-CNN object detection can achieve 5fps, which is nearly real-time object detection in video. Faster R-CNN remains the first-choice solution for two-stage object detectors.

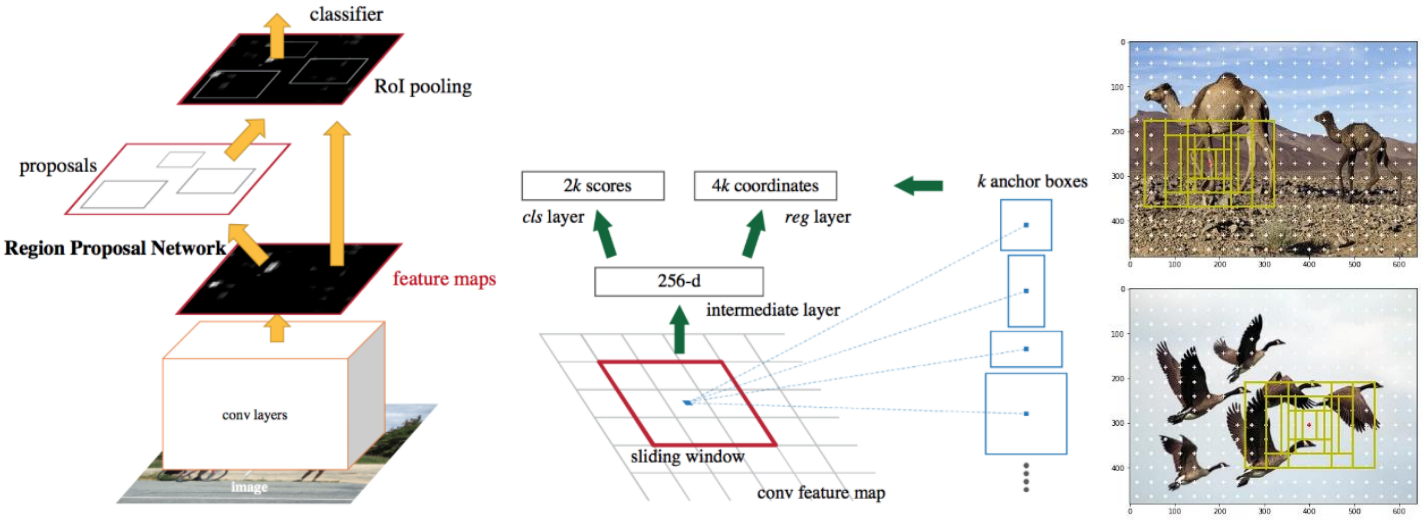


Figure 13 Illustration of Faster R-CNN (left) and the anchor boxes for object region proposals (right). Image source [32][33].

One-stage Object Detection

The increasing quality of images and videos drives the pursuit for faster object detection models without sacrificing average precision. This has led to the development of one-stage detection models such as You Only Look Once (YOLO) [34] and Single Shot Detection (SSD) [35]. In two-stage detectors, such as Faster R-CNN, the first stage is to search for region-based object proposals using RPN. The proposals generated are sparse, as potential bounding box candidates can be theoretically infinite despite being controlled by hyper-parameters. In contrast, one-stage detectors simplify the region proposal stage by directly predicting bounding boxes over a dense sampling of possible locations. This significantly improves the inference speed to over 10 times in fps compared to Faster R-CNN, while the detection average precision remains comparable to two-stage detectors.

The YOLO model was the first successful one-stage detector, offering superior inference speed while maintaining comparable average precision to two-stage detectors. It adopts an end-to-end network design consisting of 24 layers of CNN and works in a simple way. The input image is first divided into S x S cells. Each cell predicts three things: (i) the center coordinates and sizes of a predefined number of bounding boxes; (ii) a probability score indicating whether a cell contains an object; and (iii) a set of conditional probabilities that a cell contains a particular class

of object given (ii). The training process minimizes the loss arising from the object classification error and the bounding box regression error. Figure 14 illustrates the workflow and architecture of YOLO.

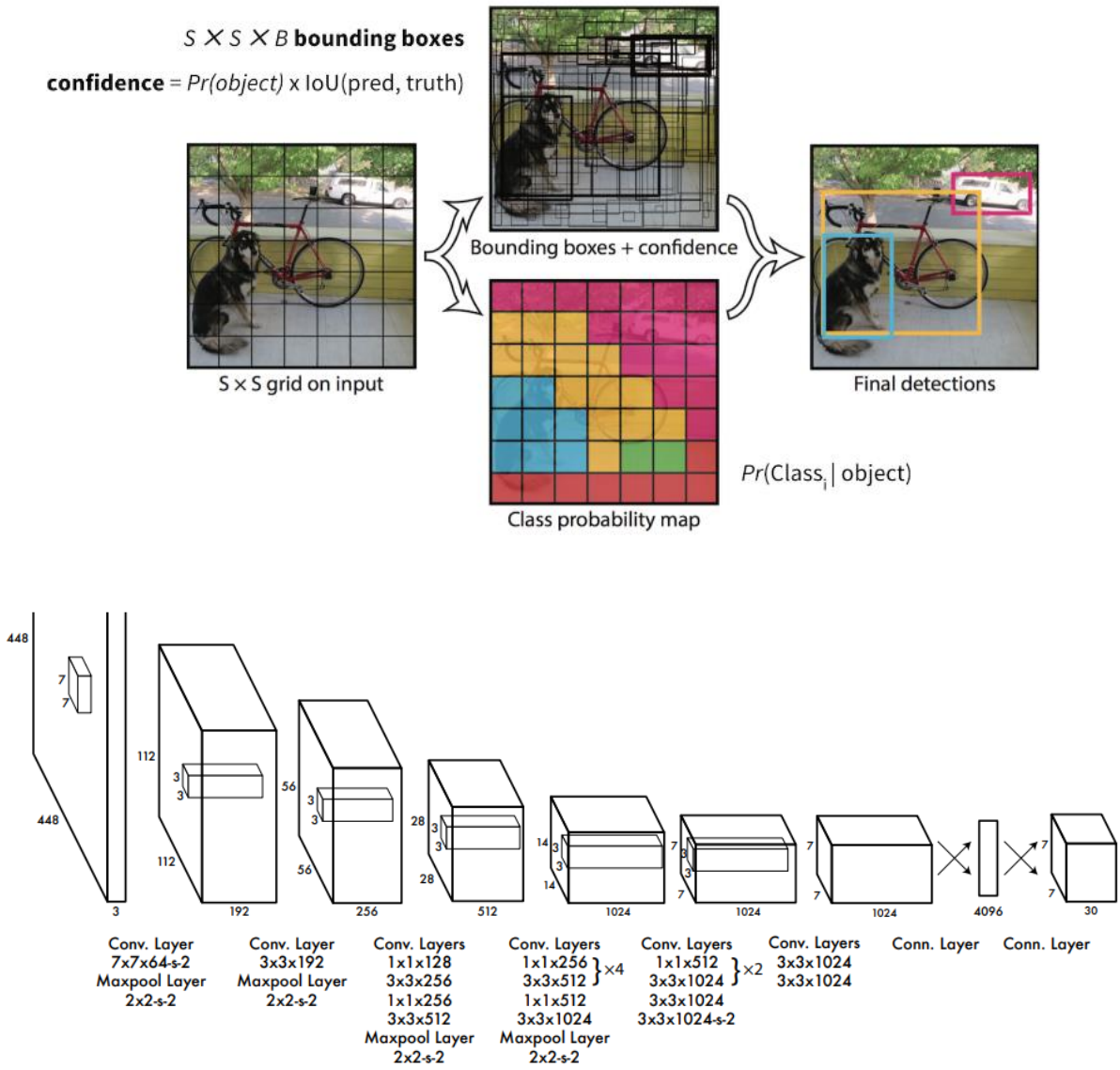


Figure 14 Workflow and model architecture of YOLO. Image source [34].

Due to the accessibility of its pre-trained model and its superior performance, YOLO became the most popular one-stage object detection model since its publication in 2016. Its architecture has undergone a number of revisions to improve its performance, and the latest version publicly available is YOLOv7, released in early 2023 [36].

Since YOLO became open-sourced with YOLOv5 in 2020 [37], variants have been added to address different needs. In particular, all two-stage and one-stage detectors, by default, adopt horizontal bounding boxes instead of oriented bounding boxes. As shown in Figure 15, adopting oriented bounding boxes is apparently more suitable for object detection, particularly in scenarios where objects are tightly packed in a rotated manner, as using oriented bounding boxes reduces the amount of overlapping. To address this problem, variants that add oriented bounding box support to the Faster R-CNN family and YOLOv5 have been developed by improving the R-CNN RPN to propose oriented bounding boxes [38,39] or generating rotated anchors in YOLO with additional rotation angle as a regression parameter [40].

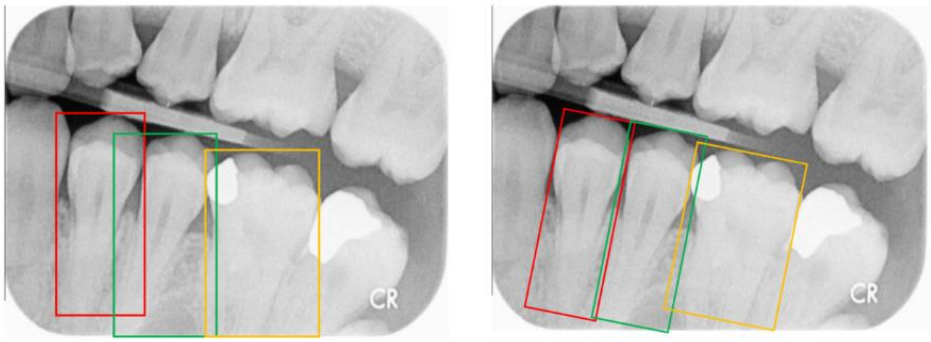


Figure 15 Use of horizontal bounding boxes (left) and oriented bounding boxes (right) in tooth detection on bitewing images.

Since adopting YOLO through transfer-learning and fine-tuning requires a smaller amount of training data and also the availability of support for oriented bounding boxes, this dissertation is going to use YOLOv5 with oriented bounding box support to address the tooth detection problem. As a large amount of labeled images is not easily available and due to the tightly packed and tilted nature of teeth, this choice will be more suitable for the task.

2.7 Research Gap Identified

The literature review of previous research works on tooth detection indicates that the application of deep learning dominates the approaches to solve tooth detection problems. However, most of these research works focus on panoramic images and periapical images. While periapical images allow dentists to look into the details of a particular tooth structure, and panoramic images

provide a holistic view to evaluate teeth conditions, these two types of images are usually taken only when a patient has complications demanding further investigation. Moreover, taking panoramic images may require more advanced imaging equipment and have longer exposure to radiation [46].

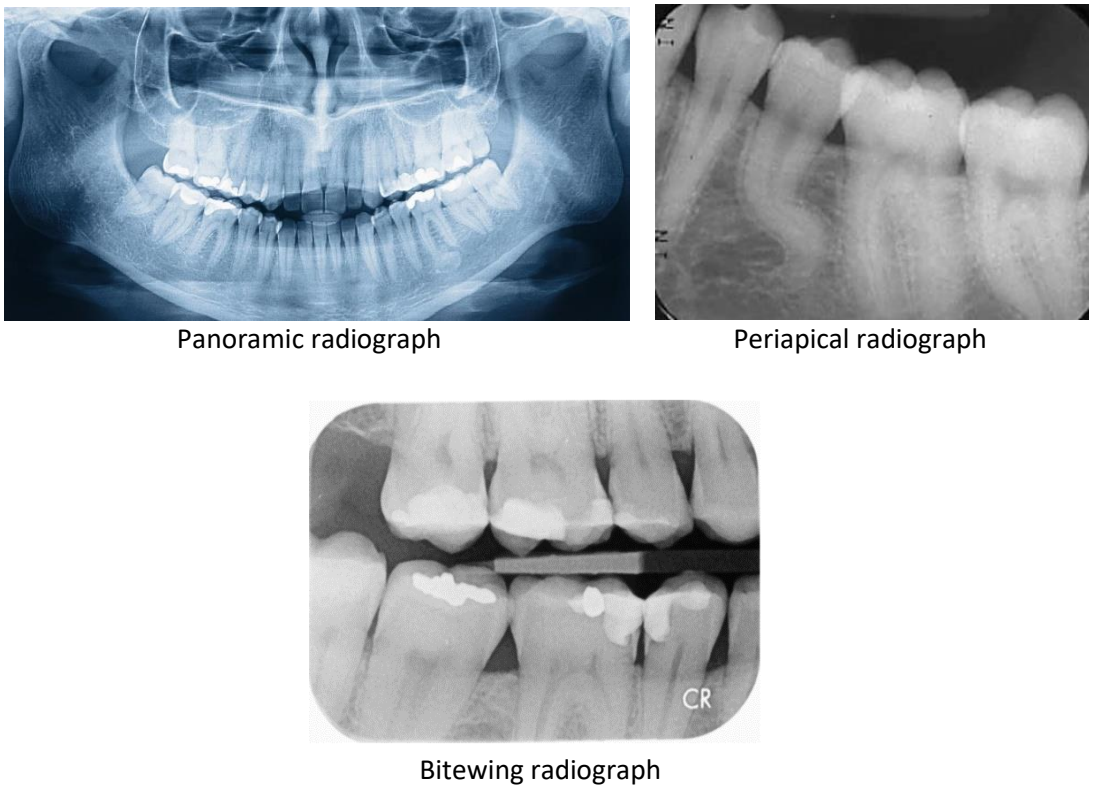


Figure 16 Example of three types of dental radiographs. Image source [1] [46].

On the contrary, bitewing images are extremely common as their capturing procedures are relatively simple. They serve the preventative purposes of detecting early signs of dental caries and examining bone levels [1, 46]. However, bitewing images cannot reveal details of tooth roots and conditions below gum lines. They also do not contain details of each tooth, as a tooth typically cannot be shown in full from its crown to its roots in bitewing images. This imposes a great challenge to deep learning object detection algorithms, as the features available to distinguish individual teeth are not as rich as those from panoramic and periapical images. Figure 16 illustrates examples of the three types of radiographs. In Figure 17, teeth are extracted from bitewing images and displayed individually. The notable differences among them are their shapes and sizes. However, it is difficult to determine their exact tooth numbers by considering

them individually. In Figure 19, their adjacent teeth are also displayed. Hints to tooth numbers can be acquired by comparing the features with their neighbors. However, it is still not certain to tell their exact tooth numbers by merely taking their neighbors into consideration, particularly when the tooth is treated with crown fillings, which may conceal its distinctive features.

Figure 17 Recognizing teeth individually without considering their spatial relationship.

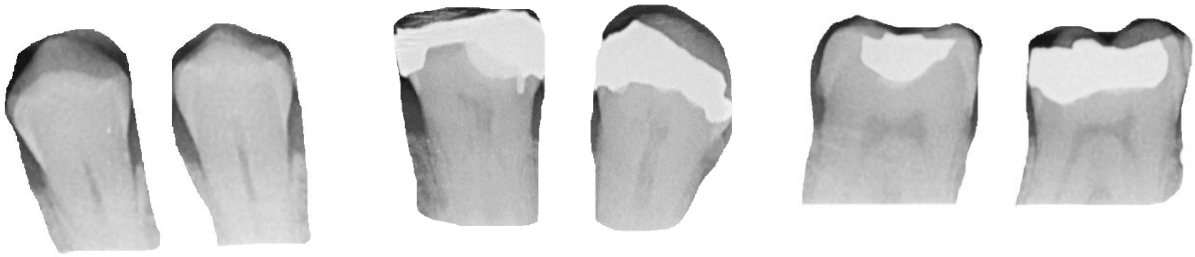


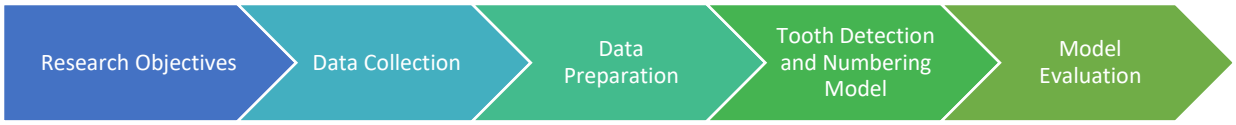
Figure 18 Recognizing teeth by considering their spatial relationship with neighbours.

In general, it is easy to distinguish between molar teeth, which are bigger in shape, and non-molar teeth, which are smaller in shape. However, it is challenging to distinguish between first molars, second molars, and third molars, as they all have similar shapes and root patterns. Similarly, distinguishing premolars, canines, and incisors is also challenging, as they have similar shapes and sizes. This can further be complicated by various treatment patterns concealing individual tooth features. Since deep learning object detection algorithms extract individual tooth features without considering the spatial relationship among teeth, it is expected that adopting such algorithms to bitewing images may not yield promising detection accuracy unless the spatial relationship among teeth can be utilized during the detection process.

In this dissertation, a stage-wise tooth detection approach is proposed with the aim of detecting tooth numbers in bitewing images. Due to the limited number of training samples, the YOLOv5 with oriented bounding box support is adopted to develop the algorithm.

Chapter 3 Research Methodology

This project adopts a 5-stage data analytic life cycle as shown below. While the research objectives have been established in Chapter 1 and Chapter 2, the details of the data collection, data preparation, and building machine learning model are elaborated in this chapter, while model evaluation is to be elaborated in Chapter 4.



3.1 Data Collection Process

A. Preliminary Image Screening

A total of 4,324 bitewing images are provided by the courtesy of the DDD project. A preliminary screening process is conducted to filter out those images with apparent image artifacts, such as incomplete images, misalignment, uneven exposure, or multiple exposures. Figure 19 shows the typical image artifacts identified. After the preliminary screening, 2,818 images are selected and uploaded to Zooniverse workflow.

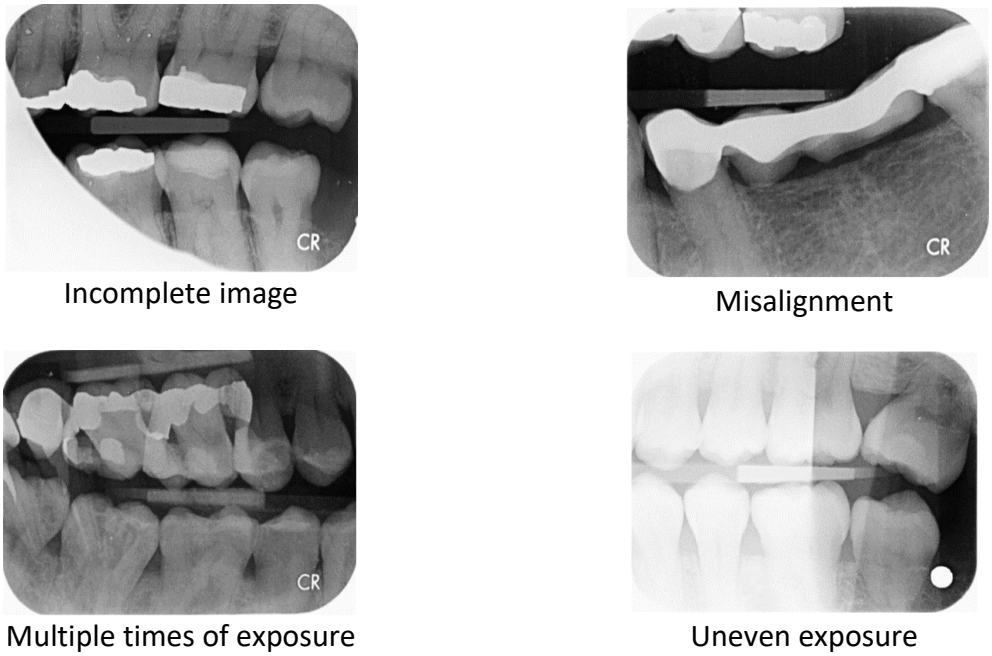


Figure 19 Examples of images with artefacts and excluded from analysis.

B. Zooniverse Crowdsourcing

The 2,818 bitewing images are all without ground-truth annotations. There is no bounding box information for the teeth in the images, nor are there corresponding tooth numbers. As a result, new tooth numbering workflows are developed and hosted under the DDD project in Zooniverse. These new workflows allow volunteers to contribute by annotating ground truth bounding boxes and tooth numbers on the images.

Design of Tooth Numbering Workflows

Internet users visiting the DDD project homepage can directly access two new workflows as shown in Figure 20.

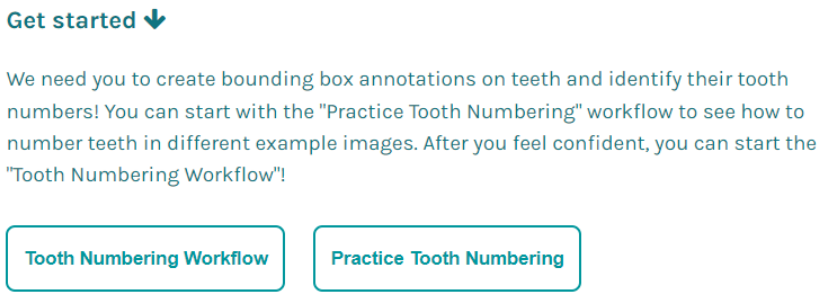


Figure 20 Homepage of the DDD project to access new workflows

The Practice Tooth Numbering workflow illustrates several examples of how to draw bounding boxes and tag tooth numbers, as shown in Figure 21. Volunteers can practice using the bounding box function as well as familiarize themselves with techniques to identify tooth numbers in this workflow. All annotated images in this workflow are not taken into account in this research. The Tooth Numbering Workflow allows volunteers to actually annotate a bounding box around each tooth and input the corresponding tooth number. A total of 2,818 images have been uploaded to the workflow subject set since June 2023. Each image is retired if it is annotated more than 10 times. Figure 22 illustrates the screenshot of Tooth Numbering Workflow.

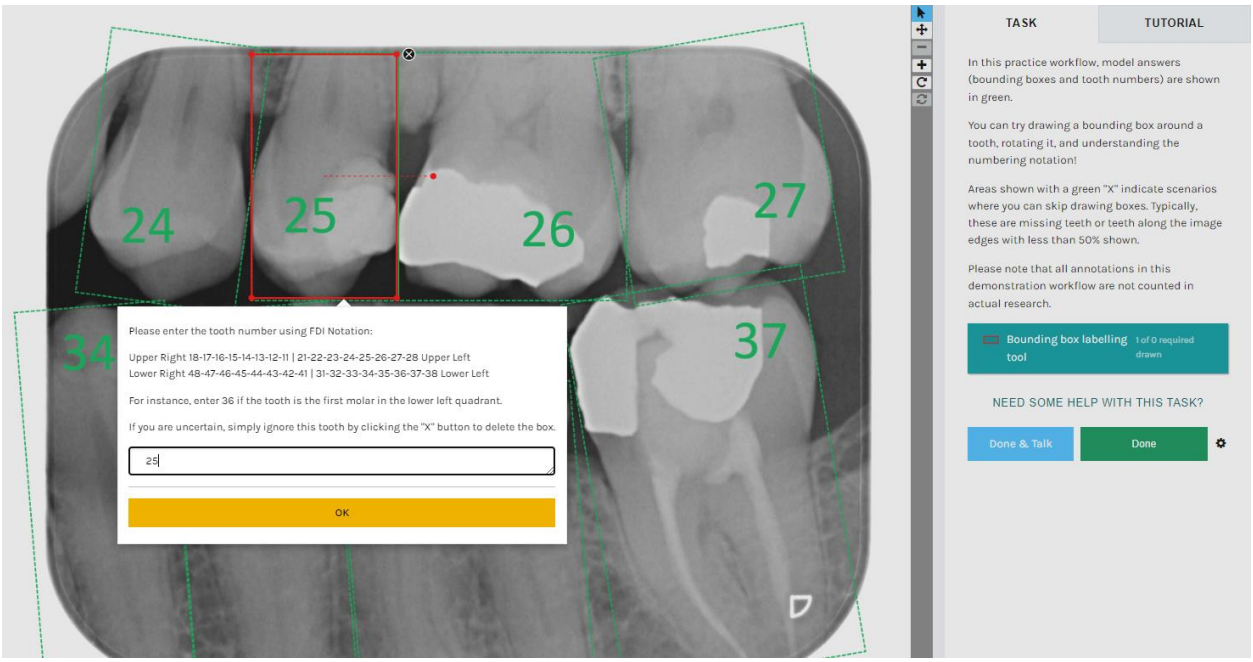


Figure 21 Screenshot of Practice Tooth Numbering Workflow

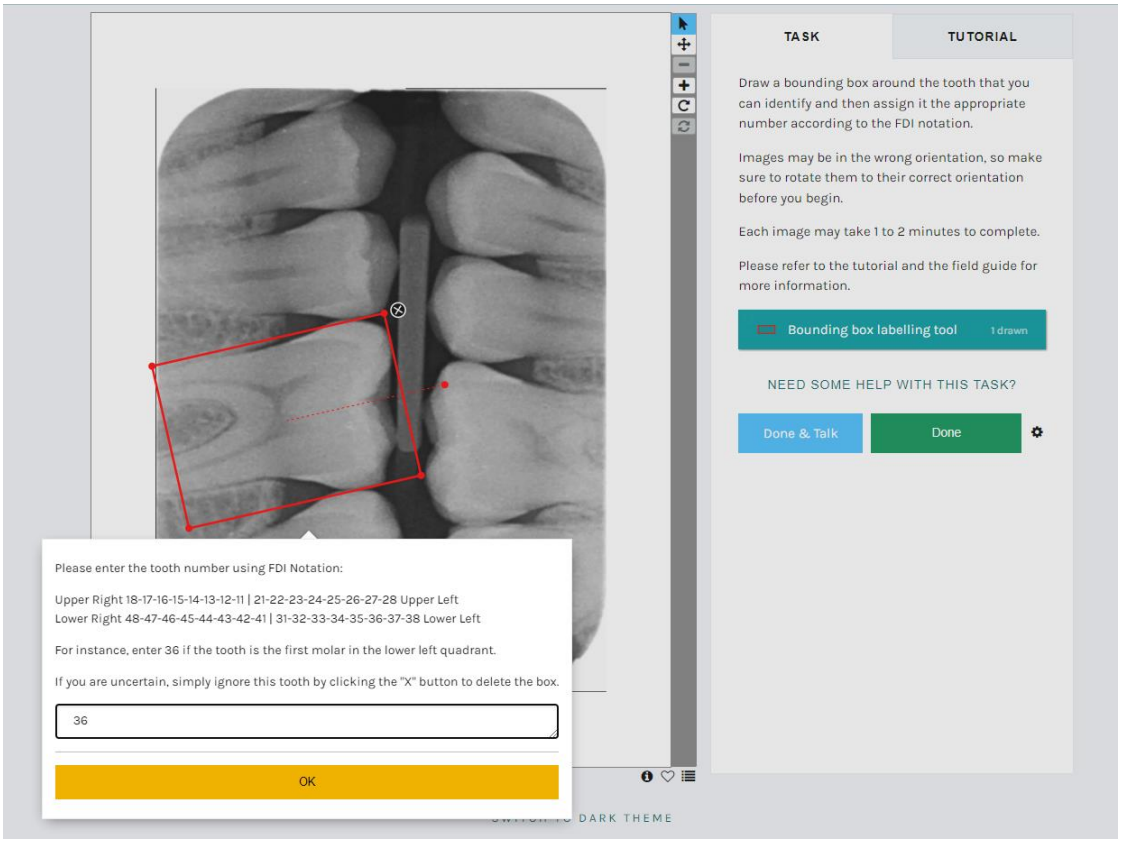


Figure 22 Screenshot of Tooth Numbering Workflow

Design of Supporting Materials

Since volunteers in the Zooniverse are not expected to have prior knowledge in dentistry, supporting materials are prepared to facilitate their contribution. The Field Guide, which is easily accessible from the workflow, contains four topics specifically selected to help volunteers gain sufficient background knowledge to complete the tooth numbering workflow task. Each topic contains concise text and/or captioned and voice-over videos to engage volunteers in the participation process.

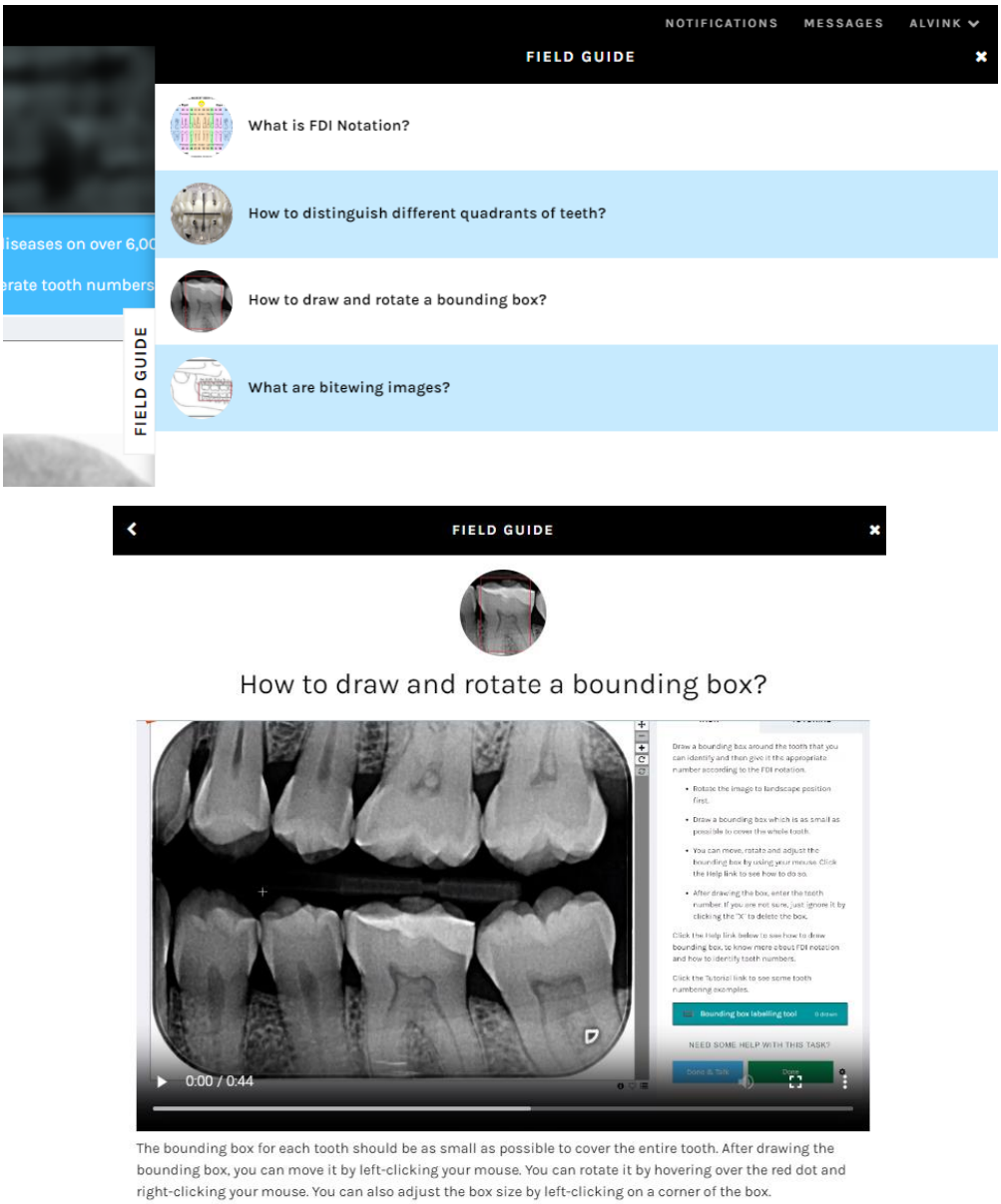


Figure 23 Screenshots of Topical Field Guides

Apart from the topical Field Guides, volunteers can also make reference to the tutorials of the workflows. Instead of elaborating on the background knowledge like the Field Guide does, the tutorials briefly describe the tips and special examples to which volunteers should pay attention. For instance, how to handle missing teeth, broken teeth, and partially displayed teeth.

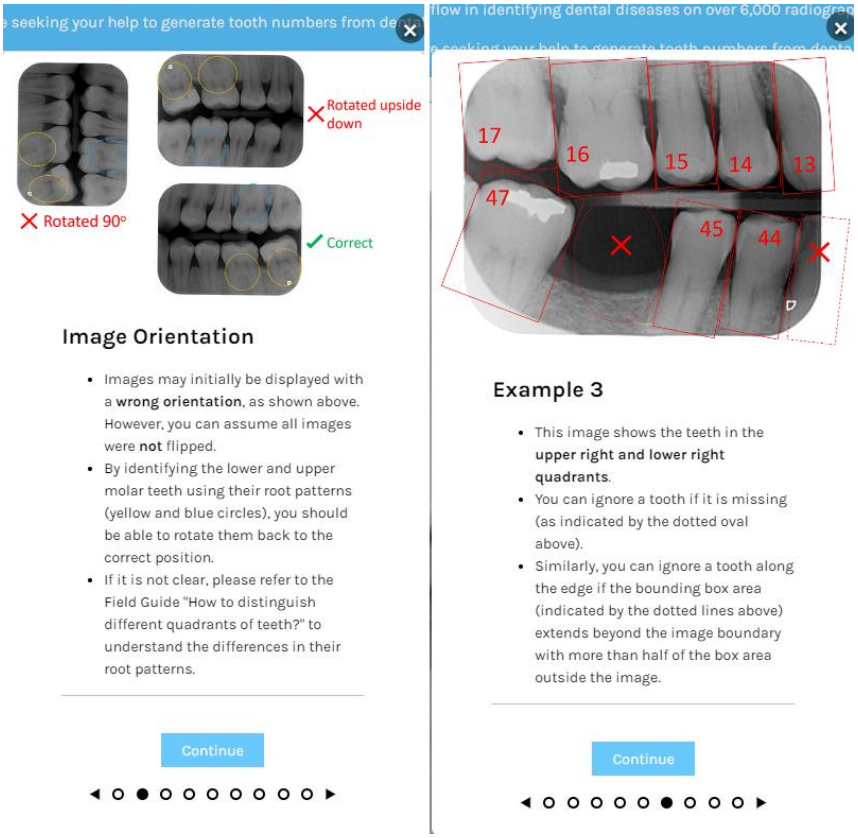


Figure 24 Screenshot of Tooth Numbering Tutorial

Engagement of Volunteers

Since the previous dental disease detection workflows under the DDD project had been completed for a few months, a newsletter announcing the release of tooth numbering workflows is sent to Zooniverse volunteers who have subscribed to the project to appeal for their contribution. In addition, the Zooniverse administrator also reactivates the DDD project, so it is shown in the active Zooniverse project pages.

C. Annotation Criteria

For each image in the workflow, volunteers are asked to tag the tooth number and draw a bounding box for each tooth they can identify. The annotation criteria are as follows:

- The bounding box should be as small as possible and have a proper rotation angle to cover the entire tooth area, starting from its crown to roots area, if its roots can be seen in the image.
- Missing or chipped off teeth can be ignored, and no annotation is required.
- Broken teeth should still be annotated.
- A tooth partially shown along the left/right edges should be annotated only if over half of the tooth has appeared inside the image.

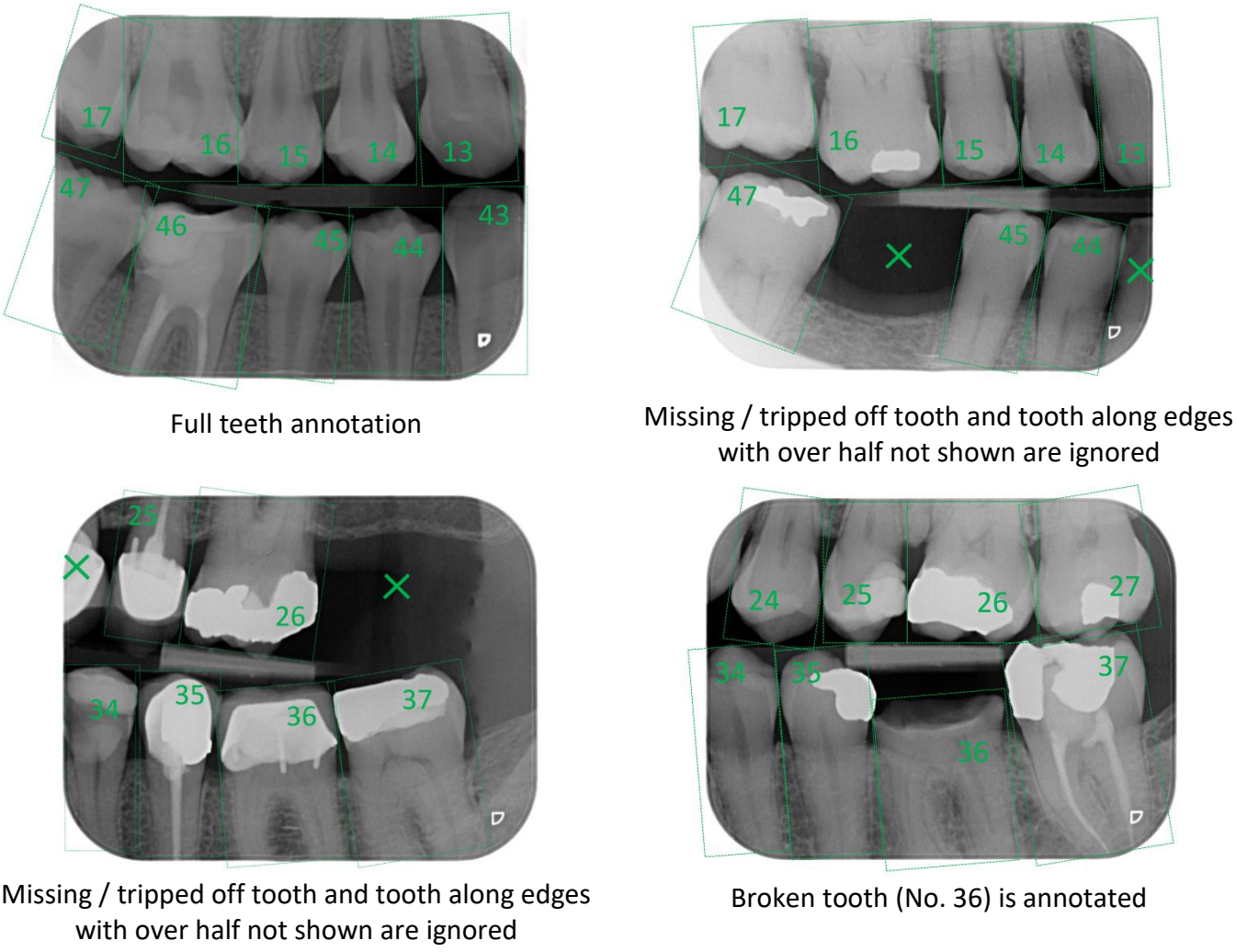


Figure 25 Illustration of Annotation Criteria

3.2 Data Preparation Process

Total 693 annotated images are received from the Zooniverse workflow at the end of writing this dissertation. The annotations undergo the following checking process to ensure their quality.

A. Data Validation and Cleansing

Label Validity Check

All tooth numbers in the annotations are checked using Python scripts to ensure they are valid according to the FDI notation described in Chapter 2.1. Invalid annotations owing to typos are rectified manually.

Quadrant Validity Check

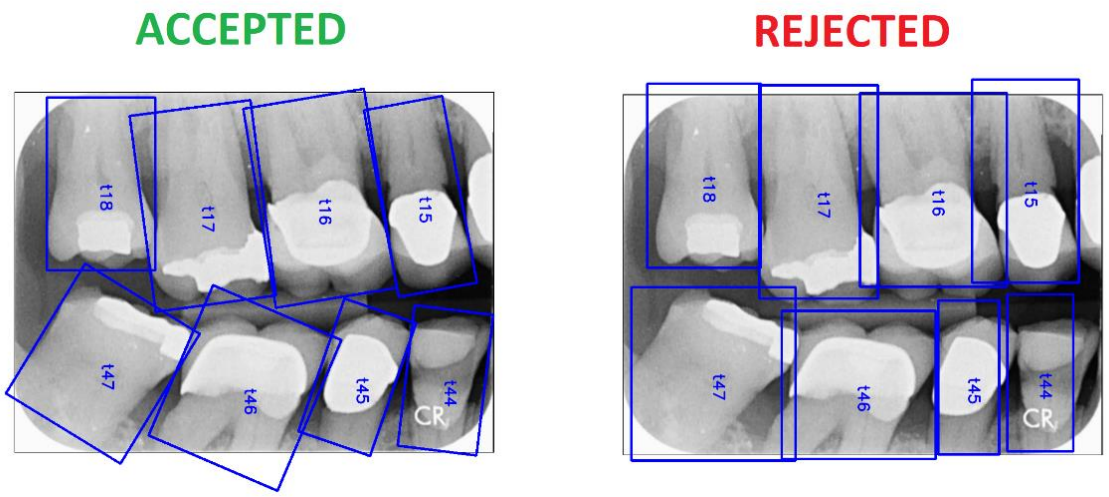
This check ensures tooth numbers appear in a consistent pair of quadrants in bitewing images. For example, tooth numbers in the upper right quadrant should only exist with the lower right quadrant but not the lower left quadrant.

Bounding Box Validity Check

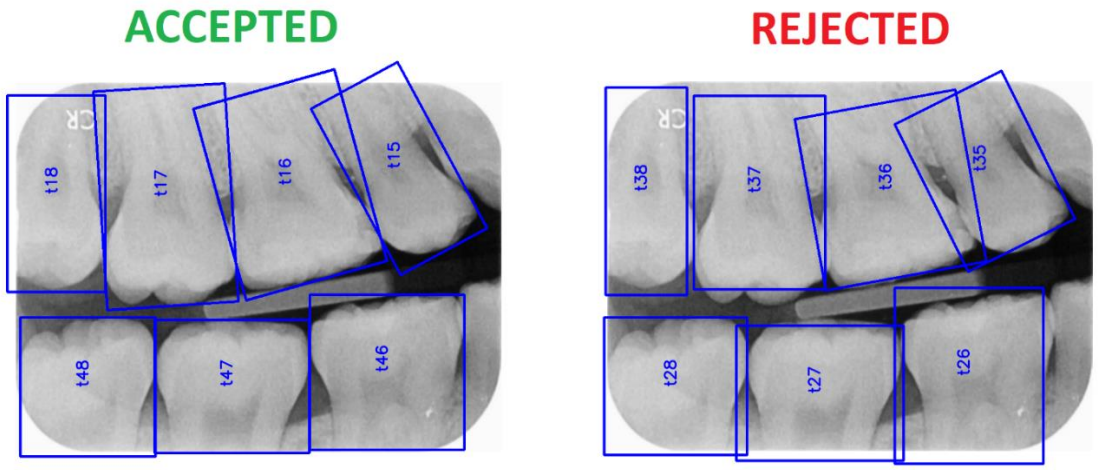
The bounding box coordinates and tooth numbers are plotted on the images. These annotated images are all checked to ensure the bounding boxes are correctly positioned on teeth. In addition, the distances between adjacent teeth are checked. This ensures that the bounding boxes are drawn in reasonable positions without overlapping and without being too far away. All invalid bounding boxes are rectified.

Data Reconciliation

Each image in the workflow is allowed to have 10 annotations before retirement, i.e., images with 10 annotations will be removed from the workflow. For images receiving multiple annotations, they are reviewed by the research team and a dentist according to the annotation criteria set out in section 3.1. Only the best annotation agreed upon by reviewers is selected as the final annotation for that image. Figure 27 illustrates two examples of annotations that are reviewed. In case 1, the volunteers correctly annotate the tooth numbers, but the one with the bounding box located closer to the teeth is accepted. In case 2, the tooth numbers are not consistent among the volunteers. The left one is accepted after reviewing by the research team (some volunteers may have been misled by the “CR” markers appearing near the image corner and mistook it as the bottom right of the actual image).



Case 1: All tooth numbers annotated are consistent.
The one with more accurate bounding box orientation is accepted.



Case 2: Bounding boxes with inconsistent annotations. The left one is accepted after reviewing by research team.

Figure 26 Examples of multiple annotations on the same image and its result of reconciliation. All teeth numbers are prefixed with the letter ‘t’ in the image.

B. Exploratory Data Analysis

Figure 26 illustrated the frequency of occurrence of each tooth number in the annotations. The incisor teeth and canine teeth (number t_{q1}, t_{q2}, t_{q3} and t_{q4} where q is the quadrant number) are the teeth least found in bitewing images. As a result, these tooth numbers are not considered in the model.

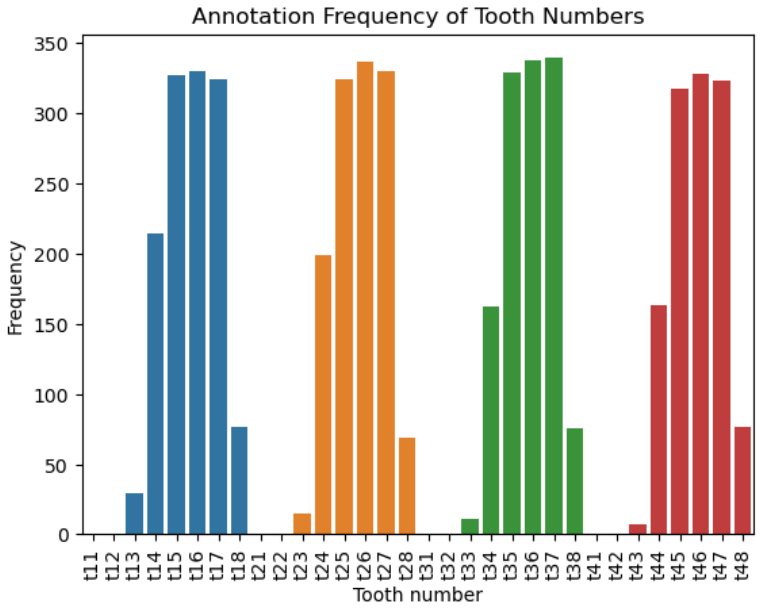


Figure 27 Frequency of tooth numbers appeared in annotations

3.3 Tooth Detection and Numbering Model

In this dissertation, a stage-wise detection model is proposed. It is inspired by the models proposed in [11], [12] and [41], where a combination of stage-wise detectors is used to refine the tooth numbering mechanism. Figure 28 illustrates the stage-wise detection model. The approach is intuitive and resembles the human decision-making process when we are required to annotate a bitewing image. As shown in the next chapter, it is found that this approach gives improved detection performance in comparison to directly training the YOLO object detector to classify 32 tooth classes, as individual tooth features are not prominent in bitewing images.

Figure 28 Stage-wise detection model in this dissertation

A. Stage 1 – Left / Right Classifier

When given a bitewing image and asked to tag its tooth numbers, the first question we should ask is whether the image refers to the left or right side of the patient. In the same token, the first stage of the model is to answer the same question. Since this classification is rather straightforward and a number of pre-trained CNNs are available, it is implemented using the popular ResNet50 [26] by following the transfer learning approach in [50] to load the pre-trained ResNet50 backbone and replace the last linear classification layer with our model having only 2 output classes (Left or Right). The learning rate is determined empirically using the learning rate finder approach [51]. The batch size of training is set to 5. Adam [52] is adopted as the optimizer. The training is conducted using one NVIDIA Quadro P4000 8Gb GPU. During the training, image augmentation by random crop and random rotation up to 180 degrees is adopted. No random flip is applied as the aim of this classifier is to determine left or right.

Since the image annotations are with tooth numbers, a new set of image labels is generated by reducing the tooth numbers into left class and right class as follows:

$$Image = \begin{cases} Right, if\ all\ teeth\ in\ Image\ \in \{t_{11,}t_{12}, \dots, t_{18}\} \cup \{t_{41,}t_{42}, \dots, t_{48}\} \\ Left, if\ all\ teeth\ in\ Image\ \in \{t_{21,}t_{22}, \dots, t_{28}\} \cup \{t_{31,}t_{32}, \dots, t_{38}\} \\ where\ t_{qk}\ is\ the\ k-th\ tooth\ in\ quadrant\ q \end{cases}$$

B. Stage 2 – Tooth Detection and Molar / Non-molar Classifier

In the previous stage, we determine whether the image comes from the left or right side of the mouth. Once we know the side, the next question is to determine which quadrant is the upper and which is the lower. Additionally, regardless of upper or lower quadrants, the most distinctive feature is the difference in sizes between molar teeth and non-molar teeth. We need to identify the molars and non-molars to start determining the tooth numbers. The Stage 2 detector is going to answer these questions. The YOLOv5 model with oriented bounding box support is adopted as the object detector serving two purposes. The first purpose is to detect the bounding box for each tooth in the image. The second purpose is to classify each tooth bounding box into four classes: upper molar, upper non-molar, lower molar, and lower non-molar. The classification result allows us to determine the locations of the upper and lower quadrants and the positions of the molar and non-molar teeth. A new set of tooth class labels is required, reducing the tooth numbers into these four classes as follows:

$$Tooth\ Class = \begin{cases} Upper\ Molar\ if\ Tooth\ Number \in \{t_{16}, t_{17}, t_{18}\} \cup \{t_{26}, t_{27}, t_{28}\} \\ Upper\ Nonmolar\ if\ Tooth\ Number \in \{t_{11,}t_{12}, \dots, t_{15}\} \cup \{t_{21,}t_{22}, \dots, t_{25}\} \\ Lower\ Molar\ if\ Tooth\ Number \in \{t_{36}, t_{37}, t_{38}\} \cup \{t_{46}, t_{47}, t_{48}\} \\ Lower\ Nonmolar\ if\ Tooth\ Number \in \{t_{31,}t_{32}, \dots, t_{35}\} \cup \{t_{41,}t_{42}, \dots, t_{45}\} \\ where\ t_{qk}\ is\ the\ k-th\ tooth\ in\ quadrant\ q \end{cases}$$

The YOLOv5 model with oriented bounding box support is used to implement this detector. Since YOLOv5 has over 30 hyperparameters, the default parameters set out in [37] are used. Additionally, since our bitewing images are grayscale and consist of teeth, which are significantly different from the object images YOLOv5 is pre-trained on, we will train the YOLOv5 model from scratch for 300 epochs. Being an efficient one-stage detector for real-time object detection, the YOLOv5 is available in different model sizes supporting different speeds of detection in terms of frame per seconds (FPS). A larger model size can give higher accuracy but requires longer training time and detection time [53]. Although the tooth detection in dental radiography is not a real-time application, a number of experiments are conducted to evaluate which model size, YOLOv5n (nano size with 213 layers and 2.0 million parameters), YOLOv5m

(medium size with 290 layers and 21.6 million parameters) and YOLOv5x (extra large size with 444 layers and 87.4 million parameters), can offer better performance. The performance of the object detection is evaluated using the following metrics:

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

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

$$F1\ Score\ (F1) = \frac{2PR}{P + R}$$

$$Intersection\ over\ Union\ (IoU) = \frac{Amount\ of\ area\ overlap\ of\ two\ boxes}{Union\ of\ area\ of\ two\ boxes}$$

$$Mean\ Average\ Precision\ (mAP) = \sum_{i=1}^{N} AP_i$$

TP, *FP* and *FN* are true positive, false positive and false negative respectively. AP_i is the Average Precision to detect and recognize the *i*-th class in the object detection problem. Average Precision is obtained from the area under P-R curve. The criterion measuring whether the detector can detect an object, i.e. the "objectness" score, is measured by the evaluating whether IoU of the predicted box and the ground truth box is over certain threshold (e.g. 0.5). If the IoU is above the threshold, then the object is detected (positive) and vice versa. The mAP@0.5 indicates the IoU threshold is set at 0.5 when measuring the P-R curve, while mAP@0.5:0.95 is the average of mAP by varying the IoU threshold over a range of values from 0.5 to 0.95 to measure a set of PR curves.

C. Stage 3 – Tooth Distances Classifier

In the classification outcomes from the previous two stages, the search space of tooth numbers is effectively narrowed down. Given the locations of molar and non-molar teeth, one can now try to estimate their tooth numbers by considering the natural tooth sequence, tooth sizes, and their relative positions. In this stage, this heuristic estimation is implemented using a number of Random Forest (RF) or Support Vector Machine (SVM) classifiers using the Scikit-Learn module [54]. Each RF/SVM classifier accepts an input vector in the following form: $v_{ij} =$

(W_i, W_j, D_{ij}), where W_i and W_j are the widths of the bounding boxes of tooth t_i and tooth t_j, D_{ij} is Euclidean distance between the centres of these bounding boxes. W_i, W_j and D_{ij} are all normalized with respect to the diagonal of the input image. Unlike the previous stage, here we denote the i-th tooth as t_i by dropping the quadrant subscript to simplify the notation without affecting the interpretation.

Since the molar and non-molar teeth in the same quadrant have been determined in Stage 2, there are six classifiers required to classify tooth pairs in the same quadrant, as summarized in Table 1.

Classifier	**Quadrant**	**Classify**	***Tooth Pairs (t_i, t_j)**
$\varphi(q, v_{ij})$	$q = Upper$	Non-molar teeth $t_i \in \{3,4,5\}$ Molar teeth $t_j \in \{6,7,8\}$	(t_3,t_6) (t_3,t_7) (t_3,t_8) (t_4,t_6) (t_4,t_7) (t_4,t_8) (t_5,t_6) (t_5,t_7) (t_5,t_8)
	$q = Upper$	Molar teeth $t_i \in \{6,7,8\}$ Molar teeth $t_j \in \{6,7,8\}$ $t_i \neq t_j$	(t_6,t_7) (t_6,t_8) (t_7,t_8)
	$q = Upper$	Non-molar teeth $t_i \in \{3,4,5\}$ Non-molar teeth $t_j \in \{3,4,5\}$ $t_i \neq t_j$	(t_3,t_4) (t_3,t_5) (t_4,t_5)
	$q = Lower$	Non-molar teeth $t_i \in \{3,4,5\}$ Molar teeth $t_j \in \{6,7,8\}$	(t_3,t_6) (t_3,t_7) (t_3,t_8) (t_4,t_6) (t_4,t_7) (t_4,t_8) (t_5,t_6) (t_5,t_7) (t_5,t_8)
	$q = Lower$	Molar teeth $t_i \in \{6,7,8\}$ Molar teeth $t_j \in \{6,7,8\}$ $t_i \neq t_j$	(t_6,t_7) (t_6,t_8) (t_7,t_8)
	$q = Lower$	Non-molar teeth $t_i \in \{3,4,5\}$ Non-molar teeth $t_j \in \{3,4,5\}$ $t_i \neq t_j$	(t_3,t_4) (t_3,t_5) (t_4,t_5)
* Tooth number 1 and 2 are ignored as they do not appear in the training data and $i, j = 3,4,\ldots,8$			

Table 1 Definition of classifiers to classify tooth pairs in the same quadrant. The input feature vector $v_{ij} = (W_i, W_j, D_{ij})$, where W_i, W_j are the widths of the bounding boxes of t_i tooth and t_j in the same quadrant, D_{ij} is distance between the centres of these bounding boxes. The output is the tooth pair label (t_i, t_j).

Given a molar and non-molar tooth pair, the $\varphi(q, v_{ij})$ is used to classify and identify the tooth numbers given they are known as molar or non-molar from Stage 2. For example, if we have an unknown non-molar tooth t_i in the upper quadrant, an unknown molar tooth t_j in the upper quadrant, the width (W_i) of bounding box t_i and the width (W_j) of bounding box t_j, and the distance D_{ij} between them. Then we input the feature vector $v_{ij} = (W_i, W_j, D_{i,j})$ to $\varphi(Upper, v_{ij})$ to get the most probable output tooth pair ($t_{i'}$,$t_{j'}$). In this way, we can have the most probable prediction of tooth numbers i' and j'. Once we identify a molar and a non-molar tooth, we can then use $\varphi(Upper, v_{ij})$ to identify the remaining teeth given we have already known a molar and a non-molar in the same quadrant.

However, if the prediction from Stage 2 fails to predict a molar and non-molar tooth in the same quadrant, we will face an ambiguity in the prediction of $\varphi(q, v_{ij})$ as we don't have the "prior knowledge". For example, Stage 2 only predicts there are two molars in the upper quadrant, and $\varphi(Upper, v_{ij})$ is used to identify the most probable molar pair as (t_7,t_8). However, we cannot further determine which one is t_7 and which one is t_8. As a result, another set of classifiers based on opposite quadrant teeth is required. By "opposite quadrant" we mean that quadrant 2 is opposite to quadrant 3 and quadrant 1 is opposite to quadrant 4.

Classifier	**Quadrant**	**Classify**	***Tooth Pairs (t_i, t_j)**
$\chi(q, v_{ij})$	$q = Upper$	A tooth $t_i \in \{3,4,\dots,8\}$ in upper quadrant. A tooth $t_j \in \{3,4,\dots,8\}$ in lower quadrant.	t_i is a tooth in upper quadrant t_j is a tooth in lower quadrant
	$q = Lower$	A tooth $t_i \in \{3,4,\dots,8\}$ in lower quadrant. A tooth $t_j \in \{3,4,\dots,8\}$ in upper quadrant.	t_i is a tooth in lower quadrant t_j is a tooth in upper quadrant
* Tooth number 1 and 2 are ignored as they do not appear in the training data and $i, j = 3,4,\dots,8$			

Table 2 - Definition of classifiers to classify tooth pairs in the opposite quadrant. The input feature vector consists of $v_{ij} = (W_i, W_j, D_{ij})$, where W_i, W_j are the widths of the bounding boxes of *i*-th tooth and *j*-th tooth in the opposite quadrants, D_{ij} is distance between the centres of these bounding boxes. The output is the tooth pair label (t_i, t_j).

There are total seventy two $\chi(q, v_{ij})$ classifiers for opposite quadrant tooth pairs. For example, given t_5 in the upper quadrant, and unknown t_j in the lower quadrant, we will compute the width (W_5) of bounding box t_5, the width (W_j) of bounding box t_j, and the distance D_{5j} between them. Then we feed the feature vector $v_{5j} = (W_5, W_j, D_{5,j})$ to $\chi(Upper, v_{5j})$ to get the classification output tooth pair (t_5,$t_{j'}$). In this way, we can have the most probable prediction of tooth number j' in lower quadrant.

The algorithm to apply the above classifiers are summarized in Process A, Process B, Process C1, Process C2, Process D and Process E below.

Process A – Identify tooth numbers of a non-molar and a molar teeth in the same quadrant which is closest together

Input:

$\boldsymbol{X_q}$ is a set of unidentified non-molar teeth t_i in quadrant q

$\boldsymbol{Y_q}$ is a set of unidentified molar teeth t_j in quadrant q. Both t_i and t_j in the same quadrant q where $q \in \{Upper, Lower\}$.

Output:

Identified non-molar tooth number i_q^* and molar tooth number j_q^* which are in the closest proximity and which maximize the probability of $\varphi(q, v_{ij})$ for quadrant q

Begin

for each *q* in *{Upper, Lower}*

Select i_{min}, j_{min} where $D_{i_{\min}\ \ j_{min}} = \min_{i,j}(D_{ij})$ for $t_i \in X_q$, $t_j \in Y_q$

Compute feature vector $v = (W_{i_{min}}, W_{j_{min}}, D_{i_{\min}\ \ j_{min}})$

Obtain i_q^* and j_q^* which gives the maximum probability output of $\varphi(q, v)$ from input vector v

end for

End

Process B – Identify tooth numbers in the opposite quadrant with prior knowledge

Input:

$\boldsymbol{X_q}$ is a set of unidentified teeth in quadrant q where $q \in \{Upper, Lower\}$.

$\boldsymbol{X_{\tilde{q}}}$ is a set of identified tooth numbers from Process A or Process D in quadrant $\tilde{q}$ where $\tilde{q} \in \{Upper, Lower\}$ is the opposite quadrant of q, i.e. $q \neq \tilde{q}$

Output:

$\boldsymbol{Y_q}$ is a set of identified teeth in quadrant q.

Begin

for each *q* in *{Upper,Lower}*

for each tooth t_i in X_q

Compute k_{min} where $D_{i\ k_{min}} = \min_k(D_{ik})$ for $t_k \in X_{\tilde{q}}$

Compute feature vector $v = (W_i, W_{k_{min}}, D_{i\,k_{min}})$

Obtain i_q^* which gives the maximum probability output of $\chi(q, v)$ from input vector v

Remove $t_{i_q^*}$ from X_q

Insert $t_{i_q^*}$ from Y_q

end for

end for

End

Process C1 – Identify non-molar tooth numbers in the same quadrant with prior knowledge

Input:

Identified non-molar tooth number i_q^* from Process A or Process B. $t_{i_q^*}$ is in quadrant q where $q \in \{Upper, Lower\}$.

$\boldsymbol{X_q}$ is a set of unidentified non-molar teeth in quadrant *q*.

Output:

$\boldsymbol{Y_q}$ is a set of identified non-molar teeth in quadrant *q*.

Begin

for each *q* in *{Upper, Lower}*

Initialize $i \leftarrow i_q^*$

for each non-molar tooth t_k in X_q where $k \neq i$

Compute k_{min} where $D_{k_{\min}\ i} = \min_k(D_{ki})$

Compute feature vector $v = (W_{k_{min}}, W_i, D_{k_{\min}\ i})$

Get k_q^* which gives the maximum probability output of $\varphi(q, v)$ from input vector v

while $t_{k_q^*}$ exists in Y_q

get next k_q^* which gives the next high probability output of $\varphi(q, v)$ from input vector v

end while

Update $i \leftarrow k_q^*$

Remove $t_{k_q^*}$ from X_q

Insert $t_{k_q^*}$ from Y_q

end for

end for

End

Process C2 – Identify molar tooth numbers in the same quadrant with prior knowledge

Input:

Identified molar tooth number j_q^* from Process A or Process B. $t_{j_q^*}$ is in quadrant q where $q \in \{Upper, Lower\}$.

$\boldsymbol{X_q}$ is a set of unidentified molar teeth in quadrant *q*.

Output:

$\boldsymbol{Y_q}$ is a set of identified molar teeth in quadrant *q*.

Begin

for each *q* in *{Upper, Lower}*

Initialize $j \leftarrow j_q^*$

for each molar tooth $\boldsymbol{t_k}$ in $\boldsymbol{X_q}$ where $\boldsymbol{k \neq j}$

Compute k_{min} where $D_{k_{\min}\ j} = \min_k(D_{kj})$

Compute feature vector $v = \left(W_{k_{min}}, W_j, D_{k_{\min}\ j}\right)$

Get k_q^* which gives the maximum probability output of $\varphi(q, v)$ from input vector v

while $\boldsymbol{t_{k_q^*}}$ exists in $\boldsymbol{Y_q}$

get next k_q^* which gives the next high probability output of $\varphi(q, v)$ from input vector v

end while

Update $j \leftarrow k_q^*$

Remove $t_{k_q^*}$ from X_q

Insert $t_{k_q^*}$ from Y_q

end for

end for

End

Process D – Identify tooth numbers without prior knowledge

Input:

$\boldsymbol{X_q}$ is a set of unidentified teeth in quadrant q where $q \in \{Upper, Lower\}$.

Output:

$\boldsymbol{Y_q}$ is a set of identified teeth in quadrant *q*.

Begin

for each *q* in *{Upper, Lower}*

Initialize j by randomly select t_j from X_q

Remove t_j from X_q

if X_q is not empty

while X_q is not empty

Select i_{min}, j where $D_{i_{min}\,j} = \min_i(D_{ij})$ for $t_i \in X_q$

Compute feature vector $v = (W_{i_{min}}, W_j, D_{i_{min}\,j})$

Obtain i_q^* which gives the maximum probability output of $\varphi(q, v)$ from input vector v

Remove $t_{i_q^*}$ from X_q

Insert $t_{i_q^*}$ and t_j to Y_q

Update $j \leftarrow i_q^*$

end while

else

Execute Process B in order to identify t_j

end if

end for

End

Process E – Main Program Flow

Input:

X is a set of unidentified teeth

Output:

Y is a set of identified teeth

Begin

Execute Process A to obtain tooth numbers i_q^* and j_q^* for all quadrants $q \in \{Upper, Lower\}$

 # For loop-1 to identify all molar or non-molar teeth if prior knowledge i_q^* and j_q^* is obtained from Process A

for each *q* in *{Upper, Lower}*

if (i_q^*, j_q^*) exists

Execute Process C1 to obtain all identified non-molar teeth in q

Execute Process C2 to obtain all identified molar teeth in q

Update *X* by removing the identified teeth

Update *Y* by adding the identified teeth

end if

end for

 # For loop-2 to identify all molar or non-molar teeth if prior knowledge is partially obtained from Process A by using opposite quadrant hints

for each pair $q, \tilde{q}$ in *{Upper, Lower}* where $\tilde{q}$ is opposite to q

if (i_q^*, j_q^*) or ($i_{\tilde{q}}^*, j_{\tilde{q}}^*$) exists

Execute Process B

Update *X* by removing the identified teeth

Update *Y* by adding the identified teeth

end if

end for

 # For loop-3 to identify all molar or non-molar teeth if no prior knowledge is obtained at all

for each *q* in *{Upper, Lower}*

if (i_q^*, j_q^*) not exists

Execute Process D to obtain i_q^* and j_q^*

Update *X* by removing the identified teeth

Update *Y* by adding the identified teeth

end if

end for

End

The main programmatic flow of the decision-making process is described in Process E. Figure 29 illustrates the decision-making process starting from finding the closest pair of non-molar and molar teeth in the same quadrant using Process A. These identified non-molar and molar teeth become the prior information for Process C1/C2 and Process B to identify the remaining teeth.

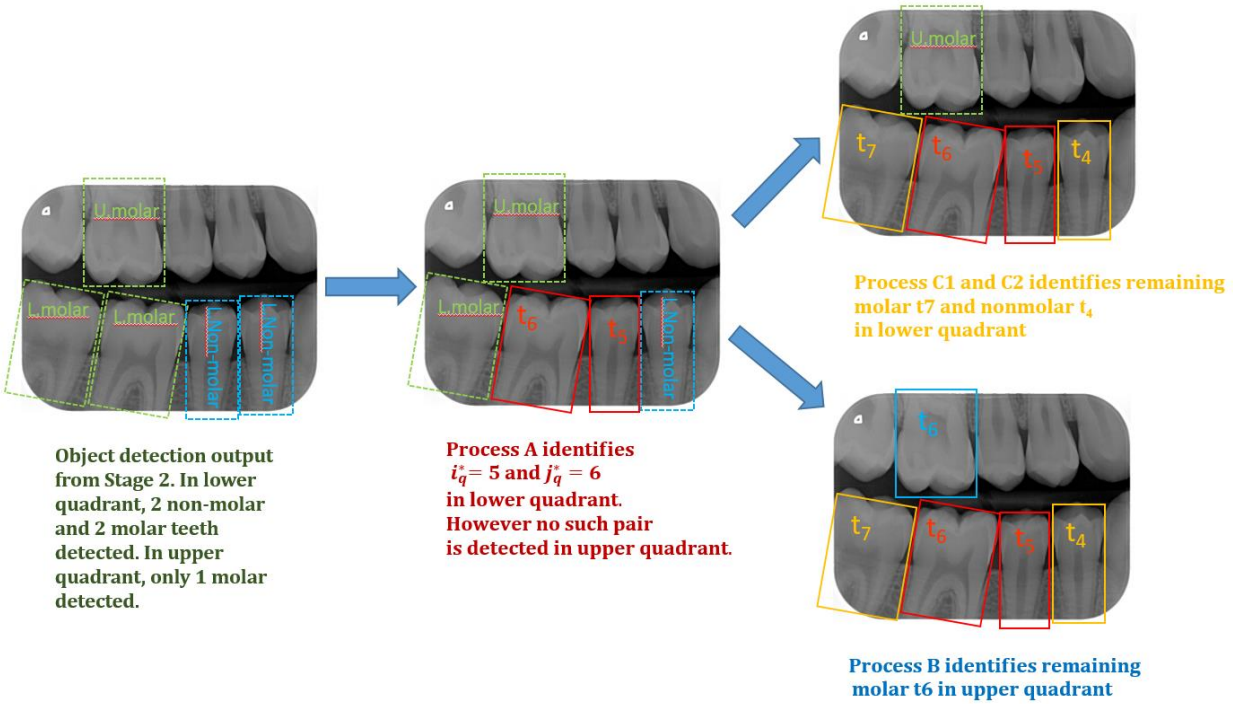


Figure 29 Illustration of Stage 3 decision making process

Figure 30 illustrates a situation when the object detection output from Stage 2 is not good enough, as no non-molar and molar teeth are detected in the same quadrant. In this case, the Stage 3 classifiers try to identify the two molar teeth in the lower quadrant based on their widths and distance. The classification may not be as accurate as identifying a molar and non-molar pair in Figure 29, as the features of two molars are generally similar to each other. After attempting to identify the two molar teeth in the lower quadrant, the process moves on to identify the solo non-molar in the upper (opposite) quadrant, given the prior information (t_6 and t_7) in lower quadrant has been identified, despite the fact that the prior may not be accurate and may lead to a wrong classification in the upper quadrant.

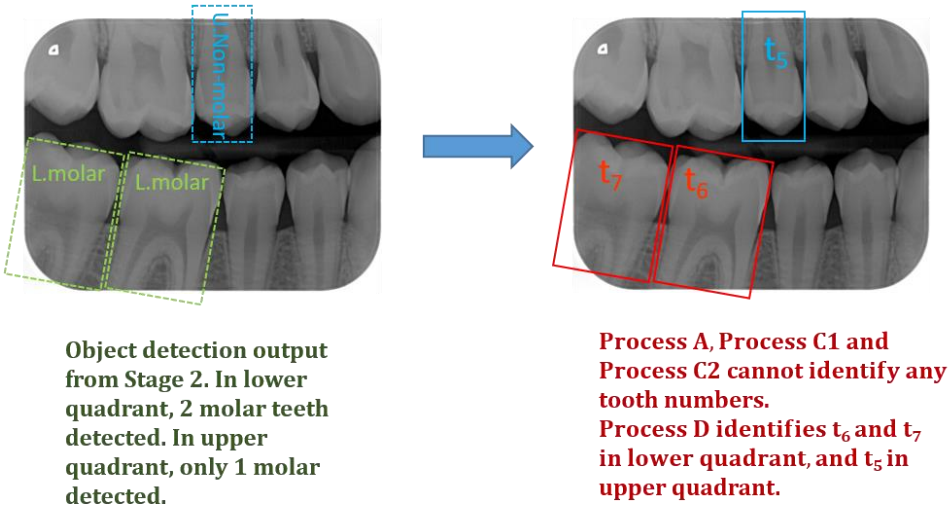


Figure 30 Illustration of Stage 3 decision making process when Stage 2 output is not optimal

The classifiers in Stage 3 are statistically based on tooth widths and the distance between them, which allows the prediction of missing teeth, as shown in Figure 31. However, this approach has limitations when only one tooth is detected in each of the upper and lower quadrants, as illustrated in Figure 32. Since the classifiers require features computed using a pair of teeth in the same quadrant, it is not possible to make a prediction if only one tooth is detected in each of the upper and lower quadrants. Therefore, the minimal case in which a prediction in Stage 3 can be made is when at least 2 teeth are detected in the same quadrant.

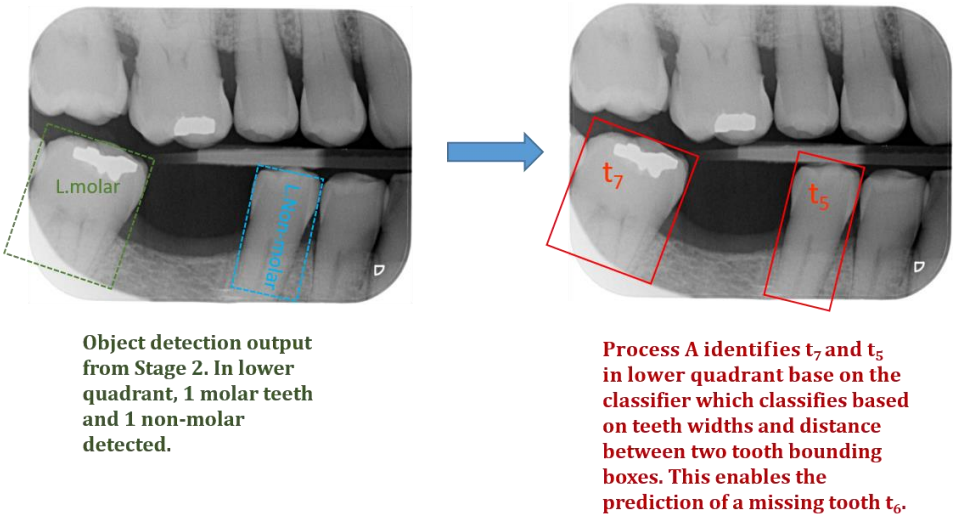


Figure 31 Illustration of Stage 3 decision making process when there is a missing tooth.

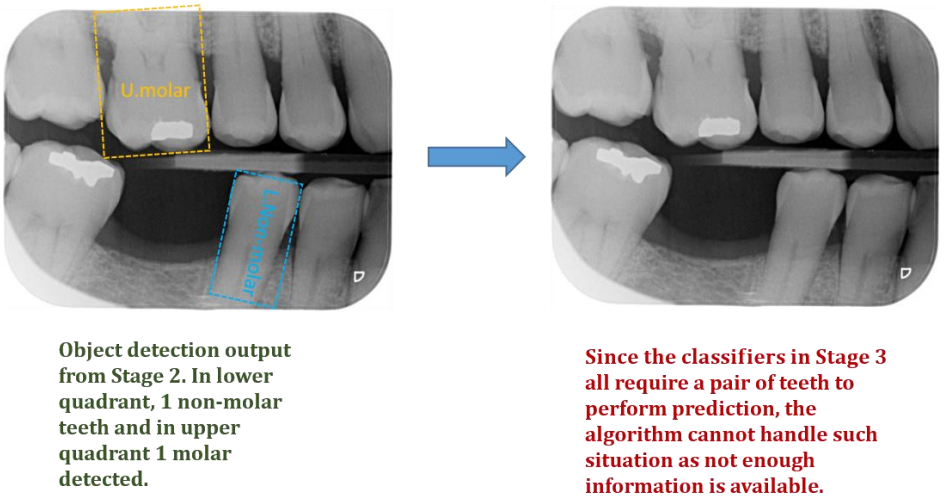


Figure 32 Illustration of Stage 3 decision making process when there are not enough teeth detected in each quadrant

D. Stage 4 – Combination of Results from Stage 1 to 3

In this final stage, the results from the previous stages will be combined to deduce the actual tooth numbers in 32 classes. The combination process is illustrated in Figure 33.

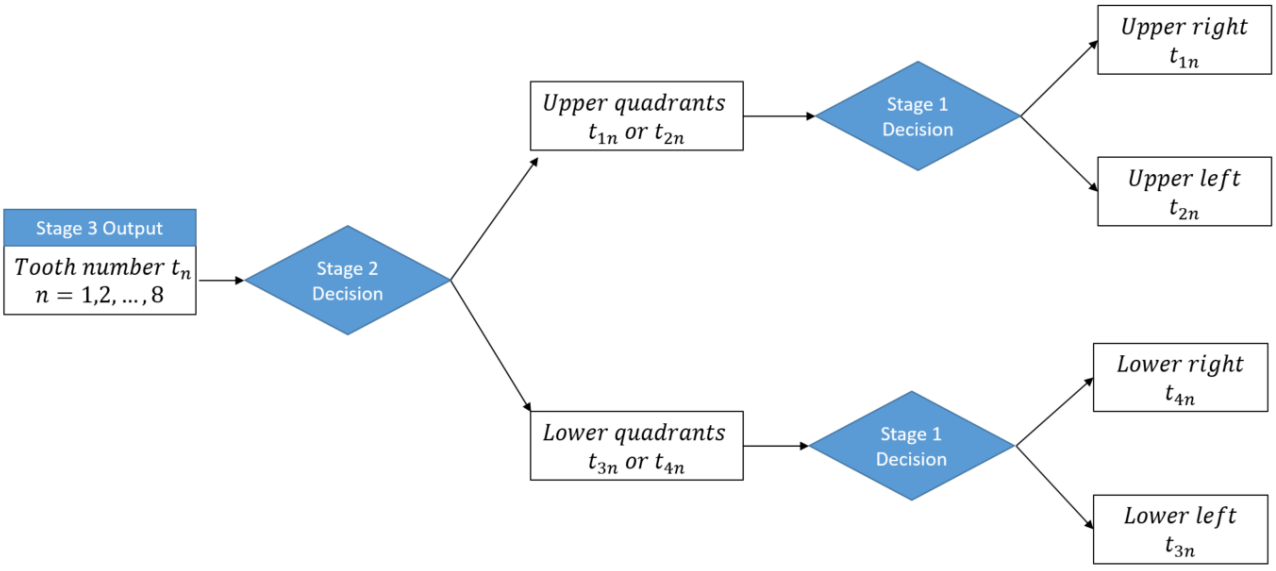


Figure 33 Illustration of combination process in Stage 4 to deduce 32 tooth numbers

Chapter 4 Presentations and Analysis Results

In this chapter, the analysis results of the approach described in Chapter 3 are presented and discussed. The stage-wise decision-making approach implies that the accuracy of the final results will be critically affected from the beginning of the decision-making process. A wrong decision in the first stage, for example, mistaking the left and right side, can lead to all subsequent tooth numbers being incorrect. Therefore, the results of each stage are evaluated, and the parameters are fine-tuned to ensure the best accuracy can be obtained at each stage.

4.1 Data Description

As of 31 July 2023, total 693 out of 2,818 bitewing images in the Zooniverse workflow are annotated. All annotations are checked through the process described in chapter 3. Figure 34 illustrates the frequency of occurrence of each tooth number in the annotations. The incisor teeth and canine teeth (number t_{q1}, t_{q2}) are not found in the annotation dataset, so these tooth numbers are not considered in the model. The distribution of quadrant sides of the image is also shown, and it is balanced between the left and right sides.

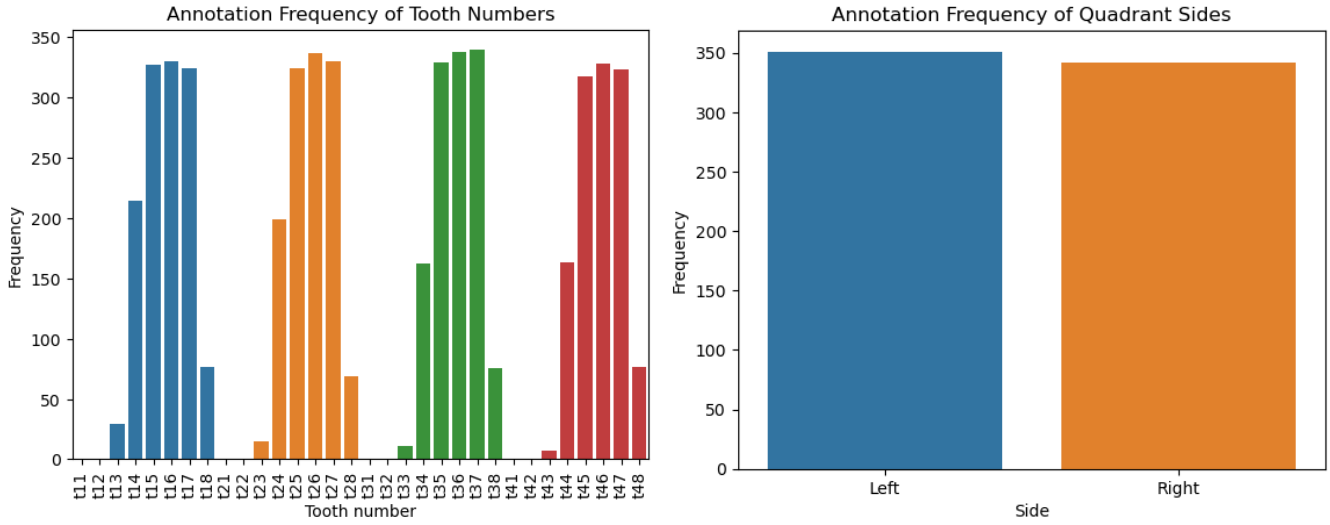


Figure 34 Annotation frequency of tooth numbers

To ensure that the bounding boxes are properly annotated in the images, the distribution of normalized distances between bounding boxes is checked, as shown in Figure 35. To simplify

the presentation, it is assumed that the left and right sides are symmetric, and thus only the distances between adjacent tooth pairs in the upper and lower quadrants are plotted.

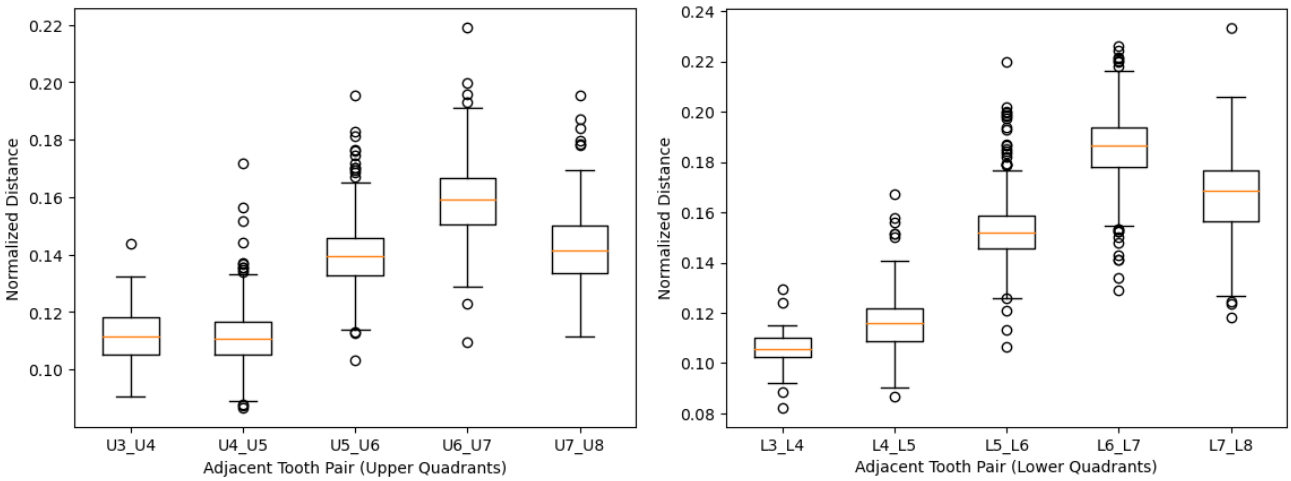


Figure 35 Boxplots of distances between adjacent tooth pairs in upper and lower quadrants

Figure 35 indicates the presence of potential outliers in the annotations. Upon further investigation, it is found that adjacent tooth pairs with exceptionally large distances are due to slight variations in the zoom levels of the images, as shown in Figure 36. Despite the normalization of the measurements by the diagonal of the images, these variations in zoom levels lead to an inflated distance between adjacent tooth pairs. However, it is important to note that such variations in zoom level are not controllable in the source images and cannot be calibrated. Furthermore, these variations do not impact the object detection in Stage 2 or the tooth pair classification in Stage 3. The reason is that the YOLO object detection in Stage 2 can accommodate different object sizes, and all the elements of a feature vector in tooth pair classification in Stage 3 are systematically scaled by the same zoom level factor.

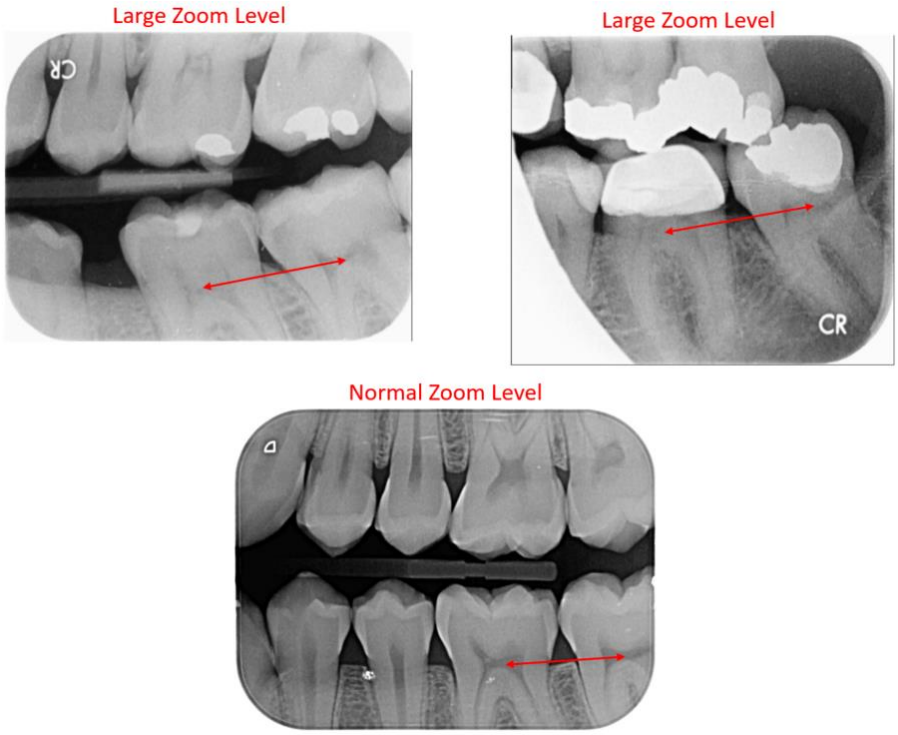


Figure 36 Images with different zoom levels.

The Intersection over Union (IoU) of adjacent bounding boxes are also checked to ensure they are not falsely overlapped, i.e. IoU should tend to zero. As shown in Figure 37, all bounding boxes have minimal overlapping (IoU is less than 10%). A slight degree of overlap is acceptable considering that the teeth are tightly packed and not perfectly aligned, as illustrated in Figure 38.

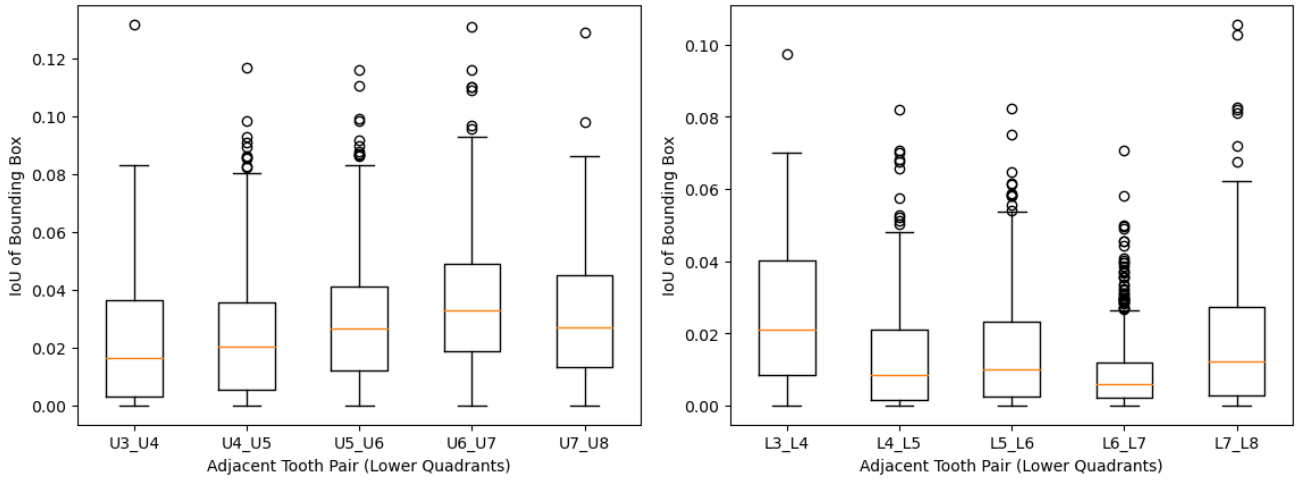


Figure 37 Boxplots of IoU between adjacent tooth pairs in upper and lower quadrants

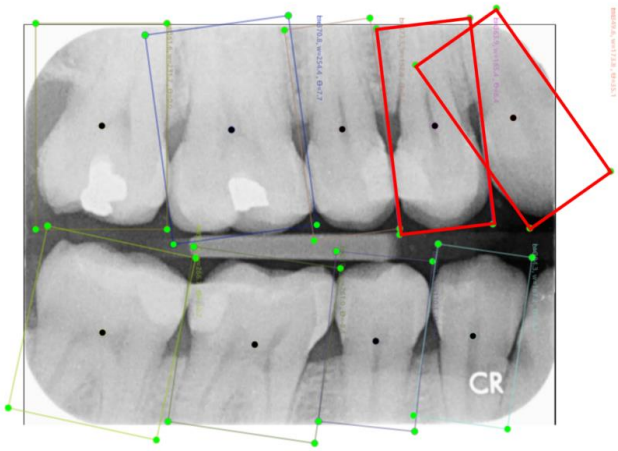


Figure 38 Overlapping of bounding boxes of adjacent teeth

The checked dataset with 693 annotations is randomly split into 485 for training and 208 for testing.

4.2 Analysis of Results in Stage 1

The left/right classifier is implemented using the pre-trained ResNet50 model, which is fine-tuned for 30 epochs. The learning rate is determined empirically using the learning rate finder approach [51], as shown in Figure 39. It is set as the point (around 10^{-3}) where the average training loss starts to increase. 10% of the training set images are selected as the validation set, and the remaining 90% are used for training.

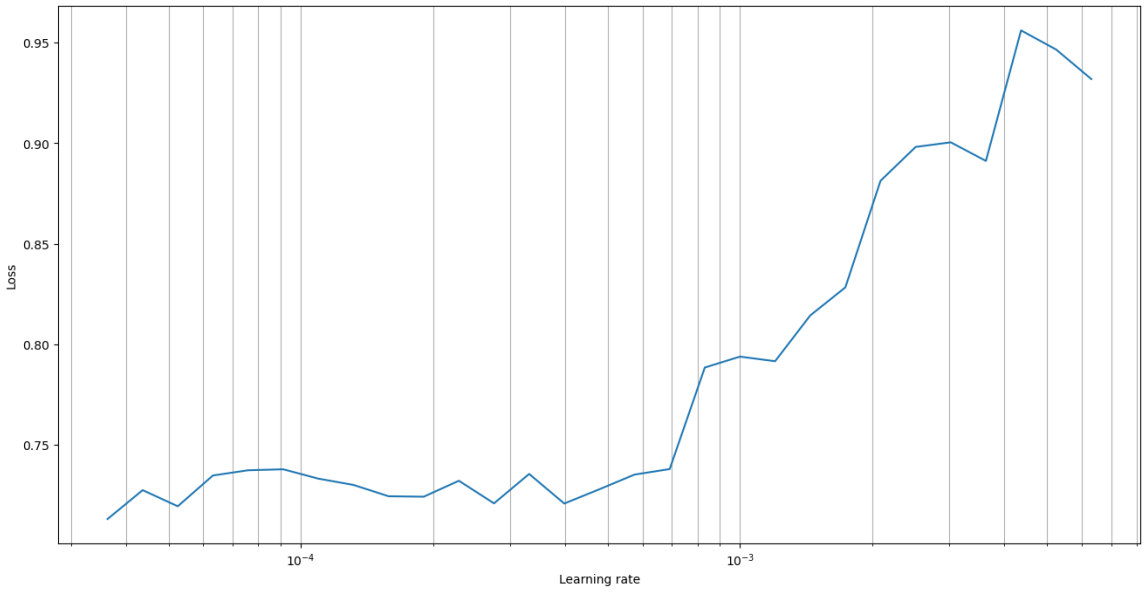


Figure 39 Change of average training loss in learning rate finder

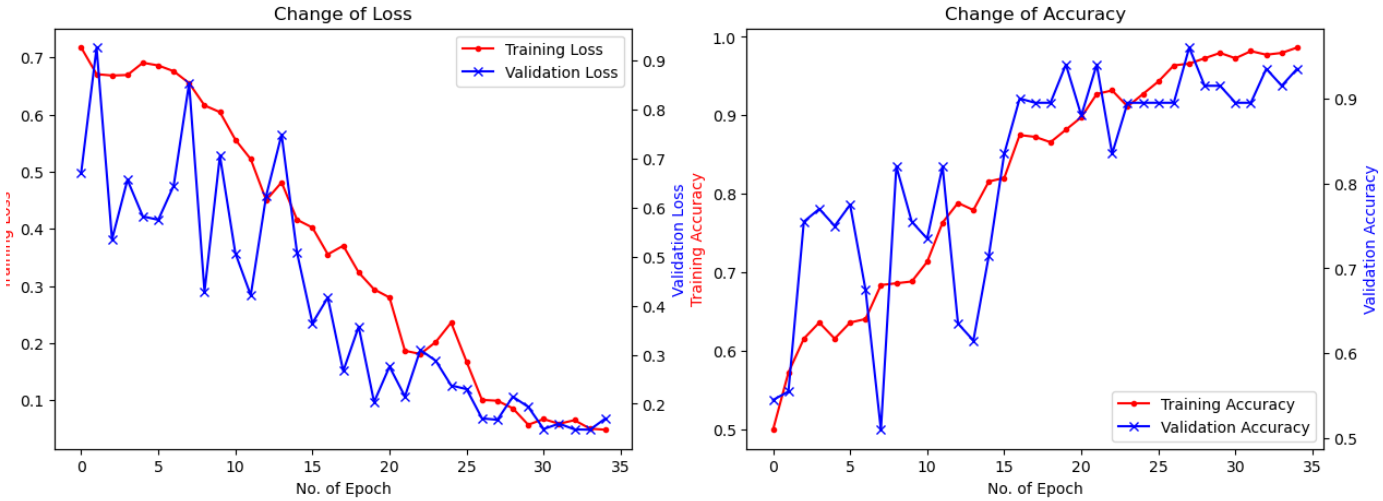


Figure 40 Training loss and validation loss of Stage 1 left/right classifiers

Figure 40 illustrated the change in training/validation loss and accuracy. The best model is saved when the lowest validation loss is obtained. The accuracy achieved in the testing dataset is 95.7%. The confusion matrix and a number of misclassified test images are shown in Figure 41. The misclassified images lack contrasting features found in the root patterns of lower molars and upper molars, and also show certain degrees of fillings and root treatments. These factors introduce difficulties for the classifier to discern the orientation of the images.

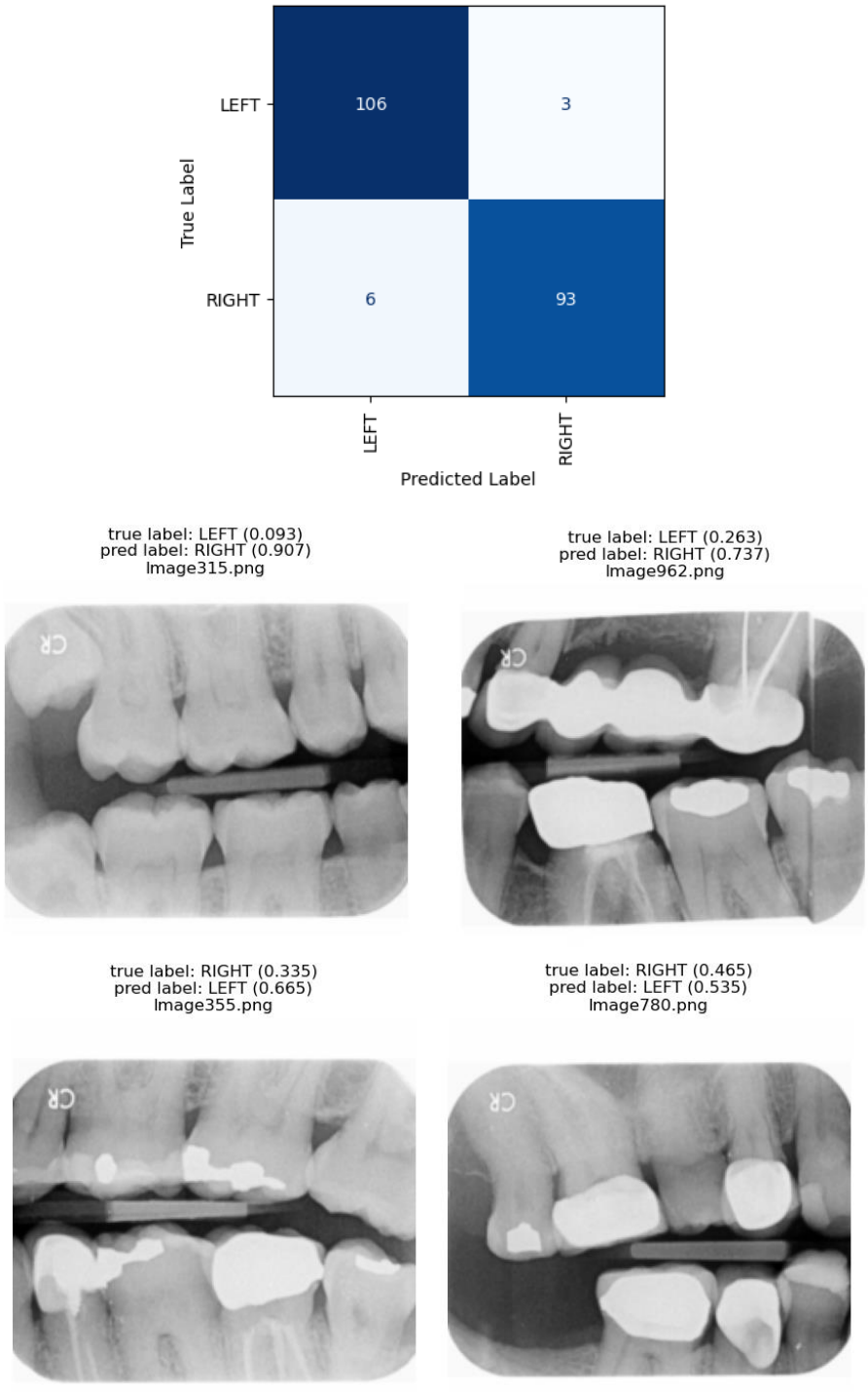


Figure 41 Confusion matrix and misclassified examples

4.3 Analysis of Results in Stage 2

In this stage, the tooth detection, non-molar, and molar classifier are implemented using the YOLOv5 model with oriented bounding box support. The YOLOv5 model is trained from scratch for 300 epochs. Three sizes of the model, namely YOLOv5n, YOLOv5m, and YOLOv5x, are trained to compare their differences in performance. Figure 42 illustrates the training and validation loss of the three YOLOv5 models. There are four types of loss: *box_loss*, which refers to the regression loss of bounding box coordinates; *obj_loss*, which refers to the loss in the "objectness" score; *cls_loss*, which refers to the classification loss in entropy; and *theta_loss*, which refers to the loss in the regression of the angle of rotation of the bounding box. When training the YOLO models with three different sizes from scratch, it takes at least 250 epochs for convergence. There is no rebound from the four types of validation losses, meaning that the models are not overfitted yet.

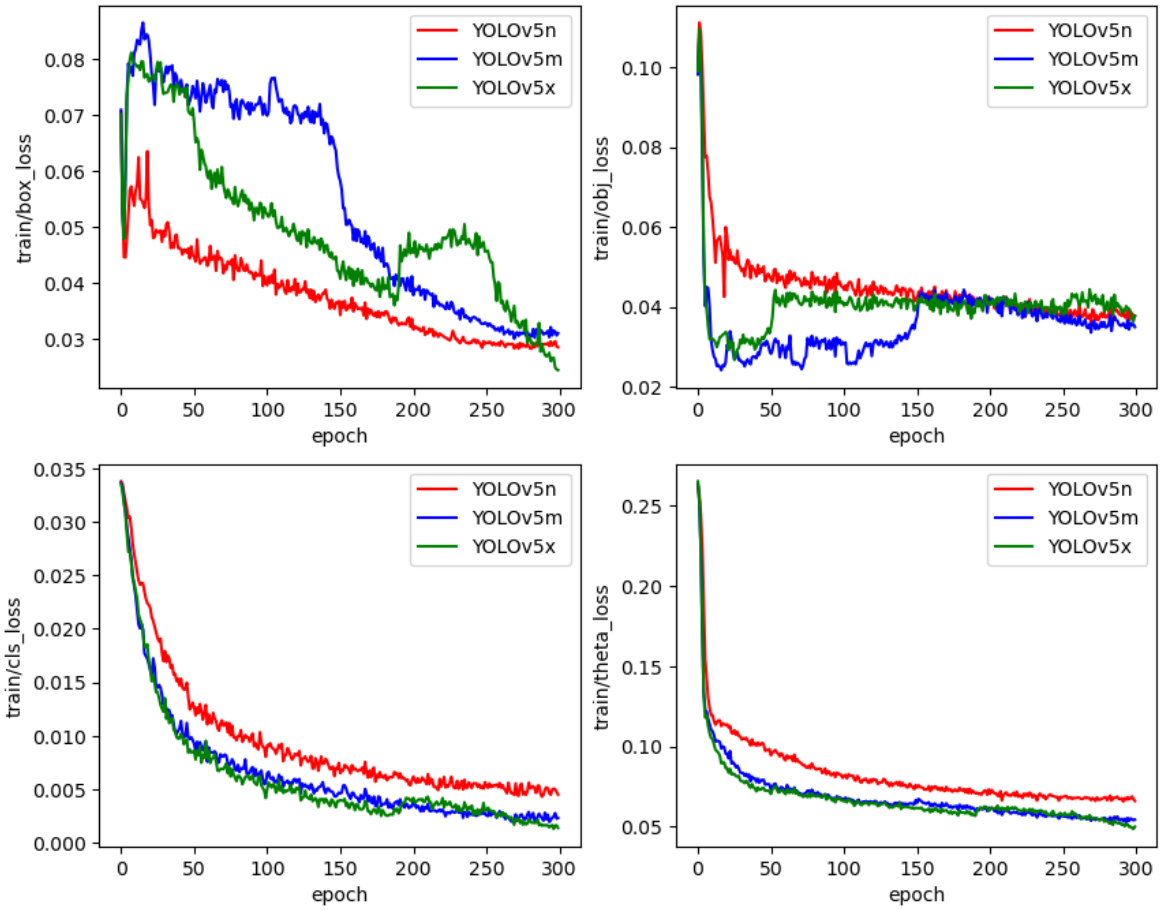


Figure 42(a) Training loss of Stage 2 Object Detector using YOLOv5

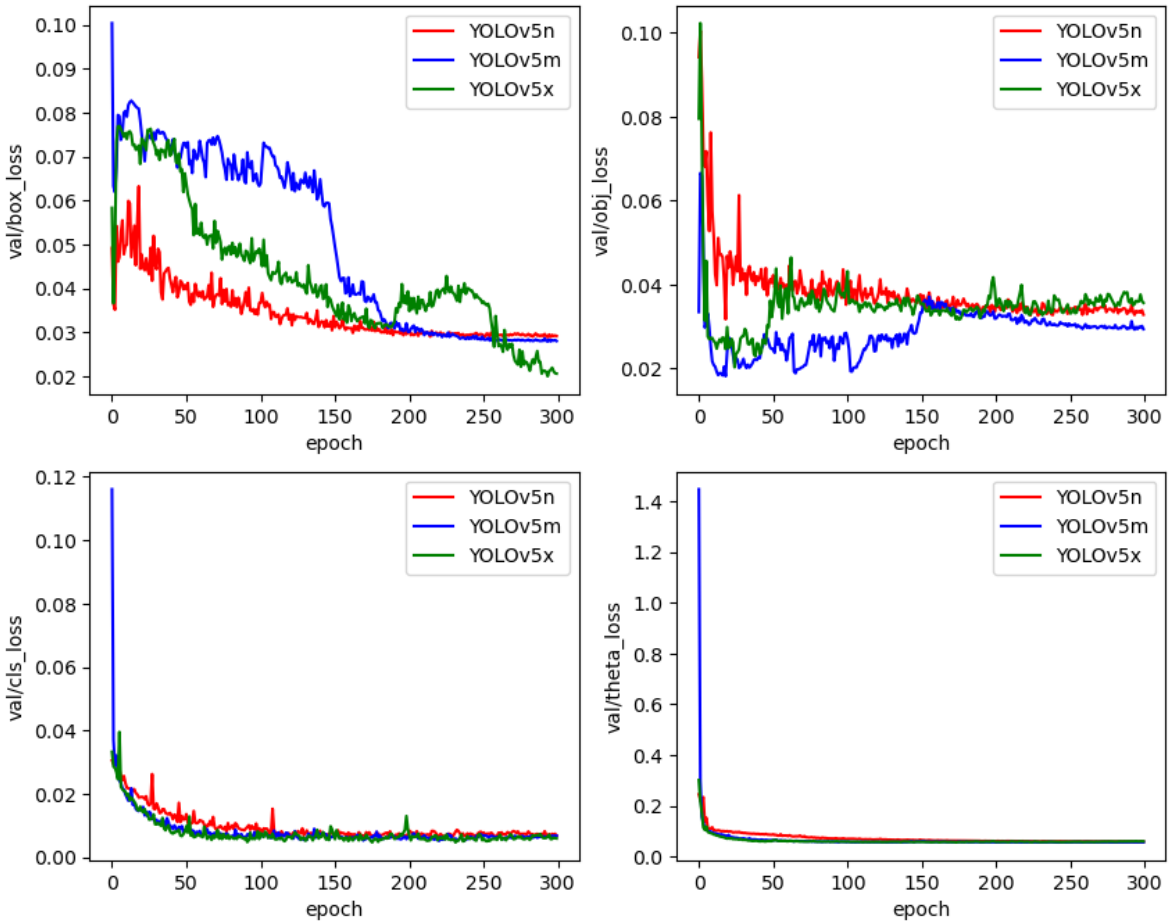


Figure 42(b) Validation loss of Stage 2 Object Detector using YOLOv5

In Figure 43, the performance metrics are plotted. The best achievable values are summarized in Table 3. YOLOv5m and YOLOv5x offer better performance, with the former achieving a slightly higher mAP. However, the differences are not significant. This finding is consistent with the general notion that larger models with more layers and parameters can yield better mAP at the cost of processing speed [53]. Nevertheless, the improvement from YOLOv5m to YOLOv5x is not substantial. In the classification of the four classes, the molar classes exhibit relatively better performance than the nonmolar classes. This disparity is due to the moderate class imbalance of around 1.38 between the molar and nonmolar classes.

Model	*No. of Training Labels*	*Precision*	*Recall*	*F1 Score*	*mAP@0.5*	*mAP@0.5:0.95*
YOLOv5n	**1515**	**0.900**	**0.872**	**0.886**	**0.918**	**0.651**
Upper Molar	441	0.949	0.93	0.939	0.955	0.672
Upper Nonmolar	336	0.843	0.917	0.878	0.922	0.598
Lower Molar	439	0.946	0.943	0.944	0.96	0.716
Lower Nonmolar	299	0.874	0.883	0.878	0.918	0.611
YOLOv5m	**1515**	**0.928**	**0.894**	**0.911**	**0.944**	**0.729**
Upper Molar	441	0.946	0.906	0.926	0.958	0.729
Upper Nonmolar	336	0.918	0.871	0.894	0.932	0.67
Lower Molar	439	0.922	0.936	0.929	0.955	0.731
Lower Nonmolar	299	0.925	0.863	0.893	0.93	0.674
YOLOv5x	**1515**	**0.930**	**0.897**	**0.913**	**0.934**	**0.675**
Upper Molar	441	0.946	0.912	0.929	0.967	0.704
Upper Nonmolar	336	0.946	0.833	0.886	0.901	0.612
Lower Molar	439	0.951	0.945	0.948	0.966	0.760
Lower Nonmolar	299	0.875	0.896	0.885	0.902	0.625

Table 3 Performance metrics of YOLOv5 with three model sizes

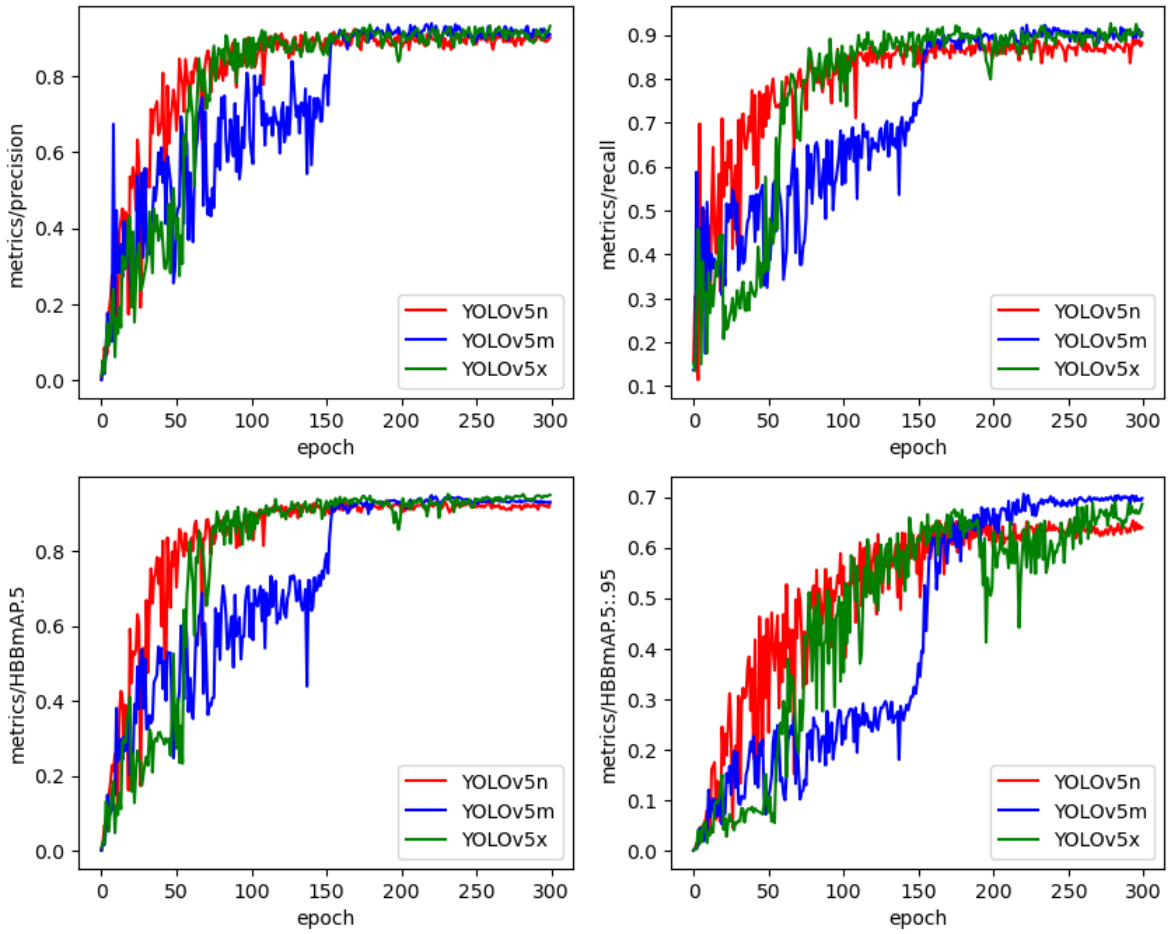
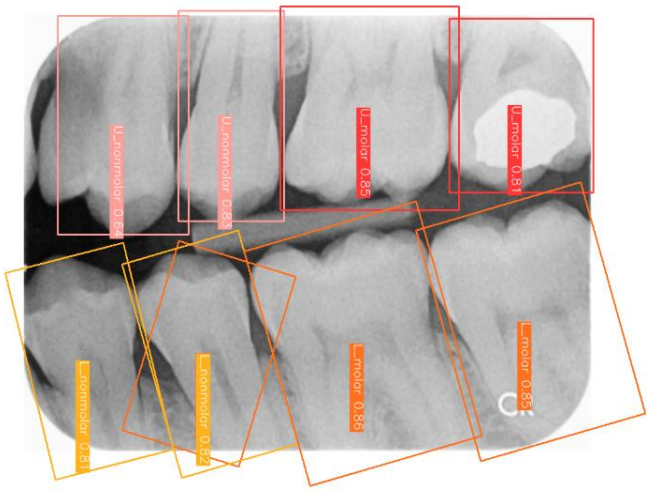


Figure 43 Training and validation loss of Stage 2 Object Detector using YOLOv5n

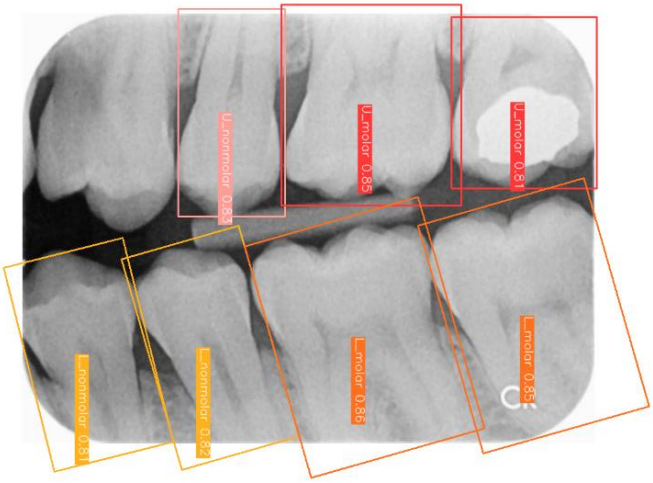
Figure **44** illustrates the inference using YOLOv5m for two images in the testing set. The inference is executed under two confidence threshold values. Given that each bounding box inferred by YOLOv5m carries an inference confidence score, the model employs the confidence threshold to determine whether a bounding box should be identified as a target object, considering that the confidence of the bounding box exceeds the threshold. Consequently, selecting a higher confidence threshold results in a reduced count of bounding boxes predicted by the model. As evident in

Figure **44**, certain bounding boxes are excluded from the final output as the confidence threshold is increased from 0.6 to 0.8.

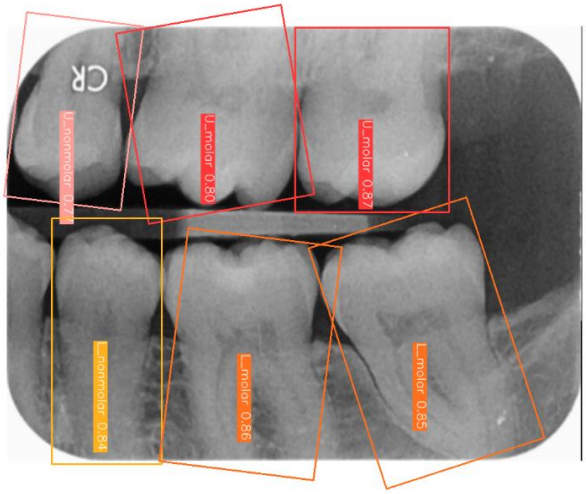
Consequently, the 0.6 confidence threshold is adopted for generating inferences in the next stage prediction to prevent the possibility of missing teeth. Instances of multiple bounding boxes for the same tooth, like the one demonstrated in Image 1 with a confidence threshold of 0.6, will be rectified by the tooth pair classifiers in stage 3.



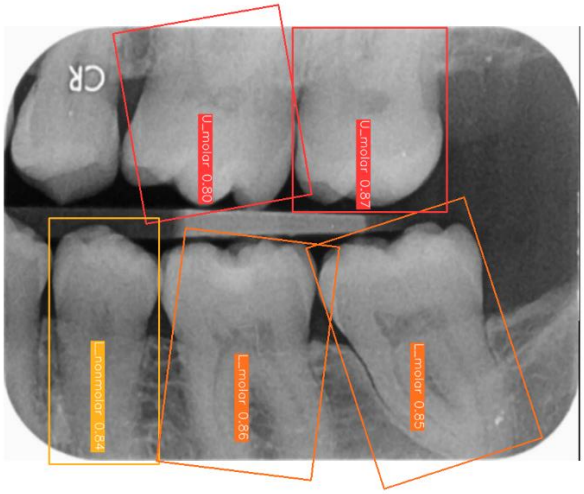
Prediction for Image 1 with Confidence Threshold = 0.6



Prediction for Image 1 with Confidence Threshold = 0.8



Prediction for Image 2 with Confidence Threshold = 0.6



Prediction for Image 2 with Confidence Threshold = 0.8

Figure 44 Training and validation loss of Stage 2 Object Detector using YOLOv5m

4.4 Analysis of Results in Stage 3

In Stage 3, several classifiers, denoted as $\varphi(q, v_{ij})$ and $\chi(q, v_{ij})$ are trained to classify pairs of teeth within the same quadrant and in opposite quadrants, respectively. The pair of teeth is represented by a feature vector $v_{ij} = (W_i, W_j, D_{ij})$, as described in Chapter 3. The tooth widths W_i and W_j and the inter-tooth distance D_{ij} are measured from the tooth bounding boxes generated in Stage 2.

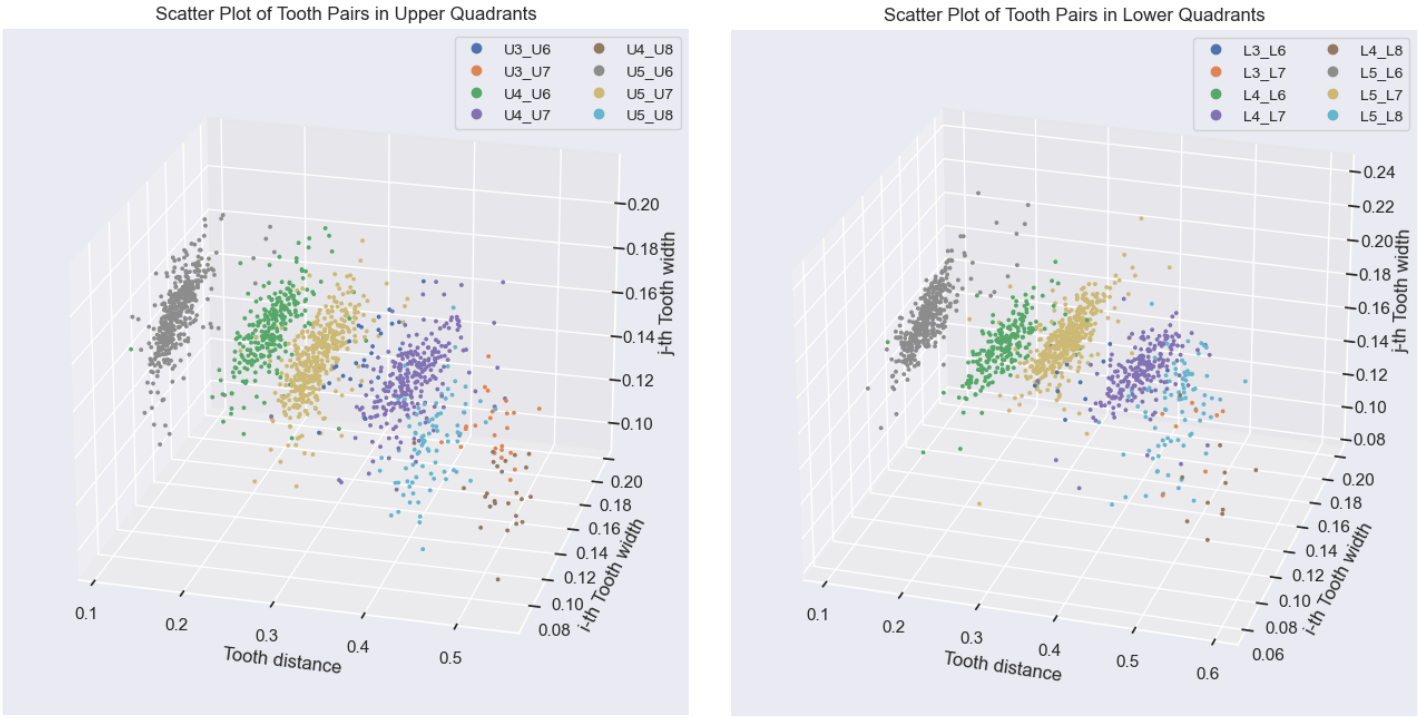


Figure 45 Scatterplots of tooth pairs. *Ui_Uj* and *Li_Lj* refer to the teeth pairs (t_i, t_j) in the upper and lower quadrants respectively. t_i denotes a nonmolar tooth and t_j denotes a molar tooth both in either upper or lower quadrants.

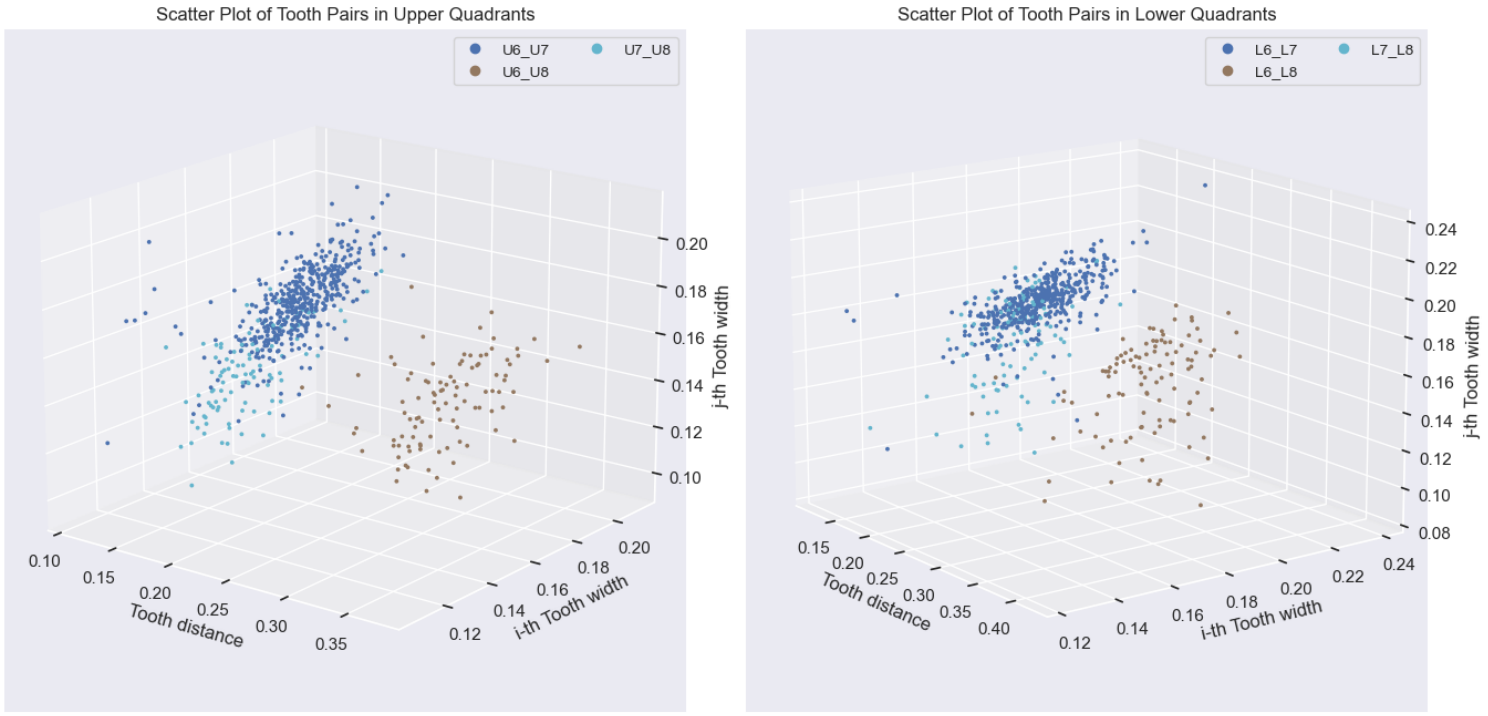


Figure 46 Scatterplots of tooth pairs. *Ui_Uj* and *Li_Lj* refer to the teeth pairs (t_i, t_j) in the upper and lower quadrants respectively. t_i and t_j denote molar teeth both in either upper or lower quadrants.

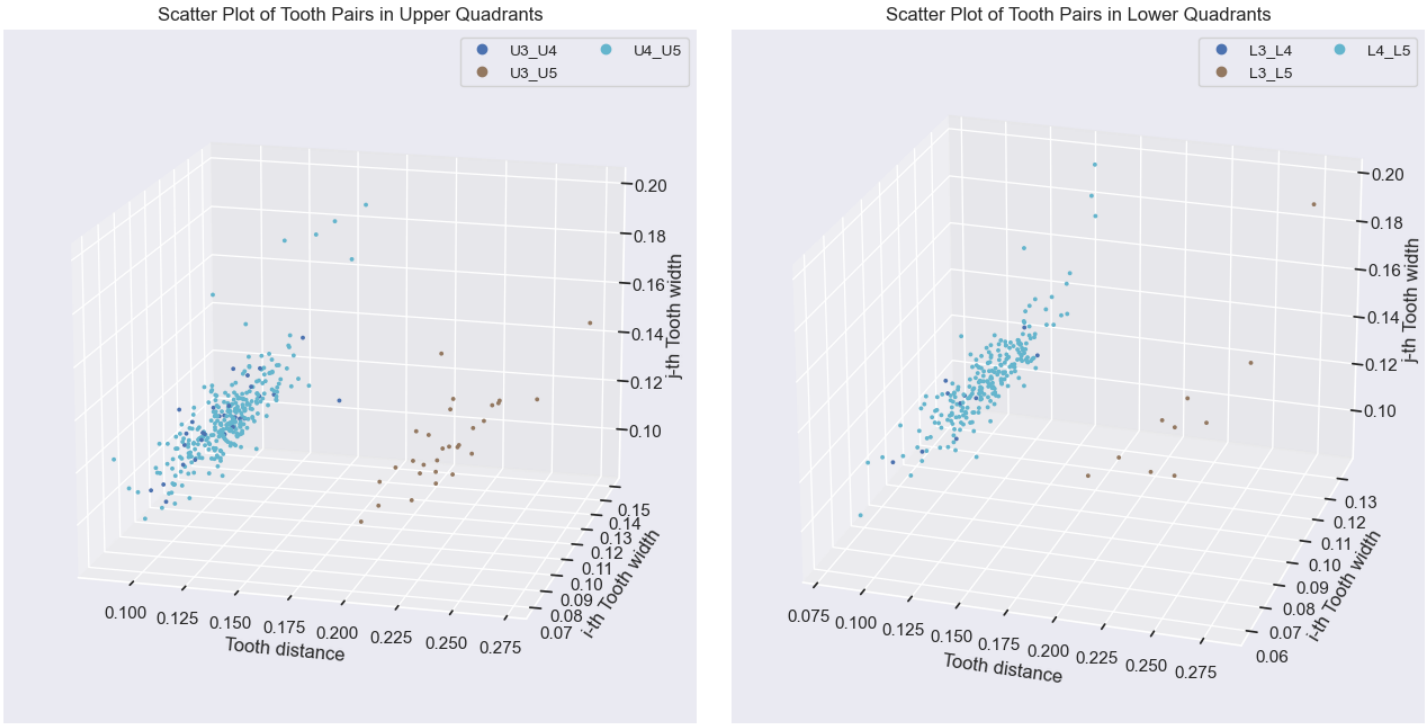


Figure 47 Scatterplots of tooth pairs. *Ui_Uj* and *Li_Lj* refer to the teeth pairs (t_i, t_j) in the upper and lower quadrants respectively. t_i and t_j denote nonmolar teeth both in either upper or lower quadrants.

In Figure 45, scatterplots of features for tooth pairs consisting of a non-molar and a molar tooth are presented. Their distributions are generally separable, particularly using the tooth distance dimension. Figure 46 displays the same scatterplots for molar teeth. Although the tooth distance dimension no longer separates adjacent tooth pairs, such as (t_6,t_7) and (t_7,t_8), the use of tooth widths can still statistically distinguish them. Figure 47 showcases the same scatterplots for nonmolar teeth. Due to the similarity in tooth widths, adjacent tooth pairs cannot be differentiated.

RF and SVM are utilized to compare their performance in fitting the classifier $\varphi(q, v_{ij})$ to classify the distributions illustrated in Figure 45 to Figure 47. The classifiers are "class-weighted" by multiplying the class with the inverse of its probability of occurrence. This approach addresses bias issues arising from class imbalance in the dataset [55, 56]. The fitting results from the testing dataset are presented in Table 4. Similar to the visualisations in Figure 45 to Figure 47, classifying a pair consisting of a nonmolar and a molar tooth yields better performance (AP@SVM=0.909 (upper) or 0.948 (lower)) than classifying a pair of nonmolar teeth or a pair of molar teeth. Classifying a pair of molar teeth achieves slightly better performance (AP@SVM=0.887 (upper) or 0.915 (lower)) than classifying a pair of nonmolar teeth (AP@SVM=0.841 (upper) or 0.908 (lower)).

Quadrant	**Tooth Pair Classes**	**Support**	**SVM**			**RF**		
			Precision	**Recall**	**Avg. Precision Score (AP)**	**Precision**	**Recall**	**Avg. Precision Score (AP)**
Upper	(t_3,t_6) (t_3,t_7) (t_3,t_8) (t_4,t_6) (t_4,t_7) (t_4,t_8) (t_5,t_6) (t_5,t_7) (t_5,t_8)	685	0.90	0.89	**0.909**	0.88	0.88	**0.914**
Upper	(t_6,t_7) (t_6,t_8) (t_7,t_8)	274	0.86	0.82	**0.887**	0.89	0.89	**0.876**
Upper	(t_3,t_4) (t_3,t_5) (t_4,t_5)	150	0.83	0.70	**0.841**	0.82	0.90	**0.800**
Lower	(t_3,t_6) (t_3,t_7) (t_3,t_8) (t_4,t_6) (t_4,t_7) (t_4,t_8) (t_5,t_6) (t_5,t_7) (t_5,t_8)	606	0.92	0.90	**0.948**	0.90	0.92	**0.943**
Lower	(t_6,t_7) (t_6,t_8) (t_7,t_8)	267	0.88	0.82	**0.915**	0.86	0.87	**0.907**
Lower	(t_3,t_4) (t_3,t_5) (t_4,t_5)	111	0.92	0.76	**0.908**	0.86	0.93	**0.900**

Table 4: $\varphi(q, v_{ij})$ classifies tooth pairs (t_i, t_j) for both t_i *and* t_j in the same quadrant and $i, j = 3,4,\ldots,8$

In Figure 48 and Figure 49, the distributions of features for tooth pairs consisting of teeth from opposite quadrants are displayed. Due to the numerous combinations, only four examples are presented. Figure 48 illustrates the distributions when examining t_5 and t_6 in the upper quadrants alongside other teeth in the lower quadrants. Figure 49 illustrates the distributions when examining t_5 and t_6 in the lower quadrants alongside other teeth in the upper quadrants. The distributions show separable mixtures, except when the tooth pairs are close to each other and consist of non-molar teeth exclusively. For instance, the boundary between the distributions of lower t_5 and upper t_5 (denoted as *L5_U5* in the figure) and that of lower t_5 and upper t_4 (denoted as L5_U4 in the figure) is not clearly defined. This is attributed to the fact that non-molar teeth generally possess less distinctive features compared to molar teeth as well as smaller number of training samples available.

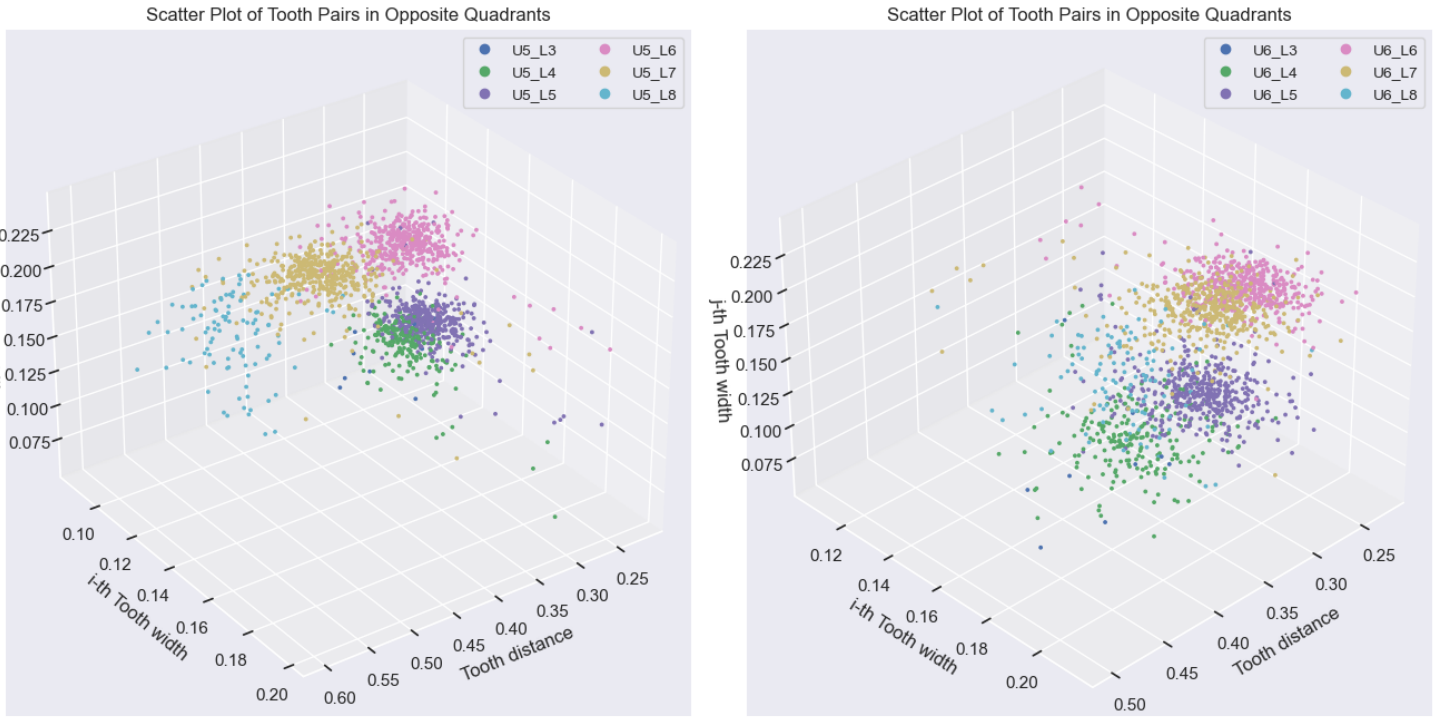


Figure 48 Scatterplots of tooth pairs. *Ui_Lj* refer to the teeth pairs (t_i, t_j) in the upper and lower quadrants respectively. Here we examine *Ui=U5* and *Ui=U6*.

Figure 49 Scatterplots of tooth pairs. *Li_Uj* refer to the teeth pairs (t_i, t_j) in the lower and upper quadrants respectively. Here we examine *Li=L5* and *Li=L6*.

Table 5 illustrates the performance of the SVM and RF implementations when fitting the tooth pair classifiers $\chi(q, v_{ij})$ to classify the distributions in Figure 48 and Figure 49. When comparing Table 4 and Table 5, the classification performance for tooth pairs in the same quadrant by $\varphi(q, v_{ij})$ is generally better than that of opposite quadrant by $\chi(q, v_{ij})$. This outcome is not unexpected, as inter-tooth distances within the same quadrant are typically shorter than those between opposite quadrants. This leads to a smaller degree of uncertainty and, consequently, more separable sample distributions.

Furthermore, the SVM and RF implementations yield similar performance in terms of precision and recall, but SVM provides slightly higher AP. As a result, SVM is adopted for subsequent analyses.

Tooth Pair Classes $j = 3,4,\ldots,8$	t_i	t_j	**Support**	**SVM**			**RF**		
				Precision	**Recall**	**Avg. Precision Score (AP)**	**Precision**	**Recall**	**Avg. Precision Score (AP)**
(t_3,t_j)	Upper	Lower	63	0.82	0.87	**0.854**	0.80	0.87	**0.900**
(t_4,t_j)	Upper	Lower	453	0.83	0.80	**0.882**	0.82	0.83	**0.864**
(t_5,t_j)	Upper	Lower	705	0.86	0.85	**0.899**	0.86	0.87	**0.885**
(t_6,t_j)	Upper	Lower	710	0.82	0.81	**0.856**	0.81	0.82	**0.828**
(t_7,t_j)	Upper	Lower	694	0.86	0.84	**0.879**	0.84	0.85	**0.881**
(t_8,t_j)	Upper	Lower	161	0.79	0.76	**0.849**	0.80	0.81	**0.838**
(t_3,t_j)	Lower	Upper	36	0.44	0.47	**0.535**	0.57	0.67	**0.761**
(t_4,t_j)	Lower	Upper	373	0.83	0.74	**0.836**	0.77	0.80	**0.827**
(t_5,t_j)	Lower	Upper	739	0.77	0.76	**0.812**	0.76	0.77	**0.787**
(t_6,t_j)	Lower	Upper	742	0.82	0.80	**0.854**	0.80	0.80	**0.834**
(t_7,t_j)	Lower	Upper	742	0.85	0.84	**0.868**	0.85	0.85	**0.846**
(t_8,t_j)	Lower	Upper	154	0.84	0.84	**0.882**	0.86	0.84	**0.877**

Table 5: $\chi(q, v_{ij})$ classifies tooth pairs (t_i, t_j) for both t_i and t_j in the opposite quadrants

4.5 Analysis of Results in Stage 4

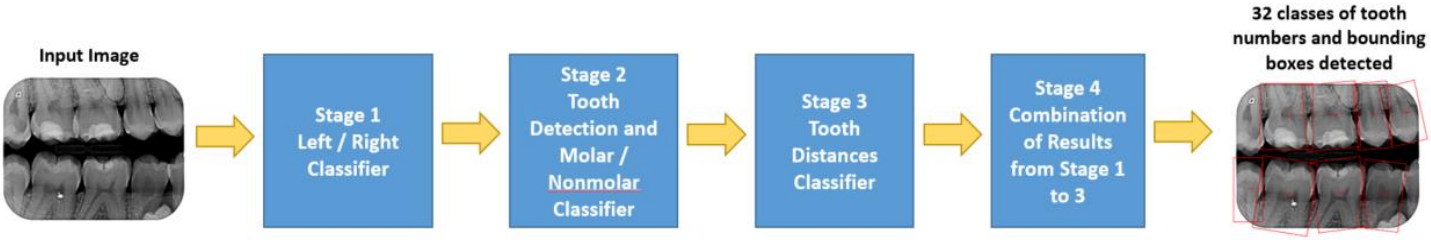
In this final stage, the stagewise results of the decision-making process shown in Figure 33 are combined to deduce the final predictions of tooth numbers. To assess any potential improvements that could be introduced by the proposed 'four-stage' classification approach, we also compare the performance when training the YOLOv5 models from scratch to:

- Directly predict 32 classes of tooth numbers. This is referred to as the 'single-stage' approach.
- Predict 16 classes of tooth numbers by disregarding the labels of left and right quadrants, and then combine the prediction results with those of Stage 1 to predict the actual 32 classes of tooth numbers. This is referred to as the 'three-stage' approach.

The comparison of these three approaches is summarized in Figure 50.

(a) Single-stage : Direct Prediction of 32 classes

(b) Three-stage : Prediction of 32 classes by combining the prediction of 16 classes and the prediction in Stage 1



(c) Four-stage: Prediction of 32 classes by combining the prediction results from Stage 1 to Stage 3

Figure 50 Three approaches to tooth object detection and tooth number classification.

		(a) Single-stage Approach			(b) Three-stage Approach			(c) Four-stage Approach		
	Support	**Precision**	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**
t_{13}	9	0	0	0	0	0	0	0.5	0.667	0.571
t_{14}	59	0.727	0.407	0.522	0.737	0.712	0.724	0.729	0.729	0.729
t_{15}	93	0.59	0.387	0.468	0.76	0.817	0.788	0.84	0.677	0.75
t_{16}	97	0.422	0.443	0.432	0.817	0.784	0.8	0.827	0.691	0.753
t_{17}	95	0.413	0.2	0.27	0.755	0.779	0.767	0.884	0.642	0.744
t_{18}	23	0	0	0	0.8	0.174	0.286	1	0.261	0.414
t_{23}	6	0	0	0	0	0	0	0.714	0.833	0.769
t_{24}	64	0.717	0.516	0.6	0.81	0.797	0.803	0.821	0.719	0.767
t_{25}	104	0.541	0.769	0.635	0.836	0.885	0.86	0.868	0.76	0.81
t_{26}	103	0.481	0.621	0.542	0.879	0.913	0.895	0.812	0.757	0.784
t_{27}	102	0.557	0.431	0.486	0.792	0.824	0.808	0.835	0.696	0.759
t_{28}	21	0	0	0	0.857	0.286	0.429	0.75	0.429	0.545
t_{33}	4	0	0	0	0	0	0	0.143	0.25	0.182
t_{34}	49	0.714	0.102	0.179	0.784	0.592	0.674	0.822	0.755	0.787
t_{35}	102	0.577	0.402	0.474	0.857	0.824	0.84	0.868	0.902	0.885
t_{36}	105	0.558	0.41	0.473	0.858	0.867	0.863	0.858	0.867	0.863
t_{37}	106	0.489	0.208	0.291	0.883	0.783	0.83	0.906	0.821	0.861
t_{38}	18	0	0	0	1	0.111	0.2	0.75	0.167	0.273
t_{43}	4	0	0	0	0	0	0	0.111	0.25	0.154
t_{44}	46	0.519	0.304	0.384	1	0.609	0.757	0.829	0.63	0.716
t_{45}	94	0.517	0.479	0.497	0.899	0.755	0.821	0.894	0.809	0.849
t_{46}	94	0.467	0.457	0.462	0.83	0.83	0.83	0.893	0.798	0.843
t_{47}	94	0.424	0.266	0.327	0.867	0.691	0.769	0.91	0.755	0.826
t_{48}	22	0	0	0	0.857	0.273	0.414	0.714	0.455	0.556
Overall (Macro Average)		**0.363**	**0.267**	**0.293**	**0.703**	**0.554**	**0.59**	**0.762**	**0.638**	**0.675**

Table 6 Performance metrics of tooth detection and tooth number classification with YOLOv5n implementation.

		(a) Single-stage			(b) Three-stage			(c) Four-stage		
	Support	**Precision**	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**
t_{13}	9	0	0	0	0	0	0	0.5	0.444	0.471
t_{14}	59	0.587	0.458	0.514	0.712	0.797	0.752	0.863	0.746	0.8
t_{15}	93	0.607	0.548	0.576	0.806	0.806	0.806	0.875	0.828	0.851
t_{16}	97	0.467	0.66	0.547	0.875	0.794	0.832	0.849	0.814	0.832
t_{17}	95	0.468	0.547	0.505	0.765	0.821	0.792	0.882	0.789	0.833
t_{18}	23	0.333	0.174	0.229	0.571	0.348	0.432	0.882	0.652	0.75
t_{23}	6	0	0	0	1	0.167	0.286	1	1	1
t_{24}	64	0.547	0.547	0.547	0.841	0.828	0.835	0.907	0.766	0.831
t_{25}	104	0.614	0.673	0.642	0.874	0.865	0.87	0.895	0.817	0.854
t_{26}	103	0.646	0.515	0.573	0.886	0.903	0.894	0.814	0.893	0.852
t_{27}	102	0.662	0.48	0.557	0.793	0.863	0.826	0.867	0.833	0.85
t_{28}	21	0.6	0.143	0.231	0.9	0.429	0.581	0.667	0.476	0.556
t_{33}	4	0	0	0	0	0	0	0.333	0.25	0.286
t_{34}	49	0.517	0.306	0.385	0.838	0.633	0.721	0.833	0.714	0.769
t_{35}	102	0.553	0.667	0.604	0.807	0.902	0.852	0.893	0.902	0.898
t_{36}	105	0.595	0.657	0.624	0.873	0.914	0.893	0.883	0.933	0.907
t_{37}	106	0.49	0.481	0.486	0.888	0.896	0.892	0.896	0.896	0.896
t_{38}	18	0.2	0.056	0.087	0.733	0.611	0.667	0.833	0.556	0.667
t_{43}	4	0	0	0	0	0	0	0.5	0.25	0.333
t_{44}	46	0.528	0.413	0.463	0.966	0.609	0.747	0.882	0.652	0.75
t_{45}	94	0.564	0.468	0.512	0.824	0.798	0.811	0.877	0.755	0.811
t_{46}	94	0.584	0.553	0.568	0.848	0.83	0.839	0.871	0.787	0.827
t_{47}	94	0.368	0.266	0.309	0.866	0.755	0.807	0.888	0.84	0.863
t_{48}	22	0.4	0.182	0.25	0.6	0.545	0.571	0.737	0.636	0.683
Overall (Macro Average)		**0.43**	**0.366**	**0.384**	**0.719**	**0.63**	**0.654**	**0.81**	**0.718**	**0.757**

Table 7 Performance metrics of tooth detection and tooth number classification with YOLOv5m implementation

		(a) Single-stage			(b) Three-stage			(c) Four-stage		
	Support	**Precision**	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**
t_{13}	9	0	0	0	0	0	0	0.75	0.333	0.462
t_{14}	59	0.588	0.508	0.545	0.842	0.814	0.828	0.891	0.695	0.781
t_{15}	93	0.62	0.473	0.537	0.862	0.806	0.833	0.867	0.774	0.818
t_{16}	97	0.659	0.557	0.603	0.886	0.804	0.843	0.786	0.794	0.79
t_{17}	95	0.5	0.632	0.558	0.793	0.768	0.781	0.922	0.747	0.826
t_{18}	23	0.364	0.174	0.235	0.688	0.478	0.564	0.929	0.565	0.703
t_{23}	6	0	0	0	0	0	0	1	0.833	0.909
t_{24}	64	0.568	0.656	0.609	0.806	0.844	0.824	0.93	0.828	0.876
t_{25}	104	0.608	0.731	0.664	0.917	0.846	0.88	0.912	0.894	0.903
t_{26}	103	0.664	0.728	0.694	0.841	0.874	0.857	0.853	0.903	0.877
t_{27}	102	0.675	0.549	0.605	0.821	0.853	0.837	0.881	0.873	0.877
t_{28}	21	0.8	0.19	0.308	0.812	0.619	0.703	0.714	0.476	0.571
t_{33}	4	0	0	0	0	0	0	0.143	0.25	0.182
t_{34}	49	0.788	0.531	0.634	0.778	0.714	0.745	0.826	0.776	0.8
t_{35}	102	0.614	0.765	0.681	0.826	0.931	0.876	0.913	0.931	0.922
t_{36}	105	0.719	0.61	0.66	0.853	0.886	0.869	0.883	0.933	0.907
t_{37}	106	0.753	0.575	0.652	0.869	0.877	0.873	0.896	0.896	0.896
t_{38}	18	0.545	0.333	0.414	0.75	0.5	0.6	0.8	0.444	0.571
t_{43}	4	0	0	0	0	0	0	0.077	0.25	0.118
t_{44}	46	0.723	0.739	0.731	0.833	0.761	0.795	0.875	0.761	0.814
t_{45}	94	0.687	0.489	0.571	0.878	0.84	0.859	0.876	0.83	0.852
t_{46}	94	0.56	0.691	0.619	0.842	0.851	0.847	0.779	0.862	0.818
t_{47}	94	0.586	0.691	0.634	0.893	0.798	0.843	0.881	0.787	0.831
t_{48}	22	0.667	0.455	0.541	0.667	0.727	0.696	0.778	0.636	0.7
	Overall (Macro Average)	**0.529**	**0.462**	**0.479**	**0.686**	**0.65**	**0.665**	**0.798**	**0.711**	**0.742**

Table 8 Performance metrics of tooth detection and tooth number classification with YOLOv5x implementation

Comparison with Single-stage Approach

The three approaches are implemented using three YOLOv5 model sizes: YOLOv5n, YOLOv5m, and YOLOv5x. The performance metrics are summarized in Table 6, Table 7 and Table 8 respectively. Overall, the performance improves when adopting the three-stage and four-stage approaches. In the single-stage approach, the relatively low performance metrics can be attributed to the fact that the YOLO model predicts teeth by mixing up left and right quadrants. As depicted in Figure 51, while the ground truth belongs to the left quadrants, the single-stage detection predicts some teeth as counterparts in the right quadrants (t_{15}, t_{16} and t_{47}).

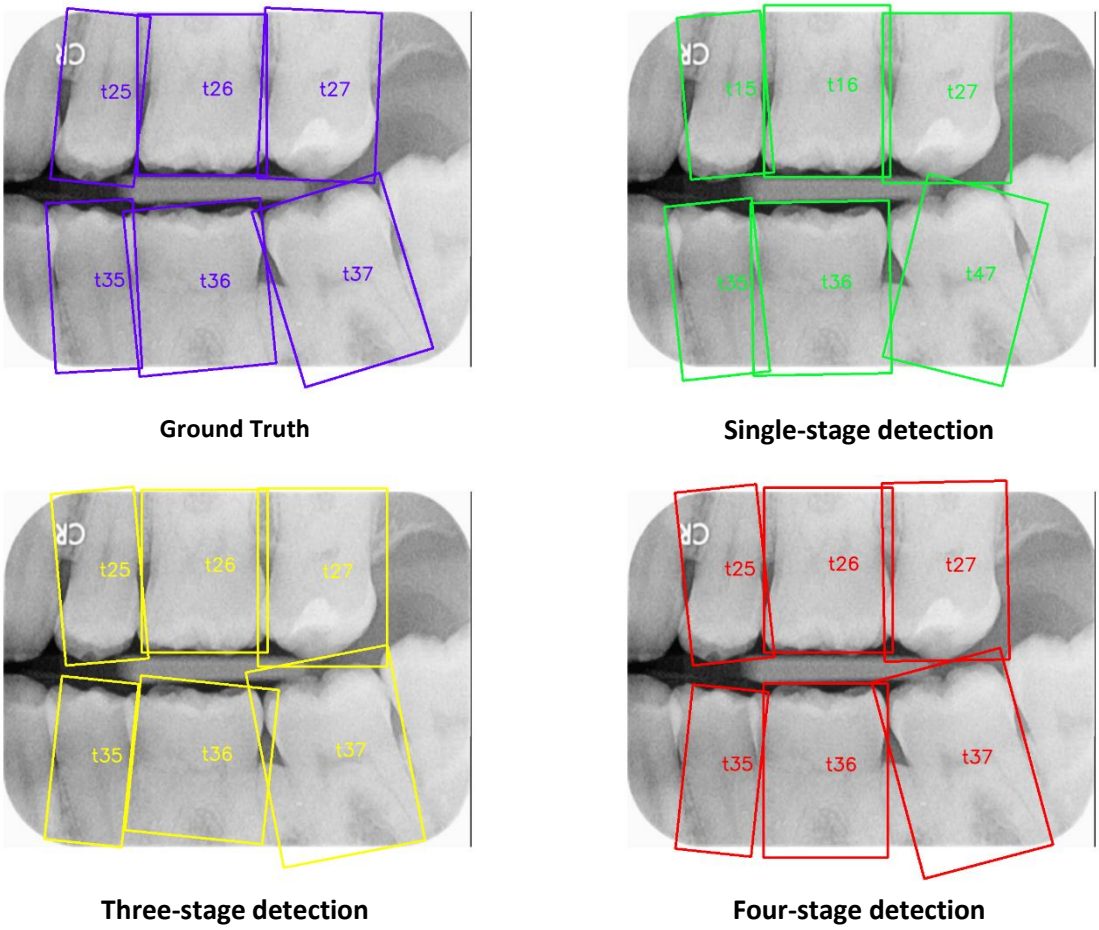


Figure 51 Detection results of the three approaches. The single-stage detector is confused by the left / right quadrants.

Comparison with Three-stage Approach

The four-stage detection approach consistently achieved notably better performance metrics than the three-stage detection approach. The most significant difference lies in the detection of teeth along the edges, such as t_{13}, t_{23}, t_{33}, t_{43}, t_{18}, t_{28}, t_{38} and t_{48}. As illustrated in Figure 52, the third-stage detector produces incorrect predictions due to the close resemblance between t_{13} and t_{14}, as well as t_{45} and t_{44}. In contrast, the four-stage detector performs tooth numbering classification based on tooth pair distances, enabling it to make corrective predictions despite the similarities in tooth features.

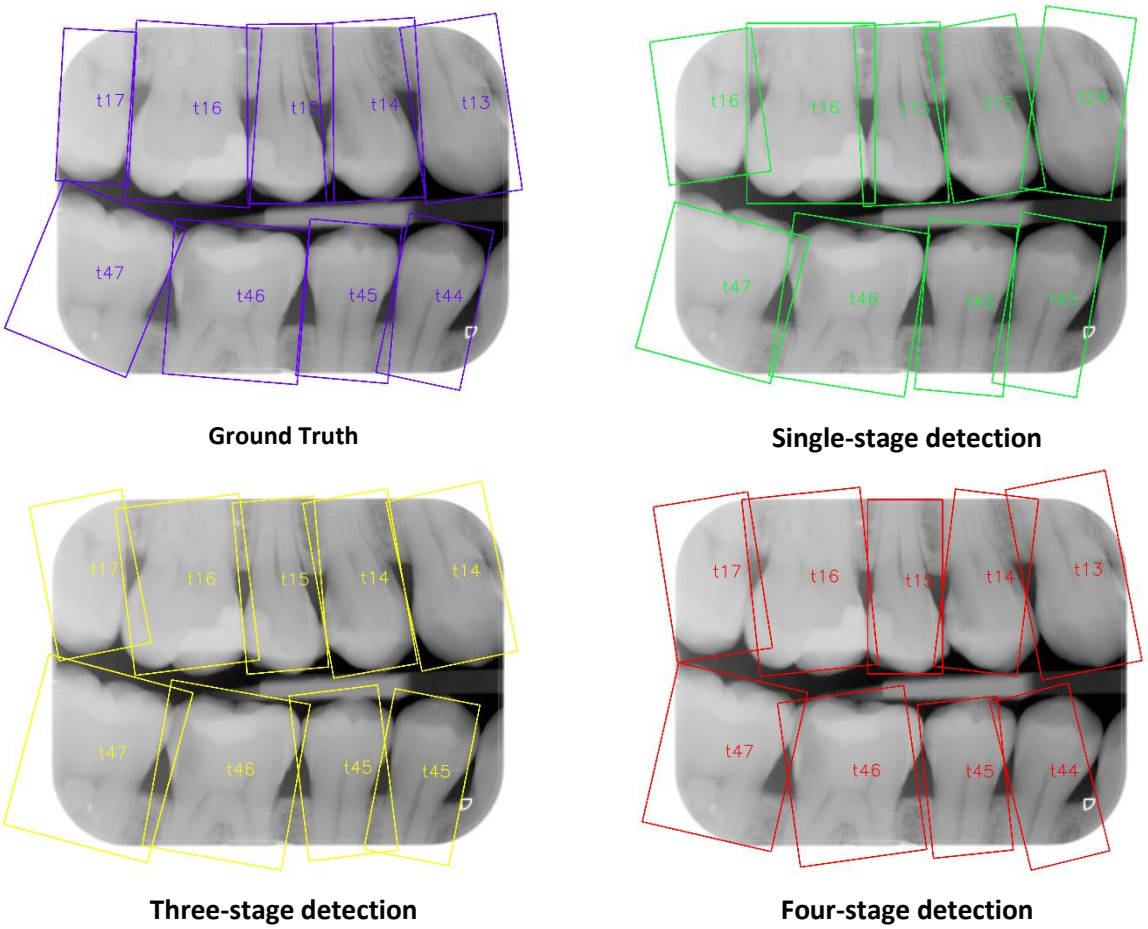


Figure 52 Detection results of the three approaches. The single-stage detector and the three-stage detector mistakes t_{13} with t_{14}, and t_{45} and t_{44}.

Another advantage of the four-stage approach is that YOLOv5 can be trained to detect a tooth "more confidently" by reducing its burden to precisely classify each tooth number, which can be challenging if the tooth features are rarely encountered in the training dataset. As illustrated in Figure 53, common tooth features in t_{25} and t_{26} are almost entirely concealed by tooth treatment patterns. This creates difficulties for YOLO object detection to assign a sufficiently high confidence score (a threshold of 0.6 is set in the analysis) to detect such tooth "objects".

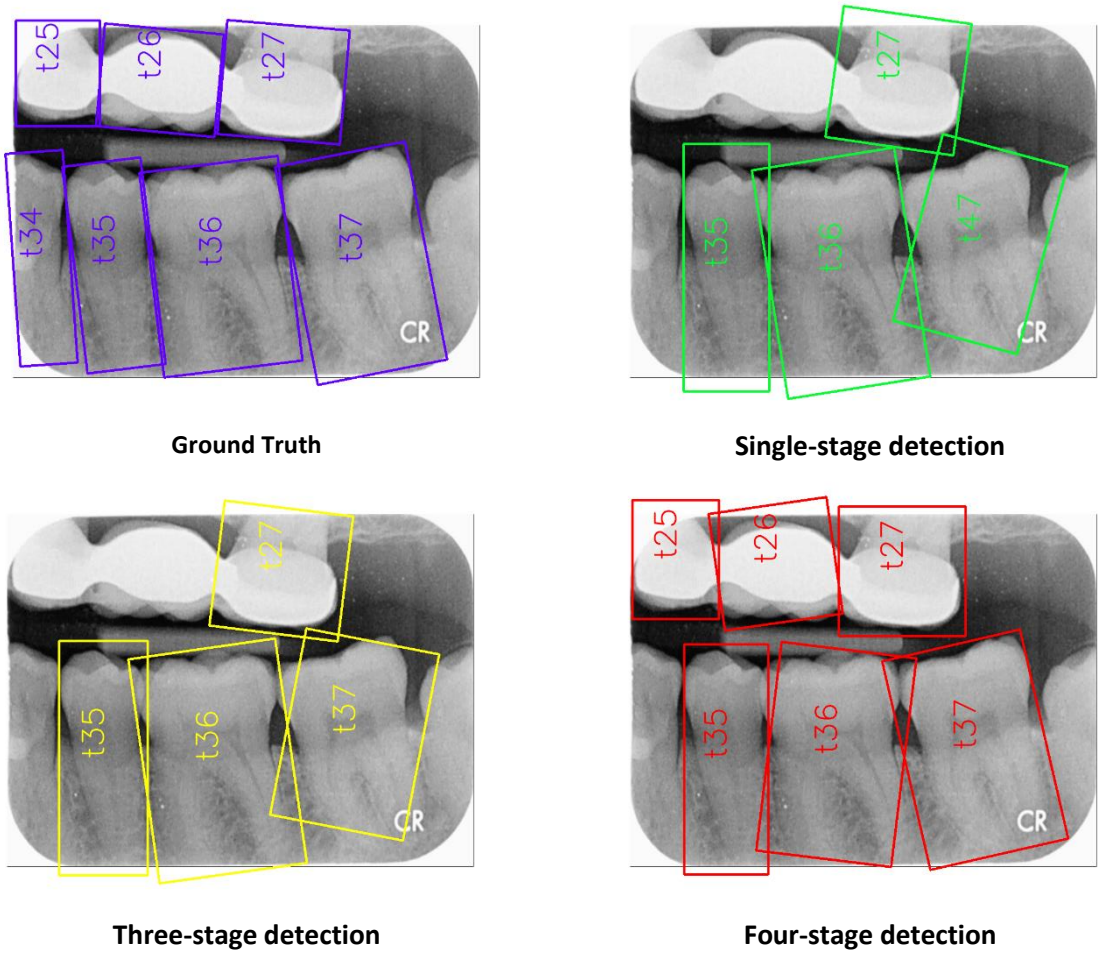
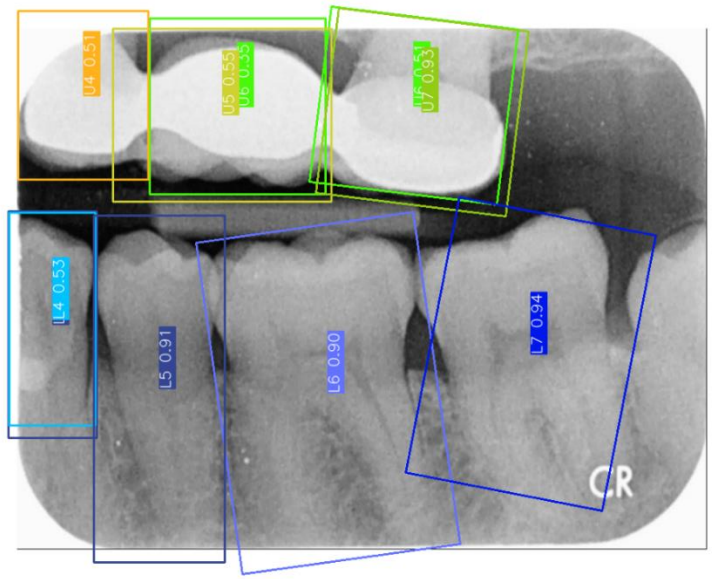
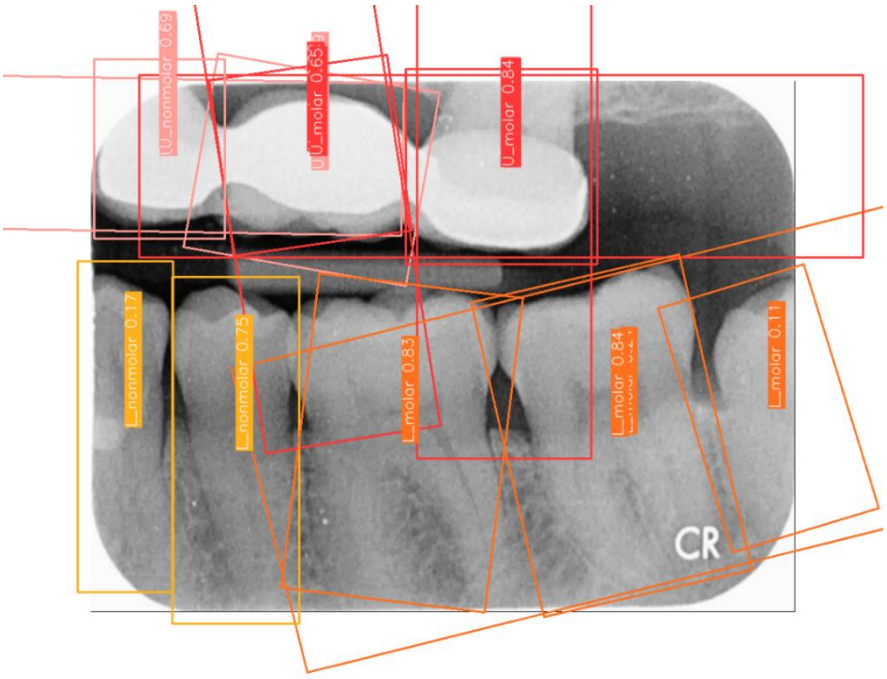


Figure 53 Detection results of the three approaches. The single-stage detector and the three-stage detector fail to detect some challenging teeth t_{25} and t_{26} as their "objectness" score is not high above the confidence threshold set at 0.6.



Three-stage detection (Output of Stage 2 to detect 8 teeth each in upper and lower quadrants)



Four-stage detection (Output of Stage 2 to detect upper/lower molar/nonmolar teeth)

Figure 54 Intermediate detection results of the three-stage and the four-stage approaches. The confidence threshold in the YOLOv5 is lowered to 0.1 to allow more bounding boxes to be shown.

To support this finding, the intermediate outputs of the three-stage and four-stage detectors are displayed in Figure 54. The confidence score threshold is lowered to 0.1 to allow more boxes to be displayed. t_{25} and t_{26} have confidence scores of approximately 0.51 and 0.55 using the three-stage detector, resulting in being suppressed in Figure 53. Conversely, the four-stage detector assigns them relatively high confidence scores at around 0.65 and 0.69, allowing them to be detected.

Limitations of Four-stage Approach

As described in Stage 3 Process A in Chapter 3.3, the four-stage detector relies on the simultaneous occurrence of a molar and a nonmolar tooth to initiate its tooth number prediction. However, it may encounter failure if a molar and a nonmolar tooth do not appear concurrently in the same quadrant, as depicted in Figure 55. Since a molar and a nonmolar pair of teeth offer the most accurate prediction of tooth numbers through the tooth distance classifier $\varphi(q, v_{ij})$, the prediction becomes more susceptible to errors if such information is missing. An extreme case is that when there is only one tooth detected in each quadrant as described in Figure 32, the four-stage detector fails completely.

Furthermore, any misclassifications by detectors in the intermediate stages can also lead to catastrophic outcomes, as illustrated in Figure 56. In this case, the Stage 2 detector fails to correctly identify the teeth in the upper and lower quadrants.

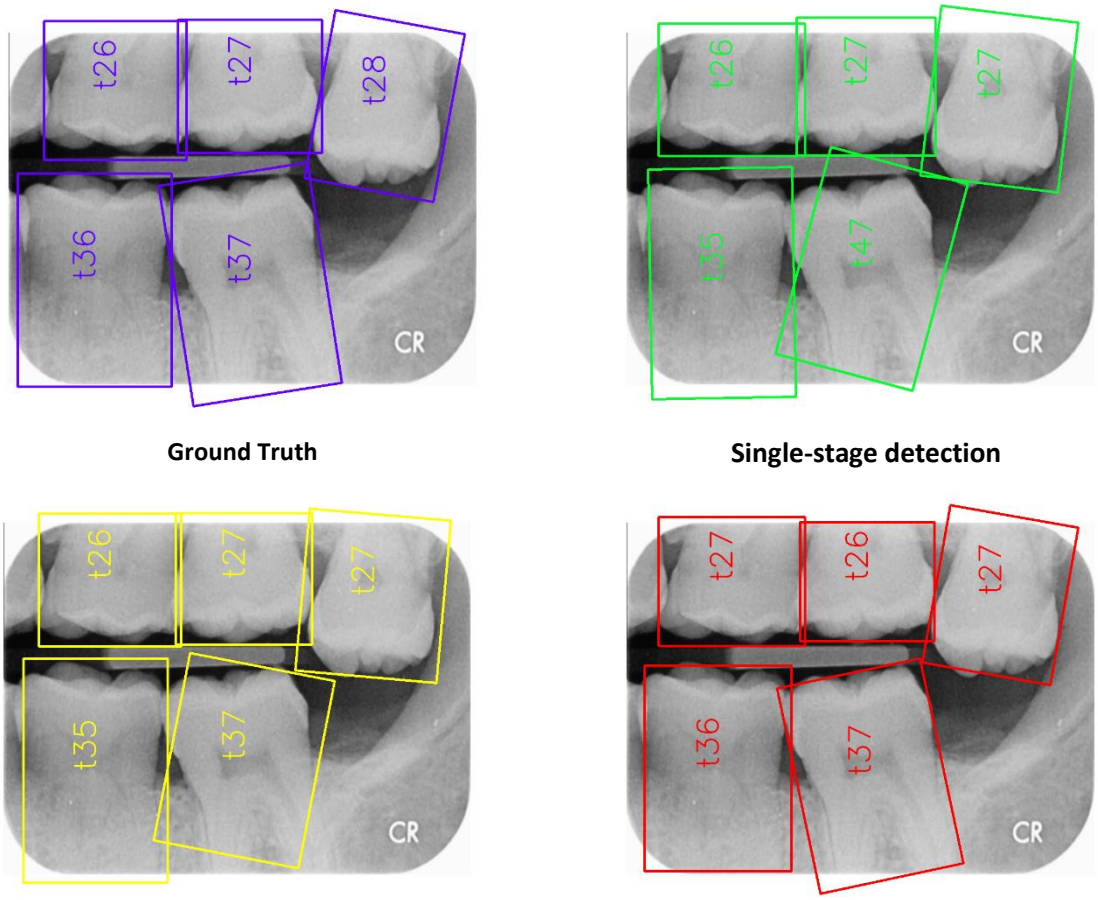


Figure 55 Detection results of the three approaches. The four-stage detection may give wrong prediction when there are no non-molar and polar teeth identified simultaneously in the same quadrant, as shown in the top left quadrant above.

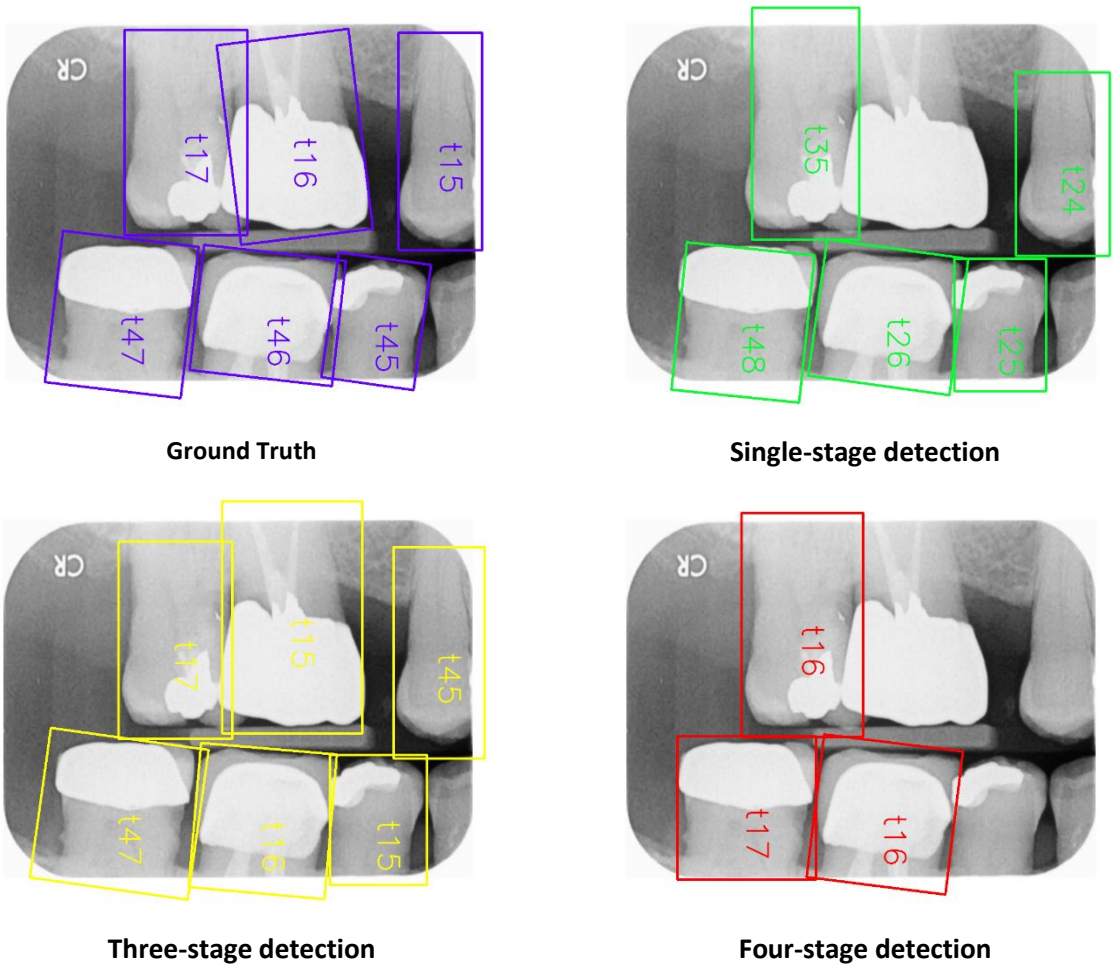


Figure 56 Detection results of the three approaches. The four-stage detection cannot make correct predictions if the detection results in any stage, e.g. stage 2 as shown above, makes wrong decision to identify upper (t_{17}, t_{15}, t_{16}) and lower quadrant teeth (t_{47}, t_{45}, t_{46}).

Chapter 5 Conclusions

5.1 Project Evaluation

The objectives of this project are to implement a novel approach for automating the detection of teeth in bitewing radiographs and identifying their corresponding tooth numbers, utilizing the data available from the DDD project. The project encompasses the following tasks within the research life cycle:

a. Data collection – A Zooniverse workflow is established to enable volunteers to contribute by annotating ground truth labels. User-friendly supporting materials are designed and published to facilitate the participation of individuals without prior dental knowledge.
b. Data cleansing – Although data collection through Zooniverse is an ongoing process, a data collection and cleansing pipeline is set up to continuously examine and refine the data for analysis.
c. Data analysis – The application of deep learning models for tooth numbering in bitewing images is a relatively unexplored area in the literature. This project employs YOLOv5 object detection models and introduces a novel stage-wise detection approach to enhance detection performance by incorporating the spatial relationships between pairs of teeth into the decision-making process. The analysis results also highlight the importance of separating the detection of left and right quadrants as a distinct step, which contributes to achieving more accurate detection performance.

5.2 Project Limitations and Future Improvements

Given the limited timeframe for preparing this dissertation, its analysis is based on a limited amount (693) of annotated images. It is expected that with the availability of more annotated images, the detection performance will improve further. In particular, the predicted bounding box angles shown in Figure 52 are not as accurate as expected and need further investigation. Moreover, teeth with special treatment patterns, as shown in Figure 53 and Figure 56 present challenges. However, the current training approach has demonstrated its potential to become more robust against such rarely encountered features.

In this dissertation, object detection is implemented using the one-stage YOLOv5 detector with oriented bounding box support. It would be worthwhile to try different implementations of two-stage detector algorithms [57] to compare their performance. Additionally, the approach proposed in this dissertation encodes the spatial relationships of bounding boxes into feature vectors for subsequent classification tasks. There are also many other methods to represent spatial relationships in the field of computer vision. Specifically, the tooth distance pair classifiers in Stage 3 only consider the distance relation between teeth, while the directional relation has not yet been taken into account [58]. It is expected that when the spatial relation descriptor includes both distance and directional elements, the type of misclassifications shown in Figure 55 can be avoided and the discencernibility between each non-molar teeth pair shown in Figure 47 can be enhanced.

Finally, the source codes of this project is publicly available at https://github.com/wongp1984/toothnumbering to facilitate collaborations and further research.

References

1. Dentalcare.com, “Intraoral Imaging: Basic Principles and Techniques”, [Online] Available at https://assets.ctfassets.net/u2qv1tdtdbbu/2Z7vLX9tbpdBT8rNpeQy6c/1f97eace95a097515c2bbcabc4fbc4bf/ce559_11-1-21.pdf

2. A. Salami, M. Al Halabi, I. Hussein, and M. Kowash (2017). “An Audit on the Quality of Intra-Oral Digital Radiographs Taken in a Postgraduate Paediatric Dentistry Setting”, [Online] Available at: https://www.longdom.org/open-access/an-audit-on-the-quality-of-intraoral-digital-radiographs-taken-in-apostgraduate-paediatric-dentistry-setting-.pdf.

3. Patil, S., Albogami, S., Hosmani, J., Mujoo, S., Kamil, M. A., Mansour, M. A., Abdul, H. N., Bhandi, S., & Ahmed, S. S. (2022). “Artificial intelligence in the diagnosis of oral diseases: applications and pitfalls”, Diagnostics, 12(5), 1029.

4. Banks, R. (2021). “Dental Disease Detection by Image Classification Of Radiographs”, Master of Data Science Thesis, University of Surrey

5. Lee, J.-H., Kim, D.-H., Jeong, S.-N., & Choi, S.-H. (2018). “Detection and diagnosis of dental caries using a deep learning-based convolutional neural network algorithm”, Journal of Dentistry, 77, 106–111.

6. Cantu, A. G., Gehrung, S., Krois, J., Chaurasia, A., Rossi, J. G., Gaudin, R., Elhennawy, K., & Schwendicke, F. (2020). “Detecting caries lesions of different radiographic extension on bitewings using deep learning. Journal of Dentistry”, 100, 103425.

7. Lin, X., Hong, D., Zhang, D., Huang, M., & Yu, H. (2022). “Detecting Proximal Caries on Periapical Radiographs Using Convolutional Neural Networks with Different Training Strategies on Small Datasets. Diagnostics”, 12(5), 1047.

8. Lian, L., Zhu, T., Zhu, F., & Zhu, H. (2021). “Deep learning for caries detection and classification. Diagnostics”, 11(9), 1672.

9. Wang, R., Naidu, A., & Wang, Y. (2021). Oral cancer discrimination and novel oral epithelial dysplasia stratification using FTIR imaging and machine learning. Diagnostics, 11(11), 2133.

10. Tuzoff, D. V., Tuzova, L. N., Bornstein, M. M., Krasnov, A. S., Kharchenko, M. A., Nikolenko, S. I., Sveshnikov, M. M., & Bednenko, G. B. (2019). “Tooth detection and numbering in panoramic radiographs using convolutional neural networks”, Dentomaxillofacial Radiology, 48(4), 20180051.

11. Zhang, K., Wu, J., Chen, H., & Lyu, P. (2018). “An effective teeth recognition method using label tree with cascade network structure”, Computerized Medical Imaging and Graphics, 68, 61–70.

12. Chen, H., Zhang, K., Lyu, P., Li, H., Zhang, L., Wu, J., & Lee, C.-H. (2019). “A deep learning approach to automatic teeth detection and numbering based on object detection in dental periapical films”. Scientific Reports, 9(1), 3840.

13. Miki, Y., Muramatsu, C., Hayashi, T., Zhou, X., Hara, T., Katsumata, A., & Fujita, H. (2017). “Classification of teeth in cone-beam CT using deep convolutional neural network”, Computers in Biology and Medicine, 80, 24–29.

14. Choi, H.-R., Siadari, T. S., Kim, J.-E., Huh, K.-H., Yi, W.-J., Lee, S.-S., & Heo, M.-S. (2022). “Automatic detection of teeth and dental treatment patterns on dental panoramic radiographs using deep neural networks”, Forensic Sciences Research, 7(3), 456–466.

15. Zooniverse, “Zooniverse Dental Disease Project”, [Online] Available at: https://www.zooniverse.org/projects/huhui/dental-disease-detection

16. Wikipedia, “FDI World Dental Federation notation”, [Online] Available at: https://en.wikipedia.org/wiki/FDI_World_Dental_Federation_notation

17. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., & others. (2015). “Imagenet large scale visual recognition challenge”, International Journal of Computer Vision, 115, 211–252.

18. Dalal, N., & Triggs, B. (2005). “Histograms of oriented gradients for human detection”, 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), 1, 886–893.

19. Uijlings, J. R., Van De Sande, K. E., Gevers, T., & Smeulders, A. W. (2013). “Selective search for object recognition”, International Journal of Computer Vision, 104, 154–171.

20. Le Cun, Y., Bottou, L., & Bengio, Y. (1997). “Reading checks with multilayer graph transformer networks”, 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing, 1, 151–154.

21. Dumoulin, V., & Visin, F. (2016). “A guide to convolution arithmetic for deep learning”, ArXiv Preprint ArXiv:1603.07285.

22. Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2021). “Dive Into Deep Learning”, arXiv preprint arXiv:2106.11342, [Online] Available at: http://d2l.ai/chapter_convolutional-neural-networks/conv-layer.html

23. CS231n: Deep Learning for Computer Vision, “CS231n Convolutional Neural Networks for Visual Recognition”, [Online] Available at: https://cs231n.github.io/convolutional-networks/

24. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). “Imagenet classification with deep convolutional neural networks”, Advances in Neural Information Processing Systems, 25.

25. Simonyan, K., & Zisserman, A. (2014). “Very deep convolutional networks for large-scale image recognition”, ArXiv Preprint ArXiv:1409.1556.

26. 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 (pp. 770-778).

27. Bouraya, S., & Belangour, A. (2021). “Deep learning based neck models for object detection: a review and a benchmarking study”, International Journal of Advanced Computer Science and Applications, 12(11).

28. Bouraya, S., & Belangour, A. (2021). “Object Detectors’ Convolutional Neural Networks backbones: a review and a comparative study”, International Journal, 9(11).

29. Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). “Feature pyramid networks for object detection”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2117–2125.

30. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). “Rich feature hierarchies for accurate object detection and semantic segmentation”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 580–587.

31. Girshick, R. (2015). “Fast r-cnn”, Proceedings of the IEEE International Conference on Computer Vision, 1440–1448.

32. Ren, S., He, K., Girshick, R., & Sun, J. (2015). “Faster r-cnn: Towards real-time object detection with region proposal networks”, Advances in Neural Information Processing Systems, 28.

33. N. Krishna (2022). “Understanding and Implementing Faster R-CNN: A Step-By-Step Guide”, Towards Data Science, [Online] Available at: https://towardsdatascience.com/understanding-and-implementing-faster-r-cnn-a-step-by-step-guide-11acfff216b0

34. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). “You only look once: Unified, real-time object detection”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 779–788.

35. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). “SSD: Single shot multibox detector”, Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14, 21–37.

36. R. Kundu (2023). “YOLO: Algorithm for Object Detection Explained”, V7 Labs, [Online] Available at: https://www.v7labs.com/blog/yolo-object-

detection#:~:text=While%20algorithms%20like%20Faster%20RCNN,a%20single%20fully%20connected%20layer.

37. Jocher, G. (2020). YOLOv5 by Ultralytics (Version 7.0) [Computer software]. https://doi.org/10.5281/zenodo.3908559 [Online] Available at: https://github.com/ultralytics/yolov5/releases

38. Ma, J., Shao, W., Ye, H., Wang, L., Wang, H., Zheng, Y., & Xue, X. (2018). "Arbitrary-oriented scene text detection via rotation proposals", IEEE Transactions on Multimedia, 20(11), 3111–3122.

39. Xie, X., Cheng, G., Wang, J., Yao, X., & Han, J. (2021), "Oriented R-CNN for object detection", Proceedings of the IEEE/CVF International Conference on Computer Vision, 3520–3529.

40. Xu, Y., & Bai, Y. (2022). "Compressed YOLOv5 for Oriented Object Detection with Integrated Network Slimming and Knowledge Distillation". 2022 3rd International Conference on Information Science, Parallel and Distributed Systems (ISPDS), 394–403. doi:10.1109/ISPDS56360.2022.9874105. [Computer software] Available at: https://github.com/hukaixuan19970627/yolov5_obb

41. Mahoor, M. H., & Abdel-Mottaleb, M. (2005). "Classification and numbering of teeth in dental bitewing images", Pattern Recognition, 38(4), 577–586.

42. Aeini, F., & Mahmoudi, F. (2010). "Classification and numbering of posterior teeth in bitewing dental images", 2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), 6, V6-66.

43. Lin, P.-L., Lai, Y.-H., & Huang, P.-W. (2010). "An effective classification and numbering system for dental bitewing radiographs using teeth region and contour information. Pattern Recognition", 43(4), 1380–1392.

44. Estai, M., Tennant, M., Gebauer, D., Brostek, A., Vignarajan, J., Mehdizadeh, M., & Saha, S. (2022). "Deep learning for automated detection and numbering of permanent teeth on panoramic images", Dentomaxillofacial Radiology, 51(2), 20210296.

45. Tan, M., Pang, R., & Le, Q. V. (2020). "Efficientdet: Scalable and efficient object detection", Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 10781–10790.

46. DR. JERRY MAYMI & ASSOCIATES A Better Life Starts with a Beautiful Smile, "6 Types of Dental X-Rays", [Online] Available at: https://drmaymi.com/6-types-of-dental-x-rays/#:~:text=Bitewings%20show%20most%20of%20the,little%20past%20the%20tooth%20root.

47. Wikipedia, “Palmer notation”, [Online] Available at: https://en.wikipedia.org/wiki/Palmer_notation

48. Wikimedia Commons, “File:Comparison of dental notations.svg”, [Online] Available at: https://commons.wikimedia.org/w/index.php?curid=90630906

49. ColoradoASDA, “Endo Access Quick Reference”, [Online] Available at: https://www.coloradoasda.org/e/endodontics/access-quick-reference

50. Trevett, B., “pytorch-image-classification”, [Online] Available at: https://github.com/bentrevett/pytorch-image-classification/blob/master/5_resnet.ipynb

51. PyTorch Lightning, “Learning Rate Finder”, [Online] Available at: https://pytorch-lightning.readthedocs.io/en/1.4.9/advanced/lr_finder.html

52. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. [Online] Available at: https://arxiv.org/pdf/1412.6980.pdf

53. R., Sovit, Gupta, V. (2022). “Performance Comparison of YOLO Object Detection Models – An Intensive Study”, [Online] Available at: https://learnopencv.com/performance-comparison-of-yolo-models/

54. ScikitLearn, “RandomForestClassifier”, [Online] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

55. Brownlee, J. (2020). “Cost-Sensitive SVM for Imbalanced Classification”, [Online] Available at: https://machinelearningmastery.com/cost-sensitive-svm-for-imbalanced-classification

56. Hughes, G. (2022). “Demystifying PyTorch’s WeightedRandomSampler by example”, Towards Data Science, [Online] Available at: https://towardsdatascience.com/demystifying-pytorchs-weightedrandomsampler-by-example-a68aceccb452

57. MMRotate Contributors. (2022). “OpenMMLab rotated object detection toolbox and benchmark” [Computer software]. https://github.com/open-mmlab/mmrotate

58. Wang, Y., Peng, H., Xiong, Y., & Song, H. (2023). “Spatial relationship recognition via heterogeneous representation: A review”. Neurocomputing, 533, 116–140. doi:10.1016/j.neucom.2023.02.053