Completely Contactless and Online Finger Knuckle Identification

Zhenyu ZHOU

June 23, 2022

1 Whole Image Translated and Rotated Triplet Loss Function

Because our neural networks were trained using the architecture of triplet network [14], we used translated and rotated triplet loss function (TRTLoss) as loss function to update convolutional kernel of our models. When we match two finger knuckle in the contactless scenario, they will not only translate but also will rotate with local deformable transformation. For solving it, we add rotation based on translation to compose a more robust transformation. As for the TRTLoss function, we rotate feature maps while translate feature maps based on the Soft-Shifted Triplet Loss Function [9].

In generally, the TRTLoss is still a variant of triple loss, so that the TRTLoss can be written as a format of triple loss function as the Equation 1. As for the N, it means the number of triplet during training iteration, and $F(I^a)$ is the output feature map of input anchor image I^a through neural network. The hard margin parameter m can determine the distance between different class cluster by pushing them away.

$$TRTLoss = \frac{1}{N}\sum_{i}^{N}[L(F(I_i^a), F(I_i^p)) - L(F(I_i^a), F(I_i^n)) + m]_+ \tag{1}$$

We add translation and rotation transformation to deal with local deformable of finger knuckle. In order to adapt to tasks with different degrees of deformation, and balance performance and speed, we set translation and rotation ranges as a hyperparameter. The $L(F_1, F_2)$ will get the minimal distance of two feature maps $D_{w,h,\theta}(F_1, F_2)$ after translation and rotation in the range $-W{\le}w{\le}W, -H{\le}h{\le}H, -\Theta{\le}\theta{\le}\Theta$. Meanwhile, the distance $D_{w,h,\theta}(F_1, F_2)$ calculates the pixel-wise MSE value when feature map is translated w pixel along x-axis and h pixel along y-axis and rotated θ angle in the Equation 3.

$$L(F_1, F_2) = \underset{-W\le w\le W, -H\le h\le H, -\Theta\le\theta\le\Theta}{min} D_{w,h,\theta}(F_1, F_2) \tag{2}$$

$$D_{w,h,\theta}(F_1, F_2) = \frac{1}{|C_{w,h,\theta}|}\sum_{(x,y)\in C_{w,h,\theta}} (F_1^{(w,h,\theta)}[x,y] - F_2[x,y])^2 \tag{3}$$

In terms of $C_{w,h,\theta}$, it represents the common region between two feature maps after one feature map shifted along x-axis with w, shifted along y-axis with h, and rotated with θ. As for the (F_a, F_p) pair, we can assume when the F_a is rotated angle of θ_{ap} and shifted with (w_{ap}, h_{ap}) pixels can get the minimal $D_{w_{ap},h_{ap},\theta_{ap}}(F_a, F_p)$, then $L(F_a, F_p) = D_{w_{ap},h_{ap},\theta_{ap}}(F_a, F_p)$. Meanwhile, with the $(w_{an}, h_{an}, \theta_{an})$, the (F_a, F_n) pair can get the minimal $D_{w_{an},h_{an},\theta_{an}}(F_a, F_m)$.

$$\frac{\partial Loss}{\partial F_i^p} = \begin{cases} 0, if(x,y) \notin C_{w_{ap},h_{ap},\theta_{ap}} \; or \; Loss = 0 \\ \frac{-2(F_i^a[[x_{w_{ap}}, y_{h_{ap}}]*M(\theta_{ap})] - F_i^p[x,y])}{N|C_{w_{ap},h_{ap},\theta_{ap}}|}, otherwise \end{cases} \tag{4}$$

The $M(\theta_{ap})$ is the rotation matrix.

$$\frac{\partial Loss}{\partial F_i^n} = \begin{cases} 0, if(x,y) \notin C_{w_{an},h_{an},\theta_{an}} \; or \; Loss = 0 \\ \frac{-2(F_i^a[[x_{w_{an}}, y_{h_{an}}]*M(\theta_{an})] - F_i^n[x,y])}{N|C_{w_{an},h_{an},a_{an}}|}, otherwise \end{cases} \tag{5}$$

As for the $F_i^a[x, y]$ derivation, because we shift and rotate the anchor in the above formula, we can inversely shift and rotate the positive and negative input feature.

$$\begin{aligned} \frac{\partial Loss}{\partial F_i^a[x,y]} = -&\frac{\partial Loss}{\partial F_i^p[[x - w_{ap}, y - h_{ap}] * M(-\theta_{ap})]} + \\ &\frac{\partial Loss}{\partial F_i^n[[x - w_{an}, y - h_{an}] * M(-\theta_{an})]} \end{aligned} \tag{6}$$

2 Experiments and Results

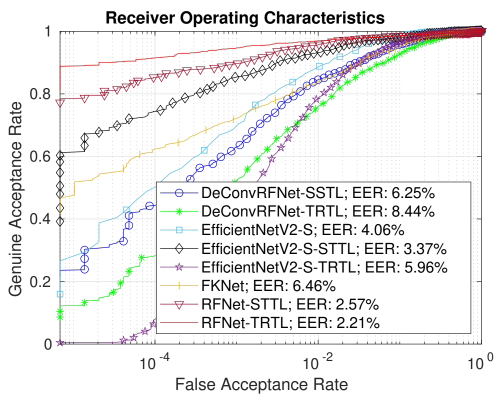
In this part, We will compare Translation and Rotation Triplet Loss (TRTL) and Soft-Translation Triplet Loss [9] performance on different finger knuckle dataset. Meanwhile, we will also compare with the state-of-the-art FKNet and EfficientNetV2. All the experiment will use the finger knuckle segmented by Yolov5 as training set and testing set, and in the Section 4, we will compare the impact of the finger knuckle segmented by YOLOV5 and the finger knuckle provided by these databases on the performance. As for the DeConvRFNet, we just change the RFNet convolution layer with deformable convolution layer. EfficientNetV2-S is the original classification model, we keep the same architecture and just change the FC layer of the head part to convolution layer for fitting TRTL and STTL. Because I want to follow the experiment protocol of FKNet, the FKNet just use one smaple of Index Finger Knuckle of Hand Dorsal Image Database to train due to classification model. However, my model uses triplet loss, so I need positive, negative and anchor samples. Firstly, all models are pre-trained on the Finger Knuckle V1 Database, and then fine-tuning on the corresponding finger knuckle database with bigger hard margin. On the Finger Knuckle V1 Database, it contains 512 subjects, and each of subjects offer 5 finger knuckle samples. In this kind of situation, I use three samples of these five samples to pre-train my models, meanwhile, as for the rest two samples as the validation dataset.

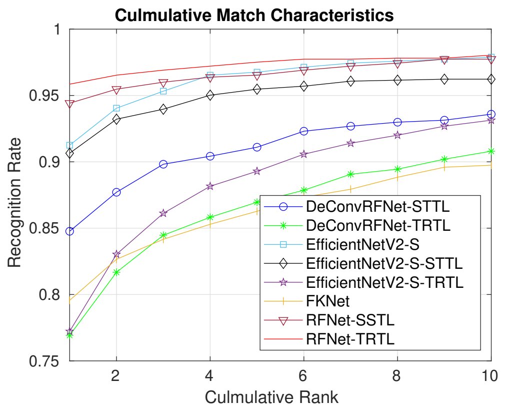
I also continue to train the DeConvRFN (change the RFN-128 convolution layer with deformable convolution) and EfficientNet, the loss converged to the local minimal point. As for the EfficientNet, I change the original EfficientNetV2-S to fit my application. It has 9 stages in totally, the 9th stage is classification task with FC layer, so I replace it with convolution layer for output feature maps. Meanwhile, I delete the stage7 and stage8, and change stage3 and stage4 with stride 1. As for the RFNet, I use different loss to train the model, such as whole image rotation and shift, image blocks rotation and shift, and whole image shift. And I also compare the performance with the FKNet.Meanwhile, for getting the original EfficientNetV2-S model performance, I also added corresponding experiments with EfficientNetV2-S. Because the EfficientNetV2-S model is a classification model, I calculate MSE of two feature vectors of last layer of model as matching score. And as for the input data, I follow the same method of FKNet, each image will rotate from $-10°$ to $10°$ to increase the amount of training data.

2.1 Within-Database Experiments

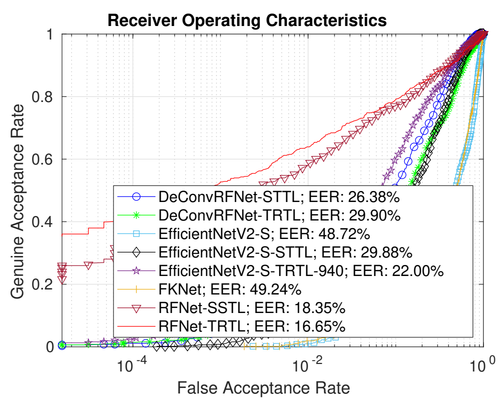
2.1.1 Finger Knuckle V3 Database with Deformation

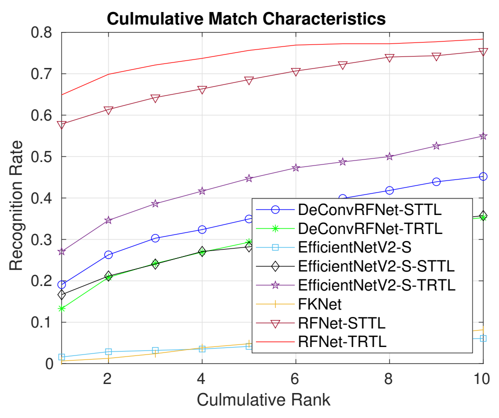
The Finger Knuckle V3 Database have 1-104 subjects that have two session samples, and the rest subjects of first session 105-221 just offer one session samples. So as the first experiment, I firstly fine-tuned my model on the second session of 1-104 subjects, and test on the first session 1-104 subjects. So it will have $104 * 6 = 624$ genuine matching scores, and have $104 * 103 * 6 = 64272$ imposter matching scores. From the below figure, if the false accept rate is below 10^{-2}, the RFN-128-WRS is better than the RFN-128-WS. I also use the FKNet to train on this database, and the performance of FKNet is not better than the RFNet depend on the ROC figure. From the CMC and ROC, each model with WRS is better than WS on this dataset. For the ROC curve, I add EfficientNetV2-S model performance.





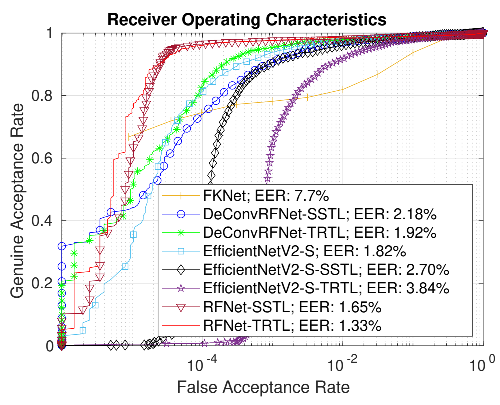
As for the two-session protocol on the database. I should fine-tune my model on the 105-221 subjects, and use two-session protocol to evaluate my model performance on the 1-104 subjects dataset. In totally, it will generate $104 * 6 = 624$ genuine scores, and $104 * 103 * 6$ imposter scores. However, the FKNet is a classification task, and the output number classes should be same when training and testing. So the two session protocol experiment is not fit for FKNet. If the FKNet train on the 105-221 subjects and test on the 1-104 subjects with two sessions, the classes is different.

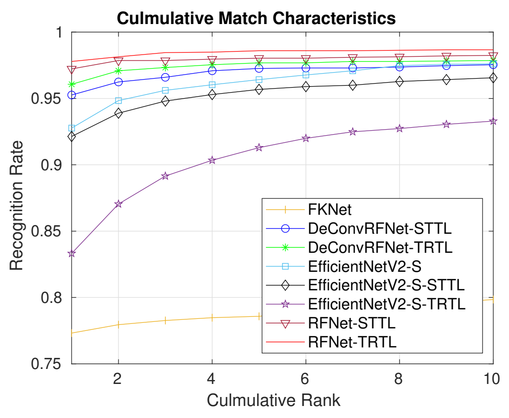




The two-session protocol will use the session1 as the probe and use the session2 as the enrollment. As for the genuine matching scores, each sample of a subject will choose the minimal matching score when compare to the rest samples. In this kind of situation, it will have $104x6$ genuine matching scores. Meanwhile, as for the imposter matching scores, it will also choose the minimal value result in $104 * 103 * 6$ imposter matching scores on the confusion matrix.

2.1.2 Index Finger Knuckle of Hand Dorsal Image Database



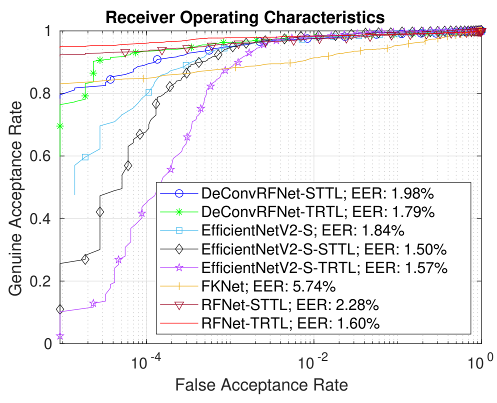


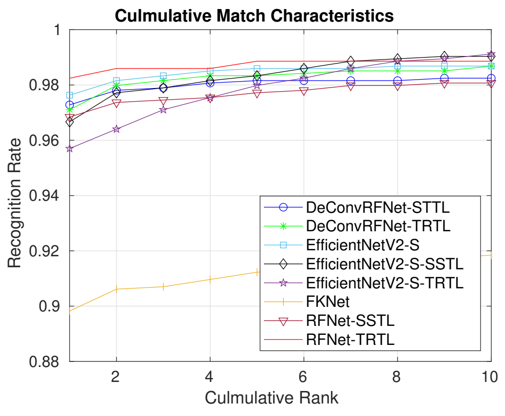
For the ROC curve, I add EfficientNetV2-S model performance. Update CMC Curve and ROC Curve with EfficientNet, DeConvRFNet and RFNet. As for the experiment, the dataset totally contains 712 subjects, and I use the segmented Index finger knuckle as my dataset. And I fine-tuned my model on the first sample of each subject, and then use the rest four sample as the testing dataset. At the testing process, it has $712 * 4 = 2848$ genuine matching scores, and has $712 * 711 * 4 = 2024928$ imposter matching scores. The performance of RFN-128-WRS and RFN-128-WS is similar, but the RFN-128-WS is slightly better than RFN-128-WRS depend on the EER value. And we can get an information that the RFNet is better than the rest network in the ROC figure, including the FKNet.

2.1.3 2D Samples of 3D Finger Knuckle Database

First experiment on the database is to use the one session 190 subjects image to fine-tune models and then to test on the another session 190 subjects. It has $190 * 6$ genuine matching scores and

190 $*$ 189 $*$ 6 imposter matching scores. From the result, we can see that these RFN-128-WRS, RFN-128-WS, EfficientNetV2 can get very high matching accuracy. Meanwhile, the RFN-WRS has the minimal EER value among these models. As for the FKNet performance, it gets a very bad result on the 2D images of 3D finger knuckle. I think I have fully trained the FKNet. Maybe the model is overfitting on the training dataset. Updated ROC Curve and CMC Curve with RFNet, EfficientNet and DeConvRFNet. And each model with WRS is better than WS. For the ROC curve, I add EfficientNetV2-S model performance.



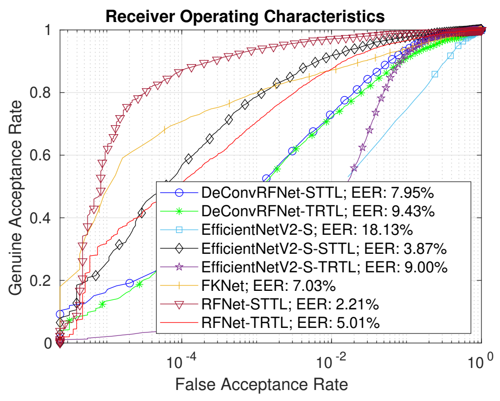


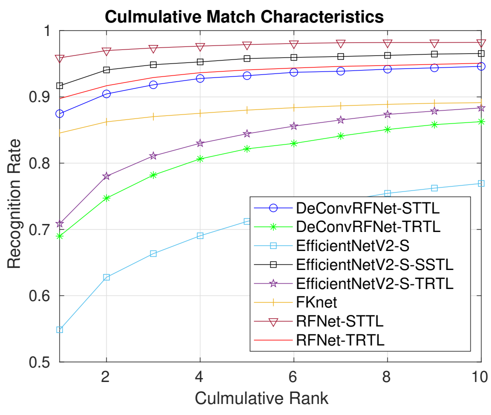
And then use the two session protocol. I use the rest samples of session1, and it has 191-228 subjects. In this kind of situation, the training dataset is too small. The two session protocol will test on the 190 subjects, these subjects can offer two session samples. Due to the training set is too small, so the matching performance is not very good. As for the FKNet, it cannot fit on two session protocol due to classification task.

2.2 Cross-Database Experiments

I firstly pre-trained my models on the Finger Knuckle V1 Database, and then fine-tuned models on the Finger Knuckle V3 Database (with deformable). I use these kind training method, and use these models to test performance on the Index Finger Knuckle of Hand Dorsal Image and Tsinghua Finger Knuckle Database as a cross database experiment. The label in the finger curve, the content in parentheses indicates the training samples. Such as RFN-WS(1-104), it uses 1-104 subjects of Finger Knuckle V3 Database to train models. Updated ROC Curve and CMC Curve with RFNet, EfficientNet and DeConvRFNet. For the ROC curve, I add EfficientNetV2-S model performance.

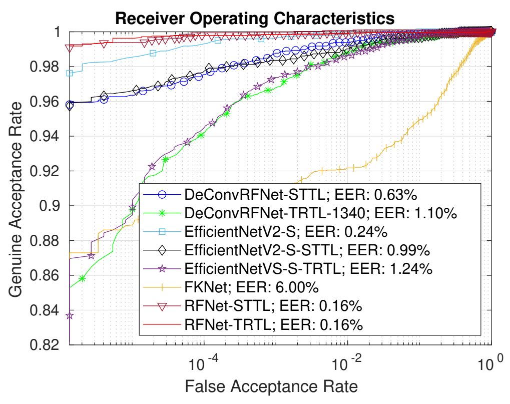
2.2.1 Index Finger Knuckle of Hand Dorsal Image

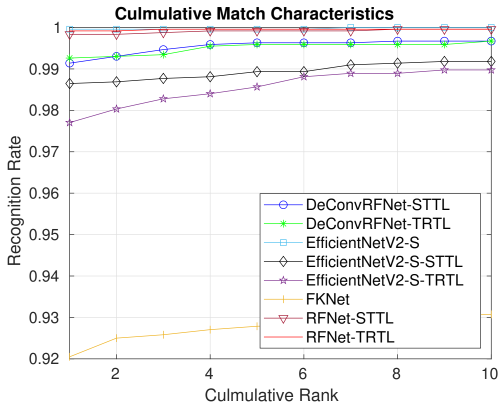




The database totally has 712 subjects, and each subject has 5 samples. Therefore, it will have $712 * 5$ genuine matching scores and $712 * 711 * 5$ imposter matching scores. From the cure, the performance of RFN-WS and RFN-WRS is similar, and the RFN-WS is slightly better than RFN-WRS while using the same training samples. Updated ROC Curve and CMC Curve with RFNet, EfficientNet and DeConvRFNet. For the ROC curve, I add EfficientNetV2-S model performance.

2.2.2 Tsinghua Finger Knuckle Database



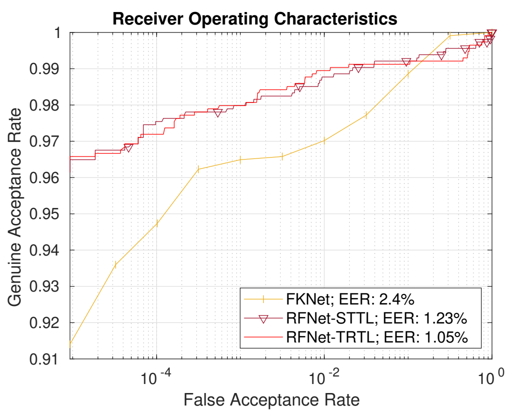


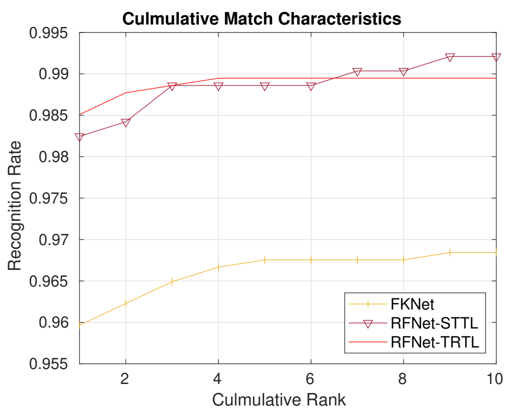
The database has 610 subjects, and each subject can offer 4 samples. Then as the cross database experiment, it will have $610 * 4$ genuine matching scores and $610 * 609 * 4$ imposter matching scores. In this database, all models can get very high matching performance from the table and figure.

3 3D Finger Knuckle of 3D Finger Knuckle Database

I have use the matlab code that offered by the FKNet to generate the 3D finger knuckle images for getting the depth information. But it is different that the input image size. The FKNet will resize the original image size $148 * 212$ to $70 * 100$ as the testing dataset, and crop from the

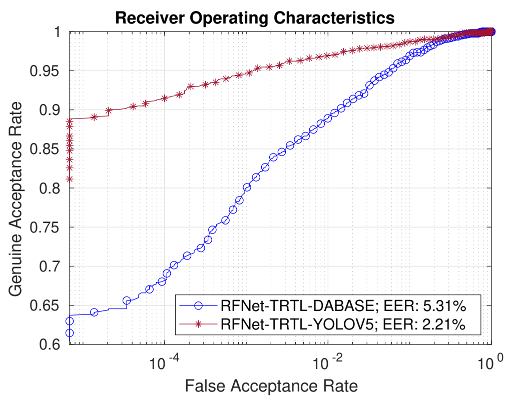
$70 * 100$ to $48 * 80$ as the training dataset. As for RFNet, I just use the original image as the input data. Then the experiment protocol will generate $190 * 6$ genuine matching scores, and $190 * 189 * 6$ imposter matching scores. From the experiment result, we can get that the RFNet is the best model for the 3D Finger Knuckle Database.





4 Compare Performance on Yolov5 Segment and Dataset Offter Finger Knuckle

This section aim to compare quality of finger knuckle between Yolov5 segmented and dataset offered. Because the segmented finger knuckle on the 3D Finger Knuckle Dataset already have high quality, I mainly test on the Index Finger Knuckle of Hand Dorsal Dataset and the Finger Knuckle Dataset V3 (with deformable).



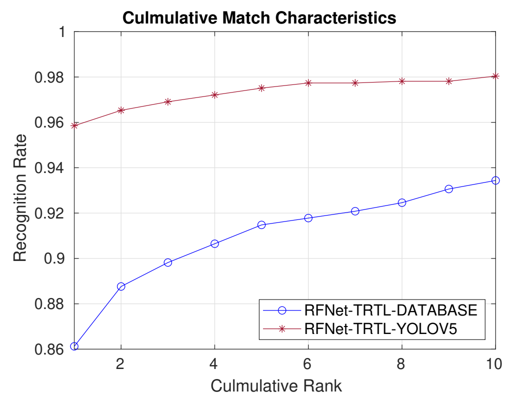


Figure 8: Compare performance on the Finger Knucle V3 Dataset (with deformable)

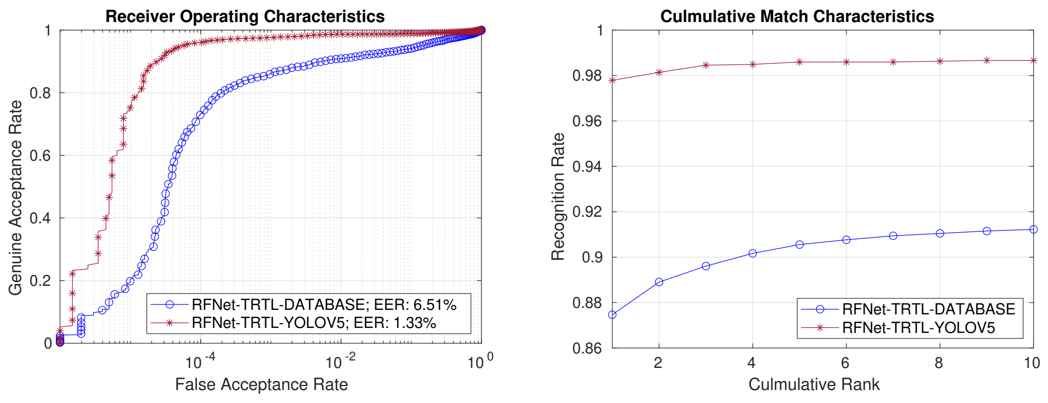


Figure 9: Compare performance on the Index Finger of the Hand Dorsal Image Database.

From the above figures, we can clearly get the conclusion that quality of segmented finger knuckle of YOLOV5 is better than the segmented finger knuckle of dataset based on the same model. Especially on the Finger Knuckle V3 Dataset, the EER value can drop from 5.31% to 2.21%. Because in the previous experiments, except for Finger Knuckle V3 and 3D Finger Knuckle Dataset, other datasets were segmented using the YOLOV5 model. But with Yolov5 segmented finger knuckle, the RFNet performance is slightly higer than the local feature descriptors based on key points matching [6], and performance higher than the paper [7]. If we want to compare different method performance, I think we should use same dataset. In this kind of situation, the method of [6] maybe will ger higer performance on the segmented finger knuckle by YOLOV5.

5 Online Contactless Finger Knuckle Identification

With TRTL loss, the RFNet [9] can outperform state-of-the-art methods. In the previous section, we have estimated its verification and identification performance on different public finger knuckle database, including within-db and cross-db experiments. As for a completely contactless and online finger knuckle identification, the finger knuckle detector is a very important module for automatically detect and segment finger knuckle region. A suitable ROI segmentation method can improve efficiency and accuracy of the matching algorithm. There is an important point is the angle of the finger knuckle,

the target object has an angular rotation problem, the matching process generally requires simultaneous matching in multiple angular ranges, and the number of pixels to be computed increases exponentially. If the algorithm of ROI segmentation is accurate enough and the accuracy of the rotation is also high enough, this will naturally improve the detection efficiency. For the problem of improving the matching accuracy, if the accuracy of the region of interest is high enough, the background interference information extracted will be correspondingly more petite, and the pixel signal of the target object obtained is enough, the signal-to-noise ratio will be high, and the matching accuracy will be improved accordingly.

However, most of the current finger knuckle segmentation approaches are based on contact finger knuckle segmentation. Even for the contactless finger knuckle segmentation problem, their [7], [3] approach is to fix the finger knuckle position in the image when taking the finger knuckle data, and if the finger appears in the image in a different position or if there are multiple fingers, the problem of not detecting the finger knuckle or missing the finger knuckle is likely to occur.

This segmentation is prone to errors, and this is not the only problem. Most importantly, the traditional segmentation algorithm cannot correctly segment the finger knuckles in the presence of complex background interference, multiple finger knuckles in the same field of view, obscured finger knuckles or bent finger knuckles. Since the subsequent operations of the finger knuckle recognition algorithm are based on the segmented image, the segmentation of finger knuckles affects the accuracy of finger knuckle recognition. It is vital to improving the efficiency of segmenting the finger knuckle region in any scenario.

5.1 Contactless Finger Knuckle Detection

This paper studies the finger knuckle region and uses the corresponding finger knuckle crease patterns as features of the finger knuckles. So the region of interest to be extracted is the part that can represent the skin crease patterns on the back of the finger. As mentioned in Section 3.1, it is difficult to use traditional object segmentation methods to automatically segment finger knuckles for applications such as in the wild. Even though there have been corresponding conventional algorithms implemented to automatically segment the finger knuckle region independent of the finger knuckle position pose [7], the method used in this paper requires fixing the position of the finger knuckles appearing in the image and is a Fixed ROI extraction.

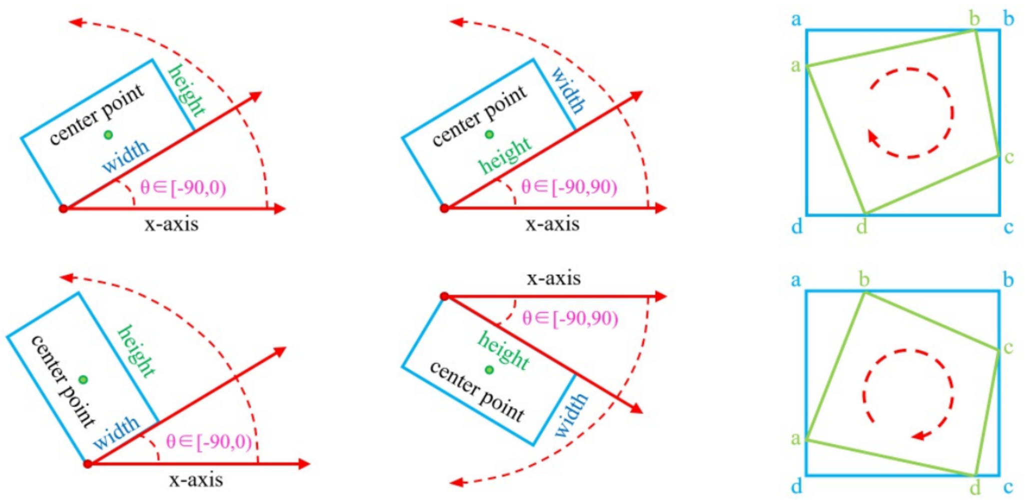
In order to solve the problem of finger knuckle detection in the real world, this paper chooses to use neural network models instead of traditional segmentation algorithms. Models of neural network models for object detection have achieved great success, whether it is the sliding window detection algorithm, the 2-stage series of R-CNN models [5], [4], [13], or the 1-stage YOLO series [12], [10], [11], [1] and SSD models [8] up to the current position, and even the anchor-free based object detection algorithm [19] as well. Each of these models has its advantages. For the 2-stage model, the object detection accuracy is guaranteed, the 1-stage based model is a speedup based on the positive accuracy, and the anchor-free is a further improvement in the detection speed. In this paper, the latest version of the YOLO model series, YOLOv5 [18], is used as the network model for finger knuckle detection because the YOLO series is famous for its fast detection speed and high accuracy. The module adopts various latest network modules, and the YOLOv5 model has a variety of model structures to meet different accuracy and speed requirements. For the YOLOv5 model, the number of layers of each submodule is varied to cope with different speed and accuracy requirements, while the overall structural component modules remain unchanged. The YOLOv5 neural network model used as a finger knuckle detection has good results in the wild, the prediction bounding boxes of YOLOv5 are based on a horizontal bounding box for regression.

Although the results have been good for finger knuckle detection, they are not sufficient for the segmentation operation of finger knuckles. There are two main problems for rotating finger knuckles segmentation. The first problem is that when the horizontal bounding box is used to predict the object, the size of the bounding box is a minimum external horizontal rectangle for the size of the object. In such a case, when there is a rotation of the object, which is not horizontal for the picture, the horizontal box will have much more background for the detected object, and in the case that the target object is crowded, it is easy to eliminate the neighbouring of the horizontal bounding boxes are eliminated. As shown in Figure **??** (a), due to the rotation of the finger and the density of the finger, the prediction bounding box of the middle finger knuckle and the major finger knuckle of the ring finger have overlapped, so it is easy to be deleted during the non-max suppressing process, even if CIOU [21] or DCOU [22] is used. As shown in Figure **??** (b), after changing the threshold of IOU in the non-max suppressing process, the major finger knuckle of the ring finger is not detected.

The second problem is that for the segmentation operation if the horizontal box is used directly to carry out the segmentation operation, it is the same as shown in Figure **??** (c)-(i), which is the region of interest of the finger knuckle corresponding to the segmentation in Figure **??** (a). However, the segmentation to the background interference has a lot, and it is not considered a good finger knuckle segmentation operation. In order to solve the above problem, a rotated prediction bounding box was then used to deal with the background region minimally while predicting the target object's position, and the rotated bounding box could be used as the orientation detection finger knuckle. The subsequent finger knuckle matching process also needs to use the finger knuckle orientation information for the correction operation.

6 Orientation Bounding Box Prediction Based on YOLOv5

6.1 Rotated Bounding Box Prediction



(a) Five-parameter method with 90° angular range.
(b) Five-parameter method with 180° angular range.
(c) Ordered quadrilateral representation.

Figure 10: Three definition of orientation bounding boxes. [20]

There are many representations of defining a rotation bounding box, but here are only a few common ways to define an arbitrary angle box. This is because the other definition methods are all the same. The three methods of defining the rotation rectangle are listed in Figure 3.3. The angle definition method in Figure 10 (a) is the definition method used by OpenCV. The angle is the acute angle formed by the rectangle and the x-axis, and the side forming the acute angle with the x-axis is defined as the rectangle's width, and the other side is defined as the length. The angle orientation is $[-90°, 0°)$. The definition method of Figure 10 (b) is also called the long side definition method, and the angle is the angle between the x-axis and the long side of the box, so the angle range is $[-90°, 90°)$, and Figure 10 (c) is listed as the ordered quadrilateral definition method, where the leftmost point is the starting point and is ordered counterclockwise. If the box itself is horizontal, the top left corner is taken as the starting point. As analyzed in this paper [20], since the prediction box of the YOLO series is regressed from the horizontal anchor, for the rotated box defined by OpenCV, there is the problem of angular periodicity and the exchangeability of the edges of the box, which leads to a relatively large loss value. While the loss of the long side definition is slightly more straightforward and only comes from the periodicity with the angle. So this paper also uses the long side representation to represent the rotated rectangle, and the data is labelled by the roLabelImg [2] tool. However, the difference is the angle x-axis of roLabelImg [2] and the clockwise angle of the long edge, ranging from $[0°, 180°)$.

The existence of angular periodicity and the exchangeability of edges in the angle regression leads to loss function values becoming large. To solve this problem, consider using classifying the angles as the categories are finite. Since there is no edge exchangeability as a loss value using the long side definition and the rotated box labelled by roLabelImg is also defined using the angle definition of $[0°, 180°)$, we can classify the angles into 180 categories. Generally, for the classification problem, we can use one-hot hard coding to solve it. Usually, if the angle of ground truth is $0°$, then the loss values predicted to be $1°$ and $179°$ should be approximately equal and not too much different. However, for one-hot, a prediction of $1°$ and $179°$ lead to a large loss. A periodic coding method should be used to compensate for the large difference between the loss values of $0°$ and $179°$. This can be solved by using the Circular Smooth Label (CSL) [20] soft coding.

$$CSL(x) = \begin{cases} g(x), & \theta - r < x < r + \theta \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

Formula 7 $g(x)$ is the window function, r is a window function of the radius. For the window function, in general, to have four properties, the first point is the periodicity, the second point is the symmetry, the third point is the maximum value to $g(\theta) = 1$, the fourth point also needs to be in the window function on the left side of the symmetry axis is an increasing function, in the right side of the symmetry axis is a decreasing function. Furthermore, in this paper, we used the Gaussian function for the Equation 7 window function, a commonly available function, and used a window radius of 6 to smooth the labels when used.

6.2 Oriented Bounding Boxes Based on YOLOv5

Output tensor channels

As used in the CSL method, the angles are classified into 180 categories using smoothing labels. For the output head of YOLOv5, the number of channels corresponding to each output tensor should be increased by 180, which becomes $3 * (4 + 1 + 2 + 180) = 561$ channels. For the class number, the task of this paper is to detect two classes of major and minor finger knuckle, so the 2 in the equation represents two classes. The output head of YOLOv5 is three tensors with different scaling, so the final output is 561 channels. If the input image is $608 * 608 * 3$ dimension tensor, which is downsampled by $1/32$ to get $19 * 19 * 561$ dimension tensor, $1/16$ to get $38 * 38 * 561$ dimension tensor, and $1/8$ to get $76 * 76 * 561$ dimension tensor.

Loss function

After the output uses the loss function to calculate the loss for network training, the original YOLOv5 loss function can have three components. The formula can be simply written as $Loss = CIOU\_Loss + Loss_{conf} + Loss_{class}$. Since the oriented bounding box is based on the modification of YOLOv5, only the angle classification is added more. For the multi-classification problem, the loss function is generally written using Cross-Entropy to calculate the loss so that for a prediction bounding box, only one classification can exist. However, there may be more than one target object falling inside a bounding box simultaneously, only the angles are different, so a lattice has to predict multiple classification problems, so Binary Cross Entropy (BCE) is used as the loss function here. So the total loss function is as expressed in Equation 8, with the addition of $Loss_{angle}$ to YOlov5. Loss can now be divided into four parts. For the bounding box used $CIOU\_Loss$, the remaining three parts of the loss are based on the loss of BCE to enter the calculation.

$$Loss = CIOU\_Loss + Loss_{conf} + Loss_{class} + Loss_{angle} \tag{8}$$

For the regression process of the bounding box, the CIOU loss used [21], CIOU is an excellent solution to the regression problem of the bounding box, increasing the aspect contrast of the bounding box, as expressed in Equation 9, where $\rho()$ is the Euclidean distance between the centre point of two bounding boxes, and c is the diagonal length of the minimum outside the rectangle of the two bounding boxes. Moreover, the α parameter is the weight parameter, and the calculation formula is derived from that shown in Equation 10, and the ν parameter is the length-width contrast coefficient of the bounding box performed.

$$CIOU\_Loss = 1 - (IOU - \frac{\rho^2(b, b^{gt})}{c^2} - \alpha\nu) \tag{9}$$

$$\begin{aligned} \nu &= \frac{4}{\pi^2}(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w^p}{h^p})^2 \\ \alpha &= \frac{\nu}{(1 - IOU) + \nu} \end{aligned} \tag{10}$$

The confidence level indicates whether there is a target object with detection in each small cell of the feature map output by the output head of YOLOv5. As listed in Equation 11, the BCE loss function is also used for the loss calculation of confidence Because each feature map needs to be traversed for each grid, S represents the size of the feature map, it is necessary to traverse S^2 times, and each grid corresponds to B anchors. Inside the Equation 11, the I_{ij}^{obj} and I_{ij}^{noobj} that for the i row j column of the grid inside whether there is a bounding box.

$$\begin{aligned} Loss_{conf} = \sum_{i=0}^{S^2}\sum_{j=0}^{B} I_{ij}^{obj}[\hat{C}_i log(C_i) + (1 - \hat{C}_i)log(1 - C_i)] - \\ \lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{B} I_{ij}^{noobj}[\hat{C}_i log(C_i) + (1 - \hat{C}_i)log(1 - C_i)] \end{aligned} \tag{11}$$

The formula for calculating the category loss for angles is the same as for categories of classes, as listed in Equation 12 and 13.The difference is that for the class loss calculation, c needs to iterate over the number of predicted categories (here twice, since major and minor finger knuckle are detected), while a needs to iterate 180 times (180 categories for the angle classification).

$$Loss_{class} = \sum_{i=0}^{S^2} I_{ij}^{obj} \sum_{c \in classes} [\hat{P_i(c)} log(P_i(c)) + \\ (1 - \hat{P_i(c)}) log(1 - P_i(c))] \tag{12}$$

$$Loss_{angle} = \sum_{i=0}^{S^2} I_{ij}^{obj} \sum_{a \in [0,180)} [\hat{P_i(a)} log(P_i(a)) + \\ (1 - \hat{P_i(a)}) log(1 - P_i(a))] \tag{13}$$

Non-Maximize Suppressing for Oriented Bounding Box

Since the prediction process of the rotating bounding box frame is based on the horizontal bounding box with an extra angle prediction, it is separated into two parts, one part is the regression process of the horizontal bounding box, and the other is the rotation angle prediction process. A serious problem will arise if the source code is used directly for non-maximal suppression based on the horizontal bounding box IOU. Because the IOU values of the two rotated bounding boxes are shallow, the IOU values of their corresponding two horizontal bounding boxes are enormous, so the correct rotated bounding boxes are eliminated in the process of non-maximum suppression. So in the prediction for the rotated boxes, the horizontal and angular information needs to be fused to form a true rotated rectangle. This process can be converted by OpenCV, which will transform the rotated rectangle expressed by five parameters to get the coordinates of four vertices of the rotated rectangle, which are the first vertices starting from the lowest point in the far right, and then rotating clockwise to get the remaining three vertices. After getting the coordinates of the four vertices of the rotated rectangle, we can use OpenCV to calculate the area size of the intersection of the two rotated rectangles so that we can use the IOU of the rotated rectangle for non-maximum suppression.As shown in Figure **??**, after using the non-maximum suppression of the rotating bounding box, the detection of 86 major finger knuckles and 22 minor finger knuckles was reduced to the final 14 major finger knuckles and 6 minor finger knuckles, respectively.

The rotated bounding box's final prediction result is the finger knuckle's prediction result, and the ROI extraction of the finger knuckle region can be performed using this rotated bounding box. Since it is a rotated rectangle, the background part is much less than the original horizontal bounding box, and the finger knuckle feature area can be increased accordingly. It is possible to achieve a high level of fully automatic ROI extraction model. Moreover, because it is a rotating rectangle, the current direction of the target object can be known, and the ROI area can be mounted using this direction angle, which prepares sufficient conditions for subsequent feature extraction and feature matching.

7 Experiments and Results

Finger Knuckle Detection Accuracy

The CSL is integrated into the YOLOv5 model to smooth the angular classification, called YOLOv5-CSL model, and the BCE is used to calculate the angular classification prediction loss. YOLOv5 has multiple sub-models, mainly by increasing the number of censored convolutional layers and residual layer modules, but the overall model framework does not change to cope with different hardware, accuracy requirements, and detection speed requirements. From the official data given, we can see the performance difference between these variations of the YOLOv5 model from the Table **??**. If speed is pursued, then the lightest YOLOv5n can be used. Conversely, if accuracy is pursued, then YOLOv5x can be used.

Model	Inference Time/ms (1024x1024)	Number of Layers	mAP^{val} 0.5	AP of Major FK	AP of Minor FK
YOLOv5x-CSL	41.395	407	**89.9**	**89.6**	**90.1**
YOLOv5m-CSL	36.252	263	85.7	88.9	80.4
YOLOv5s-CSL	32.683	213	39.6	43.9	35.4
YOLOv4	25.992	161	70.7	83.6	57.7

Table 1: Comparison of the accuracy of the different models of the YOLO series for the detection of the finger knuckle.The calculated values of mAP were measured at a detection threshold of 0.4 as well as an IOU threshold of 0.5.

This paper uses these YOLOv5s, YOLOv5m, YOLOv5x and YOLOv4 models to train the labelled data. The YOLOv4 model is a regression on the horizontal bounding box, while the remaining YOLOv5 model is a regression on the rotated bounding box, called YOLOv5-CSL. The data are obtained using a crawler dataset and partly from a publicly available dataset, with 2580 training images. Among them, 100 images were randomly selected to form the test image set. For these 2580 training images, 2347 images are obtained from the crawler and the rest from the public dataset. For the remaining 233 images, there are 169 images from the HKPolyU Finger Knuckle Database (V1.0) [15] and 64 images from the HKPolyU Hand Dorsal Database [17]. A total of 1842 of these 2580 images contain knuckles, and the remaining 738 images do not contain knuckles. According to Table 1, we can see the mAP performance of these models on the testing set after training. The mAP is calculated here with IOU equal to 0.5. Furthermore, the predicted bounding box is obtained from a confidence threshold of 0.4 and non-maximum suppression with IOU equal to 0.5. The mAP is the average of the AP values for each category. YOLOv5x-CSL has the highest accuracy with an mAP of 89.6 due to its strong learning capability. mAP for model YOLOv5s-CSL is low mainly because the model has not been sufficiently trained, and the loss has not yet reached a local minimum. Because minor finger knuckle has fewer features and smaller targets than major finger knuckle, it is more difficult to detect, as can be seen from the AP values for minor finger knuckle, which is lower than those for major finger knuckle.

Finger Knuckle Segmentaion Accuracy

Because both YOLOv5x-CSL and YOLOv5m-CSL models have mAP values above 85, they can be said to be a good finger knuckle detection model in terms of my testing data. Although the mAP of the YOLOv4 model is also good, the regression is a horizontal bounding box. Finally, both YOLOv5x-CSL and YOLOv5m-CSL models were used to test the precision and recall rate of finger knuckle segmentation. The testing procedure was performed on three publicly available collection of finger knuckle databases [15], [16], [17]. For the precision in these Table 2, 3, 4, 5, the calculation formula is based on Equation 14, for the recall, the calculation formula is based on Equation 15. For the Equation 15, the GroundTruthSegmentation are the same as the Figure 11 shows.

$$Precision = \frac{TrueSegmentation}{AllSegmentation} \tag{14}$$

$$Recal = \frac{TrueSegmentation}{GroundTruthSegmentation} \tag{15}$$

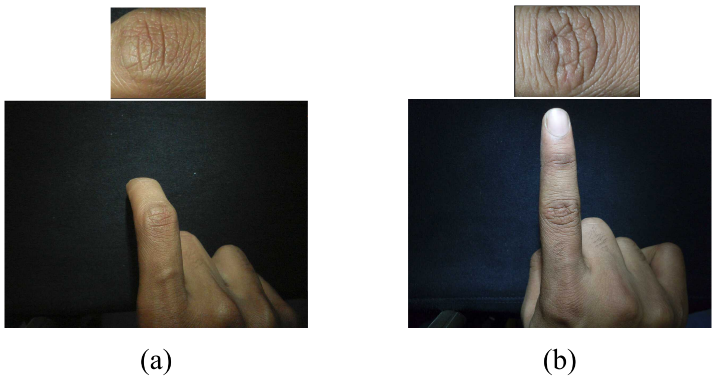


Figure 11: For the HKPolyU Finger Knuckle (V-3.0) [16] finger knuckle segmentation groundtruth image is shown. The groundtruth segmentation images are provided by the database.

Precision indicates the percentage of correctly segmented finger knuckles out of all segmented finger knuckles by YOLOv5-CSL model, while recall indicates the percentage of all finger knuckles on these databases to be segmented that were segmented. For Table 2, the finger knuckle segmentation is performed on the dorsal hand of 502-712 subjects in the HKPolyU Hand Doarsal Image Database [17], with four fingers in each test image and a total of 5368 major finger knuckles and 5368 minor finger knuckles on the 502-712 subjects' doarsal images. In Table 3, the finger knuckle segmentation test is performed on the HKPolyU Finger Knuckle Database (V-1.0) [15], each image has only one finger that has one major finger knuckle and one minor finger knuckle, and there are 2515 major finger knuckles and minor finger knuckles on the database. For Table 4, the finger knuckle segmentation is tested on the HKPolyU Finger Knuckle Database [16]. This data is more complex compared to the first two datasets because each subject provides six sample data, but these six finger images were sampled at different angles, even with the finger knuckle bent to an angle of 90 degrees.

Model	Object	Index	Confidence Threshold				
			0.2	0.3	0.4	0.5	0.6
Yolov5m -CSL	Major FKP	Total (5368)	5437	5392	5357	5312	5219
		Precision	0.983	0.988	0.990	0.991	0.994
		Recal	0.996	0.993	0.988	0.981	0.967
	Minor FKP	Total (5368)	5450	5286	5216	5152	4982
		Precision	0.969	0.992	0.998	0.999	0.999
		Recal	0.984	0.977	0.970	0.959	0.928
Yolov5x -CSL	Major FKP	Total (5368)	**5292**	**5245**	5176	5032	4602
		Precision	**0.999**	**0.999**	1.000	1.000	1.000
		Recal	**0.985**	**0.977**	0.964	0.937	0.857
	Minor FKP	Total (5368)	**5245**	**5158**	5019	4758	4253
		Precision	**1.000**	**1.000**	1.000	1.000	1.000
		Recal	**0.977**	**0.961**	0.935	0.886	0.792

Table 2: The accuracy of finger knuckle segmentation under different thresholds was tested using YOLOv5-CSL's model in hand dorsal dataset [17]. Just use 502-712 dorsal hand images for testing segmentation accuracy.

Model	Object	Index	Confidence Threshold				
			0.2	0.3	0.4	0.5	0.6
Yolov5m -CSL	Major FKP	Total (2515)	2424	2409	2368	2279	2065
		Precision	1.000	1.000	1.000	1.000	1.000
		Recal	0.964	0.958	0.942	0.906	0.821
	Minor FKP	Total (2515)	2509	2502	2494	2460	2335
		Precision	1.000	1.000	1.000	1.000	1.000
		Recal	0.998	0.995	0.992	0.978	0.928
Yolov5x -CSL	Major FKP	Total (2515)	**2503**	**2487**	2423	2263	1694
		Precision	**1.000**	**1.000**	1.000	1.000	1.000
		Recal	**0.995**	**0.989**	0.963	0.9	0.674
	Minor FKP	Total (2515)	**2512**	**2509**	2506	2498	2414
		Precision	**1.000**	**1.000**	1.000	1.000	1.000
		Recal	**0.999**	**0.998**	0.996	0.993	0.96

Table 3: The accuracy of finger knuckle segmentation under different thresholds was tested using YOLOv5-CSL's model in finger knuckle dataset [15].

Model	Object	Index	Confidence Threshold				
			0.2	0.3	0.4	0.5	0.6
Yolov5m -CSL	Major FKP	Total (1326)	1377	1249	1167	1075	975
		Precision	0.895	0.939	0.956	0.970	0.990
		Recal	0.929	0.885	0.842	0.787	0.728
	Minor FKP	Total (Unknown)	828	765	722	662	552
		Precision	0.989	0.993	1.000	1.000	1.000
		Recal	-	-	-	-	-
Yolov5x -CSL	Major FKP	Total (1326)	**1234**	**1146**	1053	989	708
		Precision	**0.964**	**0.990**	0.991	0.990	0.993
		Recal	**0.897**	**0.855**	0.787	0.738	0.530
	Minor FKP	Total (Unknown)	**626**	**576**	531	467	355
		Precision	**0.998**	**1.000**	1.000	1.000	1.000
		Recal	**-**	**-**	-	-	-

Table 4: The accuracy of finger knuckle segmentation under different thresholds was tested using YOLOv5-CSL's model in finger knuckle dataset [16]. Since the fingers are bent at different angles, the total number of minor finger knuckles cannot be estimated because they may not appear in these images on the database.

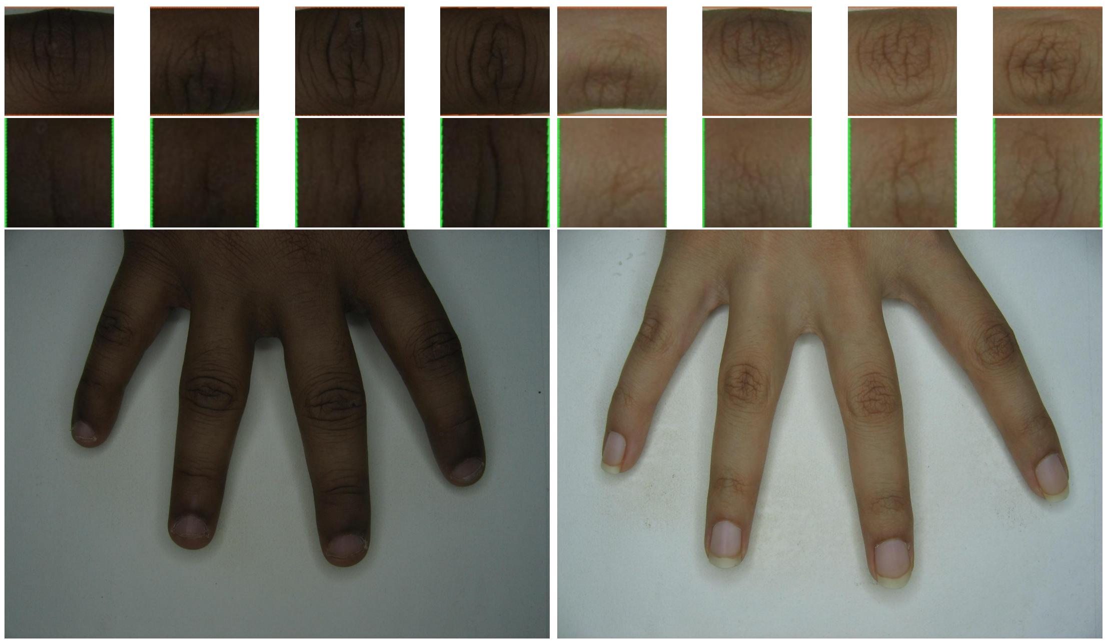


Figure 12: Some examples of finger knuckle segmentation using rotated YOLOv5x-CSL in the dorsal hand database [17].Figure (a)-(b) are images of the dorsal hands of different subjects. As for each subfigure, the first row it the major finger knuckle, and the second row is the minor finger knuckle.

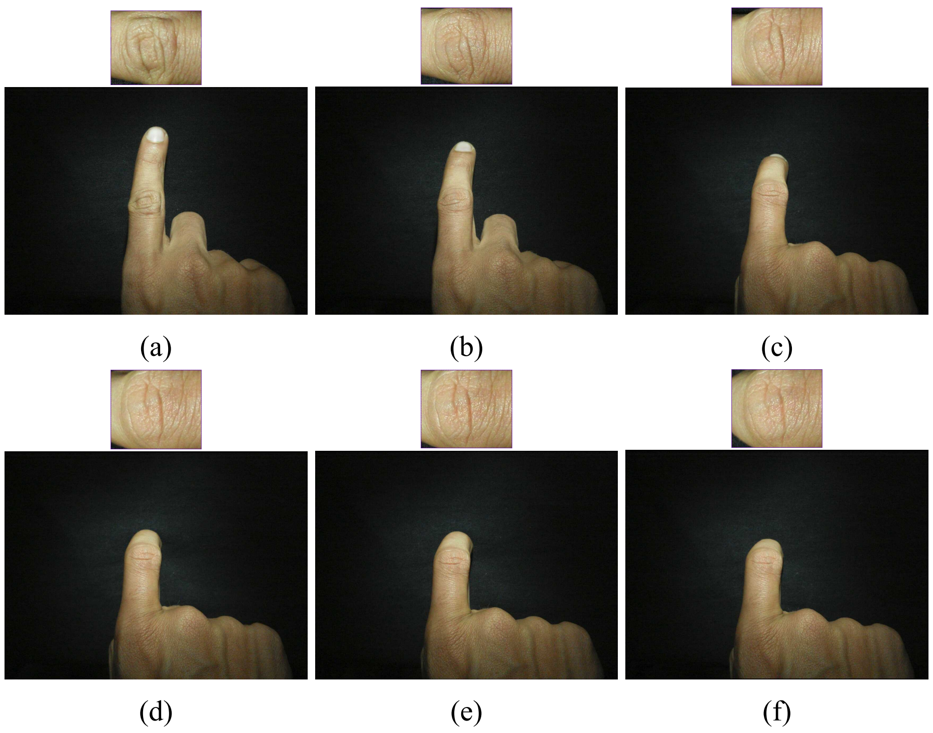


Figure 13: One subject's finger knuckle segmentation using rotated YOLOv5x-CSL in the finger knuckle database [16]. Figure (a)-(f) are images of the finger knuckle with different angle.

The results of the test were performed using different confidence thresholds for finger knuckle segmentation by using YOLOv5m-CSL and YOLOv5x-CSL. From Table 2, 3, 4, for the precision and recall values, they can hit above 0.95. Even for dataset [15], not only minor finger knuckle and major finger knuckle segmentation precision was 100% as shown in Table 3, which means that all segmented finger knuckle ROI by my model were correct. This is because in Table 3 and 2, the test databases [15], [17] used are both relatively less challenging. Both databases are sampled with fixed background interference and with the fingers extended, in which case the finger knuckle features are relatively well displayed. So the segmented precision and recall values can even reach 0.999. However, as shown in Table 4, the magnitude of the values of precision and recall in the HKPolyU Finger Knuckle Database (V-3.0) [16] is smaller than that of both precision and recall in the HKPolyU Finger Knuckle Database (V-1.0) [15] and the HKPolyU Hand Dorsal Database [17]. This shows that the segmentation model is not ideal in this dataset as in the other two datasets. However, the values of precision and recall can probably still be maintained above 90%. The analysis of the HKPolyU Finger Knuckle Database V-3.0 [16] shows that the finger knuckles are sampled under a more complex scene with different angles of deformation problems, which leads to a decrease in segmentation accuracy. Because for both precision and recall, one increases, the other has to decrease. It can also be seen from the experimental results that when the confidence threshold was raised, the precision also increased, and when the confidence threshold went above 0.4, the finger segmentation precision of these data sets was above 0.95, and the recall also decreased. This indicates that the accuracy of finger knuckle segmentation increases while more and more finger knuckles are not segmented. For YOLOv5x-CSL, the model achieves a high mAP, which is also confirmed for the segmentation test. YOLOv5x-CSL has higher precision and recall than YOLOv5m-CSL. Even for thresholds 0.2 and 0.3, recall is kept no lower than 0.85, and precision reaches 0.999.

The Figure 12 show the dorsal hand images of tow different individuals in the HKPolyU Hand

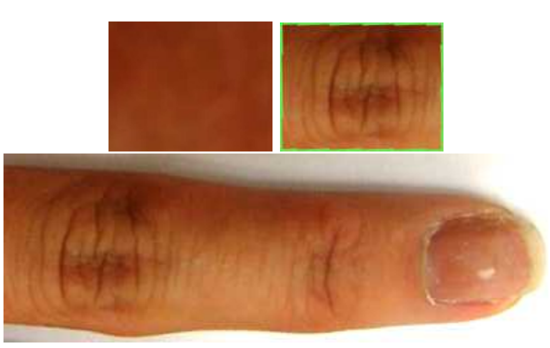
Dorsal Image Database [17], using YOLOv5-CSL oriented bounding boxes for finger knuckle prediction and segmentation operations. The first row of each subfigure is the segmentation of the major finger knuckle, the second row is the segmentation of the minor finger knuckle, and the third row is the original image, as shown in Figure 12. In the same way, the Figure 13 shows a sample of a segmentation test on HKPolyU Finger Knuckle Database (3.0) [16]. The background interference after segmentation of each finger knuckle in the image is relatively small and absent. There is now a significant improvement compared with the background interference obtained by direct segmentation using the horizontal bounding box in Figure **??**. Because these finger knuckles have some 3D angular rotation, some background information is still segmented during the segmentation.The experiment proves that the YOLOv5x-CSL and YOLOv5m-CSL model can be fully capable of performing finger knuckle segmentation as a fully automated finger knuckle segmentation model fast in the wild case, providing high efficiency for subsequent finger knuckle recognition.

Compare with Earlier Work

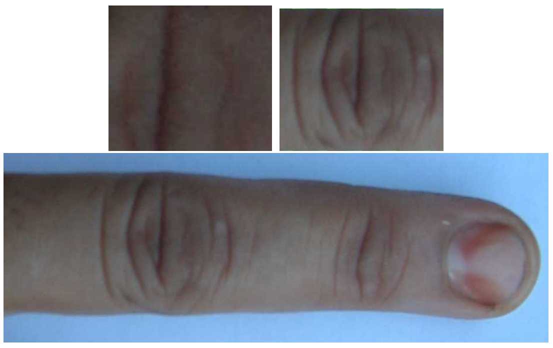
Database	FK	Precision	Recal	Totall
Finger Knuckle (V1.0) [15]	Major FK	0.982	0.982	2510
	Minor FK	0.991	0.991	2510
Dorsal Hand [17]	Major FK	0.985	0.985	4220
	Minor FK	0.985	0.985	4220
Finger Knuckle (V3.0) [16]	Major FK	0.998	0.998	1326
	MinorFK	-	-	-

Table 5: The segmentation precision provided by these public database. Finger Knuckle (v3.0) [16] does not provide segmentation images of minor finger knuckle.

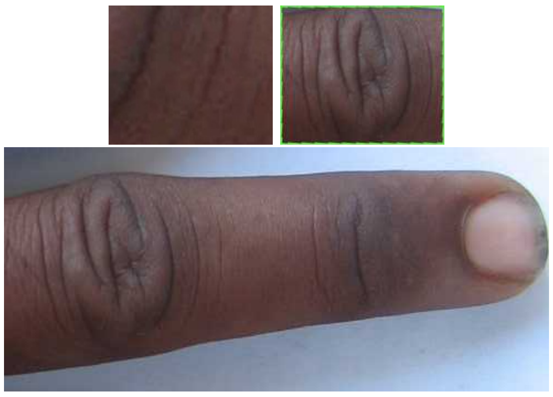
The segmented finger knuckles given in the three public datasets were counted, and the corresponding segmentation precision and recall values were obtained and listed in the Table 5. Since all three databases have segmented the corresponding finger knuckle images, there is no so-called missed segmentation, so the corresponding precision and recall values are the same. For the first database listed in the Table 5, the major finger knuckle segmentation accuracy is 0.982, and the minor finger knuckle segmentation accuracy is 0.991 for the finger knuckle v1.0 database [15], and the corresponding recall values are both equal. For the dorsal hand database [17], the precision of the major finger knuckle is 0.985, and the minor finger knuckle is 0.98. According to the precision and recall values obtained from the segmentation using our model in Table 3 and Table 2, in the case of the confidence threshold of 0.2 and 0.3, the precision and recall values obtained using the YOLOv5x-CSL model are higher than the precision and recall values of the segmented images provided by the database. Even when calculating the accuracy of the finger knuckle segmentation provided by the database, many images with misaligned segmentation positions were not determined as wrongly segmented regions. For both databases, our model's segmentation accuracy is higher, but also the segmented finger knuckle regions are more accurate, which can bring more useful information for subsequent finger knuckle feature extraction. The Figure 14 and Figure 15 shows some wrong segmentation in the corresponding public dataset and the examples that we can segment correctly. From these two figures, we can find that our segmentation model can not only segment the location of the knuckles correctly but also segment the background area well.



(a)



(b)



(c)

Figure 14: Comparison of some wrong finger knuckle segmentation images in the HKPolyU Finger Knuckle (V1.0) database [15] and the segmentation using our model. The left side of the first row of each subplot shows the incorrect segmentation in the database and the right side shows the correct segmentation by YOLOv5-CSL model.

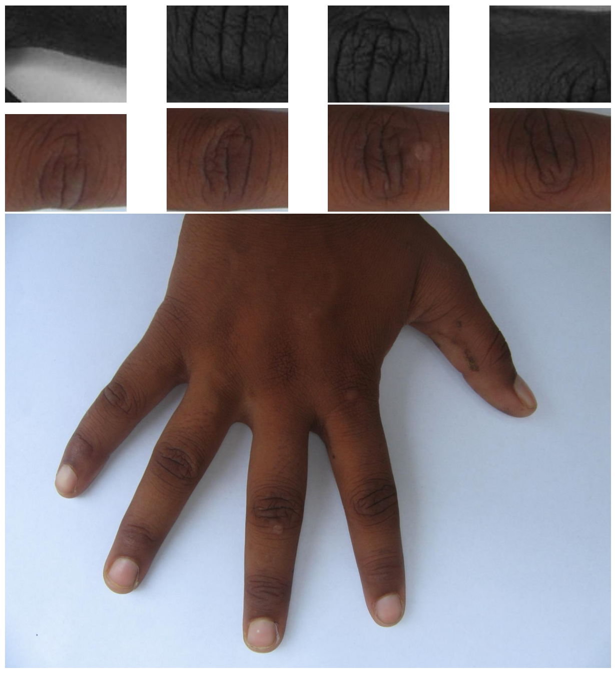


Figure 15: Comparison of some wrong finger knuckle segmentation images in the HKPolyU Dorsal Hand database [17] and the segmentation using our model. The first row shows the incorrect segmentation in the database and the second row shows the correct segmentation by YOLOv5-CSL model.

However, for the Finger Knuckle V3.0 database [16], although the finger knuckles are sampled at different angles, the position of the finger knuckles in the image is relatively fixed. The database segmentation accuracy listed in Table 5 yields a major finger knuckle of 0.998. However, from Table 4, we get a segmentation precision of 0.990, but the corresponding recall is only 0.855, which means our method has many missed segmentation. Therefore, the segmentation accuracy obtained by using the fixed position segmentation method provided in the article [7] is very high, regardless of the precision or recall values being higher than our model's accuracy for testing. This is because the corresponding finger knuckles are sampled at different degrees of curvature, and the background interference varies. The segmentation accuracy of our model is not very high for this database because few training images contain bent finger knuckles in the training set when the model is trained. The Figure 16 shows the finger knuckle regions when our model performs segmentation and the segmentation regions provided in the database. It can be seen that our model is relatively more accurate for the key region of the finger knuckle, i.e., the location of the segmented finger knuckle.

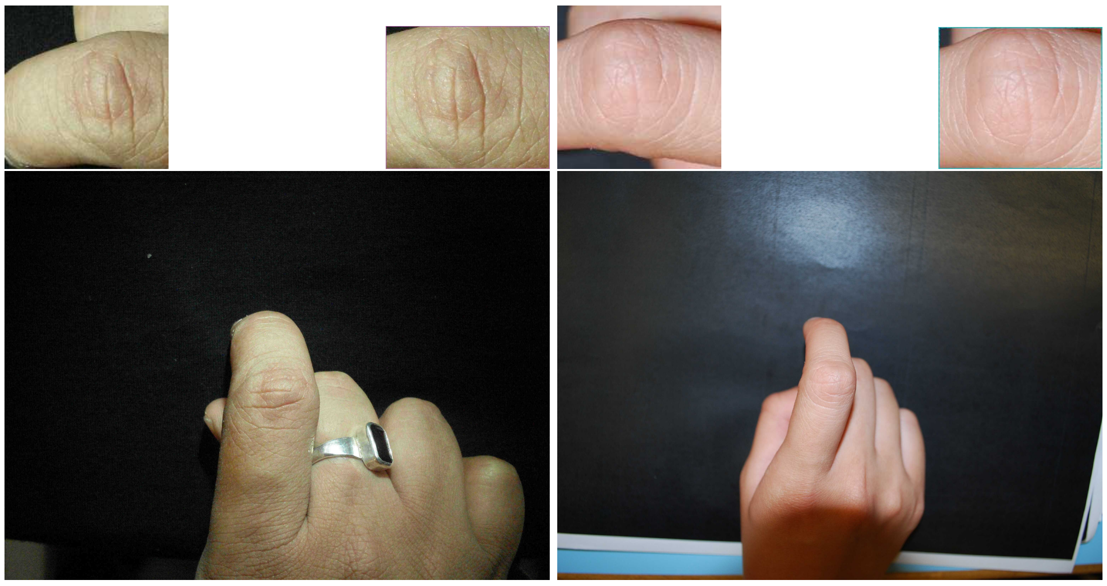

(a) (b)

Figure 16: Comparison of some wrong finger knuckle segmentation images in the HKPolyU Finger Knuckle (V3.0) database [16] and the segmentation using our model. The left side of the first row of each subplot shows the incorrect segmentation in the database and the right side shows the correct segmentation by YOLOv5-CSL model.

The results shown in Table 3, 2, 4, 5 use the detected images made available in respective publicly available databases as the ground truth or reference. However, many of these reference images themselves may not be very accurate and therefore the numbers shown in these tables can be subjective to such interpretation. From the above comparison, for both HKPolyU Finger Knuckle (V1.0) [15] and HKPolyU Dorsal Hand [17], the accuracy of our model segmentation is higher in terms of the success rate of segmentation and the correct rate of segmentation. Even comparing the segmented images 14, 15, the finger knuckle positions segmented by our model are more accurate and contain more information about the finger knuckles, which is helpful to improve the accuracy of the subsequent finger knuckle recognition. Although for the more challenging HKPolyU Finger Knuckle (V3.0) [16], the precision and recall of our model's segmentation are lower than those of the segmented images already provided in the public dataset. However, from comparing the segmented images 16, we can also obtain similar conclusions as above that our segmented finger knuckle ROI contains richer finger knuckle feature regions and less background after segmentation. Since the model training contains fewer bending finger knuckle, the segmentation accuracy is not very good for the bent finger knuckles, and the subsequent model fine-tune may achieve better results. Our model has achieved a good result for the segmentation of the finger knuckle region, both in terms of accuracy and the signal-to-noise ratio of the segmented region of interest.

8 References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. "Yolov4: Optimal speed and accuracy of object detection". In: *arXiv preprint arXiv:2004.10934* (2020).

[2] Cgvict and Wkkmike. *roLabelImg*. `https://github.com/cgvict/roLabelImg`. 7 July 2017.

[3] KamYuen Cheng and Ajay Kumar. “Contactless finger knuckle identification using smartphones”. In: *2012 BIOSIG-Proceedings of the International Conference of Biometrics Special Interest Group (BIOSIG)*. IEEE. 2012, pp. 1–6.

[4] Ross Girshick. “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[5] Ross Girshick et al. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587.

[6] Ajay Kumar. “Contactless finger knuckle authentication under severe pose deformations”. In: *2020 8th International Workshop on Biometrics and Forensics (IWBF)*. IEEE. 2020, pp. 1–6.

[7] Ajay Kumar. “Toward pose invariant and completely contactless finger knuckle recognition”. In: *IEEE Transactions on Biometrics, Behavior, and Identity Science* 1.3 (2019), pp. 201–209.

[8] Wei Liu et al. “Ssd: Single shot multibox detector”. In: *European conference on computer vision*. Springer. 2016, pp. 21–37.

[9] Yang Liu and Ajay Kumar. “Contactless palmprint identification using deeply learned residual features”. In: *IEEE Transactions on Biometrics, Behavior, and Identity Science* 2.2 (2020), pp. 172–181.

[10] Joseph Redmon and Ali Farhadi. “YOLO9000: better, faster, stronger”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 7263–7271.

[11] Joseph Redmon and Ali Farhadi. “Yolov3: An incremental improvement”. In: *arXiv preprint arXiv:1804.02767* (2018).

[12] Joseph Redmon et al. “You only look once: Unified, real-time object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[13] Shaoqing Ren et al. “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in neural information processing systems* 28 (2015), pp. 91–99.

[14] Florian Schroff, Dmitry Kalenichenko, and James Philbin. “Facenet: A unified embedding for face recognition and clustering”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 815–823.

[15] *The HKPolyU Contactless Finger Knuckle Images Database (V-1.0):* `http://www4.comp.polyu.edu.hk/~csajaykr/fn1.htm`.

[16] *The HKPolyU Contactless Finger Knuckle Images Database (Version 3.0) :* `https://www4.comp.polyu.edu.hk/~csajaykr/fn2.htm`.

[17] *The HKPolyU Contactless Hand Dorsal Images Database:* `http://www4.comp.polyu.edu.hk/~csajaykr/knuckleV2.htm`.

[18] Ultralytics. *YOLOv5*. `https://github.com/ultralytics/yolov5`. 18 May 2020.

[19] Ying Xin et al. “PAFNet: An Efficient Anchor-Free Object Detector Guidance”. In: *arXiv preprint arXiv:2104.13534* (2021).

[20] Xue Yang and Junchi Yan. “Arbitrary-oriented object detection with circular smooth label”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 677–694.

[21] Zhaohui Zheng et al. “Distance-IoU loss: Faster and better learning for bounding box regression”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 07. 2020, pp. 12993–13000.

[22] Zhaohui Zheng et al. “Enhancing geometric factors in model learning and inference for object detection and instance segmentation”. In: *IEEE Transactions on Cybernetics* (2021).