[2] Title: Automatic Fabric Defect Detection Based on an Improved YOLOv5

Jin and Nui [2] proposed a teacher-student Yolov5 model to perform defect detection upon different material and different defects. In the research done before, they found 2 main databases which could help them conduct this research (*TILDA and Xuelang Tianchi AI Challenge*).

The TILDA database had a dataset of 300 fabric images and divided into six categories, 5 different defects and 1 normal classification and images downsize to 256 x 256 pixels. 70% was used as the training set while the other 30% used as the test set. The Xuelang Tianchi AI Challenge dataset was also used, this contained 3331 labelled images, split into 2163 images containing no defect while the other 1168 images have one or multiple defects. A total of 22 types of defects were found in the dataset, this was split into 70% as a training set while the rest used as a test set.

Their results using the TILDA dataset when comparing the teacher network and student network had an area under the ROC curve (AUC) of 0.988 and 0.965 respectively and a mean average precision (mAP) of 0.451 and 0.428. The Tianchi AI dataset al so was tested on the teacher and student networks and ended up having a slightly worse AUC results of 0.981 and 0.952 while a mAP of 0.447 and 0.406.

(check if comparing with a yolov5 is good? As it is done in the paper itself and check if I add the other tables at the bottom)

[6] Title: Vision Based Topological State Recognition for Deformable Linear Object Untangling Conducted in Unknown Background

Detecting Rope: (end point and overlap labelling)

A study on ropes was done by Yu et al. [6] who developed a model which training a robot to untangle ropes. Throughout their study they used a method on how to label the rope and eventually their neural network can achieve the goal of topological state recognition. The training was done on 4 different tablecloths, as well as adding random obstacles in the image. Keeping in mind that these items never covered the camera’s sight of the target. A crossing points detection method was done using Yolov3, 800 images of a single rope and 833 images of two ropes were provided as the dataset for training the network. The program LabelImg was used to label data. As seen in Fig. 1 both region for the endpoints and where the rope overlaps with itself are marked and labelled accordingly.

Graphical user interface

Description automatically generated with medium confidence

[4] Title: Tangled: Learning to Untangle Ropes with RGB-D Perception

[5] Title: Untangling Dense Knots by Learning Task-Relevant Keypoints

Lui et al. [4] used a different method to detect the rope structure. Firstly, removing the background by place fitting and splitting the rope structure into segments as shown in Fig. 2. To obtain the order of these segments to represent the correct rope configuration a rule was set that every point must have 2 neighboring points, except the 2 end points. By using a point system, they were able compute a feature vector. A different approach building on the research of Lui et al. [4], Grannen et al. [5] also used a marking system on the ropes as seen in Fig. 3, a segmenting system would also take place in this situation but only mark the end points and overlapping parts of the rope, and not a segment every couple of centimeters. This would end up with a much smaller linear graph as seen below. These segments would get the annotations of (+) for when the rope is passing from overlapping segments and (-) for when the rope is passing from under overlapping parts.

Shape

Description automatically generated

A picture containing text

Description automatically generated

[3] Title: Domain Feature Mapping with YOLOv7 for Automated Edge-Based Pallet Racking Inspections

Hussain et al. [3] used the new state of the art Yolov7 technology to automate the classification of pallet racking of five different classes: horizontal, vertical, support, vertical damage, and support damage. To show the trained architecture’s realistic inference speed, a benchmark on its performance not only on architectural and computational performance but also on post-deployment metrics like frame-per-second (FPS).

The dataset was collected manually, and the recordings of three different warehouses' pallet racks were made using smartphones. After this annotation of the data was done carefully on the five different classes stated before. The data annotation was done on images in an occupied warehouse, stock was being places on the racks. To work around this problem only annotating the item would be done if 25% or less of the item is occluded where the occluded part would also be annotated. Where the occlusion would take more than 25% of the class, only the showing part would be marked as shown in Fig. 4.



This research architecture was designed to identify pallet racking in a wide range of warehouses. It was logical to predict that various locations would have distinct external variables to which the model-trained architecture would have to adjust. For example, warehouse A may have more lighting than warehouse B, because of the location, and it may also vary if the dataset was taken in day or night shifts. The tackle this situation a random brightness level varying between -11% and +11% would be applied to each frame of the videos. The dataset consisted of 2094 samples, where 1905 used for training, 129 used for validation and the last 60 samples were used for testing.

[9] Moving toward Smart Manufacturing with an Autonomous Pallet Racking Inspection System Based on MobileNetV2

Another research by Hussain et al. [9] used a different approach than the previous paper to solve the similar problems on pallet racking damages. Instead of using YOLOv7, the MobileNetV2-SSD architecture was researched and used. When the research was being done, no public data set of warehouse pallet racking was found, by mounting a smartphone to a forklift, videos were taken and the frames where split with a 1 second interval. For the preprocessing of the dataset was done in two parts. First, the process of remove the images with no racking and secondly, resizing all the images to 416 x 416. As this was an object detection localization problem, bounding boxes where used, by using Computer Vision Annotation Tool (CVAT) by OpenCV this was made much simpler to also calculate the predictions.

Augmentation of the images was also done to increase the model capability of predicting correctly. Images randomly selected where cropped by 0-3% and a random rotation to each was set from -6 to +6 degrees. This rotation and cropping were done to further improve the scenarios this model would be tested to. As the forklift would have different amounts of weight distribution while also moving around, this would improve how the camera installed would be moved and wobble around. A random brightness level between -5 and +5% would be applied, while adding a blur effect to some images to continue simulating real life situations. Variance in the moving speed, warehouse location and time of day these tests would be done would also be covered by this model. With the use of PyTorch, TensorFlow and Keras frameworks this was done within a couple of hours, while using the Google Collab GPUs found online. Google, Microsoft Azure, and Amazon Web Services all have these free resources to be used for investigations like this. Their different approach had slightly better results than the previous paper with a mean average precision(mAP) of 92.7% surpassing the one using YOLOv7 by 1.6% mAP.

Results Table: (Table 1)

|  |  |  |  |
| --- | --- | --- | --- |
|  | This Research  Hussain et al. [3] | Farahnakian et al. [8] | Hussain et al. [9]  Different Research same people |
| **Approach** | Object Detection | Image Segmentation | Object Detection |
| **Dataset Size** | 2094 | 75 | 19,717 |
| **Classes** | 5 | 1 | 2 |
| **Detector** | YOLOv7 | Two-Stage | Single Shot |
| [**MAP@0.5(IoU)**](mailto:MAP@0.5(IoU)) | 91.1% | 93.45% | 92.7% |

[7] Title: A real-time object detection algorithm for video

Cannot get [15] <https://www.sciencedirect.com/science/article/abs/pii/S0030402618319910>

Shengyu et al. [7] chose to develop a YOLO network upon researching different technologies. This research was aimed to detect 5 different classes: car, truck, bus, motor, and person in real-time object detection in video. This consisted of trials on videos using a vehicle monitoring data obtained from the Xiamen municipal transportation bureau. The video dataset was split into frames creating 8000 1280 x 720 images, while this was split into 75% of the images as the training set while the rest as a test set. Calculating their results of the precision, recall and frames per second (FPS), are used to compare with 6 different studies. Their findings show that their techniques outperform the original YOLO algorithm [14] as well as the other baseline methods. The results show faster detection speed at 45fps as well as better accuracy overall.

Baseline Approaches: (Table 2)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Algorithm | Precision (%) | Recall (%) | Year Published | FPS | Sizes | Dataset |
| Sliding Window [10] | 70.59 | 72.91 | 2011 | 0 | 720 x 576 | 1175 images |
| CNN [11] | 80.82 | 78.38 | 2015 | 1 | 224 x 224 | 23,953 videos |
| RCNN [12] | 84.19 | 83.69 | 2006 | 2 | Remove? | Remove? |
| Faster R-CNN [13] | 83.96 | 82.65 | 2017 | 6 | Random Sizes | 2845 images |
| YOLO [14] | 88.34 | 86.22 | 2018 | 20 | N/A | 4655 images |
| Fast YOLO [7] | 88.45 | 86.64 | 2018 | 45 | 1280 x 720 | 8000 images |
| SSD [15] | 86.82 | 84.10 | 2018 | 25 |  |  |

Baseline Approaches (Table 2)

1. Sliding Window: It can predict the classes of each object inside an image by dividing the images into grids and extracting the features of each grid.
2. CNN: The color, edge, texture, and more are extracted by a convolution operation from the images.
3. RCNN: The images are divided into a grid, and by using CNN the regions are classified as part of the object or part of the background.
4. Faster R-CNN: Same basis as RCNN and CNN but images features are only extracted once.
5. YOLO: The picture is split into many grids, and the likelihood of an object's center landing within the grid is determined. The category probabilities of the items are computed using boundary boxes.
6. SSD: The output of multiple layers is used to extract different scales of feature mapping. To establish the object categories in the grids, the mess is separated into several scales.