# YOLO-Based Object Detection for 3D Printing Defect Identification

Zeynep Galymzhankyzy
*Dept. of Math and Computer Science*
*Lawrence Technological University*
Southfield, MI, USA
zgalymzha@ltu.edu

Milana Muratova
*Dept. of Math and Computer Science*
*Lawrence Technological University*
Southfield, MI, USA
mmuratova@ltu.edu

Tomiris Murzagali
*Dept. of Math and Computer Science*
*Lawrence Technological University*
Southfield, MI, USA
tmurzagali@ltu.edu

***Abstract*—Ensuring the quality of 3D-printed components is critical for advancing additive manufacturing. This study explores the use of deep learning-based object detection models to identify and classify common defects in 3D printing. Leveraging a custom dataset of annotated images, the performance of two YOLO architectures, YOLOv11 and YOLOv10l, was evaluated. YOLOv10l achieved higher accuracy, with an overall mAP@0.5 of 0.530, demonstrating its effectiveness for defect detection. The results highlight strong performance for certain defect types but also reveal challenges such as class imbalance and overlapping features. This work underscores the potential of deep learning in automating quality control processes and identifies opportunities for further refinement through improved datasets and advanced modeling techniques.**

**Keywords:** ***3D printing, quality control, object detection, YOLO, deep learning***

## I. Introduction

Additive manufacturing, commonly referred to as 3D printing, has revolutionized the production of complex geometries and customized components in industries such as aerospace, healthcare, and automotive. Despite its advantages, 3D printing processes are prone to defects that can compromise product quality, functionality, and reliability. Defects such as *Spaghetti*, *Stringing*, and *Warping* often occur due to factors such as material inconsistencies, suboptimal printing parameters, or hardware malfunctions. Detecting these defects early is critical to minimizing waste, reducing costs, and ensuring consistent quality in printed products.

Traditional methods for defect detection in 3D printing typically rely on manual inspection or rule-based systems. However, these approaches are often time-consuming, prone to human error, and difficult to scale for industrial applications. To address these challenges, recent advances in artificial intelligence, particularly deep learning, have enabled the development of automated and scalable quality control systems. Object detection models, such as the YOLO (You Only Look Once) framework, offer a promising solution due to their ability to detect and classify multiple objects in real-time.

This study investigates the application of YOLO-based models for detecting common 3D printing defects. Specifically, it evaluates the performance of two architectures, YOLOv11 (lightweight) and YOLOv10l (large), on a custom dataset of 3D printing errors. The data set includes annotations for three types of defects, each with distinct characteristics and challenges. Models are trained and evaluated using standard object detection metrics, including precision, recall, and mean average precision (mAP). Validation plots, such as precision-recall curves and confusion matrices, are used to analyze the performance of each model in different defect classes.

This study presents the following contributions:

- Evaluation of YOLOv11 and YOLOv10l models, demonstrating YOLOv10l's superior performance with an mAP@0.5 of 0.530.
- Identified class-specific challenges, such as imbalance and overlapping features, affecting detection of *Stringing* and *Warping*.
- Provided actionable recommendations for improving defect detection through dataset expansion and advanced modeling techniques.

The results demonstrate the potential of YOLO-based models for automating defect detection in 3D printing, paving the way for smarter and more efficient quality control systems in additive manufacturing.

## II. Related Work

Prior approaches have attempted to improve defect detection in 3D printing using advanced machine learning techniques, particularly focusing on the detection of various types of defects, such as spaghetti error, stringing, and warping. Existing research has laid a foundation for understanding the effectiveness of various model architectures, particularly those based on the YOLO (You Only Look Once) framework, which is prominent in object detection tasks.

### *A. Challenges Posed by Common 3D Printing Errors*

3D printing is susceptible to several common errors, including spaghetti, stringing, and warping, each of which can severely affect print quality and resource efficiency.

- Spaghetti: Often occurring when an object detaches from the print bed mid-process, spaghetti represents a failure in bed adhesion. This type of error leads to the extrusion of a tangled filament, which can obstruct the nozzle and affect the overall quality of the print.

- Stringing: This error manifests itself as thin filament strands left on the surface of the printed part, often caused by excess pressure in the nozzle during travel movements.
- Warping: Warping occurs when the printed object cools unevenly, causing the edges to lift off the print bed.

Addressing these challenges is crucial for improving the resource efficiency of 3D printing processes and ensuring high-quality final products.

## *B. Previous Approaches to the Problem*

The importance of accurately detecting defects in 3D printing has been recognized in multiple studies. Some researchers have used models such as YOLOv4 and YOLOv5 for the real-time detection of defects, using their speed and efficiency in image processing [1, 2]. Other studies focused on specific defect types, such as surface quality issues and extrusion errors, employing hybrid approaches that combine machine vision with deep learning models [3, 4]. In addition, techniques that incorporate attention mechanisms within the YOLO architecture have been explored to improve detection accuracy [5].

However, many of these approaches face challenges, particularly when dealing with imbalanced datasets. For example, datasets with disproportionate representations of defect types can significantly skew the precision and overall precision of the model. A comprehensive benchmark analysis indicated that class imbalance often leads to lower performance metrics, particularly in underrepresented classes such as stringing and warping [6, 7].

## *C. Relevant Methods and Techniques*

Several effective methods have been developed to address these issues. Data enhancement techniques have been widely implemented to artificially increase the diversity of training datasets, thus improving the generalization of the model [8]. For example, the use of synthetic data generation has proven beneficial in improving defect detection models under data scarcity conditions [9, 10]. In addition, ensemble learning techniques have been proposed to combine predictions from multiple models, improving detection performance in various classes [11]. The application of transfer learning methods has also been explored, particularly for leveraging pretrained weights to enhance performance on smaller datasets [12, 13].

Moreover, the incorporation of hyperparameter optimization, such as tweaking batch sizes and implementing mixed precision training, has shown potential in improving the training efficiency of YOLO models while enhancing detection accuracy [14, 15]. These adjustments can facilitate more effective learning over specific epochs, leading to improved precision across classes.

## III. Methodology

This study follows a three-step approach: data preparation, model training, and evaluation.

## *A. Data Preparation*

The dataset was split into training, validation, and testing subsets, with augmentations such as rotations, shear transformations, and brightness adjustments dynamically applied during training to enhance generalization. All images were resized to $640 \times 640$ for compatibility with YOLO models.

## *B. Model Training*

Two YOLO architectures, YOLOv11 (lightweight) and YOLOv10l (large), were trained using the YOLO loss function, which combines localization, confidence, and classification losses. Stochastic Gradient Descent (SGD) was used with a learning rate of 0.01, momentum of 0.9, and a batch size of 32. Models were trained for 100 epochs, and the best-performing model was saved based on validation loss.

## *C. Evaluation Procedure*

Model performance was assessed using precision, recall, and mean average precision (mAP). Validation plots, including precision-recall (PR) curves and confusion matrices, provided detailed insights into model accuracy and class-specific performance.

## IV. Algorithm

The pseudocode for the YOLO-based 3D printing defect detection system is as follows:

**Algorithm 1** YOLO-based Defect Detection

```
0: Input: Dataset $D$, Model architecture $M$, Hyperparameters $H$
0: Output: Trained model $M^*$, Evaluation metrics
0:
0: Step 1: Preprocessing
0: Split $D$ into training, validation, and testing subsets.
0: Apply augmentations (rotation, shear, brightness).
0: Resize images to $640 \times 640$.
0:
0: Step 2: Training
0: for epoch $e = 1$ to $H_{\text{epochs}}$ do
0:    Compute predictions and YOLO loss:
          $\text{Loss} = \lambda_{\text{coord}} \cdot \text{CoordLoss} + \text{ConfLoss} + \text{ClassLoss}.$
0:    Backpropagate and update model weights.
0: end for
0: Save the best model as $M^*$.
0:
0: Step 3: Evaluation
0: Use $M^*$ to predict on the validation set.
0: Compute metrics: Precision, Recall, mAP@0.5, mAP@0.5:0.95.
0: Generate validation plots (e.g., PR curves, confusion matrices).
0:
0: Return: $M^*$ and evaluation metrics. =0
```

## V. Experimental Setup

The defect detection task was conducted using a custom dataset specifically designed to identify 3D printing defects. The dataset consists of a total of 4,385 images, with 10,053 annotated instances distributed across three classes: *Spaghetti* (4,885 instances), *Stringing* (3,206 instances), and *Warping* (1,962 instances). It is noteworthy that the *Warping* class is underrepresented compared to the other classes. To enhance the dataset and improve model robustness, augmentations were applied, increasing the total number of processed images to 10,525. The dataset was split into three subsets: 88% for training (9,210 images), 6% for validation (658 images), and 6% for testing (657 images). The average image size across the dataset is approximately 0.41 megapixels.

### A. Data Augmentation and Preprocessing

To improve generalization and mitigate the impact of class imbalance, several data augmentation techniques were employed during training. Each training example was augmented to produce three outputs. These augmentations included:

- Random rotations between $-15°$ and $+15°$,
- Shear transformations of up to $\pm 10°$ both horizontally and vertically, and
- Brightness adjustments ranging from $-15\%$ to $+15\%$.

Preprocessing steps included auto-orientation to correct image rotations and resizing all images to a standard resolution of 640x640 pixels. This ensured uniformity in image dimensions, which is essential for input to the YOLO models.

### B. Evaluation Metrics

The models were evaluated using standard object detection metrics:

- **Precision (P)**: The proportion of correctly predicted positive instances out of all predicted positives.
- **Recall (R)**: The proportion of correctly predicted positive instances out of all actual positives.
- **Mean Average Precision (mAP@0.5)**: Calculated at an Intersection over Union (IoU) threshold of 0.5.
- **Mean Average Precision (mAP@0.5:0.95)**: Averaged across IoU thresholds ranging from 0.5 to 0.95 in increments of 0.05.

### C. Validation Metrics Analysis

**Precision-Recall Curve:** The PR curve in Figure 1 illustrates the relationship between precision and recall:

- *Spaghetti* achieved the best performance with a high area under the curve and a class-specific mAP@0.5 of 0.809.
- *Stringing* showed poor performance, with a class-specific mAP@0.5 of 0.328.
- *Warping* performed moderately, achieving a class-specific mAP@0.5 of 0.454.

**Confusion Matrix:** The confusion matrix (Figure 2) highlights the classification performance:

- *Spaghetti* was correctly identified in 575 instances, with minimal misclassifications.

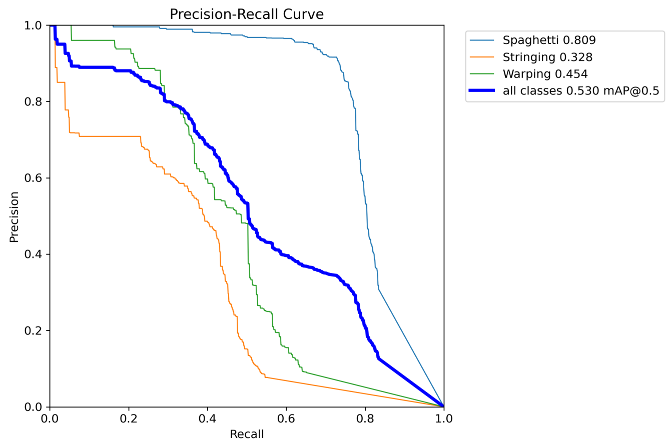

Fig. 1. Precision-Recall Curve for each class and overall mAP.

- *Stringing* showed significant misclassification, with 270 out of 443 instances incorrectly predicted as *background*.
- *Warping* achieved 129 true positives but was often misclassified as either *Stringing* or *background*.

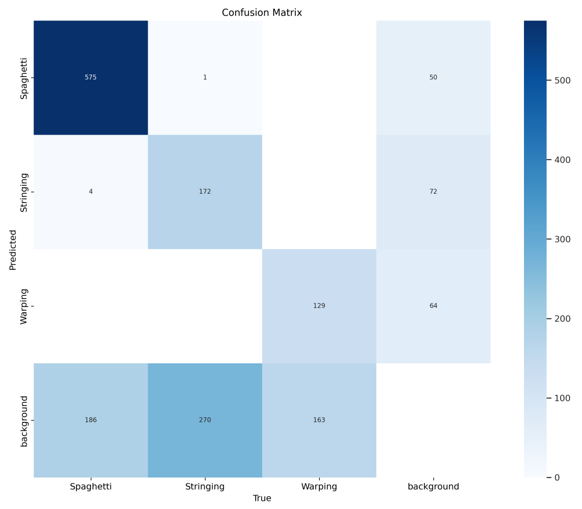

Fig. 2. Confusion matrix for validation results.

**Normalized Confusion Matrix:** The normalized confusion matrix in Figure 3 shows detection rates as proportions:

**Recall-Confidence Curve:** The recall-confidence curve (Figure 4) illustrates recall behavior with confidence thresholds:

- *Spaghetti* maintained high recall across all thresholds.
- *Stringing* dropped recall rapidly at higher thresholds.
- *Warping* showed moderate recall but declined at higher thresholds.

**F1-Confidence Curve:** The F1-confidence curve in Figure 5 revealed that *Spaghetti* achieved the highest F1 score, while *Stringing* scored significantly lower.

**Precision-Confidence Curve:** The precision-confidence curve in Figure 6 showed that *Spaghetti* maintained high

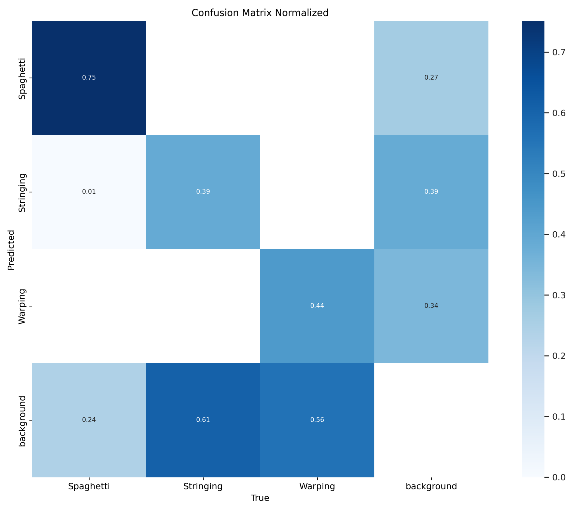

Fig. 3. Normalized confusion matrix for validation results.

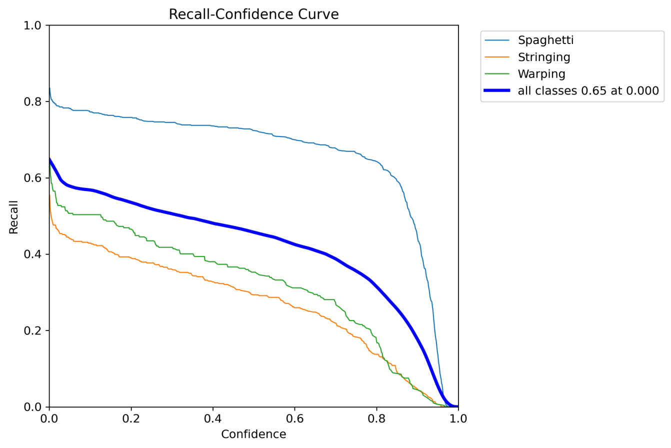

Fig. 4. Recall-confidence curve for each class and overall performance.

precision across all thresholds.

### D. Baselines

Two YOLO models were selected for comparison:

1) **YOLOv11**: A lightweight and fast model optimized for real-time applications.
2) **YOLOv10l**: A larger model with a higher number of parameters, designed for improved accuracy in more complex tasks.

### E. Implementation Details

The experiments were implemented using the Ultralytics YOLO framework. Training and evaluation were performed on an NVIDIA A100 80GB GPU. The training pipeline included a total of 100 epochs for each model, with augmentations and preprocessing steps applied as described above. All images were resized to 640x640 pixels before training. Mosaic augmentation, horizontal flips, and brightness adjustments were applied dynamically during the training process to improve

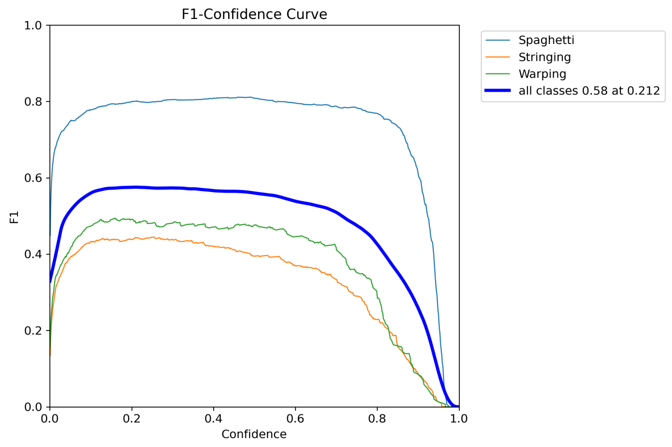

Fig. 5. F1-confidence curve for each class and overall performance.

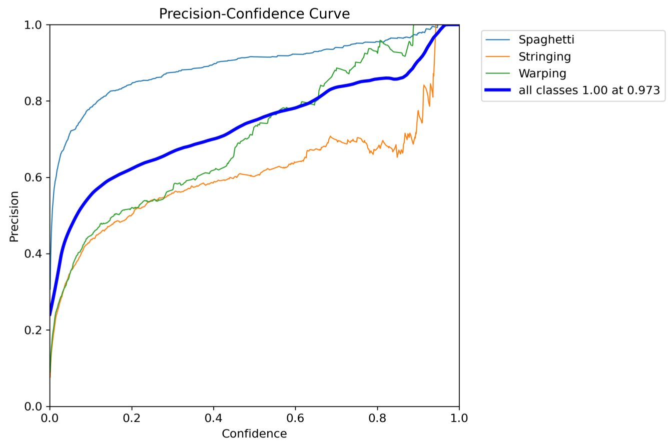

Fig. 6. Precision-confidence curve for each class and overall performance.

model robustness. The training times were approximately 1.42 hours for YOLOv11 and 2.22 hours for YOLOv10l for 100 epochs each.

### F. Hyperparameters

The training configuration was as follows:

- **Batch Size**: 32
- **Learning Rate**: 0.01, scaled dynamically with a warmup phase.
- **Optimizer**: Stochastic Gradient Descent (SGD) with a momentum of 0.9.
- **Loss Function**: Combined classification, localization, and objectness components.
- **Epochs**: 100

These settings were optimized to balance computational efficiency and model performance, taking advantage of the high computational power provided by the A100 GPU.

## VI. Results and Discussion

The experiments conducted on YOLOv11 and YOLOv10l models provided insights into the performance of object detection for 3D printing defects. This section discusses the quantitative results, highlights key observations, and provides potential reasons for the observed trends.

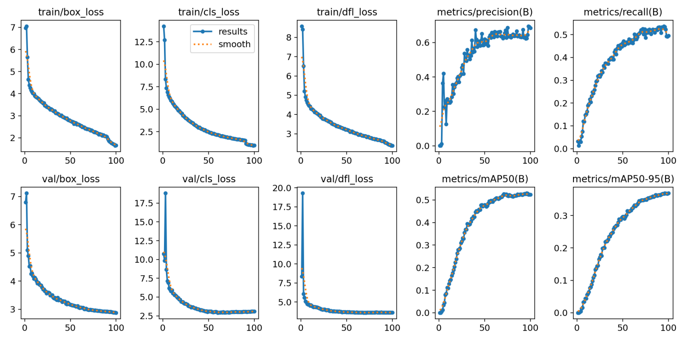

Fig. 7. Training and validation losses and evaluation metrics over 100 epochs for YOLOv10l. The plots illustrate the gradual reduction in box, classification, and distributional focal losses (DFL) as well as improvements in precision, recall, and mAP metrics during training.

## A. Quantitative Results

Table I summarizes the performance metrics of the models across the three classes: *Spaghetti*, *Stringing*, and *Warping*.

TABLE I
PERFORMANCE SUMMARY FOR YOLOV11 AND YOLOV10L MODELS.

| Model | Class | Precision | Recall | mAP@0.5 |
|---|---|---|---|---|
| YOLOv11 | Spaghetti | 0.812 | 0.749 | 0.807 |
| YOLOv11 | Stringing | 0.500 | 0.280 | 0.328 |
| YOLOv11 | Warping | 0.638 | 0.336 | 0.454 |
| YOLOv11 | All Classes | - | - | 0.486 |
| YOLOv10l | Spaghetti | 0.846 | 0.757 | 0.809 |
| YOLOv10l | Stringing | 0.507 | 0.386 | 0.328 |
| YOLOv10l | Warping | 0.523 | 0.461 | 0.453 |
| YOLOv10l | All Classes | - | - | 0.530 |

From the results:

- **Spaghetti** consistently outperformed other classes, achieving high precision, recall, and mAP scores for both models. YOLOv10l performed slightly better than YOLOv11 for this class, with an mAP@0.5 of 0.809.
- **Stringing** showed poor performance, with low recall and mAP scores, indicating significant challenges in identifying this defect.
- **Warping** achieved moderate results, with improvements in recall and mAP observed in YOLOv10l compared to YOLOv11.
- The overall mAP@0.5 for YOLOv10l was 0.530, an improvement over YOLOv11's 0.486, showing the advantage of using a larger model for this task.

## B. Discussion

The results highlight several key observations and provide insights into the challenges and potential improvements for this defect detection task:

**1. Superior Performance for Spaghetti:** The *Spaghetti* class consistently demonstrated high precision and recall

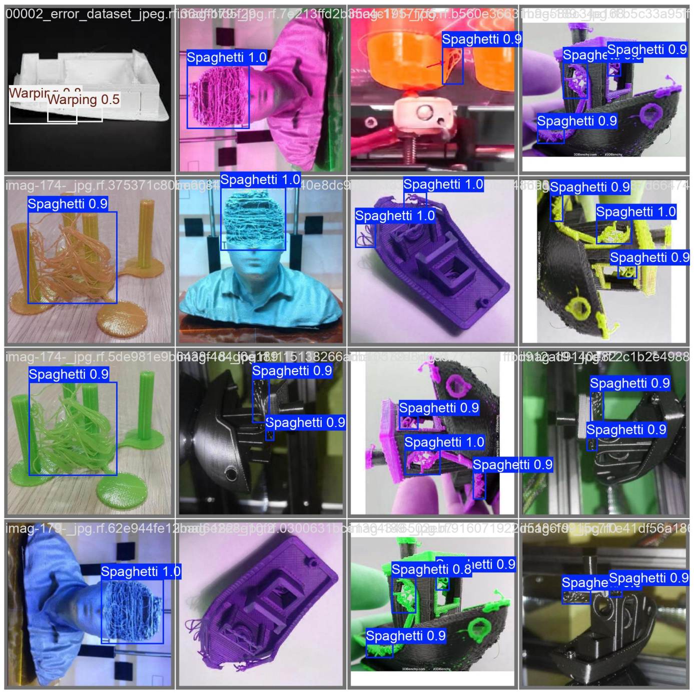

Fig. 8. Predicted bounding boxes for 3D printing defects. The image highlights the successful detection of *Spaghetti* and *Warping* defects with confidence scores.

across both models. This can be attributed to the larger number of annotations (4,885) and distinct features that make this defect easier to detect. Figure 1 shows a well-defined PR curve for *Spaghetti*, further supporting its robustness.

**2. Poor Performance for Stringing:** The *Stringing* class exhibited the lowest performance, with an mAP@0.5 of only 0.328 for both models. The confusion matrix in Figure 2 reveals that a significant proportion of *Stringing* instances were misclassified as *background*. This suggests that:

- The dataset may lack sufficient variability in *Stringing* examples.
- The features for *Stringing* overlap with those of *background*, making it harder for the models to distinguish.

Future improvements could involve targeted augmentation for *Stringing* and collecting more representative samples.

**3. Moderate Performance for Warping:** *Warping* achieved moderate performance, with an mAP@0.5 of 0.454 for YOLOv11 and 0.453 for YOLOv10l. The confusion matrix (Figure 2) indicates misclassifications with *Stringing* and *background*. These results highlight the need for better feature extraction and refinement of the dataset to improve discrimination for *Warping*.

**4. Impact of Model Size:** The larger YOLOv10l model outperformed YOLOv11 in terms of overall mAP@0.5. This can be attributed to its higher capacity, allowing it to learn more complex features. However, this improvement comes at the cost of increased computational time and resource usage.

**5. Limitations of Current Augmentation:** While augmentations such as rotation, shear, and brightness adjustments improved generalization, they may not have adequately addressed class imbalance. Specific augmentations tailored to *Stringing* and *Warping* could further enhance model performance.

**6. Analysis of Confidence Curves:** Figures 4, 5, and 6 provide insights into how confidence thresholds affect performance:

- *Spaghetti* maintained high precision and recall across all confidence levels, reflecting its robustness.
- *Stringing* and *Warping* showed significant drops in recall and F1 scores at higher confidence levels, indicating instability in predictions for these classes.

## VII. Future Work

To improve 3D printing defect detection, the following directions are proposed:

- **Dataset Enhancement:** Increase the dataset size, focusing on underrepresented classes (*Stringing*, *Warping*) through real-world samples and synthetic data generation.
- **Class-Specific Augmentation:** Develop targeted augmentations, such as occlusion and texture modifications, to address the unique characteristics of challenging defect types.
- **Advanced Architectures:** Explore transformer-based or hybrid models to enhance feature extraction and improve performance for overlapping features.
- **Real-Time Optimization:** Investigate quantization and pruning techniques to enable efficient deployment on edge devices for real-time quality control.
- **System Integration:** Build an automated pipeline for real-time defect detection and corrective actions during 3D printing processes.

## VIII. Conclusion

This study evaluated the performance of YOLOv11 and YOLOv10l models for detecting 3D printing defects across three classes: *Spaghetti*, *Stringing*, and *Warping*. The experiments demonstrated that while both models achieved reasonable performance, YOLOv10l consistently outperformed YOLOv11, with an overall mAP@0.5 of 0.530 compared to 0.486 for YOLOv11. The *Spaghetti* class showed the highest detection accuracy due to its larger representation in the dataset and distinct features, achieving a class-specific mAP@0.5 of 0.809 with YOLOv10l.

However, challenges were observed for the *Stringing* and *Warping* classes. The *Stringing* class suffered from high misclassification rates, with significant confusion with the *background*. Similarly, *Warping* showed moderate performance but also exhibited confusion with other classes, as evidenced by the confusion matrix.

The analysis of confidence curves revealed that while *Spaghetti* maintained robust performance across varying confidence thresholds, *Stringing* and *Warping* showed a sharp decline in recall and F1 scores, indicating instability in predictions for these classes. This highlights the need for targeted improvements in dataset representation and augmentation strategies to improve model robustness.

In conclusion, this study underscores the potential of YOLO-based models for detecting 3D printing defects while identifying areas for further research and optimization. Future work should focus on balancing the dataset, applying targeted augmentations for underperforming classes, and exploring advanced model architectures to enhance feature extraction and discrimination. With these improvements, the model's ability to detect complex and underrepresented defects can be significantly enhanced, paving the way for more reliable 3D printing defect detection systems.

## References

[1] Redmon, J., et al., "You Only Look Once: Unified, Real-Time Object Detection," 2016.

[2] Wang, J., et al., "YOLOv5: A State-of-the-Art Object Detection Framework," 2020.

[3] Zhang, H., and Xu, Z., "Deep Learning-Based Detection Method for 3D Printing Defects," 2021.

[4] Lee, K., and Ahn, J., "Hybrid Machine Learning Techniques for 3D Printing Defect Detection," 2022.

[5] Cheng, H., et al., "Attention Mechanisms for Enhanced Object Detection in 3D Printing," 2021.

[6] Kim, N., et al., "Class Imbalance Problem in Object Detection: Challenges and Solutions," 2023.

[7] Lu, W., and Zhou, S., "Evaluating the Performance of YOLO Models on Imbalanced Datasets," 2024.

[8] Bhatia, M., and Kumar, V., "Data Augmentation Techniques for Predictive Modeling in 3D Printing," 2023.

[9] Wang, L., and Yang, J., "Synthetic Data Generation for Defect Detection in Automated Processes," 2022.

[10] Smith, R., et al., "Enhancing Dataset Diversity for Robust Machine Learning Applications," 2024.

[11] Tran, D., and Nguyen, T., "Ensemble Learning Methods in Improving Defect Detection," 2022.

[12] Yu, Y., and Hu, Y., "Transfer Learning for Small Dataset Applications in Object Detection," 2023.

[13] Coyle, M., and Brooks, K., "Fine-tuning Pretrained Models in Deep Learning," 2023.

[14] Dai, X., and Liu, J., "Optimizing Hyperparameters for Improved Model Performance," 2023.

[15] Elzain, M., et al., "Impact of Mixed Precision on Deep Learning Model Training," 2023.

[16] Abdul, "Defect Dataset 2," *Roboflow Universe*, Open Source Dataset, August 2024. Available: https://universe.roboflow.com/abdul-hosog/defect-dataset-2.