IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Design Test Automation System Using Image Processing And Deep Learning

1.ANUJ KUMAR SINGH, 2.SUMAN GHOSH , 3.TANESHKA GIRASE

Alard college of Engineering and Management
Hinjewadi phase 1,Pune ,Maharashtra .

Abstract

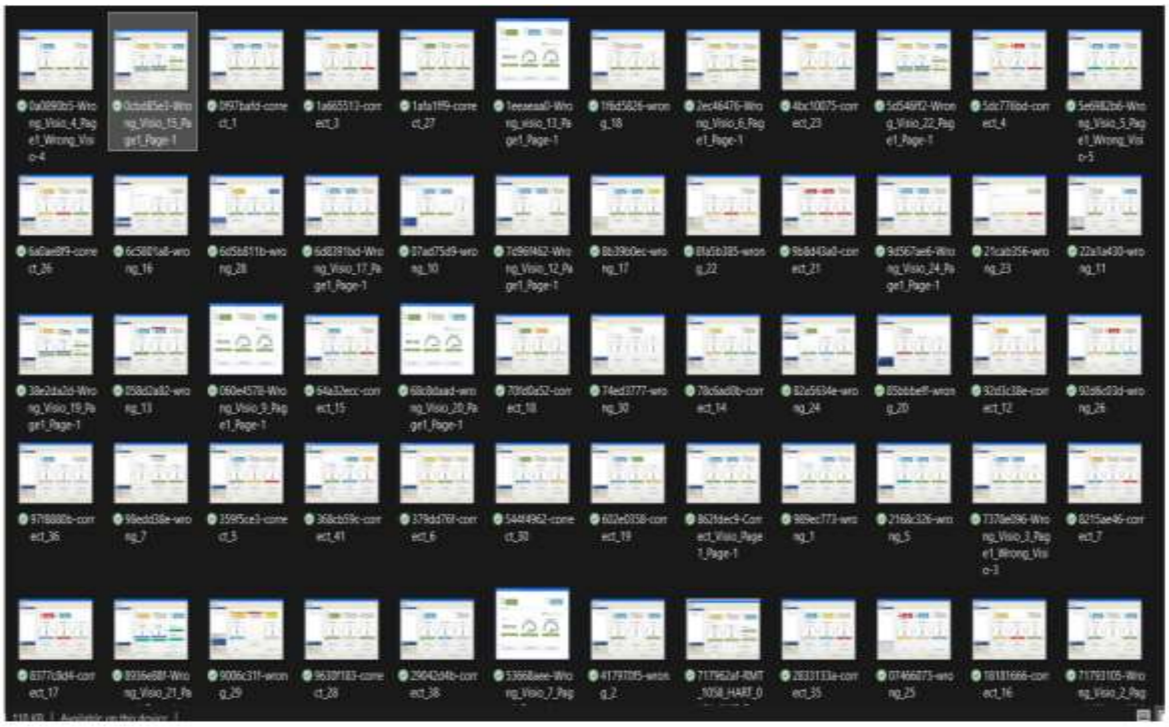
Design verification plays a crucial role in ensuring product quality and compliance with predefined standards. Conventional manual inspection methods are labor-intensive, error-prone, and unsuitable for large-scale production environments. This research proposes an automated design test system using computer vision and deep learning to overcome these challenges. The system leverages YOLOv5, a convolutional neural network–based object detection model, trained on manually annotated images using Label Studio. The trained model accurately detects predefined design components, extracts them as cropped images, and labels them with corresponding class names. These extracted components are further intended to be analyzed using a GPT-based model, which compares them against standard design templates to determine correctness and provide detailed explanations for mismatches. The proposed approach significantly improves inspection efficiency, accuracy, scalability, and interpretability.

Keywords: Design Test Automation, Computer Vision, Deep Learning, YOLOv5, Object Detection, CNN, Label Studio, GPT, Automated Inspection

1. Introduction

In the modern era of rapid industrialization and mass production, maintaining consistent design quality is a major challenge. Design test verification ensures that manufactured products conform to predefined specifications, standards, and templates. Traditionally, this task has been performed manually by human inspectors, which introduces several limitations such as fatigue, subjectivity, inconsistency, and higher operational costs.

With the advancement of artificial intelligence (AI) and deep learning (DL), automated inspection systems have become increasingly viable. Computer vision models can now analyze images with high accuracy, making them suitable for tasks such as object detection,



classification, and quality inspection. Deep learning–based object detection models like YOLO (You Only Look Once) have shown exceptional performance in real-time detection scenarios.

This research focuses on building an end-to-end **design test automation framework** that combines computer vision for visual detection and GPT-based reasoning for intelligent comparison and explanation. The system aims to reduce manual intervention, enhance reliability, and provide explainable design validation results.

2. Motivation

The motivation behind this research arises from several real-world challenges:

- Increasing complexity of product designs
- High inspection cost in manufacturing pipelines
- Human error in repetitive inspection tasks
- Lack of explainability in traditional automation systems

An intelligent automated design test system can significantly reduce these issues by ensuring faster, accurate, and consistent inspection results.

3. Problem Statement

Manual design testing methods face the following limitations:

- Time-consuming inspection processes
- Inconsistent results due to human subjectivity
- Difficulty in scaling with production volume
- Limited traceability and explanation for inspection decisions

Therefore, there is a strong need for a system that can automatically:

1. Detect design components from images
2. Verify them against standard design templates
3. Identify discrepancies
4. Provide human-readable explanations for failures

4. Objectives of the Research

The key objectives of this study are:

- To develop an automated design test pipeline using deep learning.
- To create a labeled dataset suitable for object detection tasks.
- To train and evaluate a YOLOv5 model for accurate design component detection.
- To extract detected components for further analysis.
- To integrate GPT-based reasoning for intelligent design comparison.
- To propose a deployable and scalable solution.

5. Literature Review (Brief)

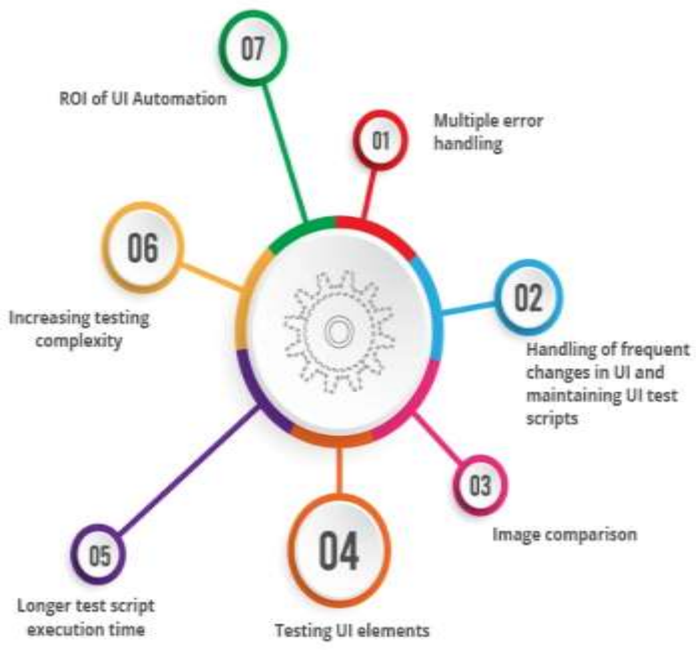
Previous studies have demonstrated the effectiveness of deep learning in quality inspection and defect detection. CNN-based architectures have been widely used for image classification and object detection in manufacturing. YOLO-based models are preferred for their real-time performance and high detection accuracy. However, most existing systems focus only on detection and lack interpretability. This research bridges that gap by integrating GPT-based reasoning to explain design mismatches.

6. System Architecture

The proposed system architecture consists of the following modules:

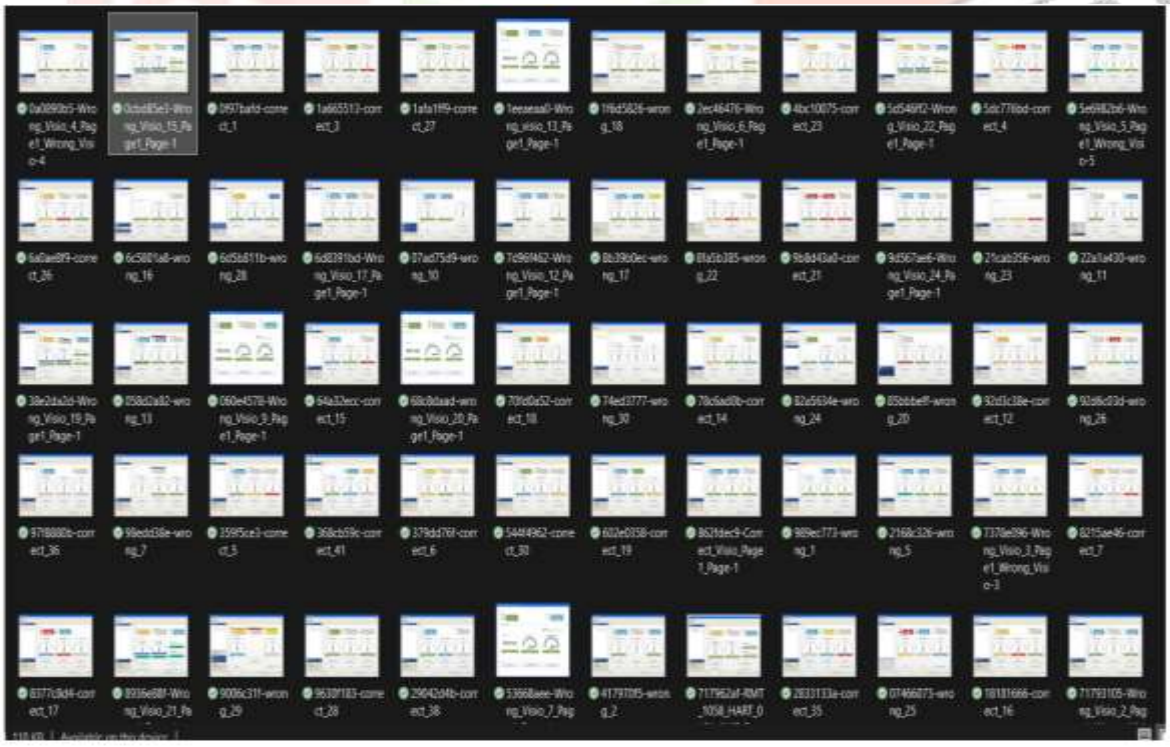
1. **Image Data Collection**
2. **Data Annotation**
3. **Model Training**
4. **Object Detection and Cropping**
5. **Design Comparison using GPT**
6. **Local Server Deployment**

Each module is interconnected to form a complete automated design test workflow.


07
ROI of UI Automation
01
Multiple error handling
06
Increasing testing complexity
02
Handling of frequent changes in UI and maintaining UI test scripts
03
Image comparison
04
Testing UI elements
05
Longer test script execution time

7. Dataset Preparation and Annotation

A dataset consisting of approximately **100 images** was prepared for this study. These images represent different design variations and components relevant to the testing process.



7.1 Annotation Process

- Label Studio was used for manual annotation.
- Bounding boxes were drawn around key design elements.
- Each bounding box was assigned a predefined class label.
- Care was taken to maintain annotation consistency and accuracy.

The annotated dataset was then exported in a YOLO-compatible format, which includes image files and corresponding label files containing class IDs and bounding box coordinates

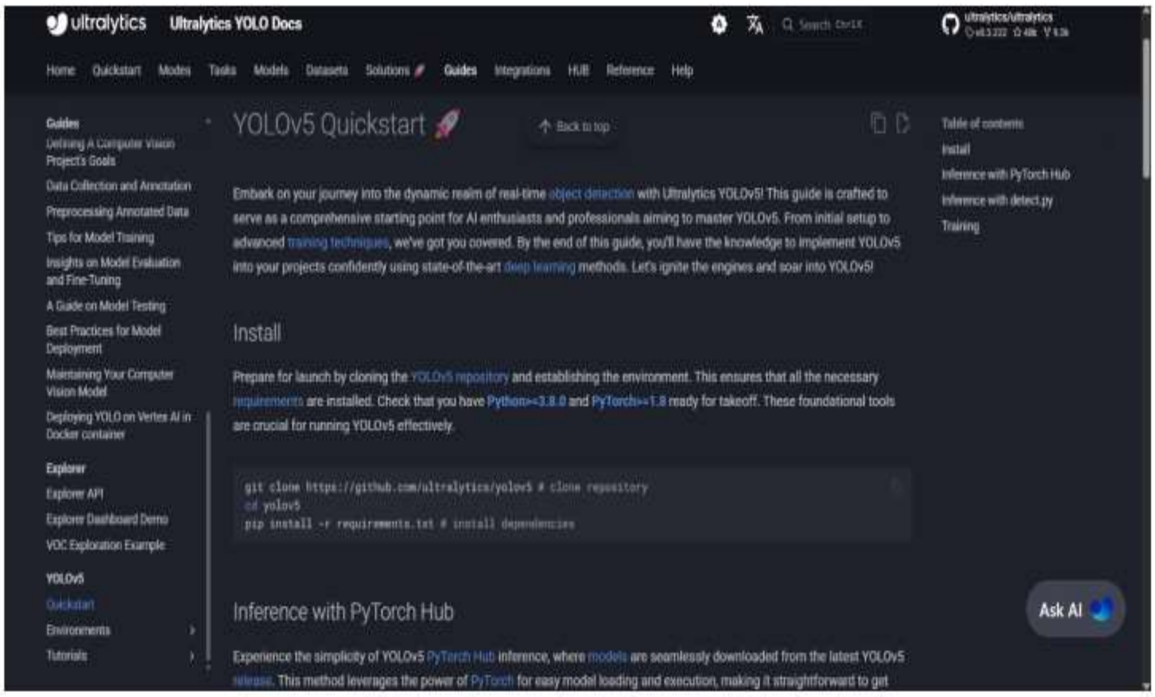
Design Test Automation

8. Model Selection

YOLOv5 was chosen as the object detection model due to the following advantages:

- Single-stage detection architecture
- High-speed inference
- Strong performance on small and medium-sized datasets
- Ease of deployment
- Open-source availability

YOLOv5 uses a convolutional neural network backbone to extract features and predict bounding boxes and class probabilities simultaneously.


ultralytics
Ultralytics YOLO Docs
Search Ctrl K
ultralytics/ultralytics
Home Quickstart Modes Tasks Models Datasets Solutions Guides Integrations HUB Reference Help
Guides
Defining A Computer Vision Project's Goals
Data Collection and Annotation
Preprocessing Annotated Data
Tips for Model Training
Insights on Model Evaluation and Fine-Tuning
A Guide on Model Testing
Best Practices for Model Deployment
Maintaining Your Computer Vision Model
Deploying YOLO on Vertex AI in Docker container
Explorer
Explorer API
Explorer Dashboard Demo
VOC Exploration Example
YOLOv5
Quickstart
Environments
Tutorials
Back to top
Table of contents
Install
Inference with PyTorch Hub
Inference with detect.py
Training
Ask AI

YOLOv5 Quickstart

Embark on your journey into the dynamic realm of real-time object detection with Ultralytics YOLOv5! This guide is crafted to serve as a comprehensive starting point for AI enthusiasts and professionals aiming to master YOLOv5. From initial setup to advanced training techniques, we've got you covered. By the end of this guide, you'll have the knowledge to implement YOLOv5 into your projects confidently using state-of-the-art deep learning methods. Let's ignite the engines and soar into YOLOv5!

Install

Prepare for launch by cloning the YOLOv5 repository and establishing the environment. This ensures that all the necessary requirements are installed. Check that you have Python>=3.8.0 and PyTorch>=1.8 ready for takeoff. These foundational tools are crucial for running YOLOv5 effectively.

```
git clone https://github.com/ultralytics/yolov5 # clone repository
cd yolov5
pip install -r requirements.txt # install dependencies
```

Inference with PyTorch Hub

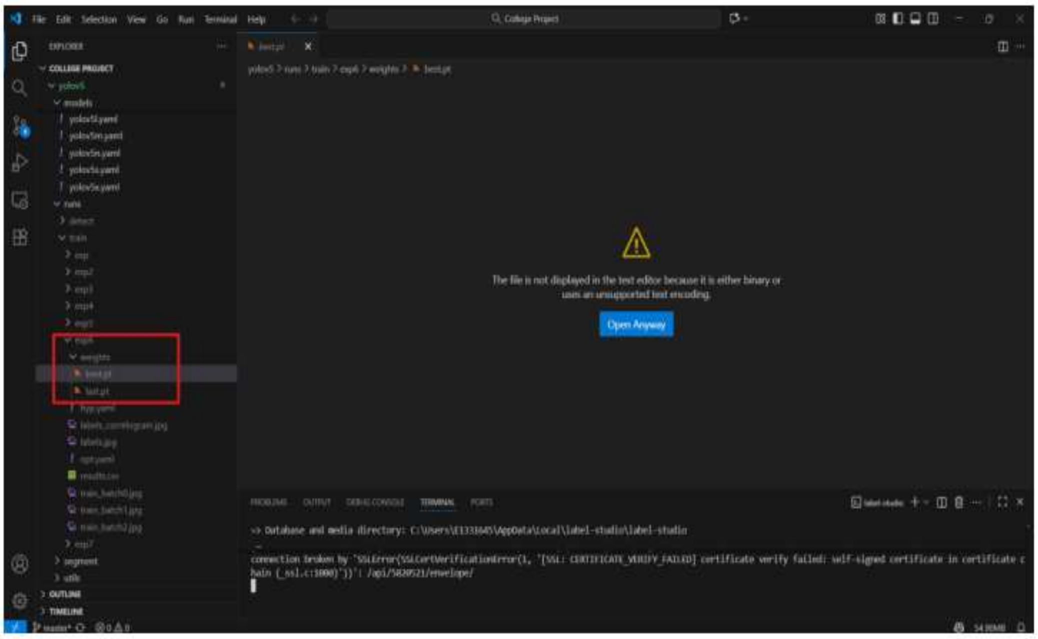
Experience the simplicity of YOLOv5 PyTorch Hub inference, where models are seamlessly downloaded from the latest YOLOv5 release. This method leverages the power of PyTorch for easy model loading and execution, making it straightforward to get

9. Model Training Methodology

The training process involved the following steps:

1. Dataset splitting into training and testing sets
2. Configuration of YOLOv5 parameters
3. Training over multiple epochs
4. Monitoring loss functions and accuracy metrics

3. The model training process produced the **'best.pt'** file as the optimal weights checkpoint.



The file is not displayed in the text editor because it is either binary or uses an unsupported text encoding.

```
=> Database and media directory: C:\Users\E1333645\AppData\Local\label-studio\label-studio
connection broken by 'SSLError(SSLCertVerificationError(1, '[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: self-signed certificate in certificate c
hain (_ssl.c:1000)'))': /api/5820521/envelope/
```

During training, the model learned to identify spatial and visual patterns associated with each design class. Upon completion, the training process generated a **best.pt** file, which represents the best-performing weights based on validation performance .

10. Model Evaluation and Testing

The trained YOLOv5 model was evaluated using unseen test images. The **detect.py** script was used to perform inference.

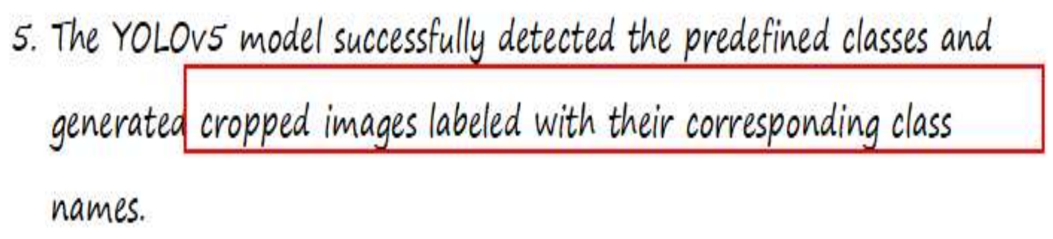
10.1 Evaluation Results

The model successfully:

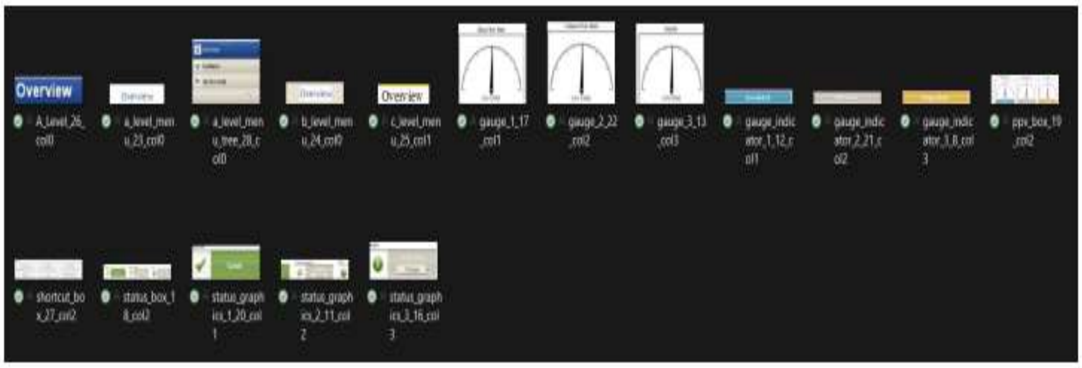
- Detected all predefined design classes
- Generated accurate bounding boxes
- Cropped detected components automatically
- Assigned correct class labels to cropped images

Design Test Automation

These results validate the effectiveness of the trained model for real-world design inspection tasks.

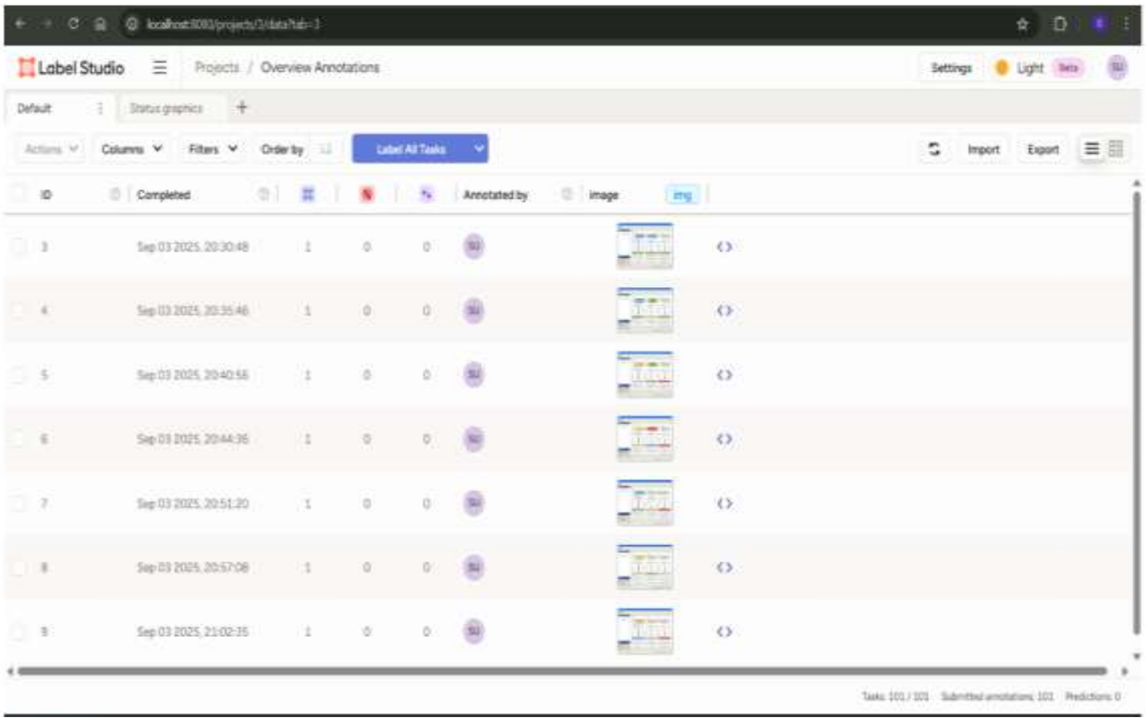


5. The YOLOv5 model successfully detected the predefined classes and generated cropped images labeled with their corresponding class names.



11. Cropped Image Extraction

Once detection is completed, each detected component is extracted as a separate cropped image. These cropped outputs serve as inputs for the next stage of analysis, ensuring modularity and flexibility in system design.



localhost:8080/projects/3/data?tab=3

Label Studio Projects / Overview Annotations

Default | Status graphics

ID	Completed				Annotated by	Image
3	Sep 03 2025, 20:30:48	1	0	0	SU	
4	Sep 03 2025, 20:35:46	1	0	0	SU	
5	Sep 03 2025, 20:40:56	1	0	0	SU	
6	Sep 03 2025, 20:44:36	1	0	0	SU	
7	Sep 03 2025, 20:51:20	1	0	0	SU	
8	Sep 03 2025, 20:57:08	1	0	0	SU	
9	Sep 03 2025, 21:02:35	1	0	0	SU	

Tasks: 101 / 101 Submitted annotations: 101 Predictions: 0

12. Design Verification Using GPT Models

The extracted cropped images are intended to be processed by a GPT-based model. This stage performs:

- Comparison between detected components and standard design templates
- Semantic and visual analysis
- Decision-making on design correctness
- Generation of textual explanations for mismatches

This approach adds **explainable AI (XAI)** capabilities to the system, which is a major improvement over traditional black-box inspection models.

13. Deployment on Local Server

To enhance usability and accessibility, the complete pipeline is planned to be hosted on a **local server**. Local deployment offers:

- Faster processing
- Reduced latency
- Improved data privacy
- Seamless integration with internal systems

14. Advantages of the Proposed System

- Automated and consistent inspection
- Reduced human dependency
- Scalable for industrial use
- Explainable results
- Modular and extensible architecture

15. Limitations

- Model accuracy depends on dataset quality
- Limited dataset size may affect generalization
- GPT-based analysis requires careful prompt engineering

16. Future Scope

Future enhancements include:

- Expanding dataset size
- Supporting real-time camera feeds
- Cloud-based deployment
- Multi-design and multi-class validation
- GPT visual reasoning integration

17. Conclusion

This research successfully demonstrates a deep learning–based design test automation system using YOLOv5 and GPT models. The system efficiently detects design components, extracts relevant information, and provides intelligent verification with explainable outputs. The proposed framework shows strong potential for adoption in industrial quality inspection and design verification systems.

18. References

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, pp. 779–788, 2016.

[2] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” *arXiv preprint arXiv:2004.10934*, 2020.

[3] G. Jocher, A. Chaurasia, and J. Qiu, “YOLO by Ultralytics,” Ultralytics, GitHub Repository, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[5] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, “Object Detection with Deep Learning: A Review,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019.