Deep Learning for Appearance Defect Inspection in Laptops: A Model Comparison

João P. O. Santiago[1], **Lucas Cabral**[1], **Lucas Sena**[1], **Joaquim Bento C. Neto**[1], **Yuri Lenon**[1], **Javam Machado**[1]

[1]Systems and Databases Laboratory (LSBD)
Computer Science Department (DC) – Universidade Federal do Ceará (UFC)
Fortaleza, CE – Brazil

{pedro.santiago,lucas.cabral,lucas.sena,
joaquim.bento,yuri.lenon,javam.machado}@lsbd.ufc.br

Abstract. *Daily, thousands of laptops pass through repair centers of electronic device manufacturers. During visual inspections, technicians manually identify potential cosmetic or structural damage, documenting findings with photos or videos. This process safeguards the repair center against claims related to damage incurred in custody. However, the manual approach is time-consuming and prone to human error. To address this, we propose a computer vision model to automatically detect appearance defects in laptops. Experiments were conducted using five deep neural networks: Mask R-CNN, SSD, Swin Transformer, YOLOv5, and YOLOv10. Our best-performing model achieved a mean Average Precision (mAP) of 70.2%, showcasing the viability of such application.*

1. INTRODUCTION

Ensuring defect-free products is vital for maintaining quality standards in the electronics industry. This not only applies to new products but is equally important during the warranty service of semi-new items. In the laptop manufacturing sector, repair centers must accurately document the condition of devices both when they arrive and before they are returned to consumers. Detailed records of any pre-existing appearance damage, completed repairs, and the final state of the product are crucial for avoiding legal disputes.

Appearance damage in laptops includes visible and often severe physical flaws that suggest rough handling. This damage may involve deep scratches, cracks, broken or missing parts, screen issues like dead pixels or burn-in, bending, hinge damage, and visible stains or discoloration. While typically not affecting the laptop's immediate functionality, appearance damage can compromise structural integrity and make the device look heavily used or neglected.

Currently, the appearance inspection documentation process is handled manually by technicians, who record laptops using video and photographs from multiple angles. This practice is widespread in electronics manufacturers' repair centers [Alves et al. 2022] where visual inspection and manual cataloging are essential for warranty management. However, this process is time-consuming, expensive, and prone to human error, especially when it comes to detecting minor appearance damages. These challenges have driven growing interest globally in leveraging image processing and computer vision techniques to automate damage detection. Recent advancements in object detection and deep learning, such as YOLO-based models and Transformers, have spurred applications in manufacturing and quality control, as evidenced by studies addressing appearance damage detection in laptops [Yang et al. 2023] and surface defects in other electronic components [Wang et al. 2023].

Traditional damage detection methods involve image processing, contour detection, feature extraction, and classification. These techniques have been widely used in industrial defect detection, as evidenced by studies such as [Lin and Landge 2021] and [Yang et al. 2023], which apply these approaches to identify surface damage in laptop cases. However, these methods often depend heavily on threshold values, which can limit their robustness in real-world applications [Lin and Landge 2021].

Recent studies suggest that defects on laptop surfaces can be identified using various data-driven approaches, such as supervised learning [Verma et al. 2021], Convolutional Neural Networks (CNN) [Lin and Landge 2021, Wang et al. 2023, Yang et al. 2023], unsupervised learning, anomaly detection [Zhu et al. 2022], or combinations of these techniques [Zhang et al. 2023]. Established methods in the literature, such as those discussed in [Yang et al. 2023] and [Wang et al. 2023], typically focus on surface damage occurring during manufacturing. However, such methods often do not address Customer Induced Damage (CID), damages occurring post-purchase, which are highly relevant for laptop manufacturers' warranty programs.

In this work, we evaluated object detection methods using models based on CNNs and Transformers [Carion et al. 2020] to identify appearance defects on laptops. More specifically, we focus on appearance defects that are CID. The dataset used in our experiments comprises images captured during the packaging and unpacking processes of laptops, documenting their condition as they enter and leave the repair center. We aim to identify and locate appearance damage on the laptops to automate the device documentation process. The images were collected in an electronics manufacturer's repair center. A fixed camera is positioned where technicians display the contents of receiving/return boxes for laptops from the repair center to document the condition of the equipment during this phase. Due to the variety of damages, the laptops may not be photographed in the same positions and orientations, adding complexity to the problem.

Most work found in the literature [Yang et al. 2023, Wang et al. 2023, Lin and Landge 2021] focuses on detecting appearance damage produced during manufacturing, thus making use of images in standardized positions acquired in a controlled environment. To the best of our knowledge, we are the first ones to address appearance damage detection for CID during warranty inspections, which implies in a less restrained scenario in terms of variability of images, due to non-standardized lighting, a wide variety of position, orientation, angle of view and damage type seen during manual inspection conducted by technicians. Our results outline the viability of using a deep learning approach for semi-automated appearance damage inspection in laptops.

The work is organized as follows: Section 2 provides the essential theoretical foundations for understanding the topic. Section 3 reviews related work, offering context and support for this research. Section 4 details the methodology and steps undertaken in this study. Section 5 presents the experiments conducted and the results obtained. Finally, Section 6 discusses the conclusions drawn from this work.

2. THEORETICAL BACKGROUND

2.1. Computer Vision and Object Detection

The field of computer vision addresses numerous challenges in visual recognition, such as classifying images, detecting objects, segmenting semantic regions, and recognizing faces [Santiago et al. 2024]. Object detection focuses on identifying objects, such as animals, vehicles, and humans, by delineating their positions within images using bounding boxes [Xiao et al. 2020].

Recently, object detection has gained significant attention due to the advancements in deep learning methods. A significant factor contributing to the advancements of deep learning-based methods is the development of CNN. Unlike traditional approaches, CNNs do not require the manual creation of feature extractors or filters and are capable of achieving impressive results on large datasets [Santiago et al. 2024]. However, CNNs require substantial amounts of data to function effectively and often struggle to achieve satisfactory results when data availability is limited [Alzubaidi et al. 2021].

2.2. Deep learning for object detection

2.2.1. SSD

Single Shot MultiBox Detector (SSD) [Liu et al. 2016] is a method for object detection in images using a single deep neural network. This method discretizes the output space of bounding boxes by defining a set of default boxes with various aspect ratios and scales for each feature map location. During prediction, the network assigns scores for the presence of each object category within each default box and makes adjustments to improve the fit of the object's shape. Additionally, it combines predictions from multiple feature maps at different resolutions to accommodate a range of object sizes effectively. Compared to methods that depend on object proposals, SSD is simpler because it removes the need for proposal generation and the following pixel or feature resampling stages, consolidating all computations within a single network.

2.2.2. YOLO

YOLO [Redmon 2016] consists of a family of state-of-the-art detectors for real-time object detection, known for their strong performance while maintaining good accuracy in object detection. In YOLO, detection is accomplished through a single stage that extracts image features using a CNN. A single CNN predicts multiple bounding boxes and the corresponding object class probabilities for each box at the same time. This approach enables YOLO to deliver high accuracy in object detection tasks. YOLO became very popular in real-time object detection due to its features, and several projects emerged as innovations to improve performance and accuracy, giving rise to a series of models based on the original YOLO. YOLOv10 [Wang et al. 2024] reveals that the dependence on non-maximum suppression (NMS) for post-processing creates challenges for end-to-end deployment and increases inference latency. YOLOv10 uses an NMS-free training approach and a holistic design strategy focused on balancing efficiency and accuracy, becoming state-of-the-art in real-time object detection.

2.2.3. Mask R-CNN

Mask R-CNN, developed by Facebook's AI Research group, is a Deep Neural Network framework designed for object detection and instance segmentation [He et al. 2017]. Building upon the Faster R-CNN architecture [Ren et al. 2016], it introduces a segmentation branch to predict object masks alongside detecting bounding boxes. The framework first extracts feature maps using a backbone network, which are then passed through the Region Proposal Network (RPN), a lightweight binary classifier that proposes regions likely to contain objects. These regions are processed by the RoIAlign layer, which aligns them with the corresponding feature map areas to generate bounding boxes. The resulting outputs are fed into fully connected layers for classification and also utilized by CNNs to predict segmentation masks.

2.2.4. Swin Transformer

The Swin Transformer [Liu et al. 2021] is a method developed by Microsoft Research Asia that employs a new vision Transformer as a general-purpose backbone for computer vision tasks. This approach adapts Transformers from the language domain to the vision domain, tackling challenges such as the significant scale variation of visual entities and the high pixel resolution of images compared to the discrete nature of words in a text. To address these challenges, the method employs a hierarchical Transformer with shifted windows. This shifted windowing mechanism enhances efficiency by confining the self-attention module's computations to non-overlapping local windows, while also facilitating cross-windows connections.

2.3. Evaluation metrics for object detection

IoU, also known as the Jaccard Index, is a key metric in computer vision for evaluating the performance of object detection algorithms, particularly in image segmentation and object localization. It measures the overlap between a predicted bounding box and an object's corresponding ground truth bounding box. Formally, IoU is defined for two convex shapes A, B as $IoU = |\frac{A \cap B}{A \cup B}|$. This metric effectively addresses damage location issues and is often employed to evaluate model performance. To evaluate the problem, IoU is applied as a threshold, along with other metrics, such as precision and recall.

The Precision metric indicates the proportion of your accurate predictions. It can be expressed as the Equation 1. Another standard metric for model evaluation is Recall, which follows Equation 2. It evaluates the effectiveness of our predictions, highlighting that False Negatives can significantly impact our performance. The IoU threshold enables us to determine the True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN) for each category in our classification task. In the literature, default values for the IoU threshold are commonly set at IoU@50 (50%) and IoU@75 (75%).

$$Precision = \frac{TP}{TP + FP}. \tag{1}$$

$$Recall = \frac{TP}{TP + FN}. \tag{2}$$

For every class in the dataset, the predictions of one model are ordered according to confidence scores. A precision-recall curve is created by adjusting the threshold for classifying a detection as positive. The area under this curve is then computed, resulting in the Average Precision (AP) for each class. AP reflects the model's effectiveness regarding precision and recall for that particular class. The AP values for all classes are averaged to calculate the mean Average Precision (mAP), a unified performance metric for the object detection model across all categories. The mAP can be expressed as the Equation 3, where N is the number of classes.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i. \tag{3}$$

3. RELATED WORK

3.1. Summary of Related Works

The study conducted by [Lin and Landge 2021] highlights the challenges of defect detection in metal laptop cases, emphasizing that traditional image processing methods, such as thresholding and contour extraction, depend heavily on threshold values. Variations in image quality across metallic surfaces can significantly affect detection outcomes, complicating the identification of defects. To tackle this issue, the authors employ advanced object detectors like YOLOv3, Faster R-CNN, and SSD, with Faster R-CNN demonstrating superior precision and recall—achieving 80% recall and 80% precision. At the same time, YOLOv3 offers the highest mAP at 83% with a 73% recall.

[Wang et al. 2023] introduce ATT-YOLO, a novel surface defect detection model tailored for electronics manufacturing, leveraging the strengths of YOLO object detectors. This innovative design features a multiscale backbone for enhanced feature extraction, enabling the model to effectively identify objects across varying scales. Additionally, ATT-YOLO incorporates a self-attention module that refines the model's focus on pertinent features within input images, resulting in improved detection accuracy.

Building on the challenges associated with defect detection, surface damage identification on laptops coated with aluminum alloy presents additional obstacles, particularly concerning the size and depth of scratches. To address these issues, [Yang et al. 2023] utilize a YOLOv5 object detection model to identify scratches on laptop surfaces effectively. Their innovative approach includes the introduction of the C3 module to enhance the YOLOv5 architecture, resulting in improved performance metrics with an accuracy of 95% and a recall of 88%.

Work found in literature about defect detection in laptops cover manufacturing defects, while CIDs are overlooked. Recent studies have demonstrated the detection of CIDs in Printed Circuit Board (PCB) using deep learning methods and object detection techniques [Alves et al. 2022, Cabral et al. 2023, Santiago et al. 2024]. Those studies point out specific challenges that arise from damage detection outside from a controlled environment of a production line, such as higher diversity of visual features, due different angle, lighting conditions and background of images.

3.2. Discussion of Related Work

Table 1 situates our research within the context of existing literature. In summary, our approach distinguishes itself from previous studies in the following key aspects:

1. Our detection process encompasses damages that extend beyond the surface of the laptop case. This includes chassis damage, hinge damage, and keyboard damage.
2. In other studies, detection typically occurs on a production line, where laptops are positioned in a fixed, standardized manner. In our approach, however, detection takes place during warranty inspections, where a technician manipulates the laptop, capturing images from varying angles and positions. This unconstrained scenario introduces additional complexity to the problem.

Table 1. Related Work on Damage Detection.

Work	**Task**	**Damage Coverage**	**Evaluation Metrics**
[Lin and Landge 2021]	Detecting surface defects on metal laptop cases using object detectors.	Deep scratches, light scratches, dots.	Precision and Recall.
[Wang et al. 2023]	Detecting surface defects during electronics manufacturing.	Dirt, particles, edge, collision, scratch, unknown.	mAP.
[Yang et al. 2023]	Identifying scratches on laptop surfaces with a focus on coating damage.	Scratches on coating.	Accuracy and Recall.
Ours	Captures images from varying angles and positions, detecting cosmetic and structural damage in semi-new laptops post-manufacturing.	Casing damage, broken hinge, keyboard damage.	mAP and Recall.

4. MATERIALS AND METHODS

4.1. Dataset

Our dataset comprises a collection of images of laptops that have undergone repair at an electronics manufacturer's service center, specifically focusing on those with at least one identified appearance damage. The images were sourced from a private company with repair centers located globally. Each image is accompanied by annotations indicating the locations of appearance defects as well as their respective classes. The dataset encompasses three distinct types of appearance damages, with a detailed description of each provided in Table 2.

Table 2. Description of dataset classes.

Class	**Description**
broken hinge	Appearance damages affecting the laptop's hinge including any deformities or functional impairments.
casing damage	Appearance damages on the laptop chassis include scratches, cracks, dents, and burn marks.
keyboard damage	Appearance damages to the keyboard include missing keys, sunken keys, loose keys, and damaged key fittings.

All annotations were marked and classified by an expert. The annotation procedure begins with the identification of the damage, followed by its delineation. Subsequently, the expert classifies the damage according to the established criteria. The dataset comprises a total of 637 instances of damage across 350 images. The distribution of the different damage classes is detailed in Table 3.

Table 3. Distribution of classes in the dataset.

Class	Amount	Class Distribution (%)
broken hinge	172	26.91
casing damage	378	59.15
keyboard damage	87	13.61
Total	637	

4.2. Experimental Procedure

Section 2.2 introduced object detection methods utilizing Deep Learning Networks. Each of these methods is assessed using the MS COCO [Lin et al. 2014] dataset, a comprehensive dataset designed for identifying and classifying a diverse array of objects, including people, vehicles, animals, and various other items. Our greatest challenge lies in addressing small damages, and we have determined that these models are well-suited for our data. They can effectively identify and classify small objects in the general-purpose COCO dataset with a high accuracy rate.

In Section 2.3, the metrics Precision, Recall, and mAP were introduced for evaluating object detection models. These metrics were utilized to assess our models' performance during both the training and prediction phases.

The dataset was created and then split into five folds for cross-validation with $K = 5$. Each fold followed a standard K-fold approach, dividing the dataset into five equal parts. In each iteration, four parts (280 images) were used for training, and the remaining part (70 images) was used for testing. This approach ensured that every image appeared in both training and testing sets across different iterations without a fixed holdout set. To maintain a balanced representation of classes across splits, stratification was applied during the dataset partitioning. We used data augmentation techniques such as cropping, flipping, rotation, etc., to increase the diversity of our data during training.

Our experiments were conducted in the Google Colab Pro+ environment, utilizing Python 3.10, PyTorch 1.13 [Stevens et al. 2020], and a GPU NVIDIA A100 of 40 GB for model training. For the Mask R-CNN, Swin, and SSD experiments, we employed the MMDetection 2.28 [Chen et al. 2019] platform, while YOLO experiments were conducted using the Ultralytics 8.3 [Jocher et al. 2023] platform. Both platforms utilize the PyTorch library to train machine-learning models. The primary motivation for selecting these platforms is their ease of use in training, managing, and configuring the various deep neural network architectures available for experimentation.

We selected and fine-tuned the hyperparameters for each of the methods based on commonly used empirical values reported in the literature, as well as preliminary grid searches on a validation set. For instance, learning rates were initially chosen from standard values (e.g., 0.001 or 0.01), then refined based on early convergence behavior and validation loss trends. Batch sizes were selected considering both model stability and GPU memory constraints. We trained each model until convergence, defined as no significant improvement in validation loss. On average, training took approximately 6 hours per model. These choices were made to balance performance and computational efficiency while ensuring fair comparison across methods. Table 4 shows the selection of hyperparameters for each of the experiments conducted in our study.

Table 4. Hyperparameters of Experimented Models.

Model	Backbone	Optimizer	Learning Rate	Batch Size	Anchor Gen.
Mask R-CNN	ResNet 101	Adam	0.0001	16	4, 8, 16, 32, 64
Mask R-CNN	Swin-S	AdamW	0.0001	16	4, 8, 16, 32, 64
SSD	VGG16 [Simonyan and Zisserman 2014]	SGD (momentum=0.9)	0.001	64	8, 16, 32, 64, 128, 256, 512
YOLOv5-X	CSP-Darknet53	SGD (momentum=0.937)	0.01	16	Scale 1: [10, 13, 16, 30, 33, 23]; Scale 2: [30, 61, 62, 45, 59, 119]; Scale 3: [116, 90, 156, 198, 373, 326]
YOLOv10-X	CSPNet [Wang et al. 2020]	SGD (momentum=0.937)	0.01	16	Not applicable

5. RESULTS AND DISCUSSION

The five object detection methods based on deep neural networks were trained and evaluated using our dataset. Table 5 presents the results of the experiments conducted with the five model architectures. Upon reviewing the results, it is clear that YOLOv10-X achieved the best object detection performance in our experiments.

Analyzing the mAP at various IoU threshold values, we observe that the architecture YOLOv10-X achieved the best results, with a mAP@0.5 of 70.28% and a mAP@0.5-0.95 of 51.68%. The second-ranked model in terms of mAP was YOLOv5-X, which recorded a mAP@0.5 of 69.44% and a mAP@0.5-0.95 of 47.42%. Swin-S achieved a commendable performance at the mAP@0.5 threshold, with a mAP@0.5 of 60.82%, placing it third overall in this category. However, its performance at the mAP@0.5-0.95 threshold was relatively low compared to the YOLO models. This decreased performance could be attributed to the fact that mAP@0.5-0.95 is calculated across multiple IoU thresholds, ranging from 0.5 to 0.95 in increments of 0.05, indicating that the model may struggle to achieve sufficient overlap at higher IoU values.

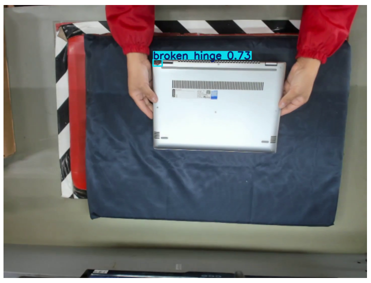
Table 5. Results of the model experiments.

Model	mAP@0.5-0.95	mAP@0.5	Precision	Recall	Inference (ms)
Mask R-CNN	0.198	0.409	0.314	0.374	44.3
SSD	0.278	0.572	0.551	0.383	14.2
Swin-S	0.343	0.608	0.361	0.527	68.1
YOLOv5-X	0.474	0.694	0.768	0.577	**3.2**
YOLOv10-X	**0.516**	**0.702**	**0.797**	**0.579**	3.3

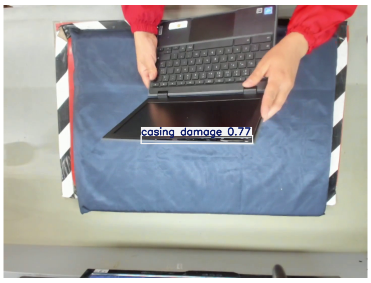
Our best model achieves a precision of 79.7%, accompanied by a recall of 57.9%. This indicates that when the model claims to have detected an object, it is generally correct, but it faces some difficulty in identifying all relevant objects. Such behavior can be explained by the challenges present in our dataset. Despite the challenges found in our dataset, the model achieves good results in detecting cosmetic damage in laptops.

In our analysis, the top-performing model detected all three types of appearance damage proposed in this study, often demonstrating strong detection capabilities and showing high confidence scores. Figure 1 presents examples of the inferences made by our best model for each of the identified damage classes.

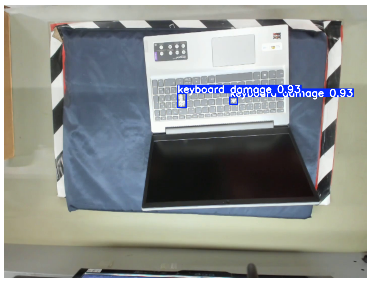
Our dataset poses several challenges for identifying patterns, including non-standardized lighting, photographs taken from various angles, small damages, and a wide



(a) Broken hinge.



(b) Casing damage.



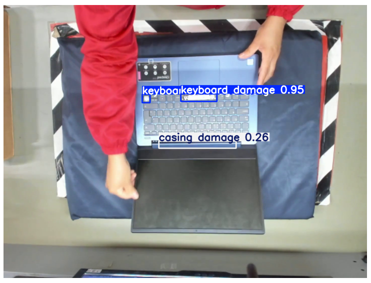
(c) Keyboard damage.

Figure 1. Cosmetic damage detections in laptops.

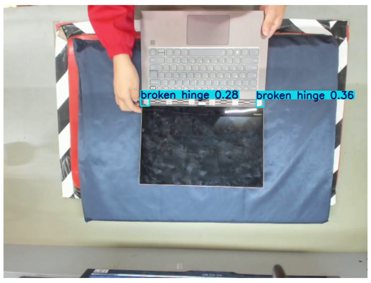
variety of damage types. These factors hinder the performance of our models. Upon analyzing the predictions from YOLOv10-X, we identified three common types of errors in its predictions within our dataset.

The first type of error occurs when a laptop is presented with its bottom casing visible, particularly when the design is uncommon in our dataset. This specific design includes cooling vents situated near the laptop's hinges. In such instances, the model may mistakenly interpret the air vents close to the hinges as a broken hinge, depending on the design of those vents. Figure 2c shows an example of when the model makes this type of error.

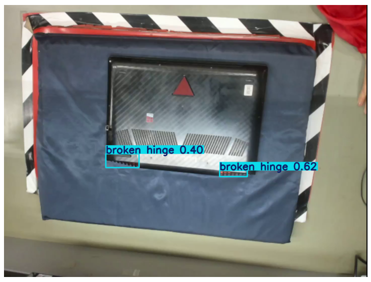
The second most common type of error involves FP in the regions of the casing between the keyboard and the laptop display. This error arises because, in certain instances of damage, this area may appear damaged or misaligned with the rest of the casing. However, due to the significant variability in our dataset and the lack of a distinctive image pattern—particularly for casing damages—the model may misinterpret design differences in this area as casing damage (when more centralized to the body of the laptop) or hinge damage (when located near the edges). This is shown in the Figures 2a and 2b.



(a) FP of casing damage.



(b) FP of broken hinge (position).



(c) FP of broken hinge (air vents).

Figure 2. Most common errors.

The final common type of error occurs when the model fails to detect actual damage in laptops, resulting in an FN. This error is particularly critical, as it may lead to undetected damage in laptops with voided warranties. Consequently, these devices could be incorrectly deemed eligible for warranty coverage, potentially causing financial losses for the manufacturer.

Overall, YOLOv10-X demonstrated excellent performance according to COCO metrics, achieving the best results across all evaluated metrics. The mAP@0.5 score is particularly suitable for addressing our challenge of automating the cataloging of laptops received and returned by repair centers. The experiments further indicate that both YOLO models perform effectively within the environmental setup employed during this study, thereby establishing YOLOv10-X as our preferred model. Additionally, the methods we proposed for detecting appearance defects on laptops can effectively automate damage detection, despite the challenges posed by our dataset, which includes significant variability in image standardization and a diverse range of damages. The detected damages may also be classified as CID and are typically not covered under the warranty for laptop repairs in repair centers.

Despite achieving good initial results, our best model still exhibits three common types of errors that negatively impact its overall performance. These errors may be associated with limitations in our dataset, which should be mitigated in the future by incorporating new data and implementing more advanced pipelines for data augmentation.

6. CONCLUSION

The proposed work presents experiments utilizing computer vision methods based on deep neural networks to detect appearance damage in laptops. The study aims to evaluate the feasibility of automating the detection of such damage for potential application in the repair centers of an electronics manufacturer. Appearance defects in laptops are an important area of study, as repair centers need to document the condition of devices both upon receipt and when returned to consumers. Furthermore, appearance damage can be linked to Customer Induced Damage (CID), which refers to harm caused by consumers or unauthorized third parties. This type of damage can result in internal damage to the printed circuit boards (PCBs) and components of the devices, ultimately leading to malfunctions. We evaluated several methods, including Mask R-CNN, SSD, Swin Transformer, YOLOv5, and YOLOv10, for detecting appearance defects in laptops, specifically targeting visible damage to the casing, keyboard, and hinges. The dataset employed in our experiments is highly challenging, characterized by a wide range of damage types, diverse image perspectives, and lighting-related issues. Three innovative aspects of the developed methodology can be highlighted: the application of Deep Neural Networks for detecting appearance defects on the casing, keyboard, and hinges of laptops; the utilization of a challenging dataset that presents issues related to lighting, laptop positioning, and a variety of small damages; and the extension of the CID classification problem to encompass appearance defects in laptops.

We selected the models for these experiments based on their reliable detection capability in computer vision tasks, as they have achieved state-of-the-art results. Analyzing our results, we can confirm that YOLOv10-X achieves the best results for the proposed application YOLOv5-X has the second-best performance regarding mAP, precision, and recall metrics. The Swin and SSD models fall in the middle of the results table, with Swin-S achieving the third-highest mAP@0.5 score across all experiments, second only to YOLO models. However, its mAP@0.5-0.95 is relatively lower than that of YOLOv10-X and YOLOv5-X, which leads to this metric. The Mask R-CNN model shows the poorest performance among all experiments, ranking last in all evaluation metrics.

We emphasize the significance of our study, which demonstrates that object detection methods can effectively identify appearance defects in laptops. This capability may assist technicians in cataloging the condition of devices upon their arrival at and departure from the repair center, thereby streamlining and enhancing the efficiency of the cataloging process. For future work, we aim to modify the YOLOv10 architecture to better align the network with our specific dataset, thereby improving model performance during training. Additionally, we plan to acquire more data using techniques such as active learning and Data-Centric AI methods to increase the diversity of our data for future model training.

References

Alves, D., Farias, V., Chaves, I., Chao, R., Madeiro, J. P., Gomes, J. P., and Machado, J. (2022). Detecting customer induced damages in motherboards with deep neural networks. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., and Farhan, L. (2021). Review of deep learning: concepts, cnn architectures, challenges, applications, future directions. *Journal of big Data*, 8:1–74.

Cabral, L., Farias, V., Sena, L., Chaves, I., Pordeus, J. P., Santiago, J. P., Sá, D., Machado, J., and Madeiro, J. P. (2023). An active learning approach for detecting customer induced damages in motherboards with deep neural networks. *Learning & Nonlinear Models*, 21(2):29–42.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer.

Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C. C., and Lin, D. (2019). Mmdetection: Open mmlab detection toolbox and benchmark.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969.

Jocher, G., Chaurasia, A., and Qiu, J. (2023). Yolo by ultralytics.

Lin, H.-I. and Landge, R. R. (2021). Comparison of deep learning algorithms on defect detection on metal laptop cases. In *2021 International Automatic Control Conference (CACS)*, pages 1–5. IEEE.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022.

Redmon, J. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, volume abs/1506.02640.

Ren, S., He, K., Girshick, R., and Sun, J. (2016). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149.

Santiago, J. P., Farias, V., Sena, L., Gomes, J. P. P., and Machado, J. (2024). Real-time detection of customer-induced damage in printed circuit boards using mobile devices and YOLO detectors. *Learning & Nonlinear Models*, 22(2):17–31.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Stevens, E., Antiga, L., and Viehmann, T. (2020). *Deep Learning with PyTorch*. Manning Publications.

Verma, R., Nagar, V., and Mahapatra, S. (2021). Introduction to supervised learning. *Data Analytics in Bioinformatics: A Machine Learning Perspective*, pages 1–34.

Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., and Ding, G. (2024). Yolov10: Real-time end-to-end object detection. *arXiv preprint arXiv:2405.14458*.

Wang, C.-Y., Liao, H.-Y. M., Wu, Y.-H., Chen, P.-Y., Hsieh, J.-W., and Yeh, I.-H. (2020). Cspnet: A new backbone that can enhance learning capability of cnn. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 390–391.

Wang, J., Dai, H., Chen, T., Liu, H., Zhang, X., Zhong, Q., and Lu, R. (2023). Toward surface defect detection in electronics manufacturing by an accurate and lightweight yolo-style object detector. *Scientific Reports*, 13(1):7062.

Xiao, Y., Tian, Z., Yu, J., Zhang, Y., Liu, S., Du, S., and Lan, X. (2020). A review of object detection based on deep learning. *Multimedia Tools and Applications*, 79:23729–23791.

Yang, Z., Yan, X., Yu, L., and Zhu, H. (2023). Laptop appearance defect detection based on improved yolov5 algorithm. In *2023 International Conference on Computer Graphics and Image Processing (CGIP)*, pages 13–18. IEEE.

Zhang, J., Li, Z., and Zhao, Y. (2023). Defect detection of laptop appearance based on improved multi-scale normalizing flows. In *2023 38th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 311–316. IEEE.

Zhu, H., Kang, Y., Zhao, Y., Yan, X., and Zhang, J. (2022). Anomaly detection for surface of laptop computer based on patchcore gan algorithm. In *2022 41st Chinese Control Conference (CCC)*, pages 5854–5858. IEEE.