Object Detection Using AutoML and YOLO: A Comparative Analysis of EasyDL, YOLOv8, and YOLOv10

Jiaxiang Chen

{34520222200859@stu.xmu.edu.cn}

Xiamen University, No. 422, Siming South Road, Xiamen, Fujian, China

Abstract. Object detection technology has been applied in multiple fields, and compared to traditional methods, using AutoML to complete object detection tasks may be more convenient and less labor-intensive. This study focuses on the performance of object detection models, comparing the AutoML platform EasyDL with the popular object detection models YOLOv8 and YOLOv10. The experiments were conducted on two datasets: mask detection and slightly larger traffic sign detection. The study compares certain performance metrics of the models and demonstrates the ease or difficulty of completing tasks. The results indicate that EasyDL provides a user-friendly and simplified workflow, allowing users without technical expertise to operate it; however, due to inherent platform limitations, its training time is relatively long. Additionally, on larger datasets, the YOLO models outperform EasyDL in terms of both accuracy and speed. Nevertheless, on smaller datasets, the performance of EasyDL is very close to that of YOLOv8 and YOLOv10. Despite the longer training time, EasyDL delivers competitive results, indicating the feasibility of AutoML in object detection tasks. This study innovatively conducts a comparative analysis of the performance dimensions of AutoML and YOLO models across different task scales, providing new insights for practitioners and non-professionals seeking efficient and accessible solutions for object detection tasks.

Keywords: AutoML, EasyDL, YOLOv8, YOLOv10, Object Detection.

1 Introduction

In recent years, with the rise of deep learning technology, the field of computer vision has seen unprecedented progress [1]. Object detection technology, as one of the most fundamental and important research topics in this field, has been widely applied in several key areas such as road condition detection in autonomous driving, facial recognition in security, cell detection and lesion image screening in the medical field, and defect detection and classification of products in industrial applications [2-5]. These scenarios and experiments have fully demonstrated the advantages of object detection technology over traditional methods in terms of convenience and accuracy.

However, despite the potential application value of rapidly developing object detection technology in many fields, its ease of use remains unsatisfactory. For individuals without relevant knowledge and experience, independently completing steps such as dataset annotation and partitioning, model selection, environment setup, coding, and model parameter adjustment and training is a daunting task. Even for experienced researchers, the repetitive tasks of feature

CONF-MLA 2024, November 21, Adana, Turkey

DOI 10.4108/eai.21-11-2024.2354621

extraction, model selection, and parameter tuning required by this technology are time-consuming and labor-intensive, with limited significance [6, 7].

To simplify this process, Automated Machine Learning (AutoML) technology has emerged. AutoML aims to reduce human intervention in machine learning tasks through automated processes, thereby lowering the technical barrier [8, 9]. Since 2014, AutoML has become a popular research direction in the field of artificial intelligence [10]. Several platforms and systems and tools, such as Auto-sklearn system,TPOT(aTree-based Pipeline Optimization Tool),Auto-Keras framework, Baidu's EasyDL and Google's AutoML platform,have been developed [11-13]. They can automate operations in data preprocessing, model selection, parameter optimization, and evaluation, helping users without relevant background knowledge efficiently build high-quality models [6].

This study focuses on the effectiveness of AutoML in the field of object detection. To identify the characteristics, advantages, and disadvantages of AutoML compared to traditional machine learning, this study will train models using the EasyDL AutoML platform and the YOLOv8 and YOLOv10 frameworks on two datasets—Face Mask Detection and Traffic Signs Detection—and compare their performance metrics.

The innovation of this study lies in its systematic evaluation of the performance of AutoML in comparison to traditional machine learning techniques within specific object detection tasks. Through comparative experiments conducted on diverse datasets, the research reveals the strengths and weaknesses of different models across varying task scales. This work addresses a gap in the existing literature concerning direct comparisons between AutoML and traditional models, providing a novel perspective on the feasibility of machine learning automation in object detection tasks. The findings of this study will offer robust theoretical foundations and practical references for future researchers and practitioners in selecting and applying AutoML platforms to generate high-quality models.

In the second section, this paper will introduce the AutoML and You Only Look Once (YOLO) algorithm frameworks and their roles in object detection. The third section will provide a detailed description of the characteristics of the datasets used, the workflows of AutoML and YOLO models, and the evaluation metrics and methods for the models. The fourth section will describe the experimental process in detail and present the results. Finally, the paper will summarize and evaluate the experiments and propose directions for future research.

2 Literature Review

In the task of object detection, comparative research between AutoML and traditional methods like YOLO has become a key focus. YOLO is a popular object detection model known for its speed, robustness, and adaptability, with the capability to achieve real-time frame rates when processing video streams. It is widely used in fields such as surface defect detection, UAV object detection, Autonomous vehicles and so on [14, 15]. Compared to YOLO, the more convenient and comprehensive AutoML systems often require more computational resources and time during the training and model optimization process, which may limit their use in applications with high real-time demands. Nonetheless, the continuous advancements in AutoML technology are gradually narrowing the gap between AutoML and YOLO in terms of performance and

practicality. Particularly in terms of automation and flexibility in model generation, AutoML has demonstrated its unique advantages.

In recent years, AutoML has become a major research direction in the field of machine learning, with its definitions and applications varying across different studies. According to relevant research, the core goal of AutoML is to reduce reliance on data scientists and domain experts, allowing users to automatically build high-performance machine learning models without extensive knowledge of machine learning or statistics [16]. Studies have also shown that AutoML significantly enhances the efficiency and flexibility of the machine learning process by automating steps such as data preprocessing, feature engineering, model selection, and hyperparameter optimization [8]. Additionally, AutoML is defined as a computational approach that automates learning configuration and optimizes model performance by incorporating task experience. These studies collectively indicate that the greatest advantage of AutoML lies in reducing human intervention and improving the efficiency of model building through automating multiple steps of the machine learning workflow.

As a typical AutoML platform, Baidu's EasyDL provides a one-stop AI development service for users without an algorithmic background or those seeking efficient development. The EasyDL platform integrates the entire process, from data annotation and cleaning to model training and deployment, significantly simplifying the traditional AI model development process. This allows enterprise users to apply AI technology to project implementation more easily without the need to master complex algorithmic details.

At the same time, the YOLO series of algorithms, as classic object detection algorithms in the field of deep learning, have been widely adopted for their efficiency and real-time capabilities. Since its initial proposal in 2015, the YOLO algorithm has undergone multiple improvements, gradually enhancing its accuracy and adaptability in object detection tasks [17]. YOLOv3 introduced the Feature Pyramid Network (FPN), which improved its ability to detect multi-scale objects, while YOLOv5 further enhanced usability and ease of deployment due to its open-source nature and widespread community support . The recently released YOLOv8 and YOLOv10 further optimized the network architecture and introduced new technologies, significantly improving object detection performance, particularly achieving higher accuracy and efficiency while reducing computational costs [18, 19].

Although the YOLO algorithm performs excellently in object detection, its model design and parameter optimization still rely heavily on expert experience. This complexity limits its use by non-expert users, and AutoML technology is gradually being introduced to the field of object detection to address this issue. Existing research has proven that AutoML can automatically complete model design, optimization, and evaluation without human intervention, achieving performance comparable to traditional methods. For example, Karimanzira et al.'s research demonstrated that an underwater object detection solution developed based on AutoML technology exhibited excellent performance on the test set, validating the effectiveness of AutoML [20]. L. Faes et al. evaluated the utility of automated deep learning software to develop medical image diagnostic classifiers by health-care professionals. They found that most automated models showed comparable discriminative performance and diagnostic properties to state-of-the-art performing deep learning algorithms [21]. J. C. O. Koh et al examined the application of AutoML for image-based plant phenotyping, finding its significant potential to

enhance plant phenotyping capabilities applicable in crop breeding and precision agriculture [22].

However, direct comparative research between AutoML and the YOLO algorithm in object detection tasks remains limited, especially in terms of performance differences across various datasets. To fill this research gap, this paper will compare the performance of the AutoML platform and YOLOv8 and YOLOv10 on multiple datasets, analyzing their respective strengths and weaknesses to provide references for future research and applications.

3 Research Methods

3.1 Selected Datasets

This study conducted comparative experiments on two different datasets to evaluate the performance of AutoML technology and the YOLO framework when dealing with datasets of varying sizes and label quantities.

The first dataset, called Face Mask Detection, is designed to identify and classify the correctness of mask-wearing. Example images are shown in Figure 1 (a). The original dataset contains 853 images across three categories (With mask; Without mask; Mask worn incorrectly), along with bounding box annotations in PASCAL VOC format. In total, there are 121 annotations for Mask worn incorrectly, 3184 annotations for With mask, and 716 annotations for Without mask [23].

Fig. 1. Examples of datasets: (a) Face Mask Detection dataset; (b) Traffic Signs Detection dataset.

The second dataset, named Traffic Signs Detection, is intended to identify various traffic signs. Compared to the previous dataset, this one has significantly more categories and images. It contains 4969 images covering multiple traffic sign categories, including Green light, Red light, Stop sign, Speed limit 10-120, and 15 other types, with a total of 5689 annotations. Example images are shown in Figure 1 (b). Although this dataset includes more categories and images, the overall background information is relatively simple, with fewer distractions, making training potentially easier [24]. Additionally, data augmentation techniques were applied to this dataset.

It is worth noting that after importing the data into the EasyDL platform, the number of valid images read was fewer than the original dataset claimed. The platform's summary showed that 848 and 4735 images were read, respectively, which is less than the originally reported 853 and 4969 images. This discrepancy might be due to the platform not importing images with empty annotation files, which were present in the original datasets.

3.2 AutoML Model Workflow

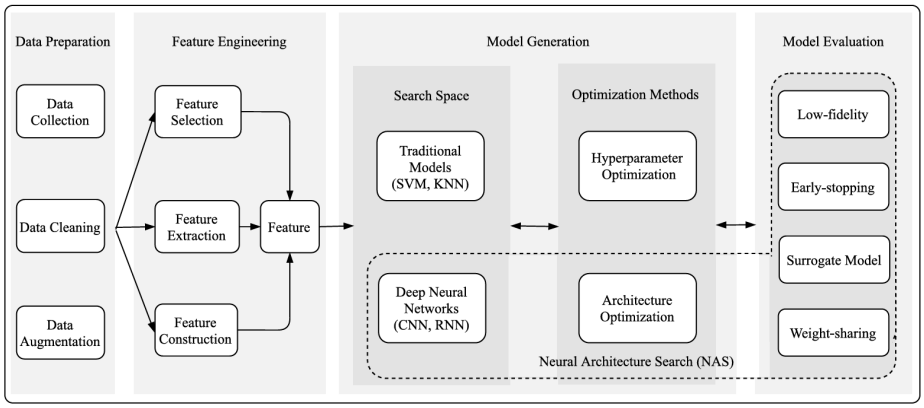

Fig. 2. AutoML Workflow [6].

As shown in Figure 2, once the raw data is imported into the AutoML model, the system can execute semi-automated data labeling based on the user's requirements. If the data is already labeled, the model will automatically perform tasks such as data cleaning, handling missing values and outliers, normalizing data, and splitting the dataset. These preprocessing steps ensure that the data is of high quality and consistent, enabling it to be effectively used for model training and testing, thereby improving the overall model performance.

Next, the system enters the feature engineering phase, with the goal of maximizing the extraction of key features from the data for the model to use. AutoML automatically handles tasks such as feature selection, feature extraction, and feature construction. It combines advanced methods like meta-learning to optimize the feature engineering process, enhancing model performance and reducing the need for manual intervention [25].

After completing feature engineering, AutoML moves on to the model selection and hyperparameter optimization phase. The system selects the optimal model through two main approaches: traditional model selection and Neural Architecture Search (NAS). In the traditional approach, the AutoML system automatically evaluates various algorithms (such as SVM, KNN, decision trees, etc.) and adjusts hyperparameters to select the algorithm best suited for the specific task and dataset. NAS, on the other hand, focuses on automatically designing neural network architectures to optimize their generalization capability and computational efficiency [26]. Once the model architecture is selected, AutoML utilizes techniques like Bayesian optimization and random search to automatically fine-tune and optimize the hyperparameters [27].

Finally, in the model evaluation phase, methods such as cross-validation are used to assess the model's performance, generating multiple performance metrics. These metrics enable users to choose the optimal model. Through this automated process, AutoML greatly simplifies the complex workflows of machine learning while improving efficiency and model performance.

3.3 YOLO Model Workflow

The YOLO model is an end-to-end object detection method that converts the object detection task into a regression problem, enabling the location and classification of objects to be completed in a single forward pass. The YOLO model workflow consists of the following key steps:

Input Image Segmentation. First, the YOLO model divides the input image into multiple grids. Each grid is responsible for detecting whether an object exists within it and predicting its bounding box and class.

Bounding Box Prediction. For each grid, the YOLO model predicts multiple bounding boxes. Each bounding box contains the following information: center coordinates (x, y), width, height, and a confidence score indicating the likelihood that the box contains an object.

Class Prediction. Simultaneously, the YOLO model predicts the probability distribution of object classes for each bounding box. These class scores are combined with the confidence scores to determine which category the detected object belongs to.

Non-Maximum Suppression (NMS). To avoid duplicate detections, YOLO uses the non-maximum suppression technique, removing redundant bounding boxes and retaining only the detections with the highest confidence scores.

Output Results. Finally, the YOLO model outputs the bounding boxes containing objects along with their corresponding class labels, which are annotated on the original image. The output includes both the location and classification information of each object, facilitating further processing and analysis.

Through these steps, the YOLO model achieves fast and efficient object detection, making it suitable for real-time applications such as autonomous driving and video surveillance [28].

3.4 Model Evaluation

Since the EasyDL platform provides only partial model evaluation metrics, this study primarily utilizes the parameters of mAP, precision, and recall, in conjunction with training time, to assess model performance.

Precision and Recall. Precision quantifies the proportion of true positives among all positive predictions, evaluating the model's ability to avoid false positives. In contrast, recall calculates the proportion of true positive predictions among all actual positive instances, measuring the model's capability to detect all instances of a certain class. This can be expressed as:

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

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

The relationships between states and decisions can be classified into four possibilities: TP (True Positive), FN (False Negative), FP (False Positive), and TN (True Negative), as shown in Table 1.

Table 1. Relationship Between State and Decision.

Decision	State	
	Positive	Negative
Positive	TP	FP
Negative	FN	TN

F1-Score: The F1-score is a metric that considers both precision and recall, allowing for the evaluation of the model's accuracy and completeness in detecting all targets. It is calculated as follows:

$$\mathrm{F1} = \frac{2\times \mathrm{Precision}\times \mathrm{Recall}}{\mathrm{Precision}+\mathrm{Recall}} \tag{3}$$

mAP (Mean Average Precision). mAP is a crucial metric for assessing the overall performance of object detection models. For each target class, a P-R curve can be computed based on its precision and recall at different IoU thresholds, with the area under the curve representing the average precision (AP) for that class. Ultimately, mAP is the mean of all class APs. In this study, we used mAP50, which is the mAP value calculated at an IoU (a measure of the overlap between predicted and actual bounding boxes) threshold of 0.50, primarily to gauge the model's overall accuracy in detection tasks.

During the final model evaluation, we assessed model performance using the precision and recall data at the highest F1-score, along with mAP50, a commonly used metric. Additionally, we compared the training times of each model to evaluate their training efficiency.

4 Experiments

4.1 Experimental Environment

Table 2. Detailed Experimental Environment.

Category	Configuration
Processor	12th Gen Intel(R) Core(TM) i7-12700H 2.30 GHz
Graphics Card	NVIDIA GeForce RTX 3050 Ti Laptop GPU
CUDA Version	12.3
YOLOv10 Version and Environment	Python3.9
YOLOv8 Version and Environment	Python 3.7
AutoML Version	EasyDL 2022.12

As shown in Table 2, the experiments were conducted on a computer equipped with a 12th Gen Intel Core i7-12700H processor, an NVIDIA GeForce RTX 3050 Ti laptop GPU, and a CUDA version of 12.3. The YOLOv10 model was run in a Python 3.9 environment, while the YOLOv8 model utilized a Python 3.7 environment. The AutoML platform used for the experiments was Baidu's EasyDL, version 2022.12.

4.2 Experimental Steps

Dataset Preprocessing. In the operation of the YOLOv8 and YOLOv10 frameworks, the dataset must contain the segmented images and the annotation information in YOLO format. The EasyDL platform does not require dataset segmentation but mandates the use of annotations in PASCAL VOC format. Therefore, the dataset needs appropriate preprocessing before application. First, we divided the two datasets into training and testing/validation sets in a 7:3 ratio for the YOLO framework.

For the Face Mask Detection dataset, since it already includes annotations in PASCAL VOC format, it can be directly input into the EasyDL platform for training. Additionally, to ensure compatibility with the YOLOv8 and YOLOv10 frameworks, we need to generate YOLO-format annotation files for this dataset.

For the Traffic Signs Detection dataset, which comes with annotations in YOLO format, it can be directly applied to the YOLOv8 and YOLOv10 frameworks after segmentation. We also need to generate PASCAL VOC format annotations for training on the EasyDL platform.

EasyDL Platform Operations. Dataset Preparation and Import. First, a data repository for rectangular box annotations needs to be created on the EasyDL platform, and the dataset should be imported according to the file format required by the platform. The import method selected is "Local Import of Compressed Package," using the XML (VOC) annotation format. Once the import is complete, as shown in Figure 4, the platform allows users to view the annotation status of the dataset and provides features for manual annotation and platform-assisted automatic annotation.

Fig. 3. View of Imported Dataset Annotations on EasyDL Platform.

Model Training. Using the platform's default data preparation configuration, the training configuration adopts the "Regular Training" method, with the training environment set to "GPU P4," while the rest of the settings remain at their default values. Training is then initiated.

Result Evaluation. Upon completion of training, a complete evaluation report is automatically generated, including metrics such as mAP, precision, and recall, which users can view directly on the platform.

YOLO Framework Operations. Dataset Preparation and Import. Place the segmented datasets (training, testing, and validation sets) in the designated directory of the YOLO framework, and create an images folder and corresponding labels folder as required by the framework. Next, write the train.py and data.yaml files, specifying the pretrained model and training parameters in the train.py file, such as epochs=100, batch=4, while keeping other parameters at their default settings. In the data.yaml file, specify the dataset path, the number of label categories, and their names.

Model Training. In the YOLOv8 environment, use the yolov8s.pt pretrained model for training; in the YOLOv10 environment, use the yolov10s.pt pretrained model. Both pretrained models are of comparable scale and exhibit high speed. The purpose of selecting a pretrained model is to simulate a scenario for small-scale personal use, thereby reducing training difficulty and resource consumption. Once preparations are complete, run the train.py file to start training.

Result Evaluation. Upon completion of training, a brief evaluation result is automatically displayed in the terminal, and the running time is recorded. Complete evaluation results can be obtained in the corresponding result file. The model.val() function can also be run to obtain evaluation results.

4.3 Experimental Results

Table 3. Comparison of Some Performance Metrics.

	P	R	mAP	Time(h)	F1-score Max
Traffic Signs Detection Dataset					
EasyDL	0.898	0.752	0.815	4	0.82
YOLOv8	0.953	0.944	0.91	3.564	0.95
YOLOv10	0.951	0.942	0.97	3.539	0.95
Face Mask Detection Dataset					
EasyDL	0.897	0.888	0.851	3	0.89
YOLOv8	0.928	0.779	0.851	0.460	0.85
YOLOv10	0.884	0.765	0.847	0.777	0.82

From Table 3, it can be observed that the performance of YOLOv8 and YOLOv10 on both datasets is similar, particularly on the Traffic Signs Detection Dataset, where YOLOv10 achieved a mean Average Precision (mAP) of 0.970, surpassing YOLOv8's 0.910. Both models are nearly identical in precision (P), recall (R), training time, and F1-score, and they generally outperform EasyDL's performance.

For the smaller Face Mask Detection Dataset, EasyDL excelled in mAP (0.851) and recall (0.888), demonstrating good performance in precision (0.897), and its performance is generally superior to the YOLO models. However, its training time of 3 hours is significantly longer than that of YOLOv8 (0.460 hours) and YOLOv10 (0.777 hours). This may be attributed to the queuing for computational resources required by the AutoML platform's training process, and the automation process could also have increased the overall training time. Additionally, the training time for YOLOv10 is slightly longer than that for YOLOv8 (0.777 hours compared to 0.460 hours), with the gap in training time narrowing for the Traffic Signs Detection Dataset. YOLOv8 demonstrates better efficiency, which may be related to differences in the features of its pre-trained models.

Overall, YOLOv8 exhibited good training efficiency and performance on both datasets, making it especially suitable for time-sensitive tasks. EasyDL shows excellent performance in certain metrics on smaller datasets, but its longer training time may limit its application in scenarios where real-time performance is crucial.

5 Conclusion

During the training of object detection models, the EasyDL platform demonstrated significant convenience, allowing users without relevant technical backgrounds to use the platform easily. Despite employing the platform's default low-configuration model and experiencing longer training times due to inherent limitations in platform management and automated machine learning algorithms, EasyDL can still match the performance of YOLOv8 and YOLOv10 models in certain tasks. However, on larger datasets (such as the traffic sign detection task), the performance of YOLO models is clearly superior to that of the EasyDL platform. Overall, the model performance achieved through EasyDL meets a high standard, indicating that applying automated machine learning in object detection tasks is feasible, particularly for users outside the computer science field.

This study systematically evaluates the performance of the automated machine learning platform compared to the classical object detection algorithms across different object detection tasks. The results demonstrate that the automated machine learning platform has advantages in ease of operation, requiring no specialized technical knowledge from users, while YOLO models necessitate manual environment configuration, coding, and parameter adjustments. On smaller datasets (such as mask detection tasks), the performance of automated machine learning and YOLO models is comparable; however, in larger datasets like traffic sign detection, YOLO models excel.

The main contributions and innovations of this paper lies in the systematic assessment of the performance of automated machine learning platforms against classical object detection algorithms in various tasks, providing a reference for non-technical users regarding automated machine learning platforms. The research findings indicate a substantial potential for automated machine learning platforms in object detection tasks, especially in simplifying operational processes and lowering technical barriers. Additionally, through in-depth comparative analysis, this study offers practical recommendations regarding the use of AutoML or traditional models in specific application scenarios, paving the way for the adoption of object detection technologies by non-expert users and small businesses. This perspective is groundbreaking in

the current research landscape, assisting users in making more informed choices between efficiency and ease of use.

Future research will further explore the following directions: training models using non-default settings of the platform, testing other automated machine learning platforms, and conducting experiments with larger-scale datasets to validate and extend the conclusions drawn in this paper.

References

[1] Y. Z. Xiao *et al.*, "A review of object detection based on deep learning, " *Multimedia Tools and Applications,* vol. 79, no. 33-34, pp. 23729-23791, Sep 2020.

[2] Z. H. Li *et al.*, "Development and challenges of object detection: A survey, " *Neurocomputing,* vol. 598, Sep 2024, Art. no. 128102.

[3] A. Vijayakumar and S. Vairavasundaram, "YOLO-based Object Detection Models: A Review and its Applications, " *Multimedia Tools and Applications,* 2024 Mar 2024.

[4] A. Raghunandan, Mohana, P. Raghav, H. V. R. Aradhya, and Ieee, "Object Detection Algorithms for Video Surveillance Applications, " in *7th IEEE International Conference on Communication and Signal Processing (IEEE ICCSP)*, Adhiparasakthi Engn Coll, Melmaruvathur, INDIA, 2018, pp. 563-568, 2018.

[5] B. Hu, Y. Liu, P. Z. Chu, M. L. Tong, and Q. J. Kong, "Small Object Detection via Pixel Level Balancing With Applications to Blood Cell Detection, " *Frontiers in Physiology,* vol. 13, Jun 2022, Art. no. 911297.

[6] X. He, K. Y. Zhao, and X. W. Chu, "AutoML: A survey of the state-of-the-art, " *Knowledge-Based Systems,* vol. 212, Jan 2021, Art. no. 106622.

[7] R. Elshawi and S. Sakr, "Automated Machine Learning: State-of-The-Art and Open Challenges, " in *14th International Conference on Research Challenges in Information Science (RCIS)*, Limassol, CYPRUS, 2020, vol. 385, pp. 627-629, 2020.

[8] A. Truong *et al.*, "Towards Automated Machine Learning: Evaluation and Comparison of AutoML Approaches and Tools, " in *31st IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, Portland, OR, 2019, pp. 1471-1479, 2019.

[9] S. K. Karmaker, M. M. Hassan, M. J. Smith, L. Xu, C. X. Zhai, and K. Veeramachaneni, "AutoML to Date and Beyond: Challenges and Opportunities, " *Acm Computing Surveys,* vol. 54, no. 8, Nov 2021, Art. no. 175.

[10] R. Barbudo, S. Ventura, and J. R. Romero, "Eight years of AutoML: categorisation, review and trends, " *Knowledge and Information Systems,* vol. 65, no. 12, pp. 5097-5149, Dec 2023.

[11] R. S. Olson, N. Bartley, R. J. Urbanowicz, J. H. Moore, and Acm, "Evaluation of a Tree-based Pipeline Optimization Tool for Automating Data Science, " in *Genetic and Evolutionary Computation Conference (GECCO)*, Denver, CO, 2016, pp. 485-492, 2016.

[12] H. F. Jin, Q. Q. Song, X. Hu, and M. Assoc Comp, "Auto-Keras: An Efficient Neural Architecture Search System, " in *25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, Anchorage, AK, 2019, pp. 1946-1956, 2019.

[13] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter, "Efficient and Robust Automated Machine Learning, " in *29th Annual Conference on Neural Information Processing Systems (NIPS)*, Montreal, CANADA, 2015, vol. 28, 2015.

[14] M. Hussain, "YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection, " *Machines,* vol. 11, no. 7, Jul 2023, Art. no. 677.

[15] E. Soylu and T. Soylu, "A performance comparison of YOLOv8 models for traffic sign detection in the Robotaxi-full scale autonomous vehicle competition, " *Multimedia Tools and Applications,* 2023 Aug 2023.

[16] M. A. Zöller and M. F. Huber, "Benchmark and Survey of Automated Machine Learning Frameworks, " *Journal of Artificial Intelligence Research,* vol. 70, pp. 409-472, 2021.

[17] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, and Ieee, "You Only Look Once: Unified, Real-Time Object Detection, " in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, 2016, pp. 779-788, 2016.

[18] G. Jocher, A. Chaurasia, and J. Qiu, *Ultralytics YOLOv8*.

[19] W. Ao, "YOLOv10: Real-Time End-to-End Object Detection, " *arXiv preprint arXiv:2405.14458,* 2024.

[20] D. Karimanzira, H. Renkewitz, D. Shea, and J. Albiez, "Object Detection in Sonar Images, " *Electronics,* vol. 9, no. 7, Jul 2020, Art. no. 1180.

[21] L. Faes *et al.*, "Automated deep learning design for medical image classification by health-care professionals with no coding experience: a feasibility study, " *Lancet Digital Health,* vol. 1, no. 5, pp. E232-E242, Sep 2019.

[22] J. C. O. Koh, G. Spangenberg, and S. Kant, "Automated Machine Learning for High-Throughput Image-Based Plant Phenotyping, " *Remote Sensing,* vol. 13, no. 5, Mar 2021, Art. no. 858.

[23] "Mask Dataset, " ed. https://www.kaggle.com/datasets/andrewmvd/face-mask-detection/data.

[24] T. S. Detection, "Traffic Signs Detection, " ed. https://www.kaggle.com/datasets/pkdarabi/cardetection/data.

[25] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks, " in *34th International Conference on Machine Learning*, Sydney, AUSTRALIA, 2017, vol. 70, 2017.

[26] Y. Jaafra, J. L. Laurent, A. Deruyver, and M. S. Naceur, "Reinforcement learning for neural architecture search: A review, " *Image and Vision Computing,* vol. 89, pp. 57-66, Sep 2019.

[27] B. Bischl *et al.*, "Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges, " *Wiley Interdisciplinary Reviews-Data Mining and Knowledge Discovery,* vol. 13, no. 2, Mar 2023.

[28] G. Yasmine, G. Maha, M. Hicham, and Ieee, "Overview of single-stage object detection models: from Yolov1 to Yolov7, " in *19th IEEE International Wireless Communications and Mobile Computing (IEEE IWCMC)*, Marrakesh, MOROCCO, 2023, pp. 1579-1584, 2023.