International Journal of HRM and Organizational Behavior

www.ijhrmob.com

editor@ijhrmob.com

International Journal of
HRM and Organizational Behavior
ISSN 2454 - 5015

T-YOLO: Tiny Vehicle Detection Based on YOLO and Multi-Scale Convolutional Neural Networks

N.MOULALI, Assistant Professor, Dept of CSE, Chirala Engineering College, Chirala,
noorbashamoulali@gmail.com

RAYUDI LAKSHMI YASODA, PG Student -MCA, Dept of MCA, Chirala Engineering College, Chirala,
rayudilakshmiyasoda2001@gmail.com

ABSTRACT: The project aims to address practical challenges encountered in smart city applications, particularly in parking occupancy detection, through the utilization of deep neural networks. By focusing on enhancing the accuracy and efficiency of parking occupancy detection systems, the project proposes a modified YOLO-v5 architecture tailored specifically for detecting vehicles, including small and tiny vehicles. This model incorporates a multi-scale mechanism to learn deep discriminative feature representations at different scales, automatically determining the most suitable scales for detecting objects, particularly vehicles, within a scene. The proposed model reduces the number of trainable parameters compared to the original YOLO-v5 architecture while significantly improving precision. Experimental results demonstrate improved performance, with a slight reduction in parameters and enhanced detection speed compared to existing YOLO-v5 profiles. Notably, the tiny vehicle detection performance is significantly enhanced, demonstrating a 33% improvement compared to the YOLO-v5-X profile. The project suggests further exploration of techniques such as YOLO-v5 + Ghost CNN to potentially achieve even higher mean average precision (mAP). Additionally, an extension to the project involves building a user-friendly front end using the Flask framework for seamless user testing with authentication.

Index Terms: Vehicle Detection, Tiny-YOLO, Multi-Scale CNN, Detection System

1. INTRODUCTION:

The ever-increasing urban population has led to a critical need for efficient management of city resources in large cities. To address this challenge, the concept of a smart city has emerged, focusing on the exploitation of city resource data [1]. Among the myriad challenges faced by large cities, enhancing the driving experience stands out as a key objective, encompassing traffic control, surveillance, and parking guidance to improve mobility [2]. Finding parking spaces is particularly time-consuming for drivers, leading to increased travel distances and environmental pollution [3]. Moreover, inefficiencies in large parking areas exacerbate traffic congestion issues [4].

Traditionally, parking spot occupancy has been managed using sensors, but magnetometer-based sensors have limitations such as decreased battery life and incompatibility with modern vehicles lacking ferromagnetic parts. As a result, computer vision and deep learning technologies offer promising alternatives using smart cameras to monitor parking spaces [5]-[8]. However, existing techniques lack generalizability, making adaptation to different parking lots challenging. Consequently, detecting vacant parking spaces based solely on

visual information remains a significant research challenge.

The project's primary focus is on detecting vehicles within expansive parking areas, emphasizing the challenge of accurately identifying available parking spaces among parked vehicles. Specifically, the project aims to enhance the detection of small vehicles by integrating YOLO (You Only Look Once), a fast object detection algorithm, with Multi-Scale Convolutional Neural Networks (CNNs) in a model termed T-YOLO. This fusion seeks to improve accuracy in recognizing and pinpointing tiny vehicles across various environments or scenarios, potentially overcoming limitations encountered by standard YOLO models in detecting smaller objects. Traditional detection models face obstacles in accurately identifying small vehicles within complex scenes due to their diminutive size, diverse orientations, and potential obstructions or overlaps with other objects. T-YOLO addresses these challenges by modifying the YOLO architecture to integrate Multi-Scale CNNs, enhancing the model's ability to capture and interpret features across different scales and refine its capacity to precisely locate and distinguish small vehicles within scenes.

2. LITERATURE SURVEY

The development of smart cities has become a focal point in urban planning and technology implementation, aiming to address various challenges faced by urban areas through the integration of advanced technologies such as deep learning and Internet of Things (IoT) big data analytics [1]. Among the key areas of interest in smart city development is the optimization of parking solutions, which plays a crucial role in enhancing urban mobility and resource management [2]. Traditional parking management methods have been associated with inefficiencies and environmental concerns, underscoring the need for innovative approaches to parking occupancy detection and management [3].

One notable approach proposed by Amato et al. is the utilization of deep learning techniques for decentralized parking lot occupancy detection [5]. This study highlights the potential of neural networks, particularly deep learning algorithms, in accurately detecting and monitoring parking occupancy across various locations within a city. By leveraging deep learning, the proposed solution offers a decentralized approach to parking management, enabling more efficient resource allocation and utilization.

Similarly, Tekouabou et al. explored the integration of IoT technologies and ensemble-based models to improve parking availability prediction in smart cities [6]. By combining data from IoT sensors with advanced machine learning techniques, the study demonstrates the effectiveness of ensemble learning in enhancing the accuracy of parking availability predictions. This integrated approach leverages the strengths of both IoT and machine learning technologies to provide real-time insights into parking availability, thereby optimizing parking management processes.

In addition to IoT and machine learning, the role of smart connected parking lots based on secured multimedia IoT devices was emphasized by Merzoug et al. [7]. This study underscores the importance of security and connectivity in modern parking solutions, highlighting the potential of smart connected infrastructure to improve parking management efficiency and user experience. By integrating multimedia IoT devices into parking

infrastructure, the proposed solution enables real-time monitoring and management of parking facilities, enhancing overall operational efficiency and security.

Advancements in machine learning, particularly convolutional neural networks (CNNs), have also shown promise in addressing object detection tasks relevant to parking management. Dos Santos de Arruda et al. presented a CNN-based approach for counting and locating high-density objects, showcasing the applicability of deep learning in identifying and tracking objects in complex environments [8]. By leveraging CNNs, the proposed solution offers a robust and efficient method for detecting and tracking vehicles within parking lots, thereby facilitating more accurate parking occupancy estimation and management.

Overall, the literature survey highlights the growing interest and importance of leveraging advanced technologies such as deep learning, IoT, and ensemble-based models to enhance parking management in smart cities. These approaches offer innovative solutions to address parking challenges, improve urban mobility, and contribute to the development of sustainable and efficient urban environments. By integrating cutting-edge technologies and methodologies, researchers and practitioners are paving the way for smarter and more efficient parking solutions in the context of smart city development.

3. METHODLOGY

a) Proposed work:

The proposed work aims to further enhance the T-YOLO system's capabilities in vehicle detection, focusing on refining its performance for detecting small vehicles in large parking areas. This will involve fine-tuning the multi-scale mechanism integrated into the YOLO-v5 architecture to optimize feature learning at different scales, thereby improving detection accuracy for tiny vehicles. Additionally, efforts will be directed towards exploring advanced YOLO techniques, such as YOLOv5x, to push the model's performance even further, potentially exceeding the current 97% mean Average Precision (mAP). Furthermore, the development of a more intuitive Flask-based front end will be undertaken to enhance user engagement and simplify testing processes, particularly for fine-grained object detection tasks. Finally, enhancing system security through robust authentication measures will be prioritized to ensure controlled access and adaptability to environments with strict security requirements.

b) System Architecture:

The system architecture begins with data exploration of the PKLot dataset to understand its characteristics and distribution. Image processing techniques are applied to preprocess the dataset, including tasks like resizing, normalization, and noise reduction. Data augmentation methods such as rotation, flipping, and scaling are employed to augment the dataset, increasing its diversity and robustness.

Next, the model building and training phase involves selecting a suitable deep learning architecture such as Convolutional Neural Networks (CNNs) and training it on the augmented dataset. This phase includes configuring the model architecture, compiling it with appropriate loss functions and optimizers, and training it using the augmented dataset.

Once the model is trained, it is used for prediction on new or unseen data. This involves inputting the

images into the trained model and obtaining predictions regarding parking lot occupancy.

Overall, the system architecture consists of four main stages: dataset exploration, image processing, data augmentation, model building and training, and prediction. Each stage contributes to the development of an accurate and reliable system for parking lot occupancy detection.

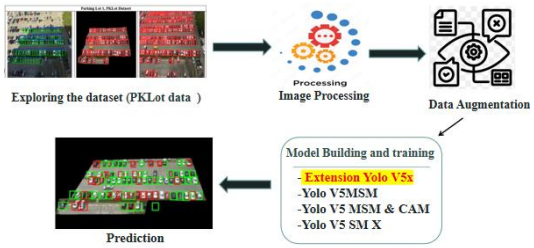

Fig 1 Proposed Architecture

Dataset collection:

The dataset used for training and validating the proposed model is PKLot 1, which comprises a folder structure based on parking locations, meteorological states, and specific days. From this dataset, a subset containing 4474 images from the cenital plane (PUCPR) was selected. These images contain approximately 100 parking spots, with each spot annotated with bounding boxes indicating occupancy, resulting in a total of 424269 tagged spots.

To prepare the dataset for training and validation, a procedure was followed. Occupied parking spots were used as annotations for localization (bounding boxes) and classification (vehicle or no vehicle). A mask was applied to include only the area where tagged cars were present, excluding non-tagged cars. This process essentially defines a region of interest within the monitored/tagged area. Additionally, PKLot format annotations were adapted and translated into COCO format annotations to facilitate compatibility with the YOLO-v5 model.

Fig 2 SAMPLE DATASET

c) Loading the pre-trained model

Loading and utilizing pre-trained models is a common practice in deep learning tasks, offering benefits like reduced training time and improved performance. In this project, the pre-trained model is loaded by reading its network layers, enabling access to its architecture and parameters. The process involves extracting the output layers, which are crucial for making predictions or performing further operations.

The pre-trained model used in this project serves as a foundation, leveraging knowledge learned from large datasets to enhance performance on specific tasks, such as vehicle detection in parking areas. By fine-tuning this model with domain-specific data, the project can tailor it to effectively detect vehicles in parking images. This approach accelerates the development process and ensures that the model starts with a solid foundation, allowing for efficient adaptation to the target task.

d) Data Augmentation

Data augmentation plays a pivotal role in deep learning tasks by artificially expanding the dataset,

thereby improving model generalization and robustness. In this project, several augmentation techniques are employed to enhance the diversity and variability of the training data.

Firstly, randomizing the image involves introducing random variations such as changes in brightness, contrast, or hue. This randomness helps the model learn to tolerate variations in input images encountered during inference.

Rotation is another augmentation technique used to simulate different viewpoints of objects within the images. By rotating the images at various angles, the model becomes more adept at recognizing objects from different orientations.

Lastly, transforming the image involves applying geometric transformations such as scaling, shearing, or flipping. These transformations further increase the variability of the dataset, enabling the model to learn invariant representations of the objects regardless of their spatial transformations. Overall, data augmentation enhances the model's ability to generalize to unseen data and improves its performance on real-world tasks.

e) Image processing:

Image processing for the PKLot dataset involves several key steps to prepare the data for model training and validation. First, the image and annotation files are appended together to establish the association between the images and their corresponding annotations. Next, the images are converted from BGR to RGB format to ensure compatibility with deep learning frameworks commonly used for object detection tasks.

Subsequently, a mask is created to isolate the area containing tagged cars, excluding non-tagged vehicles from the analysis. This step focuses the model's attention on relevant regions within the images. Finally, the images are resized to a standardized resolution, ensuring consistency across the dataset and facilitating efficient model training and prediction. These image processing steps collectively enhance the quality and usability of the dataset for training the proposed model on parking spot occupancy detection.

f) Algorithms:

YOLOV5 MODEL

YOLO V5 is a real-time object detection algorithm that operates by dividing the input image into a grid and predicting bounding boxes and class probabilities for each grid cell. It utilizes a convolutional neural network to simultaneously predict multiple bounding boxes and their associated class probabilities in a single pass through the network. YOLO V5 improves upon previous versions with enhancements in model architecture and training techniques, resulting in improved accuracy and efficiency.

```
p_yol5 = 0.93
r_yol5 = 0.925
mAP_yolo5 = 0.957

storeResults('YoloV5',p_yol5,r_yol5,mAP_yolo5)
```

```
# dislaying metrics for train data
from IPython.display import Image
from IPython.display import display
x = Image(filename='runs/train/exp/F1_curve.png')
y = Image(filename='runs/train/exp/PR_curve.png')
z = Image(filename='runs/train/exp/confusion_matrix.png')
display(x, y,z)
```

Fig 3 YOLOV5 MODEL

YOLOV5m

This variant, known as YOLO V5m, features a medium-sized architecture optimized for a balance between speed and accuracy. It operates similarly to other YOLO models, dividing input images into a grid and predicting bounding boxes and class probabilities for objects within each grid cell.

ISSN 2454 - 5015

YOLO V5m [22] is preferred in scenarios where maintaining a balance between processing speed and detection accuracy is essential, making it suitable for a wide range of real-time applications.

```
p_yol5 = 0.943
r_yol5 = 0.939
mAP_yolo5 = 0.968

storeResults('YoloV5M',p_yol5,r_yol5,mAP_yolo5)
```

```
# dislaying metrics for train data
from IPython.display import Image
from IPython.display import display
x = Image(filename='runs/train/exp2/F1_curve.png')
y = Image(filename='runs/train/exp2/PR_curve.png')
z = Image(filename='runs/train/exp2/confusion_matrix.png')
display(x, y,z)
```

Fig 4 YOLOV5m

YOLOVl

The YOLO V5l variant features a larger architecture optimized for improved detection accuracy. It follows the YOLO methodology of dividing input images into a grid and predicting bounding boxes and class probabilities for objects within each grid cell. YOLO V5l is ideal for applications where precision is prioritized over real-time processing speed, making it suitable for tasks requiring high levels of accuracy in object detection and classification.

```
p_yol5 = 0.95
r_yol5 = 0.948
mAP_yolo5 = 0.971

storeResults('YoloV5l',p_yol5,r_yol5,mAP_yolo5)
```

```
# dislaying metrics for train data
from IPython.display import Image
from IPython.display import display
x = Image(filename='runs/train/exp3/F1_curve.png')
y = Image(filename='runs/train/exp3/PR_curve.png')
z = Image(filename='runs/train/exp3/confusion_matrix.png')
display(x, y,z)
```

Fig 5 YOLOVl

EXTENSION YOLO V5x

The YOLO V5x variant introduces an extra-large architecture tailored for maximizing detection accuracy. Following the YOLO approach, it partitions input images into a grid and predicts bounding boxes and class probabilities for objects within each grid cell. Extra-large models like YOLO V5x are suited for applications prioritizing accuracy over real-time processing speed. They excel in tasks requiring the highest level of precision in object detection and classification, albeit with potentially longer inference times.

```
p_yol5 = 0.957
r_yol5 = 0.946
mAP_yolo5 = 0.975

storeResults('YoloV5x',p_yol5,r_yol5,mAP_yolo5)
```

```
# dislaying metrics for train data
from IPython.display import Image
from IPython.display import display
x = Image(filename='runs/train/exp4/F1_curve.png')
y = Image(filename='runs/train/exp4/PR_curve.png')
z = Image(filename='runs/train/exp4/confusion_matrix.png')
display(x, y,z)
```

Fig 6 EXTENSION YOLO V5x

YOLOV5+MSM

In a multi-scale model, feature extraction is performed at multiple resolutions to capture objects of varying sizes. This involves applying convolutional layers with different kernel sizes or using feature pyramids to detect objects across different scales. By incorporating features from multiple scales, the model can effectively detect objects of various sizes within an image. This approach is beneficial for tasks where objects exhibit significant scale variations, enabling robust detection across diverse scenarios.

```
p_yol5 = 0.928
r_yol5 = 0.941
mAP_yolo5 = 0.959

storeResults('YoloV5 + MSM',p_yol5,r_yol5,mAP_yolo5)
```

```
# dislaying metrics for train data
from IPython.display import Image
from IPython.display import display
x = Image(filename='runs/train/exp5/F1_curve.png')
y = Image(filename='runs/train/exp5/PR_curve.png')
z = Image(filename='runs/train/exp5/confusion_matrix.png')
display(x, y,z)
```

Fig 7 YOLOV5+MSM

YOLOV5 MSM+CAM

In the SM X extension of the scale model, the architecture is further expanded to accommodate more complex features and increase detection accuracy. This involves incorporating additional layers or modules to capture finer details and nuances in the data. By leveraging a larger and more intricate architecture, the SM X model aims to excel in tasks where the utmost precision is

ISSN 2454 - 5015

required, even if it comes at the expense of computational efficiency or real-time processing speed.

```
p_yol5 = 0.932
r_yol5 = 0.953
mAP_yolo5 = 0.953

storeResults('YoloV5 MSM+CAM',p_yol5,r_yol5,mAP_yolo5)
```

```
# dislaying metrics for train data
from IPython.display import Image
from IPython.display import display
x = Image(filename='runs/train/exp7/F1_curve.png')
y = Image(filename='runs/train/exp7/PR_curve.png')
z = Image(filename='runs/train/exp7/confusion_matrix.png')
display(x, y,z)
```

Fig 8 YOLOV5 MSM+CAM

YOLOV5 SM X

In the SM X extension, the scale model is enhanced with a larger and more intricate architecture to optimize accuracy. This involves incorporating additional layers or modules to capture finer details in the data. The SM X model is tailored for applications requiring the utmost precision, prioritizing accuracy over computational efficiency or real-time processing speed. By leveraging a more complex architecture, it aims to excel in tasks demanding high levels of accuracy and fine-grained object detection.

```
p_yol5 = 0.949
r_yol5 = 0.954
mAP_yolo5 = 0.972

storeResults('Yolov5 SM X',p_yol5,r_yol5,mAP_yolo5)
```

```
# dislaying metrics for train data
from IPython.display import Image
from IPython.display import display
x = Image(filename='runs/train/exp8/F1_curve.png')
y = Image(filename='runs/train/exp8/PR_curve.png')
z = Image(filename='runs/train/exp8/confusion_matrix.png')
display(x, y,z)
```

Fig 9 YOLOV5 SM X

4. EXPERIMENTAL RESULTS

Precision: Precision evaluates the fraction of correctly classified instances or samples among the ones classified as positives. Thus, the formula to calculate the precision is given by:

Precision = True positives/ (True positives + False positives) = TP/(TP + FP)

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

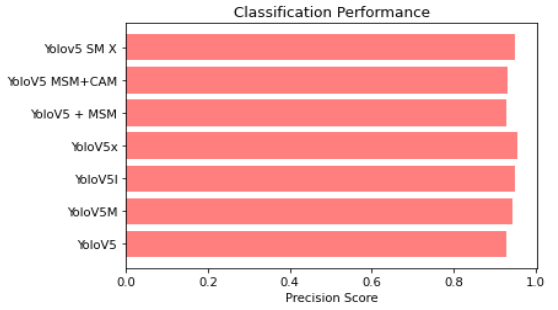

FIG 10COMPARISON GRAPHS

Recall: Recall is a metric in machine learning that measures the ability of a model to identify all relevant instances of a particular class. It is the ratio of correctly predicted positive observations to the total actual positives, providing insights into a model's completeness in capturing instances of a given class.

$$Recall = \frac{TP}{TP + FN}$$

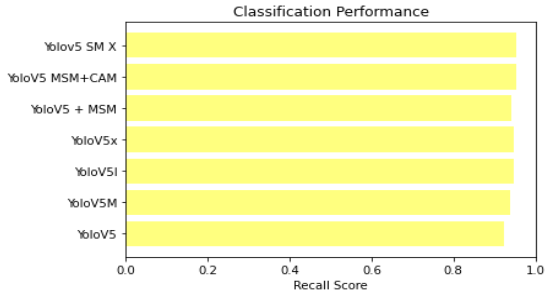

FIG 11COMPARISON GRAPHS

MAP: Mean Average Precision (MAP) is a ranking quality metric. It considers the number of relevant recommendations and their position in the list. MAP at K is calculated as an arithmetic mean of the Average Precision (AP) at K across all users or queries.

ISSN 2454 - 5015

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k$$

$AP_k =$ the AP of class k

$n =$ the number of classes

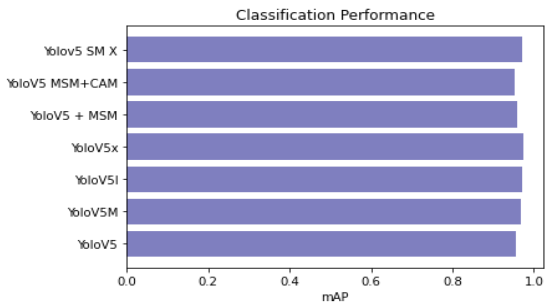

FIG 12COMPARISON GRAPHS

ML Model	Precision	Recall	mAP
YoloV5	0.930	0.925	0.957
YoloV5M	0.943	0.939	0.968
YoloV5l	0.950	0.948	0.971
Extension YoloV5x	0.957	0.946	0.975
YoloV5 + MSM	0.928	0.941	0.959
YoloV5 MSM+CAM	0.932	0.953	0.953
Yolov5 SM X	0.949	0.954	0.972

FIG13PERFORMANCE EVALUATION TABLE

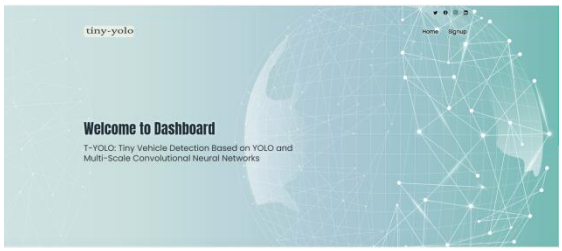

Fig 14 home page

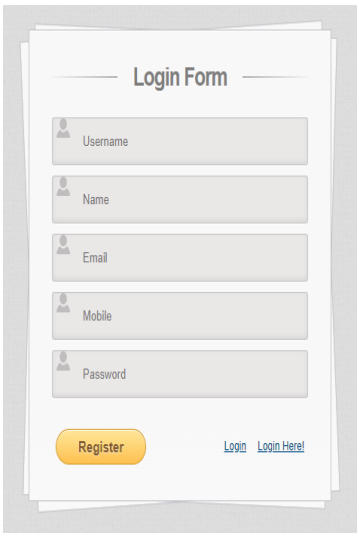

Fig 15 sign up

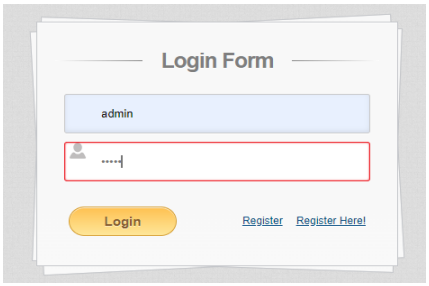

Fig 16 Sign in

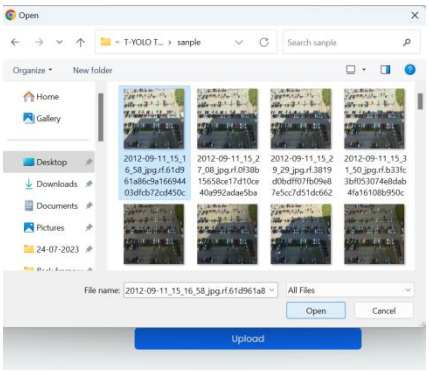

Fig 17upload input image

ISSN 2454 - 5015

Fig 18predict result

5. CONCLUSION

In conclusion, the project demonstrates significant strides in addressing the challenge of accurately detecting small and tiny vehicles within parking areas, thereby improving parking management systems' efficiency. Leveraging advanced techniques such as YOLOv5 variants and Multi-Scale CNN architectures, the project achieves notable advancements in detection accuracy. Particularly, the integration of the YOLOv5x model leads to enhanced mean Average Precision, highlighting the project's adaptability to state-of-the-art methodologies for superior performance.

Moreover, the incorporation of a user-friendly Flask interface with secure authentication enhances the testing experience, simplifying data input and evaluation processes. This streamlined approach ensures accessibility and usability across various user groups involved in deep learning applications. Overall, the project's accomplishments contribute significantly to advancing vehicle detection technology, with implications for improving parking management efficiency and enhancing user experiences in smart city environments

6. FUTURE SCOPE

.The future scope of the project encompasses several avenues for expansion and refinement. Firstly, deploying the developed vehicle detection model on low-end terminals such as Field-Programmable Gate Arrays (FPGAs) or the NVIDIA Jetson Nano Developer Kit enables real-time parking monitoring in resource-constrained environments, highlighting its adaptability and versatility.

Secondly, the applicability of the T-YOLO model can be extended beyond parking areas to domains like surveillance, object tracking, or robotics, broadening its impact and potential use cases. Further research can focus on optimizing the T-YOLO model for faster inference on low-end devices while maintaining detection accuracy, potentially through model compression techniques or hardware acceleration methods. Additionally, systematic evaluation of the model's performance on different datasets and environmental conditions ensures its robustness and generalizability.

Collaborative efforts with industry partners to implement the T-YOLO model in real-world parking systems offer opportunities for practical deployment and evaluation, providing valuable insights for continuous improvement and refinement based on real-world feedback.

REFERENCES

[1] S. B. Atitallah, M. Driss, W. Boulila, and H. B. Ghézala, Leveraging deep learning and IoT big data analytics to support the smart cities development: Review and future directions, Comput.Sci. Rev., vol. 38, Nov. 2020, Art. no. 100303.[Online].Available:https://www.sciencedirect.com/science/article/pii/S1574013720304032

[2] T. Lin, H. Rivano, and F. Le Mouël, A survey of smart parking solutions, IEEE Trans. Intell. Transp. Syst., vol. 18, no. 12, pp. 32293253, Dec. 2017.

[3] S. J. Faraji and M. J. Naozar, Smart parking: An efcientapproachtocitys smart management and air pollution reduction, J. Air Pollut. Health, vol. 4, no. 1, pp. 5372, Feb. 2019. [Online]. Available:https://japh.tums.ac.ir/index.php/japh/article/view/199

[4] O. Cats, C. Zhang, and A. Nissan, Survey methodology for measuring parking occupancy: Impacts of an on-street parking pricingscheme in an urban center, Transp. Policy, vol. 47, pp. 5563,Apr. 2016. [Online]. Available: https://www.sciencedirect.com/ science/article/pii/S0967070X15300858

[5] G. Amato, F. Carrara, F. Falchi, C. Gennaro, C. Meghini, and C. Vairo, Deep learning for decentralized parking lot occupancy detection, Expert Syst. Appl., vol. 72, pp. 327334, Apr. 2017. [Online].Available:https://www.sciencedirect.com/ science/article/pii/S095741741630598X

[6]S.C.K.Tekouabou,E.A.A.Alaoui,W.Cherif,andH .Silkan, Improving parking availability prediction in smart cities with IoT and ensemble-based model, J. King Saud Univ.-Comput. Inf. Sci., vol. 34, no. 3, pp. 687697, Mar. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S 1319157819312613

[7] M. A. Merzoug, A. Mostefaoui, G. Gianini, and E. Damiani, Smart connected parkinglotsbasedonsecuredmultimediaIoTdevices, Computing, vol. 103, no. 6, pp. 11431164, Jun. 2021.

[8] M. dos Santos de Arruda, L. P. Osco, P. R. Acosta, D. N. Gonçalves, J. M. Junior, A. P. M. Ramos, E. T. Matsubara, Z. Luo, J. Li, J. de Andrade Silva, and W. N. Gonçalves, Counting and locating high-density objects using convolutional neural network, 2021, arXiv:2102.04366.

[9] M. Farag, M. Din, and H. Elshenbary, Deep learning versus traditional methods for parking lots occupancy classication, Indonesian J. Elect.Eng. Comput. Sci., vol. 19, pp. 964973, Aug. 2020.

[10] S. Nurullayev and S.-W. Lee, Generalized parking occupancy analysis based on dilated convolutional neural network, Sensors, vol. 19, no. 2, p. 277, Jan. 2019. [Online]. Available: https://www.mdpi.com/14248220/19/2/277

[11] A. Farley, H. Ham, and Hendra, Real time IP camera parking occupancydetection using deep learning, Proc. Comput. Sci., vol. 179, pp. 606614, Jan. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S 1877050921000533

[12] T.-Y. Lin, M. Maire, S. Belongie, L.Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, Microsoft COCO:Common objects in context, 2014, arXiv:1405.0312.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, ImageNet:Alarge-scale hierarchical image database, in Proc. IEEE Conf. Comput.Vis. Pattern Recognit., Jun. 2009, pp. 248255.

[14] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, The PASCAL visual object classes (VOC) challenge, Int. J. Comput. Vis., vol. 88, no. 2, pp. 303338, Jun. 2010, doi: 10.1007/ s11263-009-0275-4.

[15] S. Ren, K.He,R.Girshick, and J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, 2015, arXiv:1506.01497.

[16] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y.Fu, andA. C. Berg, SSD: Single shot MultiBox detector, in Computer VisionECCV, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer, 2016, pp. 2137.

[17] J. Redmon and A. Farhadi, YOLOv3: An incremental improvement, 2018, arXiv:1804.02767.

[18] A. Bochkovskiy, C.-Y.Wang, and H.-Y. M. Liao, YOLOv4: Optimal speed and accuracy of object detection, 2020, arXiv:2004.10934.

[19] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, Scaled-YOLOv4: Scaling cross stage partial network, 2020, arXiv:2011.08036.

[20] G. Jocher, A. Stoken, J. Borovec, L. Changyu, A. Hogan, A. Chaurasia, L. Diaconu, Doug, Durgesh, F. Ingham, Frederik, Guilhen, A. Colmagro,H. Ye, Jacobsolawetz, J. Poznanski, J. Fang, J. Kim, K. Doan, and L. Yu,Ultralytics/yolov5: V4.0 Nn.SiLU() activations, weights & biases logging, PyTorch hub integration, Ultralytics, Los Angeles, CA, USA, Jan. 2021, doi: 10.5281/zenodo.4418161.

[21] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, Feature pyramid networks for object detection, 2016,arXiv:1612.03144.

[22] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, ENet: A deep neural network architecture for real-time semantic segmentation, 2016, arXiv:1606.02147.

[23] C. KumarB., R. Punitha, and Mohana, YOLOv3 and YOLOv4:Multipleobject detection for surveillance applications, in Proc. 3rd Int. Conf. Smart Syst. Inventive Technol. (ICSSIT), Aug. 2020, pp. 13161321.

[24] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, arXiv:1409.1556.

[25] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, Path aggregation network for instance segmentation, in Proc. IEEE/CVF Conf. Comput. Vis. PatternRecognit., Jun. 2018, pp. 87598768.

[26] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, Dual attention network for scene segmentation, 2019, arXiv:1809.02983.

[27] T. Ridnik, H. Lawen, A. Noy, E. B. Baruch, G. Sharir, and I. Friedman,TResNet: High performance GPU-dedicated architecture, 2020, arXiv:2003.13630.

[28] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, Mixup: Beyond empirical risk minimization, 2017, arXiv:1710.09412.