Using Continual Learning on Edge Devices for Cost-Effective, Efficient License Plate Detection

Reshawn Ramjattan[1], Rajeev Ratan[2] Shiva Ramoudith[3], Patrick Hosein[3], Daniele Mazzei[1]
[1]*Department of Computer Science, University of Pisa, Largo B. Pontecorvo 3, 56127 Pisa, Italy*
[2]*BPP University, London, United Kingdom*
[3]*The University of the West Indies, St. Augustine, Trinidad*
reshawn.ramjattan@phd.unipi.it, mail@rajeevratan.com, shiva.ramoudith@gmail.com, patrick.hosein@sta.uwi.edu, daniele.mazzei@unipi.it

Keywords: Deep Learning, Continual Learning, Object Detection, Edge Computing

Abstract: Deep learning networks for license plate detection can produce exceptional results. However, the challenge lies in real-world use where model performance suffers when exposed to new variations and distortions of images. Rain occlusion, low lighting, glare, motion blur and varying camera quality are a few among many possible data shifts that can occur. If portable edge devices are being used then the change in location or the angle of the device also results in reduced performance. Continual learning (CL) aims to handle shifts by helping models learn from new data without forgetting old knowledge. This is particularly useful for deep learning on edge devices where resources are limited. `Gdumb` is a simple CL method that achieves state-of-the-art performance results. We explore the potential of using continual learning for license plate detection through experiments using an adapted `Gdumb` approach. Our data was collected for a license plate recognition system using edge devices and consists of images split into 3 categories by quality and distance. We evaluate the application for data shifts, forward/backward transfer, accuracy and forgetting. Our results show that a CL approach under limited resources can attain results close to full retraining for our application.

1 INTRODUCTION

Machine learning (ML) solutions have seen overwhelmingly positive results over recent years. Even so, in real-world use, it is common for ML models to be exposed to data that is significantly different from what they were trained on. It would be ideal if those models could swiftly learn from that data and improve, but this is an immensely challenging task. As the model is changed to handle new data, old but meaningful knowledge would likely be lost and this is called catastrophic forgetting (Goodfellow et al., 2013). Continual learning (CL) is a category of ML that strives to address these issues and has been receiving a noteworthy amount of recent attention (Biesialska et al., 2020; Cossu et al., 2021; Khetarpal et al., 2020; Delange et al., 2021). CL addresses the problem of learning from a continuous stream of data while preserving and extending the acquired knowledge. This can provide significant benefits such as the quick adaptability of models to new shifts in data distribution, reduced cost in terms of computing and infrastructure overhead and more efficient use of data through forward/backward transfer of knowledge.

These benefits are especially important for the space of ML on edge devices and have inspired a specific focus on CL for the edge where computing power, memory and sometimes even internet connectivity are limited (Demosthenous and Vassiliades, 2021; Kwon et al., 2021; Piyasena et al., 2020; Pellegrini et al., 2021). Thus, we saw it fit to explore the potential of CL for one such "ML on the edge" use case, license plate detection, using real-world data.

Consider a license plate recognition system for security, comprising a plate detection model, an optical character recognition model and software for handling input, orchestrating the models and interacting with a database. The recognition system runs on a setup consisting of an Nvidia Jetson Nano module with a CSI camera mounted onto it and can be used in many environments that present varying types of input images. The unique variety of images is difficult to foresee and can affect the performance of the license plate detection step. Given the computational load, and limited computing power coming from a desire to maintain an affordable product, a low-cost CL approach to quickly respond to the data shifts of a new environment would be of great benefit.

Moreover, even if a large model by a license plate recognition product provider was successfully deployed on an edge-based setup, then, with enough data, the model could still be trained to handle these adjustments. However, for a large model to be retrained for every new environment with seemingly minor variations is inconvenient, expensive and can benefit from continual learning.

Despite the crucial advances made in open-source tools and community-building for CL (Lomonaco et al., 2021), it is still a young field. Implementing state-of-the-art approaches on embedded systems, for a task like object detection with complex models, training methods and output shapes is no minor task. Hence, exploring the potential of CL for our use case through simpler means can provide a motivating foundation of empirical evidence to drive forward further work in the space. Moreover, Gdumb (Prabhu et al., 2020) aligns well with our needs and has shown results comparable to state-of-the-art methods (Mai et al., 2022). Gdumb is a simple approach to CL that has impractical computing and memory demands but is ideal for benchmarking.

The Gdumb approach greedily stores samples in a memory buffer as they come and then trains a model from scratch using the buffer at test time. Our approach to this style fills the buffer with samples of a specific environment scenario and uses it for incremental training. In order to produce an upper bound for comparison, we also adapted this setup with a fully accumulating buffer that collects all samples over time for all scenarios. Such experiments will allow one to check the model's ability to learn incrementally and empirically observe the impact of data changes on the model.

Our work has the following research goals: (1) to observe the data shift problem for license plate detection in different environments using real data and (2) to estimate the practical benefits of using CL to address this problem. We aim to meet these goals using Gdumb-styled experiments. Our contributions, therefore, include, showing the impact of small variations on a modern deep learning model designed for the edge and exploring the potential of continual learning to ease adaptations without complete retraining by looking at a strong benchmark. To the best of our knowledge, this is also the first evaluation of Gdumb for continual object detection.

In this paper, we first review the existing relevant literature. We then describe our base model and the data and its different scenarios. Afterwards, we put forward experiments for evaluating the potential of CL and observing data shift and forgetting. Finally, we present and discuss the results before concluding.

2 RELATED WORK

Deep learning for edge devices has been well studied and widely successful (Li et al., 2018; Chen and Ran, 2019; Wang et al., 2020), especially for computer vision problems through convolutional neural networks (CNNs) (Huang et al., 2021; Sufian et al., 2021). This success has naturally extended to the case of license plate detection on edge devices (Jamtsho et al., 2021; Alam et al., 2021) through object detection algorithms based on CNNs such as YOLO (Redmon et al., 2016) and EfficientDet (Tan et al., 2020). It must be noted though that these cases train using ideal, close and clear images of vehicles and plates. Work by (Silva and Jung, 2018; Xie et al., 2018) examines license plate detection on images with oblique plates or ones from various countries, but real-world scenarios can be even more challenging with varying camera qualities, distant, moving and out-of-focus vehicles, rain or snow occlusion, image noise issues, over/under-exposure, motion blur, etc. Furthermore, these works do not explore a continual learning perspective to address the challenges that come with a product that must perform in many different environments.

There is an extensive number of CL approaches and methods and they can be loosely categorized as follows: (1) regularization-based (2) architectural (3) replay-based (Parisi et al., 2019; Delange et al., 2021). Gdumb on the other hand is a simple baseline metric. It greedily stores all samples as the scenarios change and completely retrains the base model on them. It demands heavy resources and is not suitable for use in most real-time situations.

However, it provides a useful benchmark comparison for complicated CL techniques. Moreover, studies have shown this approach outperforms many state-of-the-art CL methods (Mai et al., 2022; Prabhu et al., 2020). Our research goals are to show the impact of catastrophic forgetting in a seemingly easy-to-solve application and to observe the potential of CL in addressing it. Thus, we consider experiments around adapting Gdumb to be a simple and effective way to evaluate the potential of CL in license plate detection.

Some interesting applications of on-edge CL have demonstrated its ability to ease the path towards high-quality performance at low computational cost (Kwon et al., 2021; Shao et al., 2021). Additionally, recent investigations of CL for object detection have shown promising results with the adaptability that can be achieved (Liu et al., 2020; Wang et al., 2021).

While there has been a growing interest in CL approaches for object detection problems, there is relatively little work done in the area, both in terms of empirical evaluations and use case applications. So a simple approach to evaluating the potential benefits of CL to detection use cases can be of benefit. We evaluate such an approach to the problem of license plate detection under various challenging conditions.

3 EXPERIMENTAL SETUP

3.1 Dataset

The dataset is comprised of 1223 images of vehicles and license plates. Each image has an associated label file storing the details of the bounding box locations for each license plate. The images were gathered and labelled as part of work done for a license plate recognition product. They were acquired using various edge devices, such as smartphones and CCTV cameras, in three different types of scenarios. Most of the images were taken from oblique angles. While there are many possible scenarios, we chose three of the most likely types to occur. Examples of each are illustrated in figure 1.

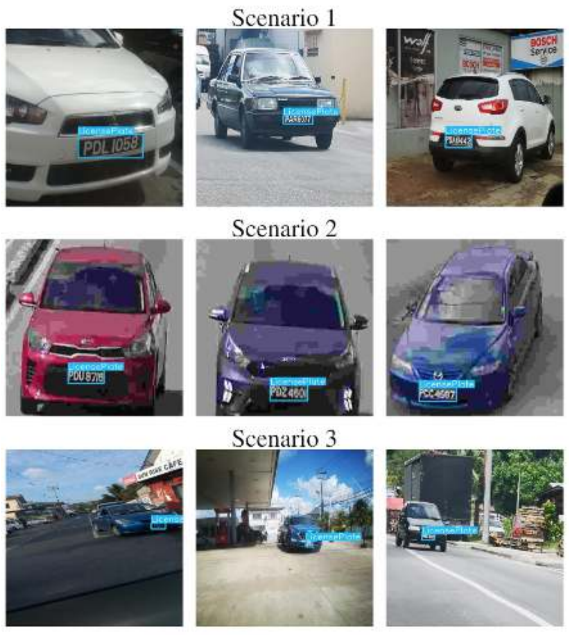

Figure 1: Samples of images under varying conditions

3.1.1 Scenario 1: High quality and close

This subset is made up of 360 images and represents the ideal situation. License plates and vehicles are captured from close range using relatively high-quality cameras that are still feasible for use in an embedded system product.

3.1.2 Scenario 2: Low quality and blurred

Scenario 2 focuses on lower-quality input that has slight motion blur and contains 300 images. Aside from the affordability and availability of lower-quality cameras, consider the following example as an illustration of why this input type might be a common occurrence. The license plate recognition system is used for allowing or rejecting vehicles at a point of entry. To secure the product, it is placed at a high point, perhaps on a pole as is commonly done for CCTV cameras. The image is then zoomed in to a key physical point to capture only the vehicle under review. With the distance and zoom ability of an affordable camera, the resulting image can be of lower quality and contain slight motion blur.

3.1.3 Scenario 3: Unisolated and distant

The third set looks at 563 photos taken from a distance where the vehicle may be one of many or only a small fraction of the entire picture. Elaborating on the prior example, if the same highly placed camera needs to now cover multiple entry points or is to be used for traffic analytics at a wider angle, then the vehicle positions can no longer be precisely anticipated. This results in multiple distant vehicles in a single image, greater skew of plates and generally less isolated vehicles.

3.2 Base Model

The object detection algorithm chosen for the experiments was YOLOv5 (Jocher et al., 2022) because of its impressive benchmark results, well-established community support and, thorough documentation. The problem of detecting license plates shares underlying similarities with problems such as detecting vehicles and recognizing backgrounds. Moreover, our dataset of scenarios contains a small number of samples. Hence, we chose to use weights pre-trained on the COCO dataset (Lin et al., 2014). Since the model needs to run on an edge device with limited resources, the YOLOv5s model version was used as its smaller size was designed for the edge.

4 METHOD AND RESULTS

We conducted three experiments to observe data distribution shifts, forget, accuracy, forward transfer and backward transfer. Each test involved multiple training steps, with access to different levels of data and tested against each scenario. At each training step, the model was trained for 50 epochs after which the optimal weights were selected. As per the style of `Gdumb`, zero hyperparameter tuning was done at any stage. Due to its ideal characteristics, images from scenario 1 (high quality and close) were used first for training at each test.

4.1 Experiment 1

We first established contextual upper-bound metrics for our models by using all 3 scenarios combined for training and testing. The YOLOv5s model gave an end result of 92.6% accuracy.

4.2 Experiment 2

The model was trained on each scenario separately. That is, the same pre-trained weights were used at each step with a single scenario's data, and the result was evaluated against the test sets of all settings. The results of this are presented in table 1.

4.3 Experiment 3

The YOLOv5s model was trained on each scenario incrementally. It was done in the order of scenarios 1 to 3 and each step used the resulting weights of the prior step. Table 1 shows the results of each step.

4.4 Experiment 4

Lastly, for a greedy approach, we trained the model on incremental scenarios while keeping old data at each step, starting from the pre-trained weights. So, first, it used scenario 1 with the initial pre-trained weights, then the second step used scenarios 1 and 2 with the initial pre-trained weights, and the third step used all three settings with the same initial weights. The results are shown in table 1.

5 DISCUSSION

The overall accuracy results for our problem in experiment 1 are positive, at 92.6% accuracy, considering the challenging cases and relatively small dataset.

The remaining experiments are geared towards evaluating the CL concerns of data distribution shift, forgetting, and forward and backward transfer. For the sake of clarity, this section will discuss the results for each of those individually.

5.1 Data Distribution Shift

Data shifts are the crux of the problem CL tries to solve. They represent a change that an existing model was not trained to handle, which translates into poor performance in production. Therefore, examining data shifts is crucial in evaluating the applicability of CL.

From experiment 2, the data shifts can be empirically seen. We observe differences in accuracy as follows: 12.3% between the close-set (scenario 1) and the blurry set (scenario 2), 41.6% from the close set to the distant set (scenario 3) and lastly, 74.7% between the blurry and distant sets. These results illustrate the difference between the close or blurry sets compared to the distant sets. Experiments 3 and 4 reveal similar evidence. The model trained on clean and blurry data incrementally with the same weights performed 57.4% worse on distant predictions and the model trained on close and blurry at the same time performed 45% worse.

5.2 Forgetting

Catastrophic forgetting is the primary challenge standing in the path of CL's ability to cope with data shifts. Thus, an estimation of this issue is also of interest to our evaluation. In experiment 3, weights were incrementally trained one scenario at a time in the order of close, blurry and distant. At stage 2 when the blurry set was used, a decrease of 0.5% on the close-set is seen. Meanwhile, when the distant set was introduced, the blurry and close results dropped by 7% and 4.8% respectively. These changes due to forgetting may seem small in comparison to the data shifts discussed, but ideally, through CL the model should retain prior knowledge and even improve with the new knowledge.

5.3 Backward Transfer

Improving on old samples due to learning from new ones is also known as backward transfer. In experiment 4, when access to data is unrestrained we can see small backward transfer improvements on prior scenarios as new ones are introduced. These improvements are 0.2% on both the close and blurry scenarios.

Table 1: Accuracy results for all experiments.

Train \ Test	Scenario 1	Scenario 2	Scenario 3	Average
Exp 2: Training from scratch for each scenario				
Scenario 1	99.3	87.0	57.7	81.3
Scenario 2	89.1	98.8	24.1	70.7
Scenario 3	92.6	62.3	79.4	78.1
Exp 3: Incrementally training the same weights without storing scenarios				
Scenario 1	99.3	87.0	57.7	81.3
Scenario 2	98.0	98.9	41.1	79.3
Scenario 3	93.2	91.9	82.1	89.1
Exp 4: Incrementally training the same weights while storing scenarios				
Scenario 1	99.3	87.0	57.7	81.3
Scenario 1 + 2	99.5	98.8	54.1	84.1
Scenario 1 + 2 + 3	99.5	99.0	79.2	92.6

5.4 Forward Transfer

Forward transfer refers to the ability to learn a new task easier because of learning another task first. If we compare the model training on the distant setting first versus training on that set after learning from the close and blurry settings, we see a 2.7% improvement. Similarly, a 0.1% increase is observed if the blurry set is used for training after the close one.

A more noticeable example of forward transfer is noted when looking at the data distribution shifts. There is a moderate shift of 12.3% between close and blurry changes to 9.7% if the blurry set is trained before the close. Likewise, the vast difference of 74.7% between distant and blurry goes to 17.1% if the distant set is seen before the blurry.

5.5 Accuracy

The average accuracy across all scenarios for the different stages of each experiment was also calculated. The highest 3-setting average for a model trained on one scenario came from the close set rather than the distant set which was the most challenging. Whereas the lowest result came from training on the blurry only. The best result attained, as one might expect, came from training on all 3 scenarios with no memory constraints at 92.6%. However, incrementally training weights on the 3 settings resulted in an average accuracy of 89.1%, only 3.5% less, which is a positive sign for the continual approaches to this problem.

6 CONCLUSIONS

A license plate recognition system in a production environment can see dynamic changes in its input data. The ideal scenario of ‘clean’ images can quickly shift to a more challenging data distribution. By procuring a dataset of 3 likely settings, Gdumb-based tests allowed us to empirically observe the data shift, forgetting, accuracy and forward/backward transfer for our plate detection problem.

The data shifts proved to be a problem that cannot be ignored with as much as a 74.7% performance drop. An adapted Gdumb approach to addressing this resulted in a final average accuracy of 89.1% versus 92.6% from regular training. On the other hand, the degree of forgetting perceived in comparison to the backward and forward knowledge transfer observed was less than ideal. Therefore, we can conclude that an incremental continual learning approach to license plate detection shows potential for low-compute on-edge solutions but still has room for improvement.

In terms of future work, while Gdumb is an excellent benchmark method and has demonstrated similar or better performance than recent CL approaches, it is still among the most resource-demanding. Our incremental training approach, with a clear comparison to unconstrained retraining on full data over time, can reduce the load of what must be collected on edge devices. Moreover, it shows the accuracy trade-off of a continual approach that is more cost-effective and efficient as a solution. A foundation that can be improved upon by leveraging more advanced CL formulations in this complex model setting. Therefore, comparing the presented results to other strong methods like iCARL or Maximal Interfered Retrieval (Rebuffi et al., 2017; Aljundi et al., 2019) will be useful. These methods can then be deployed on the Jetson Nano setup itself towards an on-edge solution that continually adapts with minimal manual intervention.

Furthermore, there have been interesting advances in the area of Unsupervised Continual Learning (Rao et al., 2019; He and Zhu, 2021; Bertugli et al., 2020). Such an approach could not only aid in solving the issues discussed in this paper but can also allow for the product using the detection model to be much more scalable to new environments without manual labelling. Hence, comparing the results and practicality of an unsupervised method to the incremental labelling approach is another area for further investigation.

ACKNOWLEDGEMENTS

This research is supported by the Ministry of University and Research (MUR) as part of the PON 2014-2020 “Research and Innovation” resources – Green/Innovation Action – DM MUR 1061/2022. This research was also supported by the Nvidia Academic Hardware Grant Program.

REFERENCES

Alam, N.-A., Ahsan, M., Based, M. A., and Haider, J. (2021). Intelligent system for vehicles number plate detection and recognition using convolutional neural networks. *Technologies*, 9(1):9.

Aljundi, R., Belilovsky, E., Tuytelaars, T., Charlin, L., Caccia, M., Lin, M., and Page-Caccia, L. (2019). Online continual learning with maximal interfered retrieval. *Advances in neural information processing systems*, 32.

Bertugli, A., Vincenzi, S., Calderara, S., and Passerini, A. (2020). Few-shot unsupervised continual learning through meta-examples. *arXiv preprint arXiv:2009.08107*.

Biesialska, M., Biesialska, K., and Costa-jussà, M. R. (2020). Continual lifelong learning in natural language processing: A survey. *arXiv preprint arXiv:2012.09823*.

Chen, J. and Ran, X. (2019). Deep learning with edge computing: A review. *Proceedings of the IEEE*, 107(8):1655–1674.

Cossu, A., Carta, A., Lomonaco, V., and Bacciu, D. (2021). Continual learning for recurrent neural networks: an empirical evaluation. *Neural Networks*, 143:607–627.

Delange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., and Tuytelaars, T. (2021). A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Demosthenous, G. and Vassiliades, V. (2021). Continual learning on the edge with tensorflow lite. *arXiv preprint arXiv:2105.01946*.

Goodfellow, I. J., Mirza, M., Xiao, D., Courville, A., and Bengio, Y. (2013). An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*.

He, J. and Zhu, F. (2021). Unsupervised continual learning via pseudo labels. *arXiv preprint arXiv:2104.07164*.

Huang, Z., Yang, S., Zhou, M., Gong, Z., Abusorrah, A., Lin, C., and Huang, Z. (2021). Making accurate object detection at the edge: review and new approach. *Artificial Intelligence Review*, pages 1–30.

Jamtsho, Y., Riyamongkol, P., and Waranusast, R. (2021). Real-time license plate detection for non-helmeted motorcyclist using yolo. *Ict Express*, 7(1):104–109.

Jocher, G., Chaurasia, A., Stoken, A., Borovec, J., NanoCode012, Kwon, Y., TaoXie, Fang, J., imyhxy, Michael, K., Lorna, V, A., Montes, D., Nadar, J., Laughing, tkianai, yxNONG, Skalski, P., Wang, Z., Hogan, A., Fati, C., Mammana, L., AlexWang1900, Patel, D., Yiwei, D., You, F., Hajek, J., Diaconu, L., and Minh, M. T. (2022). ultralytics/yolov5: v6.1 - TensorRT, TensorFlow Edge TPU and OpenVINO Export and Inference.

Khetarpal, K., Riemer, M., Rish, I., and Precup, D. (2020). Towards continual reinforcement learning: A review and perspectives. *arXiv preprint arXiv:2012.13490*.

Kwon, Y. D., Chauhan, J., Kumar, A., Hui, P., and Mascolo, C. (2021). Exploring system performance of continual learning for mobile and embedded sensing applications. *arXiv preprint arXiv:2110.13290*.

Li, H., Ota, K., and Dong, M. (2018). Learning iot in edge: Deep learning for the internet of things with edge computing. *IEEE network*, 32(1):96–101.

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 *European conference on computer vision*, pages 740–755. Springer.

Liu, X., Yang, H., Ravichandran, A., Bhotika, R., and Soatto, S. (2020). Continual universal object detection. *arXiv preprint arXiv:2002.05347*.

Lomonaco, V., Pellegrini, L., Cossu, A., Carta, A., Graffieti, G., Hayes, T. L., De Lange, M., Masana, M., Pomponi, J., Van de Ven, G. M., et al. (2021). Avalanche: an end-to-end library for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3600–3610.

Mai, Z., Li, R., Jeong, J., Quispe, D., Kim, H., and Sanner, S. (2022). Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

Pellegrini, L., Lomonaco, V., Graffieti, G., and Maltoni, D. (2021). Continual learning at the edge: Real-time training on smartphone devices. *arXiv preprint arXiv:2105.13127*.

Piyasena, D., Thathsara, M., Kanagarajah, S., Lam, S. K., and Wu, M. (2020). Dynamically growing neural network architecture for lifelong deep learning on the edge. In *2020 30th International Conference on Field-Programmable Logic and Applications (FPL)*, pages 262–268. IEEE.

Prabhu, A., Torr, P. H., and Dokania, P. K. (2020). Gdumb: A simple approach that questions our progress in continual learning. In *European conference on computer vision*, pages 524–540. Springer.

Rao, D., Visin, F., Rusu, A., Pascanu, R., Teh, Y. W., and Hadsell, R. (2019). Continual unsupervised representation learning. *Advances in Neural Information Processing Systems*, 32.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. (2017). icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788.

Shao, Y., Zhao, Y., Yu, F., Zhu, H., and Fang, J. (2021). The traffic flow prediction method using the incremental learning-based cnn-ltsm model: the solution of mobile application. *Mobile Information Systems*, 2021.

Silva, S. M. and Jung, C. R. (2018). License plate detection and recognition in unconstrained scenarios. In *Proceedings of the European conference on computer vision (ECCV)*, pages 580–596.

Sufian, A., Alam, E., Ghosh, A., Sultana, F., De, D., and Dong, M. (2021). Deep learning in computer vision through mobile edge computing for iot. In *Mobile Edge Computing*, pages 443–471. Springer.

Tan, M., Pang, R., and Le, Q. V. (2020). Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790.

Wang, F., Zhang, M., Wang, X., Ma, X., and Liu, J. (2020). Deep learning for edge computing applications: A state-of-the-art survey. *IEEE Access*, 8:58322–58336.

Wang, J., Wang, X., Shang-Guan, Y., and Gupta, A. (2021). Wanderlust: Online continual object detection in the real world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10829–10838.

Xie, L., Ahmad, T., Jin, L., Liu, Y., and Zhang, S. (2018). A new cnn-based method for multi-directional car license plate detection. *IEEE Transactions on Intelligent Transportation Systems*, 19(2):507–517.