information

MDPI

Article

Deep Homography for License Plate Detection

Hojin Yoo and Kyungkoo Jun *

Dept. of Embedded Systems Engineering, Incheon National University, Incheon 22012, Korea; yojin@inu.ac.kr

* Correspondence: kjun@inu.ac.kr

Received: 27 February 2020; Accepted: 15 April 2020; Published: 17 April 2020

Abstract: The orientation of plate images in license plate recognition is one of the factors that influence its accuracy. In particular, tilted plate images are harder to detect and recognize characters with than aligned ones. To this end, the rectification of plates in a preprocessing step is essential to improve their performance. We propose deep models to estimate four-corner coordinates of tilted plates. Since the predicted corners can then be used to rectify plate images, they can help improve plate recognition in plate recognition. The main contributions of this work are a set of open-structured hybrid networks to predict corner positions and a novel loss function that combines pixel-wise differences with position-wise errors, producing performance improvements. Regarding experiments using proprietary plate images, one of the proposed modes produces a 3.1% improvement over the established warping method.

Keywords: image processing; ALPR; deep neural networks; rectification; SVM; handwriting

1. Introduction

Automatic license plate recognition (ALPR) has had many real-world practical applications such as automatic toll collection, traffic law enforcement, parking lot access control, and road traffic monitoring. With increasing needs for intelligent transportation systems, ALPR has been studied actively despite its long history. In general, the pipeline of license plate recognition (LPR) systems has the four steps of image preprocessing, license plate detection, character segmentation, and optical character recognition (OCR). Various image conditions, such as perspective, color, font, occlusion, illumination, background, and country standards, make ALPR challenging.

ALPR is one of the active areas that are eager to benefit from the brilliant performance of deep learning methods. Such attempts began with using the Recurrent Neural Network (RNN) and regression boxes to localize texts under complicated environments [1]. Convolutional Neural Network (CNN) was the most successful solution for detecting plates typically coupled with sliding windows [2–5]. Variations of CNN such as Fast-RCNN and YOLO were proposed as well [6–8]. Recent approaches [9–12] have sought end-to-end architectures in which not only recognition but also image transformation occurs at the same time. Deep-learning-based approaches have been more efficient than traditional image processing methods, because feature decisions can be automated through the training process.

The outstanding performance of deep-learning-based ALPR is partly attributed to image-warping methods that rectify deformed plates in images. Even though image augmentation methods can help to improve the recognition accuracy and generalization of ALPR by artificially generating various types of deformed plates, warping methods have advantages over augmentation in terms of cost and computation. The warping task essentially involves matching and regression for parameters representing deformation, which can then be used for the rectification of plates, which leads to improved plate recognition.

In this paper, we propose deep learning models that are capable of predicting four-corner coordinates of deformed plate images, as shown in Figure 1. The four corners are then used to calculate

transformation matrix, called homography, that rectifies plates. Whereas affine transformation is not always successful in rectifying plates because of its limitation that keeps parallel lines as they are, homography is general enough to handle various types of projective transformation. The homography is as follows in Equation (1).

$$w\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (1)$$

where (x, y) in one image is mapped to (x', y') in a target image. Since the homography matrix has the degree of freedom of 8 with $h_{33} = 1$, we need a four pair of matching 2D coordinates to determine the matrix parameters of h_{ij}.

Figure 1. Example plate images from data set. Each image contains a single randomly oblique plate with size of 416×416. We manually mark four corners of p_i, $1 \leq i \leq 4$, for supervised training. The plate numbers are partly blurred for privacy reasons.

In this paper, we tackle the four-corner prediction problem by building and experimenting with a set of deep neural network models. We contribute by proposing an open neural network that can plug in well-known models such as Mobilenet and autoencoders in order to harness their power of feature extraction. Another contribution is a novel loss function that considers prediction errors not only in terms of coordinates but also in terms of image level. This paper is organized as follows. We review ALPR-related works in Section 2. We present our models in detail in Section 3 and discuss the experiment results in Section 4. We conclude the paper in Section 5 by discussing future work.

2. Related Work

Traditional methods for LPR have had the structure of the two steps involving text segmentation and following text recognition [13–16]. These methods, which depend mainly on the presence of discriminative characteristics such as color, shape, edge, or texture, have achieved limited success. The most notable disadvantage of the methods has been slow speed due to their inherent complex structure.

With the advent of deep learning methods, many research efforts have focused on a single-stage structure removing the segmentation step. Recurrent Neural Networks (RNNs) with Long Short-Term Memory (LSTM) are able to extract features by scanning through sequential features of license plates [17]. A fully convolutional network implements segmentation and annotation-free ALPR by using synthetic plate images and plate character samples [18]. Most segmentation-free works depend on features that are learned by using Deep CNN enabled by residual layers [19].

However, these methods usually consider as input high-quality license plate images that have little variation in terms of illumination, plate obliqueness, and intensity. Such quality requirements often lead to low performance in actual field tests. To this end, there have been works that integrate

image quality improvement in networks [11,12,20]. The main concern of their works is the rectification of slanted plates in images. A framework that includes de-noising and rectification is proposed for recognition improvement [20]. An affine transformation is estimated to align slanted plates [12]. A super-resolution technique is introduced to handle unconstrained real-world traffic scenes [11].

ALPR has been considered as a mature problem and misunderstood as easily solvable with deep learning. However, once the assumptions for input images are loosened, it poses a new level of challenges, typically in unconstrained scenes. In this work, we propose deep learning models that locate four-corner positions of plates, which are required to rectify plates. Unlike previous works that use affine transformation for rectification, our works build a homography that is a more versatile transformation.

3. Deep Models for Corner Prediction

We propose deep neural network models, as shown in Figure 2, which predict four-corner coordinates of license plates in an end-to-end manner. With the coordinates, we build a transformation matrix rectifying plates, resulting in improved character recognition accuracy in ALPR. The models share a similar two-phase structure; the first phase extracts geometric characteristics, which are then fed to fully connected layers of the second phase, which performs the prediction of the coordinates.

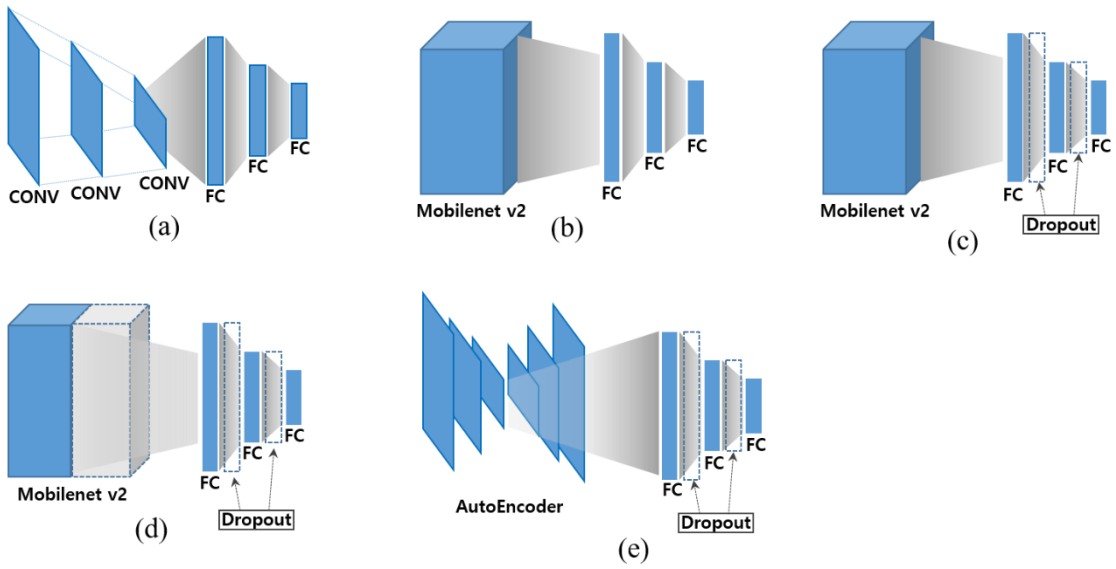

Figure 2. Four-corner prediction models. (**a**) is baseline model and (**b**) is a hybrid model with Mobilenet. (**c**) is a hybrid Mobilenet with dropout. (**d**) is a hybrid model that uses only the feature extraction layers of the Mobilenet. (**e**) is a hybrid model using autoencoder.

Figure 2a is a baseline model that consists of a set of convolution layers followed by a set of fully connected layers. The input is an image with dimension of W × H × C, and the output is a length-8 vector corresponding to four pairs of normalized (x_i, y_i), $1 \leq i \leq 4$. A hybrid model replacing the convolution layers with Mobilenet [21] is presented in Figure 2b. Its motivation is to avoid reinventing the wheel by having well-known object detection model extract geometric features. The next model of Figure 2c is the same as the hybrid model, except that the dropout layers are inserted between the fully connected layers for the model generalization. In the sense of feature extraction, it seems more appropriate to use the intermediate results of Mobilenet rather than the classification results. This observation motivates the configuration of Figure 2d that connects the output of the hidden layers of Mobilenet to the fully connected layers. The idea of Figure 2e stems from the fact that the encoders of autoencoder networks are good at capturing characteristics of images. It encourages one to feed the latent results from an autoencoder to the fully connected layers. We train the autoencoder in advance to be capable of reconstructing plates.

The models are open in the sense that, besides Mobilenet, other established networks can be plugged in the place of the feature extraction. A relatively small size but comparable performance is the main reason to use Mobilenet in our configuration, because one of the ALPR requirements is compactness for on-site installation. However, we do not limit the possibility to plug in other models such as Resnet [19] and MnasNet [22].

The second phase layers consist of fully connected layers for regression. They are different only by the dimensions of their inputs, which are dependent on the outputs from their corresponding previous layers. The regression output should be de-normalized before use by multiplying with image dimension.

As a loss function for an input image I, we use a weighted sum of two loss factors as Equation (2):

$$L(I) = \lambda \cdot L_{ssd}(I) + (1 - \lambda) \cdot \alpha \cdot L_{diff}(I) \tag{2}$$

where $L_{ssd}(I)$ is related to how far the predicted corner coordinates are from ground truth values and $L_{diff}(I)$ is about how different the rectified images based on the predictions are from ground truth image in pixel-wise. λ determines the proportion of each factor with $\lambda \in [0, 1]$, and α is a constant that is needed to compensate the volume difference between two factors.

$L_{ssd}(I)$ is the sum of squared differences between four corresponding coordinate pairs of ground truth and predicted coordinates as Equation (3).

$$L_{ssd}(I) = \sum_{i=1}^{4} (x_i^{gt} - x_i^{pred})^2 + (y_i^{gt} - y_i^{pred})^2 \tag{3}$$

where $\left(x_i^{gt}, y_i^{gt}\right)$ and $\left(x_i^{pred}, y_i^{pred}\right)$ are the normalized coordinates of ground truth and prediction, respectively, and i represents the positions.

The other loss factor of $L_{diff}(I)$ is L1 loss for the transformed input image as Equation (4)

$$L_{diff}(I) = \frac{1}{N} \sum_{(i,j) \in N}^{N} \| I_{i,j}^{r} - H(I)_{i,j} \|^1 \tag{4}$$

where $H(I)_{i,j}$ is the perspective transformation result of I based on the prediction and $I_{i,j}^{r}$ is the ground truth of rectification that can be easily obtained from ground truth four-corner positions. Having two losses helps to preserve the appearance of plates such as intensity and illumination, and leads to denoising results capable of only geometric transformation.

Having four-corner predictions, the rectification begins by determining the target coordinates to which the predicted ones are mapped, which are the four corners of the plate rectangle without obliqueness. The target coordinates can be easily decided assuming the sizes and aspect ratio of plates are known in advance. Once one-to-one correspondence relationship between the target and the predictions is established, a homography matrix is determined by which actual rectification proceeds.

4. Experiments

We use a proprietary data set of 1137 plate images, all of which are tilted. All the images are preprocessed to be gray and have a size of 416×416. Since plates have wide width that is not able to fit the whole images, the bottom part remained empty. For training purposes, we manually labelled four corner coordinates of plates in separate files. The four corners are $p_i,\ i = 1 \ldots 4$ of $(x_i,\ y_i)$; p_1 is the left top corner, and the rest are numbered clock-wise as in Figure 1. The coordinate p_is is normalized as in YOLO [23] by dividing x_i and y_i by image dimension, having $0 \leq x_i, y_i \leq 1$.

We split the data set by a ratio of 8:2, resulting in 909 images for the training set and 228 for the validation set. For training, we apply augmentation to the data set for model generalization and the diversification of plate obliqueness because most of the plate images are slanted to the right. The five

types of augmentation are shown in Figure 3; they are symmetrical transformation, skewing, or the composition of them.

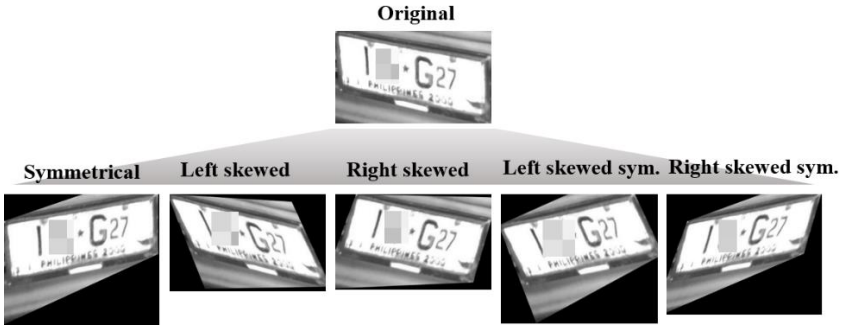

Figure 3. We use five types of augmentation, which are transformation, skewing, or their combination.

We train each of the models for 500 epochs on a same GPU, using an Adam optimizer with learning rate of 0.002 and weight decay of 0.0004. We use the batch size of 32. For the models including Mobilenet, we use the implementation of reference [24] with provided pretrained weights. Regarding the autoencoder, we train the autoencoder for 500 epochs with the data set and use only the encoder part. We implement each of the models using PyTorch [25], and source code is available at https://github.com/kyungkoo70/deep-plate-homography.

For performance comparison, we consider the WPOD-NET. However, it requires additional works before we can compare it with ours by the errors of predicted corners because the outputs of WPOD-NET are affine transformation that rectifies plates. Moreover, since WPOD-NET is tightly integrated into the end-to-end license plate recognition networks, it needs to be separated as an independent module. To calculate four-corner positions of WPOD-NET, we use an indirect way that applies the inverse matrix of the affine transformation to the four corners of rectified plates. The four corners are known because they can be calculated from ground truth corner positions. Even though it is not the perfect comparison because of errors induced during transformation, it is the best workaround to our knowledge.

Table 1 shows the average L2 distance of the predicted corners from ground truth depending on λ. Since normalized coordinates are used, the results are extremely small. We observe that the hybrid model achieves the best result, 1.25×10^{-2} at $\lambda = 0.6$ among the considered models. The hybrid outperforms WPOD-NET by 0.04×10^{-2}, which is 3.1% improvement. On other values of λ as well, we find the hybrid is superior to other models except WPOD-NET. From the results of extreme λ such as 0.0 and 1.0, we infer that the pixel-wise losses is not effective as the coordinate error based losses. However, the pixel-wise loss is complementary to the position-wise loss because it operates on the image level, exploiting nongeometric information such as intensity and illumination.

Table 1. The average differences of normalized predicted coordinates from ground truth. The models trained with the loss function with different λ were used.

Method	λ					
	0.0	**0.2**	**0.4**	**0.6**	**0.8**	**1.0**
Baseline	2.34×10^{-2}	2.04×10^{-2}	1.43×10^{-2}	1.44×10^{-2}	1.45×10^{-2}	1.46×10^{-2}
Hybrid	2.22×10^{-2}	1.95×10^{-2}	1.27×10^{-2}	1.25×10^{-2}	1.27×10^{-2}	1.29×10^{-2}
Hybrid with dropout	2.56×10^{-2}	2.34×10^{-2}	1.56×10^{-2}	1.56×10^{-2}	1.60×10^{-2}	1.67×10^{-2}
Hybrid using extracted feature	2.30×10^{-2}	2.20×10^{-2}	1.30×10^{-2}	1.30×10^{-2}	1.40×10^{-2}	1.46×10^{-2}
Hybrid with encoded	3.84×10^{-2}	3.00×10^{-2}	1.84×10^{-2}	1.84×10^{-2}	2.56×10^{-2}	2.84×10^{-2}
WPOD-NET [12]	1.29×10^{-2}					

Figure 4 shows tilted plates in the first row and the corresponding rectification results by using the homography based on the four-corner predictions from the hybrid model in the second row. We find

that the results are mostly consistent, proving that the predicted corner locations match with actual plate corners. We also observe that the blur of the plate boundaries does not affect the outcome, implicitly showing the robustness of the model against the boundary variation.

Figure 4. Plate images before the transformation (top row) and the corresponding rectified images (bottom row) by using the four-corner predictions from the hybrid model. Parts of the plates are blurred for privacy reasons.

Figure 5 shows the sizes of the proposed models. We observe the hybrid models including Mobilenet are smaller than others except the hybrid-feature model. It is approximately 27 times bigger than the hybrid, the hybrid-dropout, and the hybrid-encoder. The baseline is approximately 22 times larger than the three small-sized models. Their huge sizes are attributed to the high dimensionality of the outputs of the convolution layers.

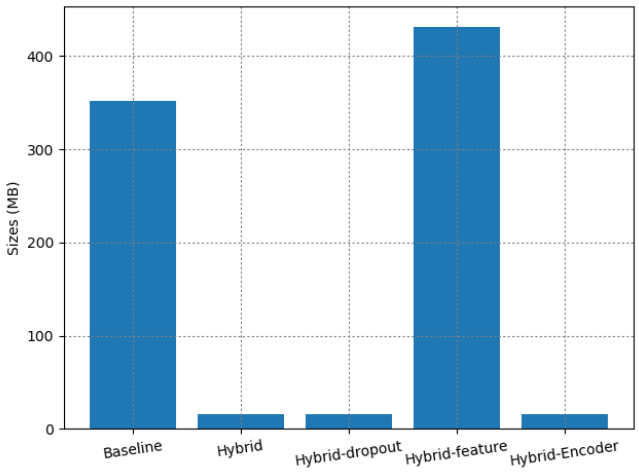

Figure 5. The model sizes. After freezing the models, we measure the stored file size of weights.

5. Conclusions and Future Works

We have introduced the deep models to estimate four corner coordinates of tilted plates. Since the predicted corners can then be used to rectify plate images, they can help improve plate recognition in ALPR. For experiments using proprietary plate images, the hybrid model brings a 3.1% improvement over the established warping method. The main contribution is a set of open-structured hybrid networks used to predict corner positions. As an additional contribution, we presented a new type of loss function that combines the pixel-wise differences with the position-wise errors, bringing performance improvements.

For future work, we want to experiment with our models with more diverse types of established detection models to observe performance variance. This would give us the opportunity to gain greater

insight and understanding about the models and the interactions between them. Moreover, we intend to explore different loss factors that can enforce the relative position between the corner predictions.

Author Contributions: Conceptualization, K.J. and H.Y.; methodology, K.J.; software, K.J. and H.Y.; validation, K.J. and H.Y.; formal analysis, K.J.; investigation, K.J.; resources, K.J. and H.Y.; data curation, K.J. and H.Y.; writing—original draft preparation, K.J. and H.Y.; writing—review and editing, K.J. and H.Y.; visualization, K.J. and H.Y.; supervision, K.J.; project administration, K.J.; funding acquisition, K.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Incheon National University Research Grant in 2014 (No. 2014-1043).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tian, Z.; Huang, W.; He, T.; He, P.; Qiao, Y. Detecting text in natural image with connectionist text proposal network. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Berlin, Germany, 2016; pp. 56–72.
2. Li, H.; Shen, C. Reading car license plates using deep convolutional neural networks and lstms. *arXiv* **2016**, arXiv:1601.05610.
3. Masood, S.Z.; Shu, G.; Dehghan, A.; Ortiz, E.G. License plate detection and recognition using deeply learned convolutional neural networks. *arXiv* **2017**, arXiv:1703.07330.
4. Naimi, A.; Kessentini, Y.; Hammami, M. Multi-nation and multi-norm license plates detection in real traffic surveillance environment using deep learning. In Proceedings of the International Conference on Neural Information Processing, Kyoto, Japan, 16–21 October 2016; Springer: Berlin, Germany, 2016; pp. 462–469.
5. Rafique, M.A.; Pedrycz, W.; Jeon, M. Vehicle license plate detection using region-based convolutional neural networks. *Soft Comput.* **2018**, *22*, 6429–6440. [CrossRef]
6. Girshick, R. Fast R-CNN object detection with Caffe. *Microsoft Res.* **2015**. Available online: https://tutorial.caffe.berkeleyvision.org/caffe-cvpr15-detection.pdf (accessed on 1 February 2020).
7. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
8. Xie, L.; Ahmad, T.; Jin, L.; Liu, Y.; Zhang, S. A new CNN-based method for multi-directional car license plate detection. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 507–517. [CrossRef]
9. Zhuang, J.; Hou, S.; Wang, Z.; Zha, Z. Towards human-level license plate recognition. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 306–321.
10. Gonçalves, G.R.; Diniz, M.A.; Laroca, R.; Menotti, D.; Schwartz, W.R. Multi-task learning for low-resolution license plate recognition. In *Iberoamerican Congress on Pattern Recognition, Havana, Cuba, 28–31 October 2019*; Springer: Berlin, Germany, 2019; pp. 251–261.
11. Lee, Y.; Jun, J.; Hong, Y.; Jeon, M. Practical License Plate Recognition in Unconstrained Surveillance Systems with Adversarial Super-Resolution. *arXiv* **2019**, arXiv:1910.04324.
12. Montazzolli Silva, S.; Rosito Jung, C. License plate detection and recognition in unconstrained scenarios. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 580–596.
13. Kim, K.K.; Kim, K.I.; Kim, J.B.; Kim, H.J. Learning-based approach for license plate recognition. In *Neural Networks for Signal Processing X, Proceedings of the IEEE Signal Processing Society Workshop (Cat. No. 00TH8501), Sydney, Australia, 11–13 December 2000*; IEEE: Piscataway, NJ, USA, 2000; pp. 614–623.
14. Anagnostopoulos, C.E.; Anagnostopoulos, I.E.; Psoroulas, I.D.; Loumos, V.; Kayafas, E. License plate recognition from still images and video sequences: A survey. *IEEE Trans. Intell. Transp. Syst.* **2008**, *9*, 377–391. [CrossRef]
15. Hsu, G.; Chen, J.; Chung, Y. Application-oriented license plate recognition. *IEEE Trans. Veh. Technol.* **2012**, *62*, 552–561. [CrossRef]
16. Zhu, S.; Dianat, S.A.; Mestha, L.K. End-to-end system of license plate localization and recognition. *J. Electron. Imaging* **2015**, *24*, 023020. [CrossRef]

17. Dorbe, N.; Jaundalders, A.; Kadikis, R.; Nesenbergs, K. FCN and LSTM based computer vision system from recognition of vehicle type, license plate number, and registration country. *Autom. Control Compu. Sci.* **2018**, *52.2*, 146–154. [CrossRef]
18. Bulan, O.; Kozitsky, V.; Ramesh, P.; Shreve, M. Segmentation-and annotation-free license plate recognition with deep localization and failure identification. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 2351–2363. [CrossRef]
19. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
20. Lee, Y.; Lee, J.; Ahn, H.; Jeon, M. SNIDER: Single Noisy Image Denoising and Rectification for Improving License Plate Recognition. In Proceedings of the IEEE International Conference on Computer Vision Workshops, Seoul, Korea, 27–28 October 2019.
21. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
22. Tan, M.; Chen, B.; Pang, R.; Vasudevan, V.; Sandler, M.; Howard, A.; Le, Q.V. Mnasnet: Platform-aware neural architecture search for mobile. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 2820–2828.
23. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
24. Mobilenet V2. Available online: https://github.com/pytorch/vision/blob/master/torchvision/models/mobilenet.py (accessed on 1 February 2020).
25. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L. PyTorch: An imperative style, high-performance deep learning library. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 8024–8035.