International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 12, Number 4 (2019), pp. 579-587

Effect of Parametric Changes in Convolutional Neural Networks for Pedestrian Detection - A Study

N K Ragesh[1] and R Rajesh[2]

[1] Tata Elxsi Ltd, Technopark, Trivandrum, Kerala, India.
[2]Dept. of Computer Science, Central University of Kerala, Periya-671316, Kasaragod, Kerala, India.

Abstract

Collision Detection and Prevention Systems (CDPS) is an important component of Advanced Driver Assistance Systems (ADAS) and will continue as an essential component of self-driving cars. In general Collision Detection and Prevention Systems (CDPS) will avoid collision with any objects/vehicles, but special attention is to be given for Pedestrian protection as the majority of accident deaths involve pedestrians. Hence, Pedestrian Protection System (PPS) is part of CDPS, which will monitor the vehicle surroundings and detect potential collision with pedestrians on the road in emergency situations. A PPS will implement techniques to detect pedestrians on the road and possibilities of a collision through tracking the movement of pedestrian and the subject vehicle by making use of camera/RADAR/LiDAR sensors mounted on the vehicle. This article analyses the use of Convolutional Neural Networks (CNN) in pedestrian detection. This paper also makes a study on the effect of parametric changes in CNN with suitable benchmarking datasets.

Keywords: Road Safety, Advanced Driver Assistance Systems (ADAS), CNN, Deep Learning, Pedestrian Detection.

I. INTRODUCTION

Deep learning is a machine learning approach where the learning is done using a deep neural network. Deep neural networks are multi-layer perceptron with many layers combined with custom layers to process 2D/3D images. There are different well known DL architectures like AlexNet [1], VGG Net [2], GoogleNet [3], ResNet [4], ResNeXt [5], RCNN (Region Based CNN) [6], YOLO (You Only Look Once) [7], SqueezeNet [8], SegNet [9], GAN (Generative Adversarial Network) [10] etc. used in different deep learning problems. As the complexity of network increases, the computational requirement and the training and testing time of the network becomes huge, which makes only high performance computers with multiple CPU and GPU cores suitable for large network training. To minimize the training time, one can transfer the learning from an already trained network using transfer learning techniques. In transfer learning, the layers of a learned network used in some other problem are copied (re-utilized) to create a new network. The convergence time for the copied network will be far less than a network training from scratch as well as the learning capacity will be more as the copied layers already have optimum learned weights.

In deep learning, the features (convolution filter coefficients and network weights) are self-learned. However, the suitability of deep learning to a given problem and its accuracy will depend on the dataset used for training the DL network as well as the network parameters used.

The most common network architecture used for deep learning is Convolutional Neural Network (CNN) [11] where, multiple layers are defined to successively filter input image through convolution filters, called Convolutional Chanel Features (CCF) [12], whose output, is fed to a neural network with multiple fully connected layers and an output layer. The filter coefficients are learned along with the network weights.

The parameters like number of layers, number of filters in each layer, size of the convolution filters etc. will affect the accuracy and learning capacity of the CNN. Very less works in this regard can be found in the literature. Hence in this paper, the effect of parametric changes in CNN for pedestrian classification is studied with suitable benchmarking datasets.

II. PEDESTRIAN DETECTION USING DEEP LEARNING

Pedestrian detection is a computer vision problem, where the output from camera is primarily used to detect pedestrians, while output from RADAR and LiDAR sensors are used to improve the robustness of the solution. Camera provides 2D colour images and ranging sensors like RADAR and LiDAR provide depth information. If multiple homogeneous or heterogeneous sensors are available, then each of these inputs are processed separately and the results are combined together using some sensor fusion technique.

However, it is not as simple as that. The appearance of pedestrians is of infinite possibilities due to the type and colour of the dress, posture due to movement etc. It is practically impossible to represent a pedestrian with a set of finite number of templates. This makes the pedestrian detection a very challenging problem and thus require some kind of intelligent/ self-learning techniques to address the pedestrian detection problem.

II.I. Pedestrian Data sets

As the pedestrian appearance have infinite possibilities, it is essential to have a representative set of sample pedestrian images to properly train a machine learning solution to detect

pedestrians. It is needed to first define a pedestrian as a person around the vicinity of a subject vehicle moving on a road. A pedestrian dataset is a subset of person dataset however, limited to automotive environment. A generic person dataset will help in identifying pedestrians, however, a more optimized automotive specific dataset is preferred for better adaptability. There are many person and pedestrian dataset available such as INRIA person dataset, Caltech pedestrian dataset, Cityscapes and particularly CityPersons pedestrian dataset etc., each providing representative pedestrian images with ground truth annotations.

In this paper, both INRIA and CityPersons dataset are used to train a CNN for pedestrian classification and compared the performance.

II.I.I. INRIA person dataset

The full test vectors provided by INRIA dataset [13] is used in the study reported in this paper. The positive training set includes 1826 individual persons from 288 training images. For nonperson images, 3648 randomly sampled non person images from 1268 negative images provided in the dataset are used. Both positive and negative training samples are scaled to 64x128 pixels size.

An expanded INRIA dataset with 2x (doubled) number of person images (3652) and 10x random non-person windows from all negative images (7296 non person windows total) are also used.

II.I.II. Cityperson pedestrian dataset

Cityperson dataset [14] is an enhanced pedestrian annotation from the Cityscapes dataset [15]. Cityperson includes visible as well as partially occluded pedestrian annotations. However, fully visible pedestrians with 128 pixels height from the full HD (1920×1080) training images are only utilized in this study. The total number of positive pedestrian samples is 3896 and non-pedestrian samples is 8064. As with INRIA dataset, the training samples are scaled to 64×128 pixel size.

II.II. Deep learning architecture for pedestrian classification

Convolutional Neural Network (CNN) is used as the base of our deep learning network. A CNN is a deep neural network with initial layers comprised of 2D/3D convolution filters. These convolution operation is followed by a rectified linear unit (ReLU) activation function, max pooling (down sampling) layer and the same trio may repeat multiple times before feeding to a fully connected layer. The last two layers of the CNN are a softmax (normalization) layer and a classification layer with the number of outputs corresponding to the number of classes. In our case, the classification layer has two outputs – a person and a non-person class. Fig. 1 shows a typical CNN architecture.

To understand the impact of CNN configurations on the pedestrian classification performance, experiments have been conducted with different CNN configurations by varying the number of convolution layers, number of filters in each layer as well as the size of the convolution filters. Experiments have been conducted by varying the size of the filly connected layer.

On the CNN network, the first layer is always image input layer which will take 64×128×3 colour image windows. The last two layers are softmax and classification. Classification layer outputs two classes – non person and person. Three combination of layer connections are used as shown in Table 1.

Different combinations of filter size, number of filters to be used, Activation functions [16], and fully connected layer configurations are used. The combinations used are listed in Table 2. We have also tried the custom filter combinations as listed in Table 3.

II.III. Performance Metrics

For evaluating the performance of CNN for pedestrian classification, standard metrics like Precision, Recall, Miss-rate and F1-Score are used (see Table 4).

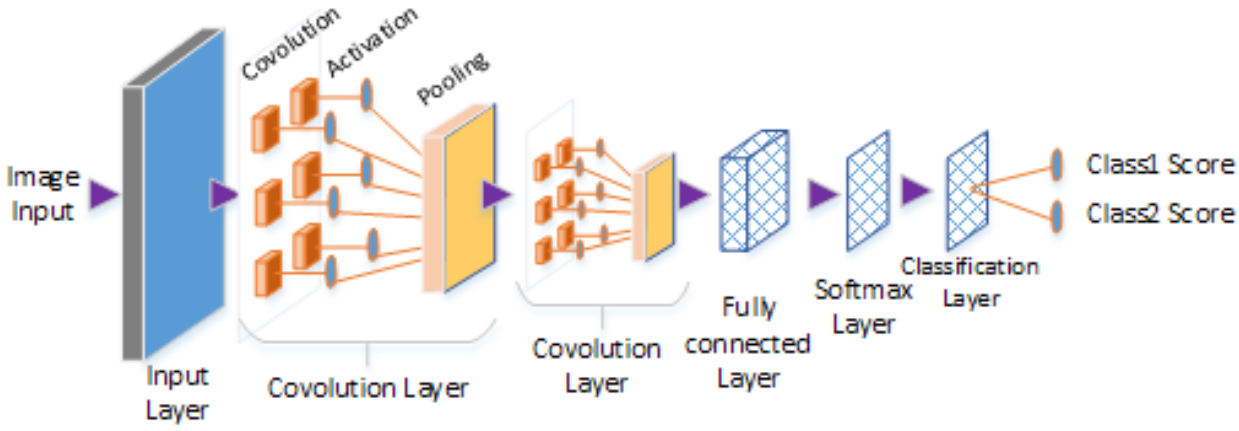

Fig. 1. CNN Network Structure

Table 1. CNN Architectures used in the experiments

CNN Layers	**2x Convolution**	**3x Convolution**	**4x Convolution**
Number of Layers	12	14	16
Input Layer	Image Input Layer [64x128x3]	Image Input Layer [64x128x3]	Image Input Layer [64x128x3]
Middle Layers	Convolution [n×n] ReLU Max Pooling Convolution [n×n] ReLU Max Pooling	Convolution [n×n] ReLU Max Pooling Convolution [n×n] ReLU Convolution [n×n] ReLU Max Pooling	Convolution [n×n] ReLU Max Pooling Convolution [n×n] ReLU Convolution [n×n] ReLU Convolution [2n×2n] ReLU Max Pooling
Final layers	Fully Connected Layer (64) ReLU Fully Connected Layer (2) Softmax Layer Classification Layer	Fully Connected Layer (64) ReLU Fully Connected Layer (2) Softmax Layer Classification Layer	Fully Connected Layer (64) ReLU Fully Connected Layer (2) Softmax Layer Classification Layer

Table 2. CNN Configuration Options

#Filters	**Filter Size**	**Fully Connected layer size**	**Activation Functions**
16, 32	[3×3], [5×5], [7×7], [9×9], [11×11], [13×13]	16, 32, 64 (default), 128	ReLU , Leaky ReLU, Clipped ReLU

Table 3. CNN Custom Filter combination

#	**Conv. Layer 1**	**Conv. Layer 2**	**Conv. Layer 3**
1	32 [3×3] filters	24 [5×5] filters	16 [7×7] filters
2	16 [7×7] filters	24 [5×5] filters	32 [3×3] filters

Table 4. Metrics used for evaluating pedestrian detectors

Metric	**Notation**	**Meaning**
True Positives	T_p	# Objects correctly detected
True Negatives	T_n	# Non-objects correctly rejected
False Positives	F_p	# Non-objects wrongly detected
False Negatives	F_n	# Objects wrongly rejected
Precision	P	$P = \frac{T_p}{T_p + F_p}$
Recall	R	$R = \frac{T_p}{T_p + F_p}$
Miss Rate	MR	$MR = \frac{F_n}{T_p + F_n}$
F1 Score	F1	$F1 = 2 \times \frac{P \times R}{P + R}$

III. EXPERIMENTAL RESULTS

This section discusses the results of experiments and analyse the impact of various network as well as training parameters on the performance of deep learning network.

III.I. Case 1: Impact of dataset selection

CNN is trained with two different datasets and their combinations to compare the relationship between the dataset complexity and the learning capability. The default network configuration is used as defined in Table 1. 32 convolutional filters of size [11×11] are used in each convolution layer. ReLU is used as the activation function. Results are shown in Fig. 2.

1) Observations

a) It can be observed from the results that the performance of a deep learning classifier is always better when testing on the same dataset than on other datasets. This shows that the classifier is highly dependent on the dataset and there is no universal dataset, which can give consistent performance on all datasets. The performance of a CNN classifier is directly dependent on the dataset.
b) Performance of classifier trained with CityPersons dataset on INRIA is better than that of a classifier trained with INRIA dataset when tested on CityPersons dataset. This shows the advantage of CityPersons dataset over INRIA as a representation of a person or pedestrian.
c) As the number of training samples increases, the classification performance also increases. This shows the importance of adding enough samples to capture a generic model of an object for better detection.

2) Inference

Selection of a good representative dataset is essential for the implementing a good object classifier. The performance improves as the number of samples increases.

III.II. Case 2: Impact of filter size on CNN performance

Performance of CNN against filter size used in convolution layers is compared in this subsection. The CNN is trained with CityPersons dataset with 60% vectors as training test. The default network configuration defined in Table 1 is used with 32 filters of different sizes in convolution layers. CNN trained and tested with both INRIA and CityPersons datasets. Fig. 3 shows the results when trained with INRIA and Fig. 4 shows the same when CityPersons dataset used for training.

1) Observations

a) Performance of a DL person classifier is consistently poor when a filter of size [3×3] is used for convolution. A filter of [11×11] is consistently performing good; even better than a filter size of [13×13]. This implies that the filter size should not be too small or too large.
b) When the CNN is trained with CityPersons dataset and tested on INRIA dataset, a filter size of [5×5] was giving the best result. This is an indication of lower object complexity in INRIA compared to CityPersons. The filter size should be matching the object complexity. Or we can say that if the object complexity is high, a larger filter may help in better performance.

2) Inference

Filter size should not be too small (poor performance) and too large (overfitting). But at the same time should be chosen based on the object complexity. A larger filter size will make the network capable of handling more complex object detection problems.

III.III. Case 3: Impact of number of filters in Convolution layers

To analyse the impact of number of filters in convolution layers, we used all three network configurations listed in Table 1 with [11×11] filters in each Convolution layers. Both 16 as well as 32 filter configurations are used. CNN trained with CityPersons dataset and tested on both INRIA and CityPersons. Fig. 5 shows the results.

1) Observations

a) When tested on same dataset used for training, two convolution layers are enough to give best classification performance. However, when tested with a different dataset, more convolution layers and more filters are giving better accuracy.
b) More layers with less number of filters are not leading to good results.
c) A 2-convolution layer network with 16 filters each is giving better performance than a 3-convolution layer CNN with 16 filters each when tested on a different dataset than used for training.
d) The performance drastically improved on same dataset but decreased on different dataset when a fourth convolution layers with 32 filters was added.

2) Inference

More number of filters may not always result in better performance. More number of convolution layers are better than using more number of filters on lesser layers.

III.IV. Case 4: Impact on custom filter size and number

Experiments are conducted with some unique combination of filter size and number of filters in different convolution layers. Both the configurations listed in Table 2 were used with a 3 convolution layer network configuration, similar to the default configuration in Table 1. A default network configuration with 32 [5x5] filters is also used for a better analysis of the impact. The two custom combinations used by us are:

Option 1: 3 convolution layers, with first layer containing 16 [7×7] filters, second layer 24 [5×5] filters and third layer 32 [3×3] filters.

Option 2: 3 convolution layers with first layer containing 32 [3×3] filters, second layer with 24 [5×5] filters and third layer with 16 [7×7] filters.

CNN is trained using INRIA dataset. The results are shown in Fig. 6.

1) Observations

a) Network with larger filters in deeper layers are performing better than networks with smaller filters in deeper layers. This can be treated as an indication that peripheral layers captures wage features and deeper layers capture detailed features, where a larger filter can perform better.
b) Constant filter size gives better performance compared to variable filter size and number of configurations.

2) Inference

Peripheral layers of a CNN capture the outline features of the object whereas the deeper layers captures more detailed features. Larger filters at deeper layers and not too small peripheral layer filters could give better performance.

III.V. Case 5: Impact on fully connected layer size

In this experiment, the impact of varying the interconnections in the fully connected layer are studied while keeping other parameters fixed in the CNN. The default network configuration as listed in Table 1 with 32 [11×11] filters in all three convolutions layers. CNN is trained using City Person dataset. The results are shown in Fig. 7.

1) Observations

More nodes on fully connected network is clearly leading to better performance.

2) Inference

Larger fully connected layers will lead to better classification performance. Beyond a fully connected layer size of 128, the improvement is minimal for the case of pedestrian detection in the proposed CNN architecture.

III.VI. Case 6: Impact on activation functions

Experiments are conducted with different configurations as listed in Table 2. The activation functions used in the experiment are Leaky ReLU with scales '0.1' as well as '0.5' and Clipped ReLU with ceiling at '10' along with standard ReLU. The formulas used for these activations functions are as shown below for a quick reference.

ReLU: $$f(x)=\begin{cases} x & x>0 \\ 0 & otherwise \end{cases} \quad (1)$$

Leaky ReLU: $$f(x,s)=\begin{cases} x & x>0 \\ s.x & otherwise \end{cases} \quad (2)$$

Clipped ReLU: $$f(x,c)=\begin{cases} \min(x,c) & x>0 \\ 0 & otherwise \end{cases} \quad (3)$$

The Network is trained using INRIA dataset and tested on both INRIA and CityPersons. 16 [11×11] filters are used in all three convolutions layers of the default network configuration as in Table 1. Results are shown in Fig. 8.

1) Observations

a) Leaky ReLU is performing better than ReLU and Clipped ReLU
b) Leaky ReLU with a larger scale value is performing better than a smaller scale value.

2) Inference

Leaky ReLU is preferred as the activation function.

III.VII. CNN configuration suggestions for better pedestrian classifier

A summary of the experiment results and the suggestions for a better CNN based DL solution for pedestrian classification are as follows:

a) Choice of dataset is important. The network needs to be trained with more test vectors representing the actual object complexity according to the target scenario for best performance.
b) Too smaller filter size leads to poor classification and too large filters will lead to over fitting the training set. A [5×5] filter is good for low complex person object classification whereas, larger [11×11] size is good for more complex pedestrian classification.
c) Too many filters will not always lead to better performance. However, more number of filters at deeper layer may produce better generalize the solution and provide better results.
d) Larger fully connected layer will help in better capturing the convolutional channel filters and produce better results for more complex objects.
e) Leaky ReLU can give better CNN classification performance. At lower values, larger scale may result in better performance. However, this needs to be investigated further.

IV. CONCLUSION

The effect of parametric changes in CNN for pedestrian classification is analysed. The results of using CNN for pedestrian detection is promising and provides 100 percent accuracy when tested with same dataset and upto 99.99 % accuracy when tested on reasonably visible and clear objects from a different dataset. However, for complex, noisy, or occluded objects better network architecture with best-representing training data needs to be used. In general, CNN itself is really promising for pedestrian detection. The bottleneck will be the computational requirements. When using the classifier as a detector, efficient methods to predict possible object locations needs to be used against standard sliding window approach for real-time performance.

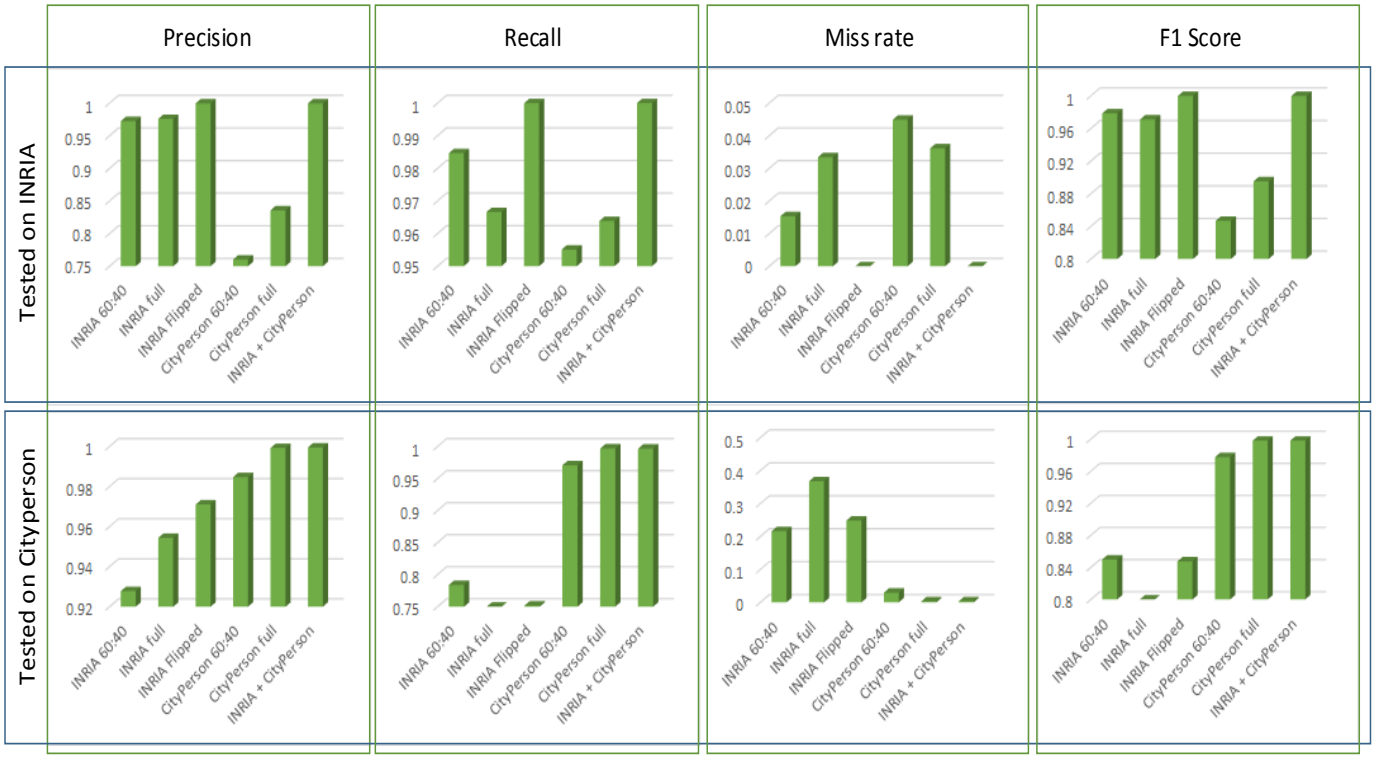

Fig. 2. Classification Performance against Datasets

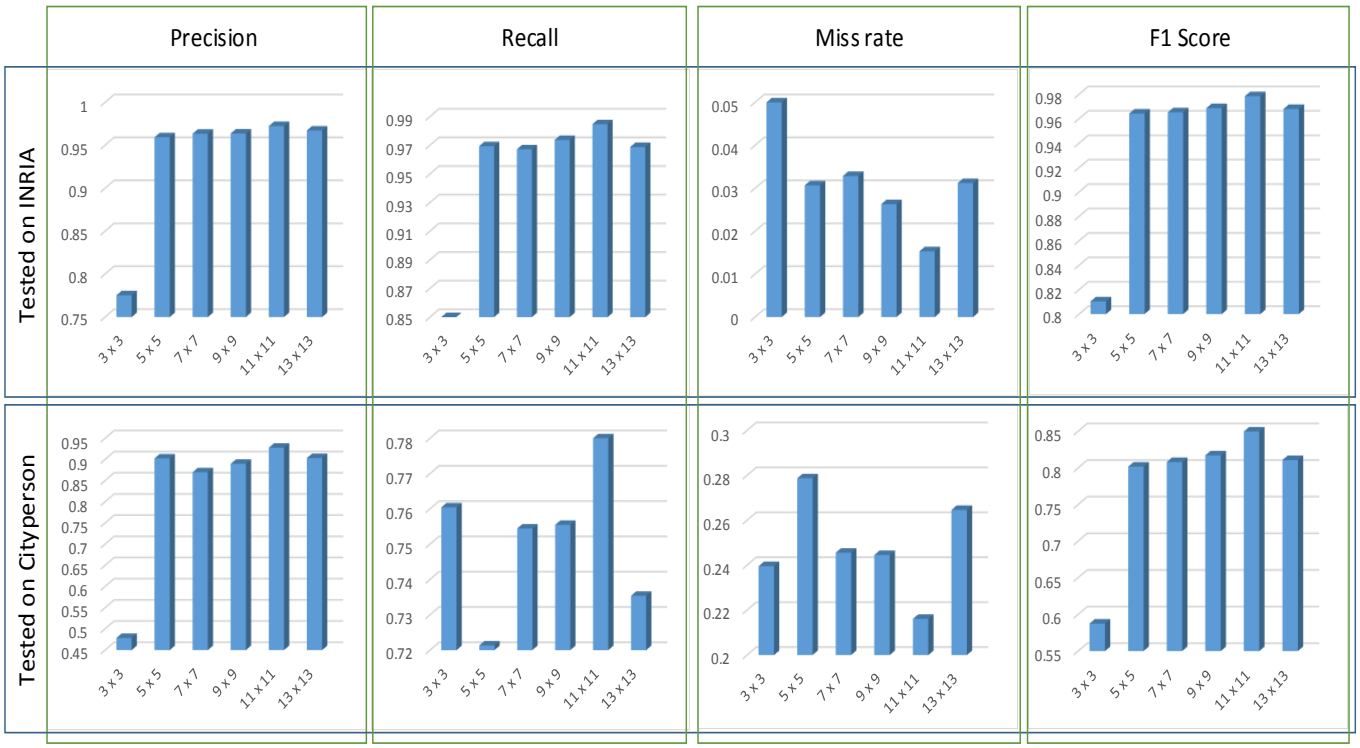

Fig. 3. Classification Performance against Filter Size. The CNN is trained with INRIA.

Fig. 4. Classification Performance against Filter Size. The CNN is trained with CityPersons

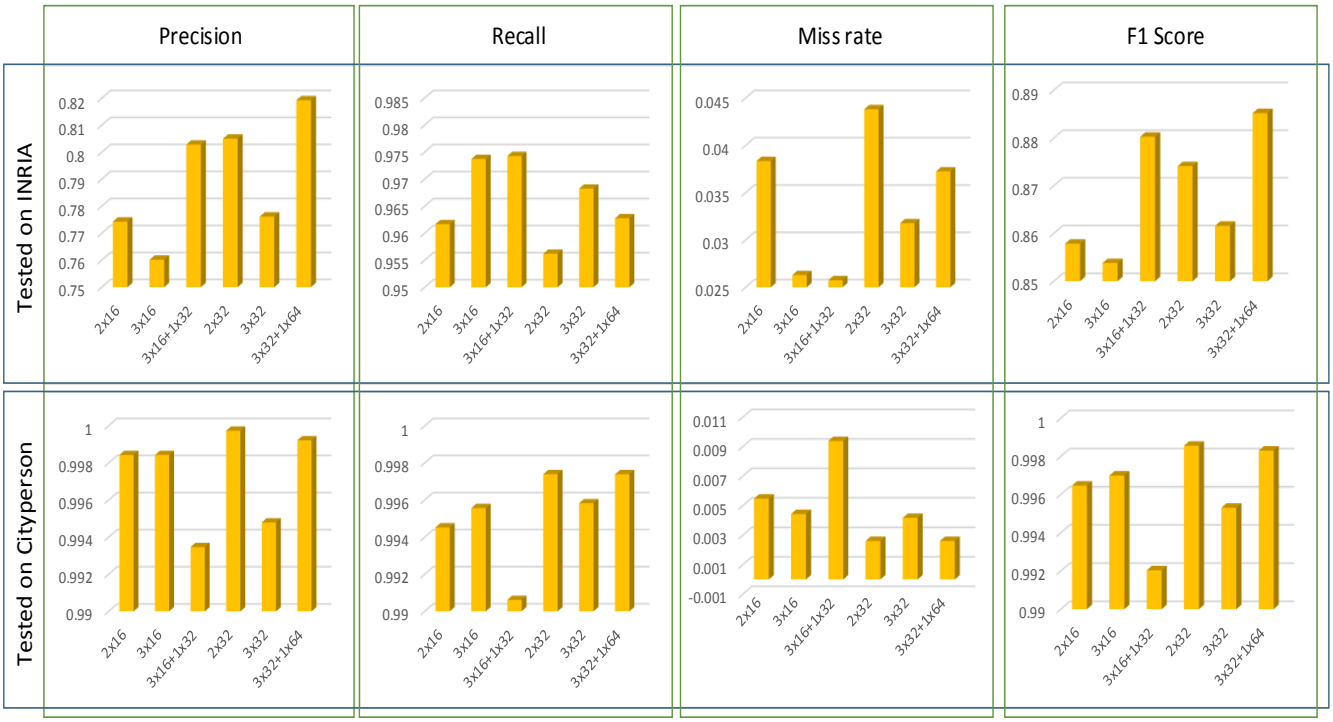

Fig. 5. Classification Performance against Number of Convolution Filters. CNN is trained with CityPerson dataset.

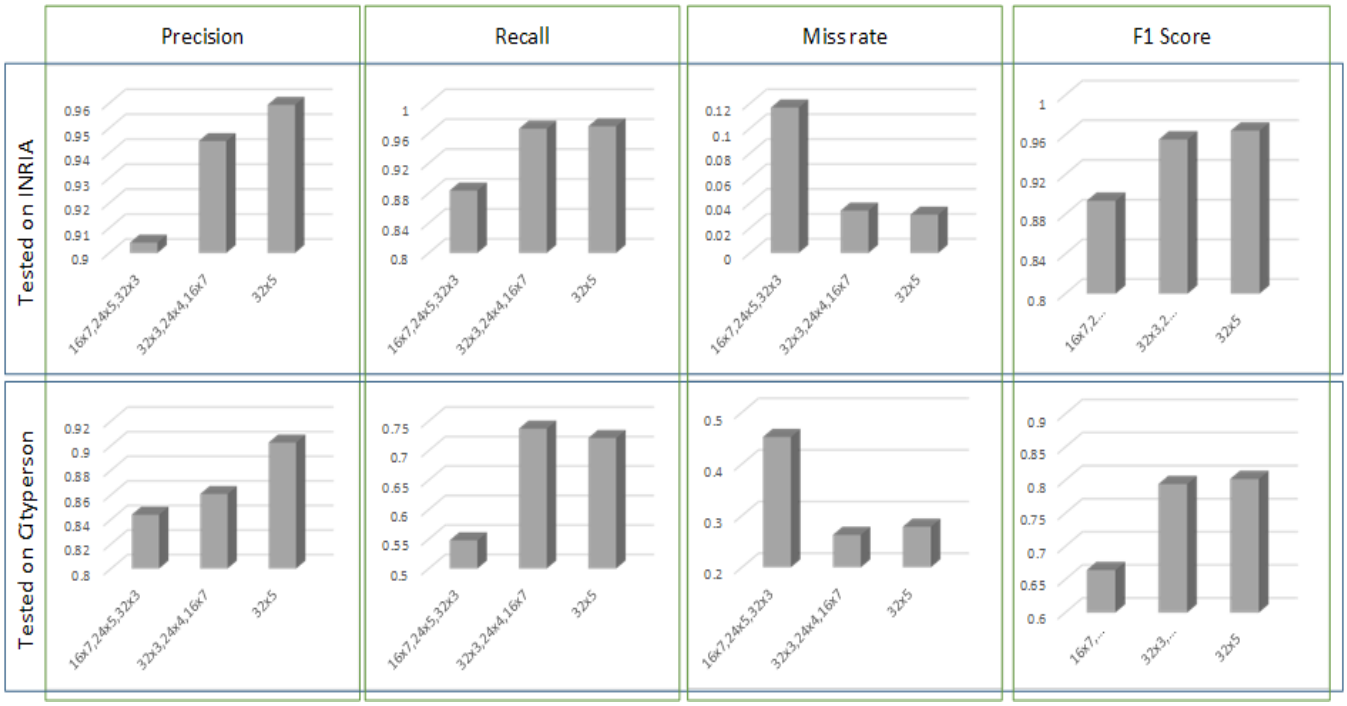

Fig. 6. Classification Performance against variable Filter size and number. CNN is trained using INRIA dataset

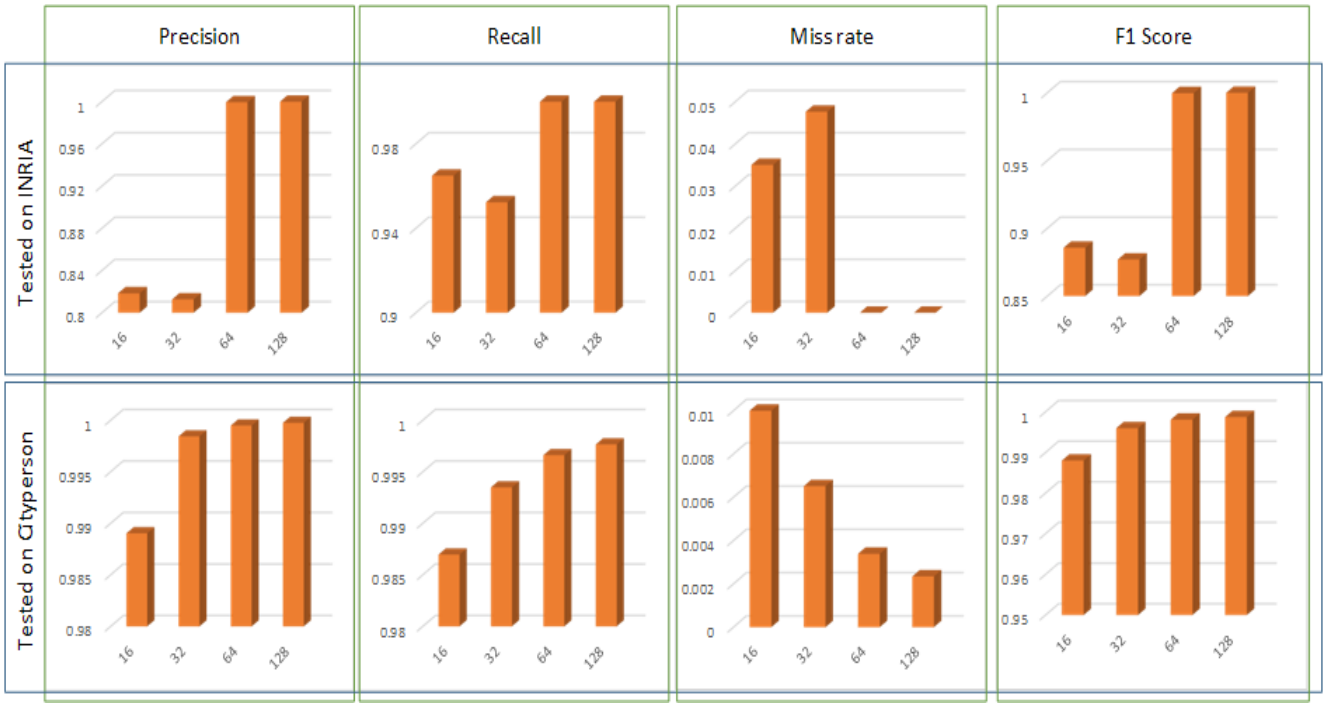

Fig. 7. Classification Performance against size of Fully connected layer. CNN is trained using City Person dataset.

.

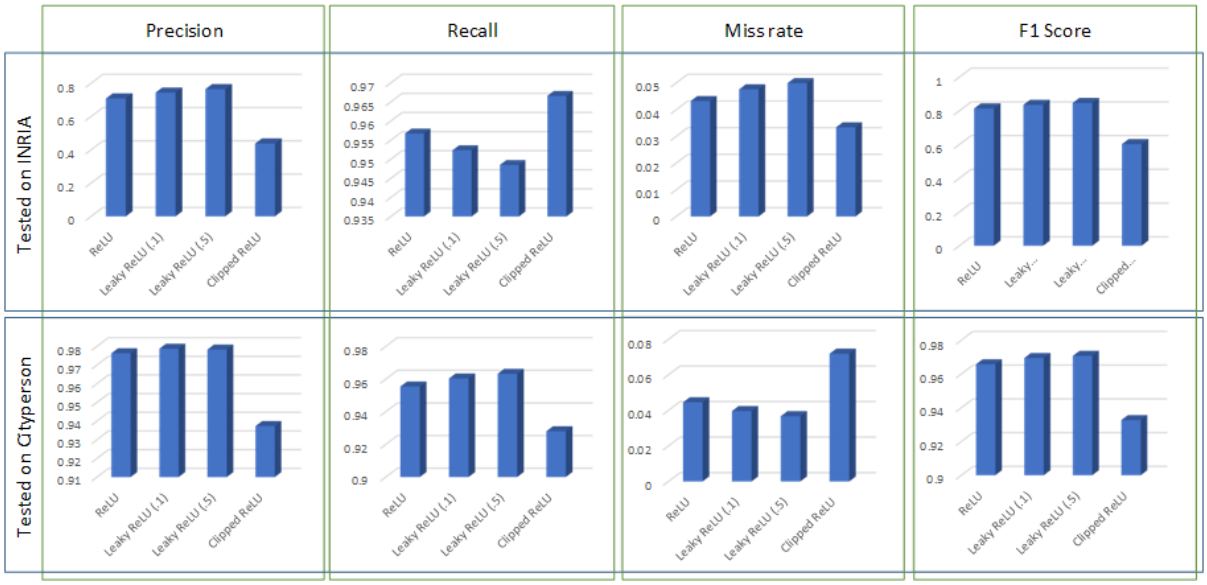

Fig. 8. Classification Performance against Activation functions. CNN is trained using INRIA dataset

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[2] A. Z. K. Simonyan, "Very deep convolutional networks for large-scale image recognition," in Computing Research Repository - arXiv:1409.1556, 2015.

[3] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," CoRR, vol. abs/1512.00567, 2015.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.

[5] S. Xie, R. B. Girshick, P. Doll´ar, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," CoRR, vol. abs/1611.05431, 2016.

[6] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, 2017.

[7] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," CoRR, vol. abs/1506.02640, 2015.

[8] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size," CoRR, vol. abs/1602.07360, 2016.

[9] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," CoRR, vol. abs/1511.00561, 2015.

[10] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in Proc. of the 27th International Conf. on Neural Information Processing Systems – Vol. 2, ser. NIPS'14, 2014, pp. 2672–2680.

[11] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun, "Pedestrian detection with unsupervised multistage feature learning," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR-2013), June 2012, pp. 3626–3633.

[12] B. Yang, J. Yan, Z. Lei, and S. Z. Li, "Convolutional channel features," in IEEE International Conference on Computer Vision (ICCV-2015), 2015, pp. 82–90.

[13] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR-2005), vol. 1, June 2005, pp. 886–893.

[14] S. Zhang, R. Benenson, and B. Schiele, "CityPersons: A diverse dataset for pedestrian detection," CoRR, vol. abs/1702.05693, 2017.

[15] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR-2016), vol. 20, June 2016, pp. 3213–3223.

[16] R. Shanmugamani, Deep Learning for Computer Vision: Expert Techniques to Train Advanced Neural Networks Using TensorFlow and Keras. Packt Publishing, 2018.