International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 06 Issue: 06 | June - 2022 Impact Factor: 7.185 ISSN: 2582-3930

Detecting Disguised Faces with Transfer Learning

Dr. A. Prabhu[1], Daphedari Kishan Prasad[2], Adabala Taraka Rama Venkata Sai Hanuman[3]and Abhinav Joel T[4]

[1] Associate Professor, Department of Computer Science and Engineering, CMR Technical Campus, Medchal, Telangana, India.

[2] UG Student, Department of Computer Science and Engineering, CMR Technical Campus, Medchal, Telangana, India.

[3] UG Student, Department of Computer Science and Engineering, CMR Technical Campus, Medchal, Telangana, India.

[4] UG Student, Department of Computer Science and Engineering, CMR Technical Campus, Medchal, Telangana, India.

ABSTRACT:

In general, a human being has the memory power due to which he/she will be able to remember whatever they have seen but as the technology is increasing the advancement is increasing in such a way that now computer is also able to recognize theX faces from its memory but in order to differentiate them, we need more advancement which leads to the development of Machine learning. Machine learning concepts developed by Arthur Manuel. There have been many techniques used over the past decade to determine the identity of a person's face, such as Eigenfaces and Principal Component Analysis (PCA), to Convolutional Neural Networks (CNN) to ensure the ability to recognize faces has become further and further. An approach to machine learning called transfer learning involves creating a model of the first training task, then testing it using the model. The difference between transfer learning and traditional machine learning is that translation involves using a pre-trained model in order to start a secondary task using the initial model. It is expected that this paper will contribute to the field of image classification by using Machine Learning algorithms to solve the problem. Transfer learning significantly improves the performance of VGG models Based on these results, we conclude that VGG Models are the best choice for recognizing faces using ImageNet weights

Key words: *Image, Memory, Machine Learning, Techniques, CNN models, Face Recognition, Deep Learning, Transfer Learnings*

1-INTRODUCTION:

Computer systems use algorithms and statistical models to automate task performance without being explicitly programmed through machine learning (ML). Almost every application we use daily uses learning algorithms. Google works so well because a learning algorithm is trained to rank web pages every time someone searches the internet with the help of a search engine. A variety of applications can be performed with these algorithms, including data mining, image processing, predictive analytics, etc. Its main benefit is that algorithms can learn to interpret data automatically once they learn what to do with it. An overview of machine learning algorithms and their future prospects is presented in this paper where we are using the transfer learning for recognizing face in disguise. Several methods for identifying people's faces have been developed over the last decade, including Eigenfaces and Principal Component Analysis (PCA), as well as Convolutional Neural Networks (CNN), and the capacity to recognize faces has improved dramatically. Transfer learning is a machine learning technique in which the first training problem produces a model, and then the second test is performed using the model from the first training assignment.[1] Transfer learning varies from typical machine learning in that it includes employing a pre-trained model to launch a secondary task. Using Keras, an open-source neural network library written in Python, we compare a few pre-trained CNN architectures. A total of six architectures were used: VGG16, VGG19, ResNet50, ResNet152 v2, InceptionV3, and Inception-ResNet V2. Based on the results of this research, we expected to see the best Pretrained Architecture model with the highest degree of accuracy, as well as the most cost-effective function within the optimal hyperparameters. First, we construct a deep learning framework for detecting 14 facial key-points, and then we utilize those points to identify disguised faces based on these key points. Because deep learning architectures require large annotated datasets for training, two annotated face key-points datasets are presented. For each key point, the efficiency of the facial main point detection method based is demonstrated. A comparison of the important detection method based with other deep networks demonstrates its advantages. When compared to state-of-the-art face disguise classification technique, the efficiency of classification performance is also proved.

2- System Analysis:

The critical phase of the system design is system analysis. The system is thoroughly investigated and assessed. The process of determining the optimum solution to a problem is known as analysis. The process of learning about existing problems, defining variables and requirements, and evaluating solutions is known as system analysis. It is a style of thinking about an organization and the difficulties it faces, as well as a combination of technologies that aid in the resolution of these issues. Feasibility studies are crucial in system analysis because they provide an objective for development and design.[2][1] We expected the finest Pretrained Architecture model with highest level of accuracy and the lowest cost function in the ideal hyperparameter state to emerge from this study. A data collection of 75 photographs of a person's face taken with a disguised tool such as a bandana, masker, fake moustache, fake beard, spectacles, and so on. Due to the countless changes that can be created using various disguises, disguised face identification (DFI) is an incredibly difficult subject. We provide a deep learning system that first detects 14 facial key-points before performing disguised face detection.

3- Existing system:

The PCA method is used in the present structure to recognize disguised faces. Face photos with considerable fluctuations in light direction and facial expression are better recognized by the PCA algorithm, which divides the face images into smaller sub-images. Each of these inter - and intra is subjected to the PCA method. Because certain of an individual's local facial features do not change regardless of stance, lighting direction, or facial emotion. Using typical face databases, the accuracy of a conventional PCA approach and the modular PCA method is assessed underneath the conditions of varied expression, illumination, and position. • PCA takes a long time to find the eigenvectors and eigenvalues. Each face image's size and location must be consistent. The PCA (Eigenface) technique maps features to major subspaces that contain the most energy. When a facial expression algorithm identifies a face in an image or an even now from a video recording, the face's relative size compared to the entire picture size influences how effectively the face is recognized. Principal Component Analysis (PCA) is an algorithm (PCA).

4- Proposed system:

We compare the common Pre-Trained CNN Model Architectures given by Keras, an opensource neural network toolkit written in Python, in the suggested system. VGG16, VGG19, ResNet50, and InceptionV3 were the architectures we employed. Then we split it into two parts: using the vector to train the classifier model and evaluating the accuracy and cost function of the classifier model. We expected seeing the best Pre - trained models Architecture model with the highest level of accuracy and the lowest cost function in the optimal hyperparameter state in this research. Transfer learning is when a machine learning model that has already been trained is used to solve a separate but related problem. For instance, if you trained a basic algorithm to classify whether a picture contains a bag, you'd be able to predict whether the image contains a backpack may apply the knowledge obtained during the model's training to distinguish additional objects, such as sunglasses.[3] Because the model has been pre-trained, a good machine learning model may be generated with relatively little training data via transfer learning.

• This is especially useful in natural language processing, where huge labelled datasets require a lot of expert knowledge.

• In addition, training time is lowered because training a deep convolutional neural network from start on a complex job can take days or weeks.

5- Requirements:

The logical properties of each interface between both the software system and the system's hardware components are defined by hardware interfaces. Some hardware requirements are listed below.

• Processor: i9

• Hard Drive: 500 GB

• Input Devices: Keyboard, Mouse

The logical properties of each interface and software component of the system are specified in Software Requirements. Some software requirements are listed here.

• Operating Systems: Windows 10/MacOS

• Programming Languages: Python, Django

• PyCharm CE is the programme you'll be using.

6- Proposed Methods:

Convolutional Neural Network (ConvNet):

Convolutional Neural Networks (CNN), often known as ConvNet, are a type of Artificial Neural Network (ANN) that is currently thought to be the best technique for solving object recognition and digit detection problems. [5]There are various models being built in a deep neural network like CNN, but we will only focus on three alternative model architectures in this research.

1) AlexNet is a website that connects people.

This architecture, which was created in 2012, is the first neural network that can accurately categories some objects in the Imagenet database, compared to standard approaches that existed before AlexNet. this network consists of 5 convolutional layers following 3 fully linked layers.

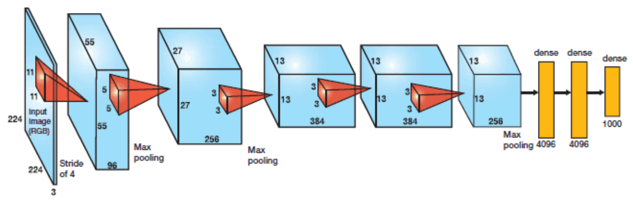


FIG -1 Network architecture of AlexNet.

2) VGG16

The VGG group, Oxford, designed this architecture in 2013. By substituting huge kernel filters (11 and 5 during the first and second convolutional layers, respectively) with some [5] 3x3 kernel filters, VGG was able to improve on the AlexNet design. Small-sized kernel that are layered are better than large-sized kernels for a given receptive field because numerous non-linear layers improve the depth of the network, allowing it to learn more complicated features.

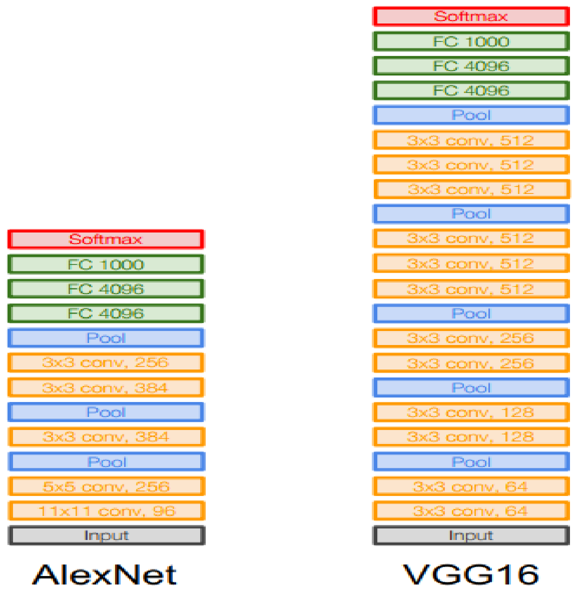


FIG – 2 Network architecture of VGG16.

3) ResNet

According to what has been mentioned thus far, in order to improve network accuracy, the layer depth must be increased as long as the network can continue to over-fit. However, just adding layers will not increase the depth of the network. Deep networks are challenging to implement because of the issue of vanishing gradients, which occurs when gradients are repeated to the preceding layer, resulting in a very small gradient.[5] As a consequence, as the network expands, performance gets saturated, if not rapidly degraded. ResNet (Residual Network) was founded in 2015 with the goal of introducing a "identity shortcut link" that passes via one or more levels.

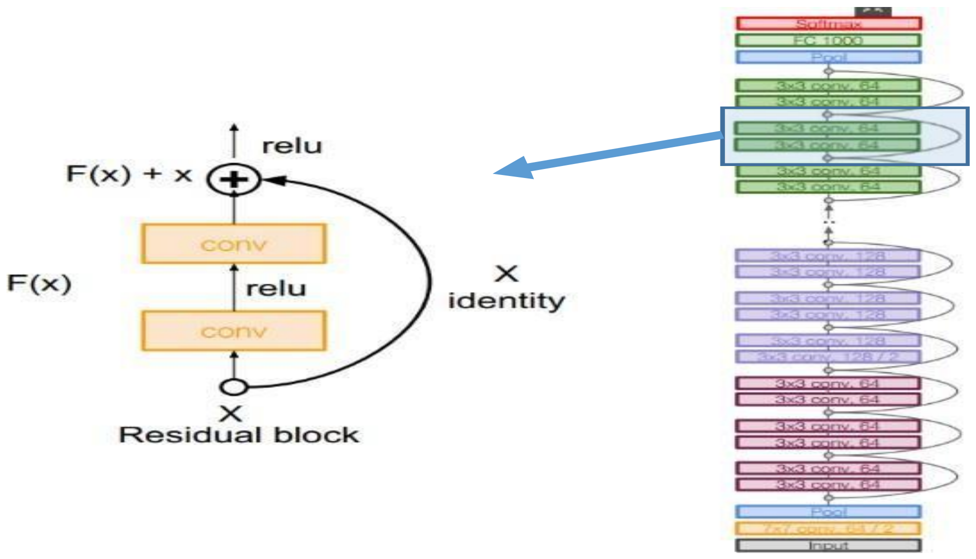
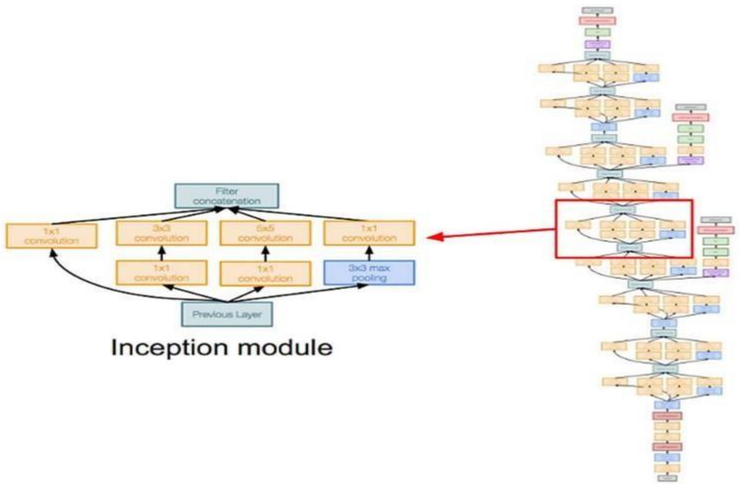


FIG -3 Network architecture of ResNet.

4) Inception V3

VGG attained exceptional precision in the ImageNet dataset, however its application, despite the GPU, necessitates a lot of work (Graphic Processing Unit). Due to the wide breadth of the convolution layers utilised, this has become inefficient. GoogLeNet is based on the premise that most deep network activations are either unnecessary (zero value) or excess due to the association between them. As a result, the most effective deep network architecture will have few connections among activations, meaning that none of the 512 output channels will be connected.[5] GoogLeNet created the Inception module, which was numbered in the manner of a thin CNN with a solid architecture. As previously stated, only a small fraction of a neurons are effective, hence the size of convolutional filters is limited.

FIG-4 Network architecture of GoogLeNet/Inception.



7- Architecture:

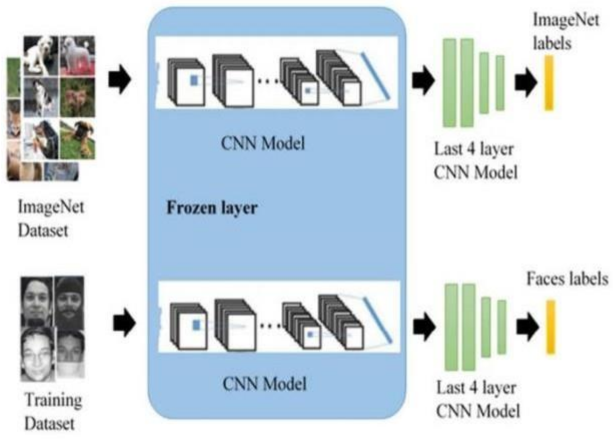


FIG- 5 ARCHITECURE REPRESENTATION

We set up a simple Sequence Model Architecture for the base layer, and then added the input layer to the VGG16, VGG19, ResNet50, ResNet152 v2, InceptionV3, and Inception-ResNet V2 Pre-trained Models we used in this study. Then, to our input model, we apply a pre-trained weight "ImageNet" that could be used for Transfer Learning. On our first attempt, we use VGG16 and proceed in the same manner with our other Input File.[6] We use 30 epochs for all trained models in each setup. And, with the exception of the InceptionV3 Model, each model receives a separate parameter modification after unfreezing the last four layers.

CNN Model	Freezing all layer (parameter)	Unfreezing last 4 layer (parameter)	CNN Model	Loss Value	Accuracy (%)	Validation Loss Value	Validation accuracy (%)
VGG16	8,466,507	15,545,931	VGG16	3.41	20.49	3.14	45.24
VGG19	8,466,507	15,545,931	VGG19	3.91	11.93	3.81	26.19
ResNet50	102,838,347	103,893,067	ResNet50	1.74	53.6	4.66	1.6
ResNet152 v2	102,838,347	103,893,067	ResNet152 v2	1.57	61.0	6.16	1.6
Inception v3	52,506,699	52,506,699	Inception v3	3.51	16.3	3.43	3.9
Inception-ResNet v2	100,741,195	103,937,611	Inception-ResNetv2	4.01	7.3	4.15	1.6

TABLE -1 TRAINABLE PARAMETER OF CNN MODEL

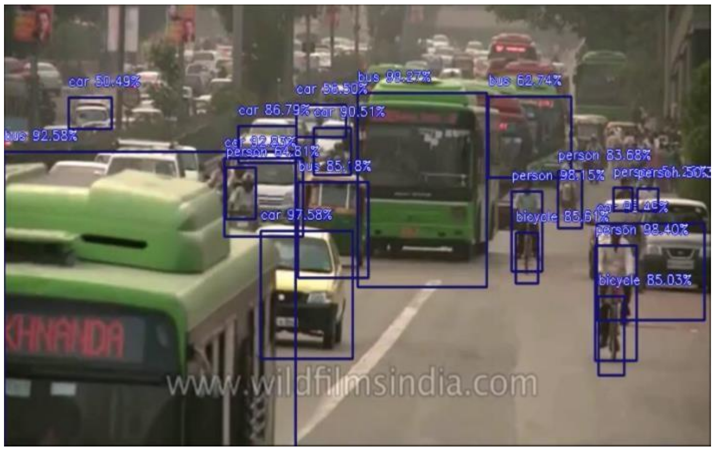
To avoid overfitting the model, we utilize a data pretreatment approach before training it. The technique entails downsizing all images to 224x224 pixels, as well as flipping and rotating them.

FIG -6 Output image after preprocessing

For training, we employ two setups: Freezing all layers in the CNN Model (setup 1) & Unfreezing the last four layers (setup 2). (setup 2). We are using the weight from the prior training to Cnn Architecture, which in our case comes from Keras trained in the ImageNet dataset, which has a lot of annotated images. The setting 2 means that we froze all layers but the last four, and then train the last four to improve accuracy on the dataset we used. This technique is known as Transfer Learning. In the CNN Model, all layers are frozen (Setup 1), We want to know the initial given impact for applying pre-trained weight on CNN Model, which is ImageNet, in this training configuration. In the CNN Model, unfreeze the final four layers (setup 2)In this training scheme, we first freeze all levels but the last four to our Pre-Trained CNN Model, which implies we only train the CNN Model's last four layers. We apply this to the entire Model, regardless of layer type. Then we have to re-fit our model.

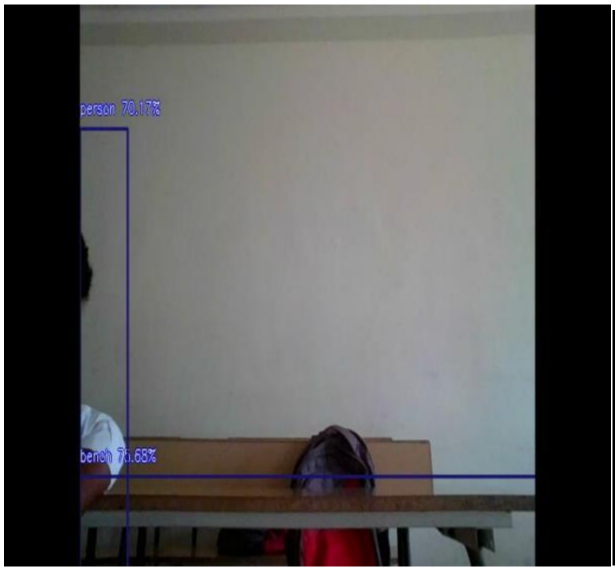
8- Results:

8.1 - OUTPUT - 1



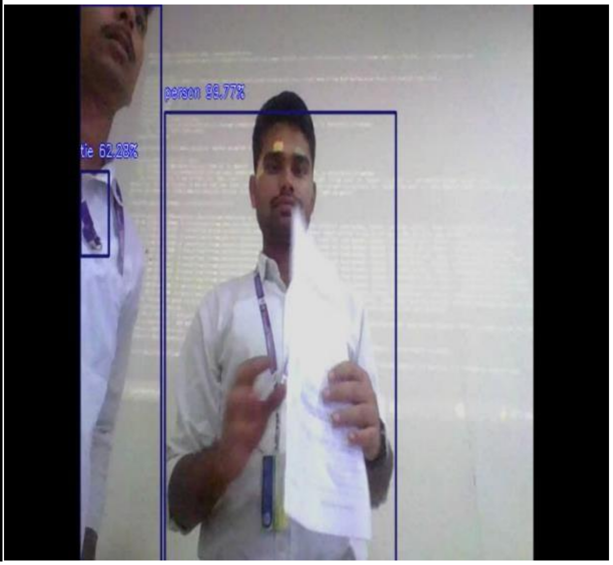
Screenshot 8.1: Output-1

8.2 OUTPUT – 2



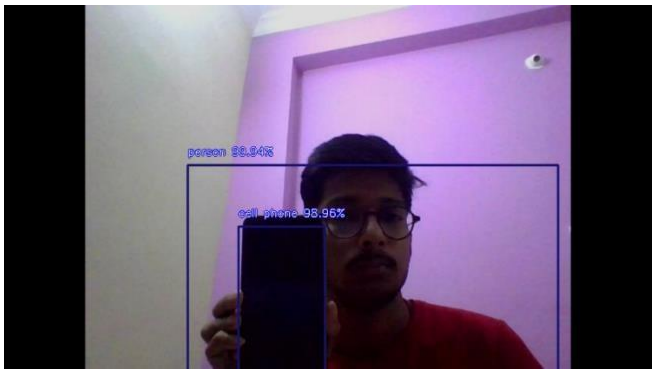
Screenshot 8.2: Ouput-2

8.3 OUTPUT-3



Screenshot 8.3: Output-3

8.4 OUTPUT-4



Screenshot 8.4: Output-4

9- Testing and Test case used:

Unit testing is designed to ensure that each path of a business process performs accurately. Integration tests demonstrate that although the components were individually satisfactory[8], as shown by successfully unit testing, the combination of components is correct and consistent.

Functional testing is centered on the following items:

Valid Input	:	Identified classes of valid input must be accepted.
Output	:	Identified classes of application outputs must be exercised.

S.no	Test Case	Excepted Result	Result	Remarks(IF Fails)
1.	User Has to locate the image path	User can keep training images, testing images and validation images under data folder	Pass	If images not available then failed
2.	Start the system cameras or pre video files	Detect the objects from given video files	Pass	CNN Model required
3.	Detected object play again the video	All frames playing again.	Pass	Video file .avi format required
4.	Create object of VGG16 Model	VGG16 Model object creates and process.	Pass	Vgg16 has to install in the system
5.	Generate the h5 weights files	Loaded Model .H5 file will generate and stores in system drive	Pass	If model not trained the failed
6.	Load VGG19 Model	VGG19 Model loaded	Pass	Vgg16 Model ahs to Load
7.	Load the ResNet50 Models	ResNet50 Model loaded and object created	Pass	ResNet has to install.
8.	Load InceptionV3 model	Inception V3 Model has to load	Pass	First Inception V3 Model has to load
9.	Test user images	User has to test its own images	Pass	To test created models weight required
10.	Confusion Matrix generated	Confusion Matrix for Vgg16 and Vgg19 Models	Pass	If model not trained the failed

10- Conclusion and Future Scope:

We provide a comparison of six common CNN Models for detecting a disguised person's face using the "Recognizing Disguised Faces" datasets in this project, and the outcomes show how Transfer Learning may be employed in the Face Verification problem. The VGG model has a balanced accuracy of training and validation in the training set, whereas the ResNet152 v2 model has a greater accuracy than VGG in the train set.The Convolutional Neural Network's success is also one of the reasons why Deep Learning CNN has become such a popular topic in recent years. We intend to encode and include the idea of familiarity in automated algorithms as a future research topic, which may increase performance. Furthermore, we anticipate that studying how masking unique facial features affects face presentations may lead to improved strategies to attenuate these differences.

Reference:

[1] A. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," IBM Journal of Research and Development, vol. 3, no. 3, p. 210–229, 1959.

[2] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, p. pp. 586–591, 1991.

[3] J. Yang, D. Zhang, A. F. Frangi and J. Yang, "Two dimensional PCA : a new approach to appearance-based face representation and recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 1, p. 131–137, 2004.

[4] Y. Taigman, M. Yang, M. Ranzato and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," CVPR, p. 1701–1708, 2014..

[5] J. West, D. Ventura and S. Warnick, "A Theoretical Foundation for Inductive Transfer," Spring Research Presentation, 2007.

[6] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama and a. T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in ACM MM, 2014.

[7] D. Tomé, F. Monti, L. Baroffio, L. Bondi, M. Tagliasacchi and S. Tubaro, "Deep Convolutional Neural Networks for pedestrian detection," Signal Processing: Image Communication, vol. 47, pp. 482-489, 2016.

[8] Z. Deng, H. Sun, S. Zhou, J. Zhao and H. Z. Lin Lei, "Multi-scale object detection in remote sensing imagery with convolutional neural networks," ISPRS Journal of Photogrammetry and Remote Sensing, vol. 145, no. Part A, pp. 3-22, 2018.