Dataset Pre-Processing and Artificial Augmentation, Network Architecture and Training Parameters used in Appropriate Training of Convolutional Neural Networks for Classification Based Computer Vision Applications: A Survey

Abhishek Kumar Annamraju

Alumni at Department of EEE, Bits Pilani KK birla Goa Campus, Goa, India

Abstract— *Training a Convolutional Neural Network (CNN) based classifier is dependent on a large number of factors. These factors involve tasks such as aggregation of apt dataset, arriving at a suitable CNN network, processing of the dataset, and selecting the training parameters to arrive at the desired classification results. This review includes pre-processing techniques and dataset augmentation techniques used in various CNN based classification researches. In many classification problems, it is usually observed that the quality of dataset is responsible for proper training of CNN network, and this quality is judged on the basis of variations in data for every class. It is not usual to find such a pre-made dataset due to many natural concerns. Also it is recommended to have a large dataset, which is again not usually made available directly as a dataset. In some cases, the noise present in the dataset may not prove useful for training, while in others, researchers prefer to add noise to certain images to make the network less vulnerable to unwanted variations. Hence, researchers use artificial digital imaging techniques to derive variations in the dataset and clear or add noise. Thus, the presented paper accumulates state-of-the-art works that used the pre-processing and artificial augmentation of dataset before training. The next part to data augmentation is training, which includes proper selection of several parameters and a suitable CNN architecture. This paper also includes such network characteristics, dataset characteristics and training methodologies used in biomedical imaging, vision modules of autonomous driverless cars, and a few general vision based applications.*

Keywords— artificial dataset augmentation, autonomous vehicle navigation, biomedical image processing, convolutional neural networks, pre-processing.

I. INTRODUCTION

Convolutional Neural Networks are being extensively used in computer vision applications such as pedestrian detection, face recognition, biometric applications, biomedical imaging, etc. Training a classifier based on CNN for object detection and localization is a complex process. Proper understanding with regards to the steps like selection of dataset, pre-processing of dataset and artificial augmentations to the dataset is extremely important. When creating the network, the network structure, layer alignment in the network, and branching of layers in the network is important too. Once the network structure is decided, the parameters in training such as learning rate, momentum value, weight initializers, weight decay rate, number of iterations, dynamic alterations to learning rate, etc., play a very crucial role in proper training of the classifier. Without prerequisite knowledge over these factors, the trained network may succumb to overfitting, over-false positives generation in classifications etc.

To give a keen understanding as to how researchers select these factors, a large set of prominent researches in this field were studied and a detail review is presented. The following paper includes different pre-processing techniques used to remove noise and add variations to the dataset, in section 2. The next section, section 3, presents algorithms used to artificially augment the dataset. This augmentation is specifically done to avoid the problem of overfitting. Section 4, describes the network architecture and the training parameters used with respect to researches in biomedical imaging applications involving

CNNs. Similarly, Section 5 and Section 6 describe these factors when the applications are related to autonomous vehicle navigations and other vision based applications in general, respectively. Section 7 concludes the paper.

II. PRE-PROCESSING

Image pre-processing [1] is a very important part of computer vision applications. It is required before image-processing techniques are applied over the images for analysis. It involves geometry based corrections, like resizing, cropping, resampling [2], correcting geometric-distortions [3], etc. On the other hand it also includes techniques like image enhancements, contrast and brightness manipulation and kernel based filtering, etc. The authors in [4] applied a contrast-extracting layer [5] that resulted in increase in the recognition rate. Local Contrast Normalization was used in [6] and [7] before feeding the images into the network. Another normalization technique mentioned in [7] was ZCA whitening [8], it is an algorithm which makes local patches to have zero-covariance. The effect of ZCA whitening was completely discussed in [9] as this pre-processing step is highly accoladed for removing redundancies in training images. In [10], the input image was normalized using local divisive normalization [11]. Authors in [12] subtracted a constant value, 128, from image pixels and then used the absolute values of the result. In [13][14][15][16][17][18][19], the input training images were normalized to have zero mean and unit variance.

The authors in [20][21] experimented using Contrast Limited Adaptive Histogram Equalization (CLAHE) [22], low-pass filtering(LPF) and high-pass filtering(HPF) and concluded that contrast equalization and noise removal using LPF resulted in increase in classification accuracy. Authors in [23] made use of the color representations, instead of the usual Red, Green and Blue Channels, they incorporated L from LAB color space, L and U from LUV color space and blue ratio. Blue ratio as explained in [23] is ratio of blue value to the sum of green and red values for every pixel. In one of their other approaches they went with image contrast enhancement [24]. In a similar way, authors in [25] converted image originally in RGB color space to HSV space. In [26] image enhancement filters were applied to the lesions in the image to get improved sensitivity and specificity in the pixel based segmentation applications mentioned in the paper.

Authors in [27] used color enhancement and hybrid patching to denoise the underwater images being used. In hybrid patching, same patch was cropped from three different scales of the image, rescaled the patches to the biggest one to ensure randomization and blurring effects in training patches. In [28], the normalization was done with mean set as the average pixel value of the image and image norm as 100. Authors in [21], experimented on three kinds of normalization, normalizing the three channels to mean around average pixel intensity and standard deviation of |1|, second, normalizing the three channels to mean around average pixel intensity and standard deviation of |2| and final, CLAHE. In [29] images were pre-processed using lightning correction and histogram equalization. One the most common methods of pre-processing used was subtracting the mean image of the training set from the dataset [30]. Authors in [31] classified pre-processing into two parts, pixel level and feature level. In pixel level they applied image deblurring, edge detection and image enhancement whereas in feature level it was image restoration. The paper mainly targeted the influence of pre-processing techniques in CNNs. Three different pre-processing techniques were applied in [32], first, histogram equalization, second, scaling intensities between 0 and 1, and third, normalizing the data to zero mean and unit standard deviation. To remove the correlations between raw pixels, authors in [33] first converted the image RGB space to YUV space and applied zero mean and unit invariance normalization.

Authors in [34] claimed to arrive at their best results when normalization was done by rescaling the image between 0 and 1 and converting the RGB space to gray color space. Authors in [35] used eight pre-processing algorithms, and fed an eight channel pre-processed input to the network. The first three maps were ZCA whitened color channels, and the next five were HOG180 [35] features. In [36] normalization was applied in the sense that the pixels are scaled between 0 and 1 with respect to the minimum and maximum pixel values. In [37], the only pre-processing done was to convert RGB image to grayscale color space. Authors in [38] converted the images from RGB color space to YUV space in order to ensure that the brightness and color properties are separated. Herein, Normalization in the Y channel was local and over the other two channels it was implemented globally. The normalization criteria were zero mean and unit variance. A three step patch pre-processing was applied in [39], which included histogram equalization, linear lighting removal [40] and intensity normalization to unit variance.

III. ARTIFICIAL DATASET AUGMENTATION

The size and quality of image training dataset has many constrained restrictions like, lack of proper resources to capture data, very specific data available online, or limited time constraints in obtaining the data. This makes the data very inclined towards limited variations available. For a classifier to generalize effectively, it must

be provided with different variants of possible scenarios/data it is expected to learn. To comply with this issue, the training datasets are usually augmented artificially using geometric, affine, and perspective transformations, aided with filtering, blurring, smoothing, and color-space conversion type algorithms. Authors in [41][42][23][43][44][45][18] augmented the dataset by mirroring and randomly cropping patches from the training dataset after applying different amounts of padding. Authors in [46] used affine and geometric transformations to augment the dataset with more images. In [14] instead of cropping, flipping was used with mirroring.

In [4] it was said that the error rates decreased when translation, rotation, shearing, scaling and elastic distortions were used for the 2D training images. However it was also mentioned that the augmentation had less effect when the training set consisted of 3D data. A random translation of about 50 pixels in either x or y direction accompanied with random degree of positive and negative rotation was applied to the dataset in [47] and the augmented data was resized to a size appropriate for the network. The error rate decreased by almost 4% when the training set was augmented in [7] with random translations and applying horizontal reflection transformations. Translation and rotation were also applied by authors in [48] to increase the variants of images in the training dataset. Authors in [49] augmented the data by adding two datasets, the original had 1000 classes and they added another dataset with 512 classes resembling the ones in the original set and the performance of the classifier was improved. Authors in [20] focused on the importance of training data augmentation, as in their approach, it resulted in over 50% decrement in error rate. They have used random cropping with five translations over the cropped patches, followed by mirroring of the translated data. Random gaussian noise was added to images and PCA transformation was applied to the training dataset in [50], before giving it to the network. In [51], translations by +-10 pixels in both x, y directions, rotation by +-10 degrees and scaling (to 1.4x zoom in 8 steps) were applied to increase the dataset for better generalization which improved the quality of the classifiers.

For the 3D convolution neural network [52], rotation was applied to the 3D dataset by a full 360 degree in intervals of 30 degrees, resulting in 12 images. In [53], authors through experimentation showed that using augmentations caused the test error rate to drop from 24% to 20%. Similarly by adding random translations applied to training images in [54], the error rate dropped to 20% from 28% and by adding rotations it further decreased to 17%. Authors in [55], with the translations and horizontal flipping, the RGB channels values were manipulated to get different contrast and brightness variants of the same image. Three approaches were adopted in [32] for augmenting the training dataset, first by artificially applying affine and geometric transformations on the image, then by adding more images from other datasets with similar type and finally by adding contrast based distortions and intensity based variations on the original data and this reduced the testing error by almost 5%. Horizontal flips, rotations, scaling, brightness and contrast modifications were applied on the images in [56] to increase the dataset.

In [57], the image dataset was increased 8 times the original set, by using random translations, affine transformations, rotations in both clockwise and counter clockwise. A training dataset of around 100000 images in [39] was increased upto 2M using flipping, translations, rotations, scaling aided with zooming. Similarly, the set containing 30k negative images was rolled upto 10 billion images. Augmentations were applied to the dataset used in [58] with intensity variations by multiplying the set with a constant values 0.8, 0.9, 1.1, 1.2. Translations were also applied in both x and y directions by shifting the image with -2, -1, 1, 2 pixel strides. The images were scaled by values 0.93 and 1.05 and cropping was applied for the upscaled ones.

IV. CNNs IN BIOMDEICAL IMAGE PROCESSING

Authors in [32] used CNN for classification of Human Epithelial Type 2 Cells. The dataset used was ICIPR2013 [59]. This dataset contained images of 1455 cells subjected to 9 classes. The architecture used in had three convolution layers, each followed by a pooling layer of stride and size 2. The first pooling layer was pooled using maximum value from the window of fixed size, whereas the others took the average. The first two convolution layers had 5x5 sized 32 kernels and the third one had 64 filters. The final pooling layer was connected to a Fully Connected (FC) output layer with 8 neurons and a dropout layer in between the output and last pooling layer. The accuracy achieved was about 80.25%. The architecture was pre-trained on a large image dataset and then was used to train the ICIP2014 dataset. A set of 721 images were artificially augmented and used for the training. To avoid overfitting a weight decay of 0.0004 was applied in the training. The pre-training was done on a dataset with 120000 iterations and the network was learnt on the dataset for 12000 iterations with a fixed learning rate of 0.001. A 64x64 sized image was given to the architecture. The larger source dataset had 472080 training images obtained after augmentations. In the

confusion matrix, a larger set of fine speckled class was classified as homogeneous class cells.

Diagnosis of multiple Sclerosis [60] based on MRI scans was addressed by authors in [61] using CNN. The dataset used had data from 150 patients and they were characterized into two classes, having the diseases, and normal conditioned. The deep architecture used here had a set of four convolution layers each followed by a pooling layer with stride 2 and kernel size 2x2 and a hyperbolic tangent activation layer. An input of 240x180 sized image was given to the network. The first convolution layer had 6 7x7 sized filters, the next one had 16 8x8 sized filters. The third layer had 30 8x7 sized filters and the final one had 50 filters of size 7x8. The fourth pooling layer was connected to a hidden FC layer with 120 neurons and it being finally connected to an output layer with 2 neurons. The images were resized to 256x256 before giving it to the training. The learning rate was fixed to a value of 0.005 for the entire 100 epochs of the training. Herein, the batch size used was 20 patches.

Authors in [62] used CNN for clustering biomedical images. They used architecture with two convolution layers each followed by a 2x2 sized pooling layer. Each kernel, in both the convolution layers, was sized 15x15, with 20 filters in the first layer and 50 in the second. The output of the second pooling layer was connected to a FC layer with 500 neurons and this in turn was connected to a logistic regression classifier. A set of 128x128 sized patches were extracted from original image of resolution 2492x1984.The batch size used in training is 128, and a total of 212 batches were used. Every pixel was assigned a label based on the surrounding pixels in that patch. The authors gave segmented ground truths for the training and tested four different configurations of CNNs and selected the best one out of them. A CNN based methodology was used in [16] for predicting semantic descriptors from biomedical images. The dataset compiled had 157 SD-OCT [63] volumes of the retina. These had size of 512x128x1024. A two-branched network was used for extracting out semantic details. The first branch took 35x35 sized patches from the input voxel while the second used 71x71 sized patches. The first branch had three convolution layers with 2x2 pooling layers. The first convolution layer here had 32 filters of size 6x6, the next one had 64 filters with size 4x4 and the final had 3x3 sized 128 channels. In the second branch there were two convolution layers with 2x2 sized pooling layers. The first convolution layer had 8x8 sized 32 filters and the second one had 64 with 5x5 size. The output of these branches was stacked into a FC layer with 2048 neurons. This was then connected to another hidden FC layer with 64 neurons which in turn was followed by the output layer. For segmenting out the required regions of the image slides, non-overlapping image patches of size 81x81 were taken out for training and testing. The ground truth patches were manually annotated. The number of patches for training was 82883 and for validation the number was 31352.

Authors in [64] used CNN for retina vessel segmentations. The training and testing dataset was chosen from the DRIVE dataset [65]. The architecture had four convolution layers, each followed by a max pooling layer of kernel size 2x2 and a stride of 2 px. Each convolution layer had 48 kernels. The kernel sizes of the convolution layers were 6x6, 5x5, 4x4, and 2x2, respectively. The final pooling layer was connected to a hidden FC layer with 100 neurons, followed by an output FC layer consisting of 2 neurons. An application for segmentation of bone structures from X-ray images were presented in [66] which used CNN for pixel based classification. The solution to automatic detection of invasive ductal carcinoma was discussed by the authors in [35]. The solution obtained was based on CNNs. The training dataset consisted of data from around 162 patients. Two convolution layers were included in the architecture presented in [35], the first having 16 filters and the second having 32 filters with each having filter kernel size of 8x8 window. An average pooling layer was used after every convolution layer, each having a stride of 2px and kernel size of 2x2. The final pooling layer was connected to a FC layer with 128 neurons and in turn was linked to the output layer with 2 neurons. The classifier used is log-softmax [67].

Computational mammography was addressed by the authors in [58] using CNNs. A CNN was used to classify regions into semantically coherent tissues. Herein, authors used three convolution layers, each accompanied by a max pooling layer of kernel 3x3 and stride 2, and an rectified linear unit (ReLU) activation layer. Each convolution layer had 16 filters of sizes 7x7, 5x5, 5x5 respectively. The final pooling layer was connected to FC layer having 128 neurons with a bridge of the dropout layer with probability parameter 0.5. This FC layer was connected to another hidden layer with 16 neurons and finally to an output FC layer with 4 neurons. Approximately 8000000 patches were extracted out to train the classifier to predict one of the four mentioned classes. Stochastic gradient descent [68] algorithm was used to train the classifier with a batch size of 256 pixels and initial learning rate of 0.001. At 30000 and 60000 iterations the learning rate was dropped by a factor of 10. The momentum parameter was set to 0.9 and weight decay was assigned with a value of 0.0005. The weights of the filters were initialized using a normal distribution, having a mean 0 and 0.01 as the variance for the

convolution layers and mean 0 and 0.1 as the variance for the FC layers.

Authors in [69] used CNN for counting bacterial colony present on the microbiological cultural plates. A dataset was compiled with around 17000 RGB color-space images. These images were to be classified into 7 colonies depending upon the number and type of colonies in the image. The architecture took an input image of size 128x128 and each of the three color channels. The network had 4 convolution layers with first two having filters of size 5x5 and the last two having filter of size 4x4. The numbers of filters in these layers were 20, 50, 100, and 200 respectively. These layers were accompanied with 2x2 kernel sized pooling layers with stride of 2 px and a ReLU activation layer. The fourth pooling layer was followed by a hidden FC layer with 500 neurons and then accompanied an output FC layer with 7 neurons. Before both the FC layers a dropout layer was introduced. The network weights were initialized using a Xavier distribution. Herein, stochastic gradient descent optimization solver was used on a batch size of 64. The weight decay parameter was set to 0.0005 and the momentum parameter to 0.9. Initial learning rate was set as 0.01 and was decreased by 0.01% of the current learning rate after every iteration. A total of 500000 iterations took around 3 hours on an Nvidia Titan

Black GPU. The task of cell counting in microscopic images was addressed by the authors in [70] using CNN. Authors presented a very unique network to segment out the cells from the image. Two different networks were presented in the paper. In the first network, three convolution layers, one FC layer and three deconvolution layers were involved. The activation function used with the layers is the rectified linear units function. The convolution layers were followed by max pooling layers of kernel size 2x2 and stride 2. The deconvolution layers had parameters in such way that the outermost layer resembled the first convolution layer and so on. The convolution layers had 32, 64 and 128 fillters respectively with filter sizes as 3x3 for each of them. The second network had four convolution layers, one Fc layer and one deconvolution layer. The deconvolution layers in both the networks were preceded by an upsampling layer. The convolution layers had 32, 64, 128, 256 filters with the first three having kernel size of 3x3 and the final one having 5x5 sized filter kernels. The FC layer had 256 neurons and the deconvolution layer parameters were same as that of the fourth convolution layer. The authors found that the first network produced better results than the second. A set of 500 patches overlapping of size 100x100 were extracted out from a 500x500 image. The learning rate was initialized at 0.01 and was reduced by a factor of 10 every 5 epochs, with weight decay parameter as 0.0005 and momentum as 0.9.

V. CNNs IN AUTONOMOUS VEHICLE NAVIGATION MODULES

In [38], the background data was also incorporated into a set of classes to label the scenes with the objects. The datasets used were Stanford Background [71] and the SIFT flow [72]. The former had 9 scene classes ranged over 715 images of size 320x240. The later, had, 33 object classes in 288 images of size 256x256. They created a small architecture, trained it weakly and then added it parallely to a bigger architecture. The smaller net had three convolution layers, each, except the last one, followed by a TanH type activation function layer and a 2x2 pooling layer with stride as 2. The convolution layers had a size of 7x7 with 16,9 and 64 filters respectively. The final convolution layer was connected to a FC layer with 9 neurons. The bigger net was not trained separately as the weak one. The bigger net's first two layers had similar properties except the second convolution layer, it had 55 filters. The output of the second layer from both the networks was concatenated to get a 64 channelled 7x7 features. This was passed through a final convolution layer and was converted to a FC layer with 9 neurons. This weak classifier was trained with all the classes and this classifier is then incorporated into a bigger architecture. The weak classifier was trained specifically to differentiate the context-scene labels and the augmented net is for both the scene as well as the objects. Of all the images from the Stanford Background dataset, a total of 40 M patches were extracted each of size 46x46, while from the object database (SIFT FLOW) a total of 140 M patches were extracted.

A weakly supervised object segmentation method was introduced in [56] using CNNs. The CNN was applied over two classes of objects, namely dog and cat, from the ImageNet [73] dataset, and the third class being the background. The network had three convolution layers, followed by two FC layers. Each convolution layer, except the third one, was followed by a pooling layer of size 4x4 and stride of 4 px and the activation function used is hyperbolic tangent. The hidden FC layer had 320 neurons and the final layer had 2. The first convolution layer had 40 kernels of size 8x, the second one had 80 filters of size 3x3 and the final one 160 filters of size 5x5. The second architecture presented in the paper had similar architecture with slight changes in the kernel size and the number of kernels in the convolution layers which they used on Pascal VOC 2012 dataset [74]. From each image 9 patches of size 100x100 are extracted out. The extracted patches are then normalized to fit the range [-1, 1]. They used the softmax classifier to arrive at the pixel based

inference, and then club the super pixels in post processing.

In [75], a convolution network was presented for classification of objects and scenes separately. For scene classification, Places dataset [76] was used; it had 2.5M images of 205 different classes of scenes. A very deep architecture was presented with five convolution layers. The first two convolution layers were followed by a local contrast normalization layer and a pooling layer with stride 2 and kernel size of 2x2. The first convolution layer had 96 filters and kernel size of 7x7, and the second one had 256 filters with kernel size of 5x5. The next three convolution layers had 3x3 sized kernels and 512 kernels. The fifth convolution layer was connected to a FC layer with 4096 neurons and this was in turn connected to another hidden FC layer with 2048 neurons. Each hidden layer was followed by a dropout layer with probability 0.5. This final layer was connected to the output layer. A pre-training methodology was used for recognizing objects and scenes. The CNN for object was trained on the ImageNet dataset, and for the scene classification it was trained on Places Dataset. A batch size of 256 images was used for training and the momentum value was set to 0.9. A layer specific learning rate scheme was deployed, the learning rate of the hidden layers was set to be 0.01 times of the output layer. Initial learning rate was set to 0.01 and was reduced to 0.001 after 1.4K iterations and then to 0.0004 after 2.8K iterations and was kept fixed till the training was stopped after 2.8K iterations.

CNN was used in [17] for eye detection to estimating remote gaze. Authors claimed to achieve a detection rate of 100% with a false alarm rate of 0.000265%. The architecture presented in [17] had two convolution layer each followed by a subsampling layer. The first convolution layer had 4 filter maps of size 5x5 and the second one had 15 maps of size 3x3 each. The second subsampling layer was connected to 15 neuron FC layer which was finally connected with an output GC layer having 2 neurons. A softmax activation function was used here. A Stochastic Diagonal Levenberg Marquardt [68] optimization solver was used to train the classifier. A total of 25K positives and 25K negatives were given to the network for 56 epochs.

Authors in [77] used CNN for the task of detecting pedestrians. A sliding window based approach is used to classify the current window as background or a pedestrian. The step size of this slider is 3 px. The window resolution is 30x60 pixels. The obtained multiple detections are then merged. A two convolution layer architecture was presented, each followed by a subsampling layer. The input to this network was an image of size 30x60. The kernel size of the convolution layer was 5x5 and the number of filters per layer was 15. The second subsampling layer was connected to a 40 neuron hidden FC layer. This FC layer was then connected to an output layer with 1 neuron that predicts the score for the presence of pedestrian in the image. An imbalanced training set was given to the network with 3699 positive images containing the pedestrians and 30000 negative images containing the negative samples.

Localization and classification of speed signs was carried out by the authors in [78], using CNNs in real time at a rate of 35 fps. The dataset constructed had 713 images for 7 classes of speed signs. The background images used were other speed signs and scenes. The architecture used for speed sign classification, included two convolution layers and two subsampling layers. The convolution layers had 5x5 sized kernels with 6 kernels in the first one and 16 kernels in the other. The subsampling layer had 2x2 sized window and a stride of 2 px. The final subsampling layer was connected to a FC layer with 80 neurons. The activation function used in the network was hyperbolic tangent function. Iterative boosting [79] was applied for increasing the accuracy of the classifier.

In this step, during the validation cycle, the entire false positive results beyond a certain threshold. Authors have used sliding window based approach for localizing the speed signs. Learning and predicting pedestrian attributes was addressed in [80] using CNNs. The datasets used were VIPeR [81] and GRID. The attributes used were like gender, hair color, shirt color, presence of backpacks etc.

Authors in [80] presented a architecture with three convolution layers each with 16 filter kernels. The kernels used had sizes 7x7, 5x5 and 3x3 respectively. Every convolution layer was followed by a pooling layer with kernel size 2x2 and stride 2, and a rectified linear unit activation layer. The third subsampling layer was connected to the output FC layer. To get the pedestrian attributes from the image, the image was divided into 15 overlapping patches. Each patch was labelled manually as per the attribute, and was given to the net.

The problem of detecting and mapping crowds from the image through a first person view was addressed by authors in [82] using CNN. The architecture had three convolution layers, each followed by a max pooling layer. The input to this architecture was a RGB image of size 50x50. The first convolution layer had 15 filters each of kernel size 8x8, the second had 7x7 sized 90 filters and the third one had 180 filter kernels of size 5x5. The third pooling layer was connected to a FC layer with 420 neurons and that in turn was connected to a output FC layer with 9 neurons. This was then given to a softmax classifier layer. To segment out crowd from the background, patches of crowd and background to the network as positives and negatives respectively. The positives were further classified as three types of crowds

depending upon the approximate distance of the crowd from the camera. Similarly the background class was classified into 6 subclasses.

The task of vehicle type classification was carried out by authors in [83] using CNN. A dataset BIT-Vehicle dataset was compiled by the authors, which included 9850 images in frontal-view. Authors in [83] presented a branched network. The input to the architecture was a 143x143 sized input. This was connected to the first convolution layer with kernel size of 9x9 and 64 filters. This layer was followed by an absolute rectification layer, a local contrast normalization layer, an average pooling layer with kernel size 10x10 and stride 1. This pooling layer was then connected to a subsampling layer with stride and kernel size 5x5. The output of this layer was split into two branches. The first branch was connected to convolution layer with same properties as the first convolution layer except the number of filters here were 256. This layer was followed by an absolute rectification layer, a local contrast normalization layer, an average pooling layer with kernel size 6x6 and stride of 1 px, and a subsampling layer of kernel size 4x4 and stride of 4 px. The output of this was connected to a 4096 neuron hidden FC layer. The second branch was connected to another pooling layer with kernel size 6x6 and stride 1 px, followed by another subsampling layer of size 4x4 and stride 4 px. This layers output was connected to a hidden Fc layer with 2304 neurons. The hidden FC layers of the branches were concatenated to get a hidden FC layer with 6400 neurons. This was finally connected to a FC layer with 6 output neurons. The classifier used is softmax. An unsupervised learning algorithm, sparse Laplacian filter learning (SLFL), was presented which makes the convolution filters learn the weights. This learning methodology was used to classify vehicles into one of the mentioned classes.

Authors in [84] used CNN for recognising the vehicle color. Chen car dataset [85] was used which had 15601 images of cars spread over 8 classes of colors. Every pooling layer used in the architecture presented in [84] has a stride of 2 px and kernel size of 3x3 pixels. The network architecture was very deep and branched. The input image of size 227x227x3 was given to two identical branches. The first convolution layer had 48 filter kernels of size 11x11 and stride 2 followed by a pooling layer. The output of this layer was split into two parts, each having 24 channels. These were then passed through convolution layers with 64 kernels of size 3x3 and pooling layers, separately. The output of these two layers were concatenated and passed through a convolution layer with 192 kernels of size 3x3. This output now with
192 kernels was split into two halves and each branch is passed through two convolution layers each with 3x3 kernel, with the second convolution layer having a pooling layer after it. The output from these two split short branches was then connected to two FC layers each having 1024 neurons. Similarly two more FC layers were obtained from the other main branch and all these branches were concatenated to one single FC layer with 4096 neurons. This hidden FC layer was connected to another FC layer having 4096 units via a dropout layer and in turn was connected to the output layer with 8 neurons. The classifier used was softmax. Stochastic gradient descent optimization algorithm was used to train the network. A mini batch size of 115 images was used with momentum parameter set to 0.9 and weight decay parameter set to 0.0005. The learning rate was initialized to 0.01 and was reduced by a factor of 10 after every 50K iterations until the training converged at 200K iterations. The weights of the classifier were initialized using a gaussian function with 0.01 as the standard deviation and 0.1 fixed values for the bias.

Authors in [18] devised a system to predict eye fixations in an image using CNN at different resolutions of the image. The authors have used four different datasets, MIT dataset [86], the Toronto dataset [87], the Cerf Dataset [88], and the NUSEF dataset [89], with each image being viewed by 25 subjects on an average to label the eye fixations. The architecture presented in [18], took one patch each from the three rescaled images and fed them into three parallel branches. Each branch was identical. Each branch consisted of three convolution layers and pooling layers. The pooling layers have kernel of size 2x2 and stride of 2 px. The convolution layers had 96, 60 and 288 filters respectively and each of the kernels had sizes 3x3. The activation function used was rectified linear units. The output of the final pooling layer was fed into a hidden FC layer having 512 neurons. This FC layer from each of the three branches was concatenated and connected to another hidden FC layer with 512 neurons. This layer was ultimately connected to an output neuron. In the approach taken by the authors, the image was rescaled to three different resolutions for predicting eye fixations. These resolutions were 150x150, 250x250, and 400x400. From each of these images a 42x42 sized patch was extracted out and giving it to the classifier for training and testing. The data given for the training included 10 fixation locations and 20 non-eye fixation locations. Weight decay was set to 0.0002 and unlike other approaches; the momentum was linearly increased from 0.9 to 0.99. The network was first trained on the MIT dataset and then the trained weights were transferred for the training of the images from other sets. The learning rate was kept 0.0001 for the first set of images and was reduced by 10 for the rest of them.

The detection and classification of croatian traffic signs was discussed in [15] using CNNs. The dataset used was FER-MASTIF TS2010 [90] for training and testing the classifier. Authors presented two architectures for MNIST [91] and FER-MASTIF TS2010 datasets. The one for the MNIST set had three convolution layers and two pooling layers one each for the first two convolution layers. The convolution layers had 6, 16, and 100 feature maps respectively, each having kernels of size 5x5. The pooling layers had kernel size of 2x2 and stride of 2 px. The third convolution layer was connected to a hidden FC layer with 80 neurons and output layer layer with 10 neurons. The second architecture had four convolution layers and three pooling layers attached to the first three convolution layers. These layers had 10, 15, 20 and 40 feature maps respectively with kernel sizes 7x7, 3x3, 3x3 and 3x3 respectively. The fourth convolution layer was connected to hidden FC layer with 80 neurons and that in turn was connected to output FC layer with 9 neurons. The traffic sign classifier was trained for 54000 iterations using the stochastic gradient descent algorithm. The input to the first architecture for MNIST data was of size 28x28 and that for the other it was 48x48.

VI. CNNs in OTHER COMPUTER VISION APPLICATIONS

In [37], authors used CNN to recognize 100 faces when given training images compiled from the AR [92] and FERET [93] face databases. They used a network which had two convolution layers and two FC layers for recognizing faces. The convolution layers were followed by pooling layers with kernel size of 2 and a stride of 2. The convolution layers had 15 and 45 filters respectively and each of the filters had a size of 6x6. The number of neurons in the hidden FC layer was 130 and the output layer had 100 neurons for recognizing 100 faces from the dataset. The input to this architecture was a one dimensional grayscale image of size 56x46. The architecture used in case of AR database was slightly different from the one in case of FERET database. A set of 3000 training images were compiled for the face recognition application. Of this set, 80% was used in training and the rest for validation. The solver used in the training process was Stochastic Diagonal Levenberg Mar-quardt (SDLM) [68]. The activation function used was TanH. Input to the architecture was an image of height 56 and width 46 px. To find the best possible combination of number of filters for each layer, a set of experiments were done with random combinations of the number and the best one among them was selected. Weights were initialized using a gaussian distribution with zero mean and 0.01 as standard deviation after experimenting with three more ways of weight initialization, namely, uniform, Fanin , and nguyen distributions. This approach led to a 99.5% and 85.13% accuracies on AR and FERET test datasets respectively.

Authors in [94] proposed a classifier integration model using CNN. A Classifier integration model is a cascade of local classifier with individual importance as weights. Their approach involved integrating multiple three layered CNNs and stacking their results to predict the class of the object. It was tested on 101 Caltech object dataset [95]. Herein, the CNNs have the same depth of three layers. The only differences were in the filter sizes and the number of filters per layer. These CNNs had three convolution layers, two pooling layers and the final output was a FC layer. A set of 4 CNNs were trained for about 1000 epochs. The architecture was trained till convergence was obtained with the Mean Squared Error (MSE) Loss function layer. The activation function used here was hyperbolic tangent function.

To recognize facial expression in [57], a multi-scale CNN network was proposed. The dataset used was JAFFE facial expression database [96]. This dataset had faces from 10 subjects with 6 different expressions. Each image was a grayscale image of size 256x256. The architecture presented had two convolution layers each followed by a pooling layer of kernel 2x2 and stride 2. The first convolution layer had 6 feature maps, each obtained after convoluting with a kernel of size 11x11. The next one had 9 maps when passed through a kernel of size 8x8. The output of the second pooling layer was connected to an output FC layer with 6 neurons, each one representing a facial expressions probability. Hyperbolic tangent function was used for activation. The image was resized to 64x64 sized resolution and was given to the CNN as the input.

Authors in [39] used Deep CNNs to accomplish the task to face detections in different views. The network was designed to differentiate a face from background (non-face) image, localize the face and estimate its pose. The data was compiled from various face datasets, FERET Dataset, PIE dataset [97], BioID dataset [98]. The compiled data had 43000 frontal faces, 43000 half profiles and 25000 full profiles. Negative dataset was about 30K images. A cascade based multiview face detector assigned a label face/non-face to the input image. If it were a face, then a 32x32 sized patch was sent to a Deep CNN for more accurate decision. The proposed architecture had a very peculiar way of making the network as a multi-task cnn. The CNN classified, localized and estimated the approximate pose of the face. The network had two convolution layers; each had a pooling layer attached to it with a size of 2x2 and stride 2. The first convolution layer had 32 kernels of 5x5 size, whereas the second had 32 kernels of size 3x3. The final

pooling layer was connected to a FC hidden layer with 512 neurons. This layer was connected to three FC hidden layers in parallel with 128 neurons in the first two, and 256 in the third branch. The first branch was connected to a 2 neuron output, one indication presence of face and other non-faces. Similarly, the second layer was connected to a second output layer with 5 neurons each indicating a probability of a particular pose of the face. The final branch was connected to a 196 neuron hidden layer which in turn was connected to the third output layer with 14 neurons. These 14 neurons gave positions of 7 facial landmarks. ReLU activation was used in here. The hidden FC layers had dropout layers, with probability parameter as 0.5, in between them. The authors used a stochastic mini-batch size of 128 batches. The decay rate for the weight was set to be 0.00005 and the initial learning rate of 0.01. This rate was manually decreased by 50% if the loss didn't reduce after 5 epochs. A set of 1.88M positive and 570K negative patches were used for training and it was stopped after 100 epochs.

An image scanning methodology was presented in [99] using CNN. This paper presented image segmentation and classification using deep CNNs. Herein, the authors presented a 4 convolution layer network followed by two FC layers. Each pooling layer used after the convolution layers had kernel of size 2x2 with a stride of 2. Each convolution layer had 48 filters of size 5x5, expect for the third one which had 5x5 sized kernels. The hidden FC layer had 200 neurons and the output FC layer had 2 neurons.

Gender classification task was addressed by the authors in [34] using CNNs. In the paper, they compared training and testing with three color spaces, RGB, YUV and grayscale and concluded that the best results were obtained with the grayscale color space. The dataset used was MIT pedestrian dataset, containing 988 images of size 64x128. The dataset contained images of 600 males and 288 females. Authors presented an architecture with two convolution layers, each followed by a subsampling layer, and two FC layers. The first convolution layer had 10 filter maps each of size 5x5 and the second one had 20 filter maps of the same size as of the filters in the first one. The subsampling layer was of type max-pooling and had kernel size of 2x2 and a stride of 2. The hidden FC layer had 25 neurons and the final output layer had 2 neurons representing the genders. The images from the MIT pedestrian dataset were cropped to 54x108 by removing the border from all the four sides equally. This cropped image was resized to 40x80 resolution. The weights in the network layers were initialized using uniform distribution, between the range [-sqrt(6/f), sqrt(6/f)]. The hyper-parameter f, was the sum of number of input connections and the output connections of a layer. The network was trained for 500 epochs.

Authors in [100] presented a CNN based classification of maritime vessel images. A dataset name E2S2-Vessel was created using the images from the web. This architecture had a very deep network with 5 convolution layers. The first two convolutions were followed by a 3x3 pooling layer with stride 2 px, an activation layer layer with rectified linear unit functions and a local contrast normalization layer. The next two convolution layers were followed just by the activation layer. The fifth one was followed by a pooling layer and the activation layer.

This pooling layer was connected to hidden FC layer having 4096 neurons. This FC layer was connected to another hidden FC layer with 4096 neurons through a dropout layer which in turn was connected to the output layer with 35 nodes. The classifier used was softmax layer. The convolution layers had 96, 256, 384, 384, and 256 filters respectively with sizes 11x11, 5x5, 3x3, 3x3, and 3x3 respectively. The stride of the first convolution layer was 4 px and for the others it was 1 px. A stochastic gradient descent solver was used to optimize the training process in reaching the convergence point. The mini-batch size used was 80 and the solver had momentum parameter set to 0.9 and weight decay parameter set to 0.0005. The learning rate was initialized to 0.01 and was dropped by a factor of 10 after every 40K iterations. The dataset collected had 130000 images spread over a 35 classes of maritime vessels.

VII. CONCLUSION

The review presented in the paper thus included the important factors required in successfully training a CNN based classifier. The pre-processing techniques used before the training in different cases is presented. The data augmentation processes used by researchers to append the dataset with artificial variations is also presented. The review prominently includes the network architecture and training parameters used for training a CNN based classifier in computer vision based applications. Thus, the paper comprehensively explained the factors stated above.

REFERENCES

[1] Francesco Virili and Bernd Freisleben, "Nonstationarity and Data Preprocessing for Neural Network Predictions of an Economic Time Series," in Proc. IEEE - INNS - ENNS Int. Joint Conf. on Neural Networks, 2000, pp. 129-136. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] Jonathan Sachs, "Image Resampling," unpublished, Available: http://www.dl-c.com/Resampling.pdf

[3] R. L. L. I. Setyawan, "Human perception of geometric distortions in images," unpublished, Available: http://prb.tudelft.nl/sites/default/files/SPIE-SanJose2004-1.pdf

[4] D. C. Cire san, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in Proc. 22nd Int. Joint Conf. on Artificial Intelligence, Vol. 2, 2011, pp. 1237-1242.

[5] K. Fukushima, and H. Shouno, "Deep convolutional network neocognitron: Improved interpolating vector," in proc. Int. Joint Conf. on Neural Networks, IJCNN, Jul 2015, pp. 1-8.

[6] J. Jin, A. Dundar, and E. Culurciello, "Robust convolutional neural networks under adversarial noise," CoRR. Available: http://arxiv.org/abs/1511.06306

[7] I. J. Goodfellow, D. Warde-farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," in Proc. 30th Int. Conf. on Machine Learning, June 2013, pp. 1319-1327.

[8] X. L. Nouman Qadeer, "Empirical evaluation of whitening and optimization of feature learning," in Proc. UKSim-AMSS 16th Int. Conf. on Computer Modelling and Simulation (UKSim), IEEE, 2014, pp. 36-39.

[9] W. L. Zuhe Li, and Yangyu Fan, "The effect of whitening transformation on pooling operations in convolutional autoencoders," EURASIP J. Adv. Sig. Proc. , 2015. Available: http://dx.doi.org/10.1186/s13634-015-0222-1

[10] C. Farabet, B. Martini, P. Akselrod, S. Talay, Y. LeCun, and E. Culurciello, "Hardware accelerated convolutional neural networks for synthetic vision systems," in Proc. Int Symp. on Circuits and Systems (ISCAS 2010), France, 2010, pp. 257-260.

[11] S. Lyu, and E. P. Simoncelli, "Nonlinear image representation using divisive normalization," in Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR 2008), Alaska, June 2008.

[12] D. Kiela, and L. Bottou, "Learning image embeddings using convolutional neural networks for improved multi-modal semantics," in Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP), Qatar, 2014, pp. 36-45.

[13] P. H. O. Pinheiro, and R. Collobert, "Recurrent convolutional neural networks for scene parsing," CoRR. Available: http://arxiv.org/abs/1306.2795

[14] G. Hu, Y. Yang, D. Yi, J. Kittler, W. J. Christmas, S. Z. Li, and T. M. Hospedales, "When face recognition meets with deep learning: An evaluation of convolutional neural networks for face recognition," in Proc. IEEE Int. Conf. on Computer Vision Workshop, ICCV Workshops, Chile, 2015, pp. 384-392.

[15] V. Vukoti´c, J. Krapac, and S. Segvi´c, "Convolutional neural networks for croatian traffic signs recognition," in Proc. The Croatian Computer Vision Workshop, Zagreb, 2014, pp. 15-20.

[16] T. Schlegl, S. Waldstein, W.-D. Vogl, U. Schmidt-Erfurth, and G. Langs, "Predicting Semantic Descriptions from Medical Images with Convolutional Neural Networks," in Inform. Processing in Medical Imaging, Vol. 9123 of Lecture Notes in Computer Sci., 2015, pp. 437-448. Available: http://dx.doi.org/10.1007/978-3-319-19992-4 34

[17] J. C. L. Lam, and M. Eizenman, "Convolutional neural networks for eye detection in remote gaze estimation systems," in Proc. Int. MultiConference of Engineers and Computer Scientists, 2008, pp. 601-606.

[18] N. Liu, J. Han, D. Zhang, S. Wen, T. Liu, "Predicting eye fixations using convolutional neural networks," in Proc. IEEE Conf. on Computer Vision and Pattern Recognition, CVPR, Boston, 2015, pp. 362-370

[19] P. H. O. Pinheiro, and R. Collobert, "From image-level to pixel-level labeling with convolutional networks," in Proc. IEEE Conf. on Computer Vision and Pattern Recognition, CVPR, Boston, 2015, pp. 1713-1721.

[20] R. F. Nogueira, R. de Alencar Lotufo, and R. C. Machado, "Fingerprint liveness detection using convolutional neural networks," IEEE Trans. Inform. Forensics and Security, vol.11(6), pp.1206-1213, 2016.

[21] D. C. Ciresan, U. Meier, J. Masci, and J. Schmidhuber, "A committee of neural networks for traffic sign classification," in Proc. The Int. Joint Conf. on Neural Networks, IJCNN, 2011, pp. 1918-1921.

[22] S. Mukhopadhyay, S. Mandal, S. Pratiher, S. Changdar, R. Burman, N. Ghosh, and P. K. Panigrahi, "A comparative study between proposed hyper kurtosis based modified duo-histogram equalization (HKMDHE) and contrast limited adaptive histogram equalization (CLAHE) for contrast enhancement purpose of low contrast human brain CT scan images," CoRR abs/1505.06219. Available: http://arxiv.org/abs/1505.06219

[23] M. Veta, P. J. van Diest, S. M. Willems, H. Wang, A. Madabhushi, A. Cruz-Roa, F. A. Gonz´alez, A. B. L. Larsen, J. S. Vestergaard, A. B. Dahl, D. C. Ciresan,

J. Schmidhuber, A. Giusti, L. M. Gambardella, F. B. Tek, T. Walter, C. Wang, S. Kondo, B. J. Matuszewski, F. Precioso, V. Snell, J. Kittler, T. E. de Campos, A. M. Khan, N. M. Rajpoot, E. Arkoumani, M. M. Lacle, M. A. Viergever, and J. P. W. Pluim, "Assessment of algorithms for mitosis detection in breast cancer histopathology images," CoRR abs/1411.5825. Available: http://arxiv.org/abs/1411.5825

[24] R. Maini, and H. Aggarwal, "A comprehensive review of image enhancement techniques," CoRR abs/1003.4053. Available: http://arxiv.org/abs/1003.4053

[25] Y. Wu, Y. Liu, J. Li, H. Liu, X. Hu, "Traffic sign detection based on convolutional neural networks," in Proc. The 2013 Int. Joint Conf. on Neural Networks, IJCNN, 2013, pp. 1–7.

[26] K. Suzuki, "Pixel-based machine learning in medical imaging," Int. J. Biomedical Imaging 2012, Available: http://dx.doi.org/10.1155/2012/792079

[27] M. Elawady, "Sparse coral classification using deep convolutional neural networks," MSc dissertation, Dept. Computer Architecture and Tech., Girona Univ., 2014

[28] Y. Tang, "Deep learning using support vector machines," CoRR abs/1306.0239. Available: http://arxiv.org/abs/1306.0239

[29] S. Roth, A. R. T. Gepperth, and C. Igel, "Multi-objective neural network optimization for visual object detection," in Proc. Multi-Objective Machine Learning, 2006, pp. 629-655.

[30] Peterson C., and Anderson J. R., "A mean field theory learning algorithm for neural networks," Complex Systems, vol.1, pp. 995-1019, 1987.

[31] P. T. Bharathi, and P. Subashini, "Optimization of image processing techniques using neural networks: A review," WSEAS Trans. Info. Sci. and App., vol. 8(8), pp. 300-328, 2011.

[32] N. Bayramoglu, J. Kannala, and J. Heikkil¨a, "Human epithelial type 2 cell classification with convolutional neural networks," in Proc. 15th IEEE Int. Conf. on Bioinformatics and Bioengineering, BIBE, Serbia, 2015, pp. 1-6.

[33] Z. Shi, L. He, K. Suzuki, T. Nakamura, and H. Itoh, "Survey on neural networks used for medical image processing ," Int. J. Comp. Sci., vol.3(1), pp. 86-100, 2009.

[34] B.-M. G. Choon-Boon Ng, and Yong-Haur Tay, "Comparing image representations for training a convolutional neural network to classify gender," in Proc. 1st Int. Conf. on Artificial Intelligence, Modelling and Simulation, IEEE, 2013, pp. 24-28.

[35] H. Schulz, and S. Behnke, "Object-class segmentation using deep convolutional neural networks," in Proc.of the DAGM Workshop on New Challenges in Neural Computation 2011, pp. 58-61.

[36] S. P. Hohberg, "Wildfire smoke detection using convolutional neural networks," Ph.D. dissertation, Freie Univ., Berlin, 2015.

[37] A.R.Syafeeza, M.Khalil-Hani, S.S.Liew, and R.Bakhteri, "Convolutional neural network for face recognition with pose and illumination variation," Int. J. of Eng. and Technology, vol. 6(1), pp. 44-57, 2014.

[38] T. Kekec, R. Emonet, E Fromont, A. Tr´emeau, and C. Wolf, "Contextually constrained deep networks for scene labelling," in Proc. British Machine Vision Conf., BMVC, Nottingham, 2014.

[39] C. Zhang, and Z. Zhang, "Improving multiview face detection with multi-task deep convolutional neural networks," in Proc. IEEE Winter Conf. on Applications of Comput. Vision, 2014, pp. 1036-1041.

[40] H. Faraji, and W. J. MacLean, "CCD noise removal in digital images," IEEE Trans. Image Processing, vol. 15(9), pp. 2676-2685, 2006.

[41] M. Castelluccio, G. Poggi, C. Sansone, and L. Verdoliva, "Land use classification in remote sensing images by convolutional neural networks," CoRR abs/1508.00092. Available: http://arxiv.org/abs/1508.00092

[42] K. Lin, H.-F. Yang, and C.-S. Chen, "Flower classification with few training examples via recalling visual patterns from deep cnn," unpublished. Available: http://www.csie.ntu.edu.tw/ r01944012/cvgip15 flower.pdf

[43] D. C. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in Proc. 26th Annu. Conf. on Neural Inform. Processing Systems, United States., 2012, pp. 2852-2860.

[44] T. Pfister, K. Simonyan, J. Charles, and A. Zisserman, ``Deep convolutional neural networks for efficient pose estimation in gesture videos," in Proc. 12th Asian Conf. on
Comput. Vision, 2014, pp. 538-552.

[45] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, and V. D. Shet, "Multi-digit number recognition from street view imagery using deep convolutional neural networks,"CoRR abs/1312.6082. Available: http://arxiv.org/abs/1312.6082

[46] A. Dosovitskiy, J. T. Springenberg, and T. Brox, "Unsupervised feature learning by augmenting single

images," CoRR abs/1312.5242. Available: http://arxiv.org/abs/1312.5242

[47] J. Redmon, and A. Angelova, "Real-time grasp detection using convolutional neural networks," CoRR abs/1412.3128. Available: http://arxiv.org/abs/1412.3128

[48] H. R. Roth, L. Lu, J. Liu, J. Yao, A. Seff, K. M. Cherry, L. Kim, and R. M. Summers, "Improving computer-aided detection using convolutional neural networks and random view aggregation," IEEE Trans. Med. Imaging vol.35(5), pp. 1170-1181, 2016.

[49] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in Proc. IEEE Conf. on Comput. Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 2014, pp. 1717-1724.

[50] S. Li, and A. B. Chan, "3d human pose estimation from monocular images with deep convolutional neural network," in Proc. 12th Asian Conf. on Comput. Vision, Singapore, Nov 2014, pp. 332-347.

[51] J. Margeta, A. Criminisi, R. Cabrera Lozoya, D. C. Lee, and N. Ayache, "Finetuned convolutional neural nets for cardiac MRI acquisition plane recognition," J. of Comput. Methods in Biomechanics and Biomedical Eng.: Imaging and Visualization, 2015. Available: https://www.microsoft.com/en-us/research/publication/finetuned-convolutional-neural-nets-for-cardiac-mri-acquisition-plane-recognition/

[52] D. Maturana, and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS, Germany, Sept, 2015, pp. 922-928.

[53] S. He, R. W. H. Lau, W. Liu, Z. Huang, and Q. Yang, "Supercnn: A superpixelwise convolutional neural network for salient object detection," Int. J. of Comput. Vision, vol.115(3), pp. 330-344, 2015.

[54] D. C. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in Proc. IEEE Conf. on Comput. Vision and Pattern Recognition, USA, June, 2012, pp. 3642-3649.

[55] J. Wan, D. Wang, S. C. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li, "Deep learning for content-based image retrieval: A comprehensive study," in Proc. of the ACM Int. Conf. on Multimedia, 2014, pp. 157-166.

[56] P. H. O. Pinheiro, and R. Collobert, "Weakly supervised object segmentation with convolutional neural networks," in Proc. IEEE Conf. on Comput. Vision and Pattern Recognition, 2015.

[57] B. Fasel, "Multiscale facial expression recognition using convolutional neural networks," in Proc. of the Third Indian Conf. on Comput. Vision, Graphics and Image Processing, India, Dec 2002.

[58] A. Dubrovina, P. Kisilev, B. Ginsburg, S. Hashoul, and R. Kimmel, "Computational mammography using deep neural networks," Comput. Methods in Biomechanics and Biomedical Eng.: Imaging and Visualization vol.0(0), pp. 1-5, 2015.

[59] "Icipr 2013 dataset," Available Online, http://mivia.unisa.it/icip-2013-contest-on-hep-2-cells-classification/

[60] J. Acquarelli, M. Bianchini, and E. Marchiori, "Discovering potential clinical profiles of multiple sclerosis from clinical and pathological free text data with constrained non-negative matrix factorization," in Proc.
Applications of Evolutionary Computation - 19th European Conf., Portugal, March 2016, pp. 169-183.

[61] M. Maleki, P. M. Teshnehlab, and D. M. Nabavi, "Diagnosis of multiple sclerosis (ms) using convolutional neural network (cnn) from mris," Global J. of Medicinal Plant Research vol. 1(1), pp. 50-54, 2012.

[62] D. Lyndon, A. Kumar, J. Kim, P. H. W. Leong, and D. Feng, "Convolutional neural networks for medical clustering," in Proc. Working Notes of CLEF 2015 - Conf. and Labs of the Evaluation forum, Toulouse, Sept 2015.

[63] A. A. Khanifar, and S. Farsiu, "Spectral-domain oct in practice," unpublished. Available: http://people.duke.edu/sf59/khanifar08.pdf

[64] M. Melinscak, P. Prentasic, and S. Loncaric, "Retinal vessel segmentation using deep neural networks," in Proc. 10th Int. Conf. on Comput. Vision Theory and Applications, vol.1, Germany, March, 2015, pp. 577-582.

[65] \ J. Staal, M. D. Abr` amoff, M. Niemeijer, M. A. Viergever, and B. van Ginneken, "Ridge-based vessel segmentation in color images of the retina," IEEE Trans. Med. Imaging vol. 23(4), pp. 501-509, 2004.

[66] \ C. C. Glavan, and S. Holban, "Segmentation of Bone Structure in X-ray Images using Convolutional Neural Network (Periodical style)," Advances in Electrical and Comput. Eng., vol. 13(1), pp. 87-94, 2013.

[67] M. Auli, and A. Lopez, "Training a log-linear parser with loss functions via softmax-margin," in Proc. Conf. on Empirical Methods in Natural Language Processing, EMNLP 2011, pp. 333–343.

[68] A. Al-Mayyahi, W. Wang, and P. Birch, "Levenberg-marquardt optimised neural networks for trajectory

tracking of autonomous ground vehicles," Int. J. of Mechatronics and Automation, vol. 5(2/3), pp. 140-153, 2015.

[69] A. Ferrari, S. Lombardi, and A. Signoroni, "Bacterial colony counting by convolutional neural networks," in Proc. 37th Annu. Int. Conf. of the IEEE Eng. in Medicine and Biology Society, Milan, 2015, pp. 7458-7461.

[70] W. Xie, J. A. Noble, and A. Zisserman, "Microscopy cell counting with fully convolutional regression networks," in Proc. MICCAI 1st Workshop on Deep Learning in Medical Image Analysis, 2015.

[71] S. Gould, R. Fulton, and D. Koller, "Decomposing a scene into geometric and semantically consistent regions," in Prof. IEEE 12th Int. Conf. on Comput. Vision, ICCV 2009, Kyoto, Japan, Sept 2009, pp. 1-8.

[72] C. Liu, J. Yuen, and A. Torralba, "SIFT flow: Dense correspondence across scenes and its applications," IEEE Trans. Pattern Anal. Mach. Intell., vol. 33(5), pp. 978-994, 2011.

[73] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, "Imagenet large scale visual recognition challenge," CoRR abs/1409.0575. Available: http://arxiv.org/abs/1409.0575

[74] "Pascal visual object classes recognition challenge 2012," Available: http://www.pascal-network.org/?q=node/874

[75] L. Wang, Z. Wang, W. Du, and Y. Qiao, "Object-scene convolutional neural networks for event recognition in images," in Proc. 2015 IEEE Conf. on Comput. Vision and Pattern Recognition Workshops, CVPR Workshops, Boston, 2015, pp. 30-35.

[76] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in Proc. Annu. Conf. on Neural Inform. Processing Systems, Dec, 2014, pp. 487-495.

[77] M. Szarvas, A. Yoshizawa, M. Yamamoto, J. Ogata, "Pedestrian detection with convolutional neural networks," in Proc. Intelligent Vehicles Symp., IEEE, 2005.

[78] P. Maurice, M. Bart, and C. Henk, "Speed sign detection and recognition by convolutional neural networks," unpublished. Available: http://parse.ele.tue.nl/system/attachments/11/original/paperspeedsigncnn.pdf.

[79] H. Fu, X. Kong, and J. Lu, "Large-scale image retrieval based on boosting iterative quantization hashing with query-adaptive reranking," Neurocomputing vol.122, pp. 480-489, 2013.

[80] J. Zhu, S. Liao, D. Yi, Z. Lei, and S. Z. Li, "Multi-label CNN based pedestrian attribute learning for soft biometrics," in Proc. Int. Conf. on Biometrics, ICB 2015, Thailand, May, 2015.

[81] D. Gray, S. Brennan, and H. Tao, "Evaluating appearance models for recognition, reacquisition, and tracking," in Proc. IEEE Int. Workshop on Performance Evaluation for Tracking and Surveillance, Rio de Janeiro, 2007.

[82] J. S. Olier, C. S. Regazzoni, L. Marcenaro, M. Rauterberg, "Convolutional neural networks for detecting and mapping crowds in first person vision applications," in Proc. Advances in Computational Intelligence - 13th Int. Work-Conf. on Artificial Neural Networks, IWANN Spain, June 2015, pp. 475–485.

[83] Z. Dong, Y. Wu, M. Pei, and Y. Jia, "Vehicle type classification using a semisupervised convolutional neural network (Periodical style)," IEEE Trans. Intelligent Transportation Systems vol. 16(4), pp. 2247-2256, 2015.

[84] R. F. Rachmadi, and I. K. E. Purnama, "Vehicle color recognition using convolutional neural network," CoRR abs/1510.07391. Available: http://arxiv.org/abs/1510.07391

[85] L. Yang, P. Luo, C. C. Loy, and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," in Proc. IEEE Conf. on Comput. Vision and Pattern Recognition, 2015, pp. 3973-3981.

[86] T. Judd, K. A. Ehinger, F. Durand, and A. Torralba, "Learning to predict where humans look," in IEEE 12th Int. Conf. on Comput. Vision, ICCV, Japan, 2009, pp. 2106-2113.

[87] J. Han, K. Li, L. Shao, X. Hu, S. He, L. Guo, J. Han, and T. Liu, "Video abstraction based on fmri-driven visual attention model (Periodical style)," Inf. Sci., vol. 281, pp. 781-796, 2014.

[88] M. Cerf, J. Harel, W. Einh¨auser, and C. Koch, "Predicting human gaze using low-level saliency combined with face detection," in Proc. of the 21st Annu. Conf. on Neural Inform. Processing Systems, Vancouver, 2007, pp. 241-248.

[89] S. Ramanathan, H. Katti, N. Sebe, M. S. Kankanhalli, and T. Chua, "An eye fixation database for saliency detection in images," in Proc. 11th European Conf. on Comput. Vision, Greece, 2010, pp. 30-43.

[90] S. Segvic, K. Brkic, Z. Kalafatic, and A. Pinz, "Exploiting temporal and spatial constraints in traffic sign detection from a moving vehicle," Mach. Vis. Appl. vol. 25(3), pp. 649-665, 2014

[91] "Mnist digit classification dataset," Available: http://yann.lecun.com/exdb/mnist/.

[92] L. Ding, A. M. Martínez, "Features versus context: An approach for precise and detailed detection and delineation of faces and facial features," IEEE Trans. Pattern Anal. Mach. Intell. , vol. 32(11), pp.2022-2038., 2010.

[93] "Feret dataset," Available: http://www.nist.gov/itl/iad/ig/feret.cfm.

[94] Y. Zeng, X. Xu, Y. Fang, K. Zhao, "Traffic sign recognition using deep convolutional networks and extreme learning machine," in Proc. 5th Int. Conf., IScIDE China, June, 2015, pp. 272-280.

[95] F. Li, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," Comput. Vision and Image Understanding, vol. 106(1), pp. 59–70, 2007.

[96] M. J. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expressions with gabor wavelets," in Proc. 3rd Int. Conf. on Face & Gesture Recognition (FG '98), Japan, 1998, pp. 200–205.

[97] T. Sim, S. Baker, and M. Bsat, "The CMU pose, illumination, and expression database," IEEE Trans. Pattern Anal. Mach. Intell., vol. 25(12), pp. 1615-1618, 2003.

[98] "Bioid face dataset," Available: https://www.bioid.com/About/BioID-Face-Database

[99] A. Giusti, D. C. Ciresan, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Fast image scanning with deep max-pooling convolutional neural networks," in Proc. IEEE Int. Conf. on Image Processing, ICIP 2013, Australia, 2013, pp. 4034-4038.

[100] C. Dao-Duc, H. Xiaohui, and O. Mor`ere, "Maritime vessel images classification using deep convolutional neural networks," in Proc. of the 6th Int. Symp. on Inform. and Commun. Technology, Vietnam, 2015, pp. 276-281.