IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 27, 2019, accepted May 23, 2019, date of publication May 28, 2019, date of current version June 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2919656

Vision System for Quality Assessment of Robotic Cleaning of Fish Processing Plants Using CNN

EMIL BJØRLYKHAUG AND OLAV EGELAND

[1]Department of Ocean Operations and Civil Engineering, Norwegian University of Science and Technology, Aalesund, 6009 Aalesund, Norway
[2]Department of Mechanical and Industrial Engineering, Norwegian University of Science and Technology, Trondheim, 7491 Trondheim, Norway

Corresponding author: Emil Bjørlykhaug (bjorlykhaug@ntnu.no)

This work was supported by the Research Council of Norway under Grant 245613/O30.

ABSTRACT A vision system has been developed for automatic quality assessment of robotic cleaning of fish processing lines. The quality assessment is done by detecting residual fish blood on cleaned surfaces. The system is based on classification using convolutional neural networks (CNNs). The performance of different convolutional neural network architectures and parameters is evaluated. The datasets that simulate various conditions in fish processing plants are generated using data augmentation techniques. Tests using further augmented training data to increase the performance of the neural network are performed, which results in a substantial increase in performance both compared to the color thresholding technique and the same neural network architecture without augmented training data. The performance of the system is validated in experiments in an industrial setting.

INDEX TERMS Aquaculture, robot vision system, machine learning, computer vision.

I. INTRODUCTION

Process hygiene is important to ensure product quality in a fish processing plant. Microbiological control of spoilage bacteria such as *Pseudomonas* and *Shewanella* is critical for the quality and shelf life of fresh fish [1], [2]. Additionally, *Listeria monocytogenes* contamination during production may introduce a health risk to the consumer.

To cope with the risk of bacteria contamination, processing plants must be thoroughly and frequently cleaned [3]. In the case of salmon processing, production plants are cleaned every day. Currently this is done manually by cleaning crews at night after the production lines have been shut down. There are strong incentives for automating this process as it is repetitive and physically demanding, and the costs are significant for a salmon processing plant. To ensure that an automatic cleaning system will give a satisfactory result, it is important to have a system for assessment of cleaning quality. Moreover, to fully take advantage of an automatic cleaning system, also the quality assessment system should be automatic. A potential solution is to use computer vision, as also manual quality assessment is primarily based on visual inspection. It is considered to be straightforward for human workers to decide if the cleaning quality is acceptable. In this visual inspection, critical tasks include the detection of the presence of blood, fish debris and fish slime.

An interesting technique for visual inspection of fish processing lines is vision systems based on Convolution Neural Networks (CNN). CNN is well-established as a tool for deep learning in computer vision, and is well suited for classification of visual data. Typical applications for CNN are image classification, object detection, object tracking, pose estimation and natural language processing [4]. The method was proposed by LeCun *et al.* in 1990 [5], who developed the method further in [6]. In 2005, Steinkrau *et al.* showed that Graphic Processing Units (GPUs) could be used to speed up training times. This was followed up in 2006 [7]–[9] and 2007 [10]. Recently CNN has received much attention in computer vision, as it has given impressive results on visual interpretation tasks, e.g. the ImageNet Classification problem by Krizhevsky *et al.* [11].

Deep learning techniques rely on training where large dataset are needed for high accuracy. For this reason, data augmentation has been adopted to artificially inflate the size of the datasets and to achieve a more accurate network. This was done by Flusser *et al.* where moment invariants were used for pattern recognition [12]. Another solution was proposed by Yaeger *et al.*, who used small random changes in skew, rotation, and linear and quadratic scaling to augment training data [13]. A more recent example is

The associate editor coordinating the review of this manuscript and approving it for publication was Bora Onat.

Tao *et al.*, who augmented a 2D image sign language dataset by viewing 3D images from multiple angles and then converting the viewpoint to 2D images [14]. Such data augmentation techniques makes the CNN less sensitive to changes in position, orientation, stretching and scaling. A comparison of various data augmentation schemes applied to a simple CNN was performed by Taylor and Nitschke [15]. State-of-the-art classification networks include GoogleNet [16], AlexNet [11], VGG16 [17] and YOLO [18]. These networks perform well on large classification problems, and will handle large datasets for training. However, a too large network may be prone to overfitting if the size of the dataset is not sufficient.

For smaller classification problems with less training data available, it may be advantageous to develop specialized CNN classification networks. This has been done in research on the classification of images containing blood. This includes Ferreira *et al.*, who used CNN and a texture descriptor for detection and diagnosis of glaucoma [19], while Tiwari *et al.* used CNN for classifying white blood cell type [20]. Vogado *et al.* used pre-trained state-of-the-art CNN models and Support Vector Machines for diagnosing leukemia in blood slides [21]. An alternative solution was used in Penna *et al.*, who used color thresholding and filtering techniques for detecting blood in intestines [22], while Mackiewicz *et al.* used color histograms in the HSI colorspace, which was classified with a Support Vector Classifier for images of intestines [23]. A system for computer vision in quality inspection was presented by Benalia *et al.*, who used Principal Component Analysis (PCA) and Partial Least Squares – Discriminant Analysis for sorting dried figs based on color [24]. Another example was presented by Iglesias *et al.*, who inspected the quality of slate slabs by 3D color images [25]. The MangoNet system, based on CNN, for detecting and counting fruit in images was proposed by Kestur *et al.* [26]. However, no specialized CNN has been developed for classification of images containing blood for a cleaning system.

In this paper we develop a system for detecting blood on a fish processing line using CNN as the classification algorithm. The proposed solution uses data augmentation to generate a sufficiently large dataset and to avoid overfitting of the CNN. The data augmentation includes a generic method to artificially increase the size of the dataset without changing the complexity, and several methods to increase the complexity of the dataset. Here the increased complexity of the dataset is used to make the classification algorithm more robust, which improves the performance of the algorithm in the classification of an unseen image which deviates from the original dataset. A comparison with color thresholding techniques is performed. We also test the influence of network architectures on this problem and how further augmenting the training set can reduce overfitting of the network.

The rest of the paper is organized as follows: Section II presents the cleaning system. In Section III the proposed approach, the data augmentation techniques and the datasets are presented. Simple color threshold techniques for establishing a baseline are presented in Section IV. The experiments and their results are presented in Section V and the conclusions from the results and proposed future work are found in Section VI.

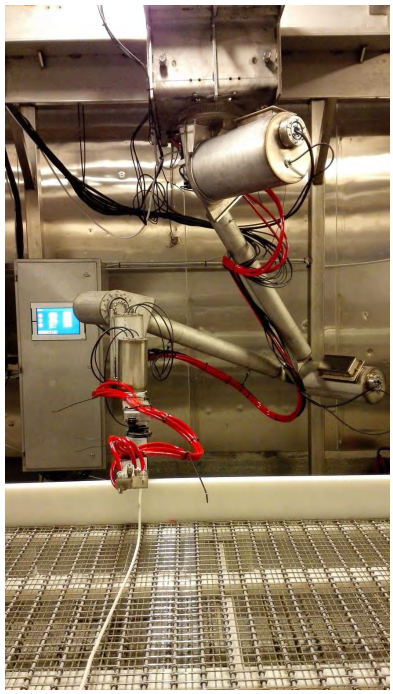


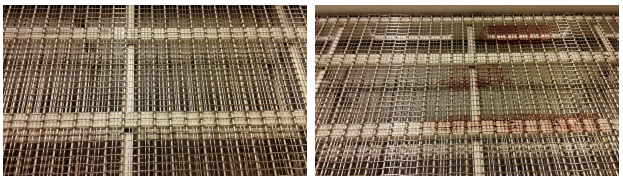
FIGURE 1. The robotic cleaning system.

II. SYSTEM OVERVIEW

The robotic cleaning system studied in this paper was documented in [27] and is shown in Figure 1. The system consists of a robotic manipulator where the kinematics has been optimized for the task, and where the design is made to withstand the harsh environment that is present during cleaning of fish processing plants. The manipulator is mounted to a horizontal axis placed underneath the ceiling. The manipulator will be parked outside of the processing plant during production, and when the daily production has stopped it will enter the production facility and start the cleaning process. After the cleaning has been completed, the vision system will assess if the requirements for cleaning quality are satisfied. The vision system can be mounted at the end effector of the manipulator, which enables the system to locate where the cleaning quality is not sufficient. The cleaning system can then choose if the whole cleaning process should be repeated, or if just the unclean area should be cleaned once more. Alternatively, the vision system can be arranged with cameras mounted on predefined positions inside the processing plant.

III. PROPOSED APPROACH

We propose to use CNN with augmented data for processing the images. Due to the power of the CNN solution,



(a) (b)

FIGURE 2. Example of image from the two different classes, one with blood present and one without. (a) Image without blood. (b) Image with blood.

we propose to use an area scan camera for acquiring the images. For testing the proposed approach a dataset was collected with a Sony IMX214 image sensor. The image sensor has a resolution of 4160×2336 pixels. The lens used was a F2.0 28mm wide angle lens. The images were captured with a distance of approximately 800 mm. The equipment used for testing the proposed approach was a conveyor and an electric stunner, two common machines in fish processing lines. Both machines have stainless steel belts and plastic elements as guides and walls which get contaminated with blood during production. An example of images captured is shown in Figure 2.

A. CONVOLUTIONAL NEURAL NETWORK

CNN is a feedforward neural network architecture that is a combination of convolution, pooling/subsampling and fully connected layers. The output of a convolution is given by

$$\mathbf{S}(i,j) = \sum_m \sum_n \mathbf{I}(i-m, j-n)\mathbf{K}(m,n) \tag{1}$$

where $\mathbf{K}$ is the convolution kernel and $\mathbf{I}$ is the input from the previous layer. The output is calculated at locations (i, j) that are shifted with the *stride*. After the kernel has traversed the whole input, the complete feature map is built. A layer may have multiple kernels, which gives multiple output feature maps. When multiple feature maps are used as input to a convolution layer, one output of layer l is:

$$\mathbf{I}_l(i,j) = \sigma\left(b + \sum_k \sum_m \sum_n \mathbf{I}_{l-1}(i-m, j-n, k)\mathbf{K}_l(m,n,k)\right) \tag{2}$$

where k is the index of the feature maps in the input $\mathbf{I}_{l-1}$ from the previous layer, b is the bias and σ is the activation function.

The convolution is usually followed by a nonlinear activation function, like a Rectified Linear Unit (ReLU), which reduces computation time without reducing accuracy [11], [16], [17], [28]. The ReLU function is:

$$\sigma(x) = \begin{cases} x, & x \geq 0. \\ 0, & x < 0. \end{cases} \tag{3}$$

Pooling is a technique to down-sample the feature maps. Additionally, the pooling technique makes the network become less sensitive to translations of the input [4]. Max pooling [29] is a common choice of pooling technique. Pooling works by sliding a window over the feature maps, and in the case of max pooling, the maximum value from that window will be kept. The size of the window and the stride determines the size of the output maps.

The fully connected layers in a CNN are equal to the layers in a Multilayer Perceptron (MLP). The output of neuron j of the intermediate layer l is

$$a_j^l = \sigma(z_j^l) \tag{4}$$

where σ is the activation function, and where the argument is the weighted sum

$$z_j^l = b_j^l + \sum_k w_{jk}^l a_k^{l-1} \tag{5}$$

of the outputs a_k^{l-1} from neuron k in layer $l-1$. Here w_{jk}^l are weight factors, and b_j^l is the bias. In the first layer $l = 1$, and the input is $a_j^0 = x_j$, where x_j, $j = 1, \ldots, N$ are the inputs of the neural network. As the layer is fully connected, the weighted sum of the outputs of layer $l-1$ is over all neurons k.

The technique that enable neural networks to generalize on training data is the supervised learning technique backpropagation. Backpropagation is a method that calculates the error of the network prediction and then minimizes it by calculating the gradient backwards through the network and updating the weights so that the error decreases. We use the loss function in terms of the cross entropy [30]

$$C = -\sum_{j=1}^{n} y_j \log(\hat{y}_j) \tag{6}$$

where $\hat{y}_j = a_j^L = \sigma(z_j^L)$ is the predicted probability value for class j in the final layer L, and y_j is the true probability. For our classification problems the activation function in the output layer is the softmax function [4]:

$$\sigma(\mathbf{z})_j = \frac{\exp(z_j)}{\sum_{k=1}^{K} \exp(z_k)} \tag{7}$$

The true probability has a one-hot encoding, i.e. $y_j = 1$ for the correct class, while $y_j = 0$ for all other values of j [4]. This error is propagated backwards in the network and the weights of the neurons are updated to minimize the error. The flow in a CNN is shown in Figure 3.

B. WEIGHT INITIALIZATION

The initialization of the weights in the network has a high influence in the networks ability to train, especially as networks get deeper. Too small weights and the signal will shrink as it passes through each layer until it is too small to be useful. Too large weights will cause the signal to explode in amplitude as it goes deeper. Glorot and Bengio expanded upon this problem and derived the Xavier weight initialization, which is given by [31]

$$\mathrm{Var}[w] = \frac{2}{n_{in} + n_{out}} \tag{8}$$

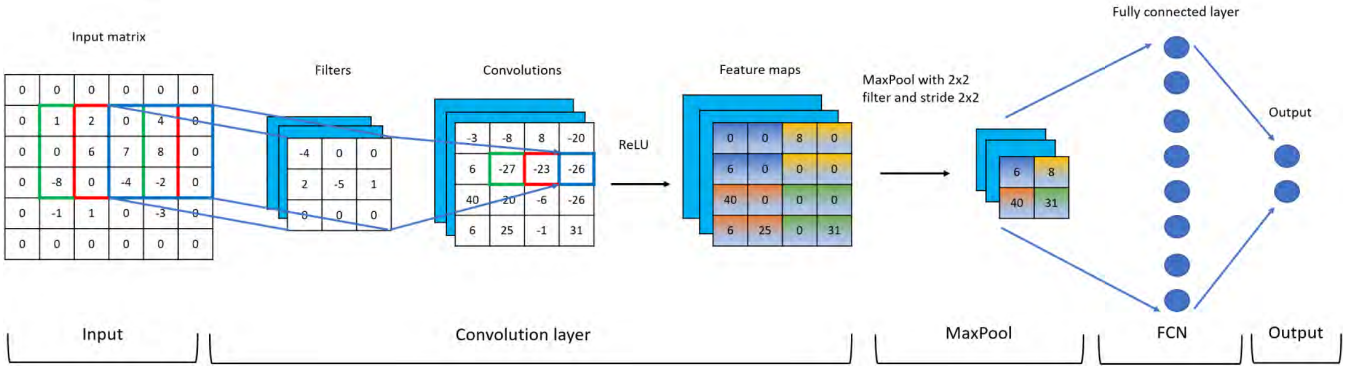


FIGURE 3. Calculations for a 6 × 6 × 1 input in a simple Convolutional Neural Network consisting of one convolution layer with 3 3 × 3 filters with ReLU activation, one MaxPool layer with 2 × 2 filter and stride 2 × 2, one fully connected layer with 8 neurons and one output layer with 2 neurons. Calculations for the fully connected layers are omitted. The light blue planes represent additional filters, convolutions and feature maps.

where n_{in} is the number of connections in to a neuron, and n_{out} is the number of connections out of a neuron. The equation is often simplified to

$$\text{Var}[w] = \frac{1}{n_{in}} \tag{9}$$

However, the weight initialization is based on the assumption that the activation is linear, and have been proven to work fairly well for traditional nonlinear activation functions such as hyperbolic tangent and the sigmoid function. For more recent activation functions, such as ReLU and PReLU, this assumption no longer applies and the weight initialization must be tailored according to the activation function, which makes it possible to ensure that even deeper CNNs converge [32]. The ReLU weight initialization is

$$\text{Var}[w] = \frac{2}{n_{in}} \tag{10}$$

due to a rectified linear unit outputting zero for half its inputs when the distribution has zero mean.

C. DROPOUT

As proposed by Srivastava *et al.*, dropout may facilitate the ability of a network to generalize training data and avoid overfitting [33]. Dropout is a regularization technique, but instead of adding a penalty to weights in a neuron, it cuts a certain percentage of the neurons in a layer during a forward pass. Typically, the layers with dropout will need an increased number of neurons, which increases the training time.

D. DATA AUGMENTATION

For some applications it may be difficult to get sufficient labelled training data. A solution is to use data augmentation, which makes it possible to artificially increase the training data, which enables the neural network to train better and achieve higher accuracy on cross validation and test sets. Initially the training dataset for our approach consisted of 43 images of class 1 and 57 images of class 2. Since our image classification is not dependent on the position and rotation of the characteristic separating the two classes, some simple steps can be applied to increase the dataset. First, the dataset was extended by rotating the images. Each images was rotated 3 times by 90 degrees, and for each rotation a copy was made, thereby increasing the dataset fourfold. Then, each of the images was cropped four times, retaining 80% of the corners. The augmentation process is shown in Figure 4. This process increases the dataset without increasing the complexity [34].

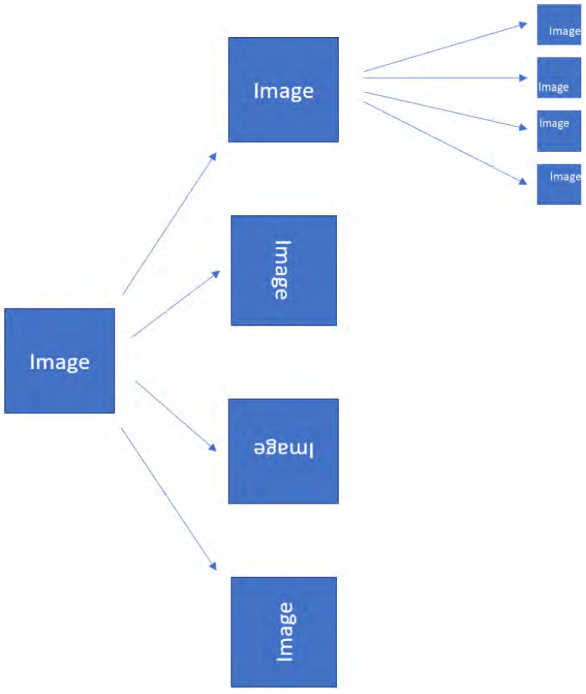


FIGURE 4. The data augmentation process. The end result is a ×16 increase in dataset size.

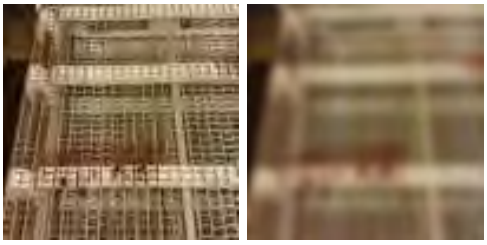
E. BLURRING

Since the environment in the processing plant after cleaning is very wet and humid, the lens of the camera may get tainted. In some cases there can be a visible mist in the plant, which will reduce the quality of the images. For this reason,

a dataset with blurred images is also used for training [35]. The blurring technique is Gaussian filtering [36]

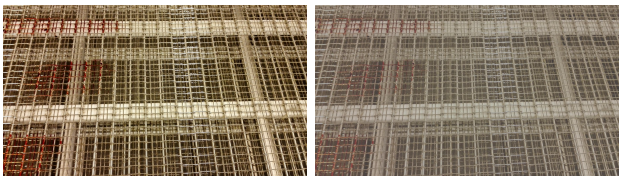
$$I(x, y) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{d^2}{2\sigma^2}\right) \tag{11}$$

where $d = \sqrt{(x - x_c)^2 + (y - y_c)^2}$, (x_c, y_c) is the center of the filter and σ is the standard deviation. For our experiments $\sigma = 2$ is used. An example of an original and blurred image is shown in Figure 5.



(a) (b)

FIGURE 5. Comparison of an original image, and one which is blurred. (a) Not blurred. (b) Blurred.



(a) (b)

FIGURE 6. Comparison of an original image, and one with reduced contrast. (a) Original contrast. (b) Reduced contrast.

F. CONTRAST SHIFT

In order to make the CNN more resistant to varying conditions, we generate a dataset using contrast shift [37]. Images are generated with both higher and lower contrast. For the images with the lower contrast the contrast is reduced to 50% of the contrast in the original image. For the images with increase contrast the images are scaled such that the bottom and upper 1% of the pixels are saturated. An example of reduced contrast is shown in Figure 6.

G. ILLUMINATION

The light conditions inside processing plants may vary significantly from plant to plant. Additionally, the potential mist inside the plant after the cleaning procedure may affect the illumination. In order to achieve tolerance to variation in lighting conditions, the lighting conditions in the images in the dataset are manipulated [38]. Both brighter and darker images are generated. New images can be obtained by

$$I_{out}(x, y) = I_{in}(x, y) + \gamma \tag{12}$$

where γ is the change in illumination. The effect of changing the illumination is shown in Figure 7.

(a) (b)

FIGURE 7. Comparison of a bright and a dark image. (a) Bright image. (b) Dark image.

H. IMAGE TRANSFORM

To increase the robustness of the trained network and reduce the problem of overfitting on the training set, experiments were done using further data augmentation on the training data. This is hereafter referred to as Image Transform (IT). The Image Transform consisted of flipping and warping images [15], [34]. The Image Transform steps are chained together with different probabilities:

$$I_{out} = I_{in} * \text{Flip}(x_1) * \text{Flip}(x_2) * \text{Warp}(x_3) \tag{13}$$

where x_1 is the probability of flipping the image by the first transform, x_2 is the probability of flipping it the second time, and x_3 is the probability of warping the image. The flip transforms has a equal chance of not flipping it, horizontally flipping it, vertically flipping it or flipping it both vertically and horizontally. For the warp transform the function is

$$I_{out}(x, y) = I_{in}\left(\frac{M_{11}x + M_{12}y + M_{13}}{M_{31}x + M_{32}y + M_{33}}, \frac{M_{21}x + M_{22}y + M_{23}}{M_{31}x + M_{32}y + M_{33}}\right) \tag{14}$$

where M is the transformation matrix

$$\begin{bmatrix} t_i x_i' \\ t_i y_i' \\ t_i \end{bmatrix} = M \cdot \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \tag{15}$$

where (x_i, y_i) are the quadrangle vertices

$$\begin{bmatrix} 0 & \text{cols} & \text{cols} & 0 \\ 0 & 0 & \text{rows} & \text{rows} \end{bmatrix}$$

where *cols* and *rows* are the number of columns and rows in the image, respectively, while (x_i', y_i') are the quadrangle vertices

$$\begin{bmatrix} \delta * r_1 & \text{cols} + \delta * r_3 & \text{cols} + \delta * r_5 & \delta * r_7 \\ \delta * r_2 & \delta * r_4 & \text{rows} + \delta * r_6 & \text{rows} + \delta * r_8 \end{bmatrix}$$

where r_i, $i = 1 \ldots 8$, are random numbers that are uniformly distributed in $[-1, 1]$ and δ is the maximum deviation from the original position.

I. EARLY STOPPING

A very simple technique to avoid overfitting is early stopping. Early stopping means that the training is terminated when the accuracy on the cross validation data starts to decrease.

J. DOWN-SAMPLING

Before all the data manipulation steps, the images are down-sampled to 100×100 by bicubic interpolation [39] to reduce the computational cost.

K. DATASETS

Using the techniques described, several datasets are generated. Dataset 1 is the base dataset where rotation and cropping has been applied to increase the size, without increasing the complexity of the dataset. Dataset 2 consists of blurred images obtained by the technique explained in Subsection III-E. Dataset 3 has both images with higher and with lower contrast as explained by Subsection III-F, and thus has twice as many elements compared to dataset 1 and 2. Similarly, dataset 4 also has twice as many elements as dataset 1 and 2, using the technique described in Subsection III-G. Dataset 5 is a combination of dataset 1, 2, 3 and 4, and thus the largest and most complex. An overview of the datasets is shown in Table 1.

TABLE 1. Datasets.

Name	Description	Elements of Class 1	Class 2
Dataset 1	Base augmented dataset	688	912
Dataset 2	Blurred dataset	688	912
Dataset 3	Contrast manipulation	1376	1824
Dataset 4	Illumination	1376	1824
Dataset 5	All	4128	5472

L. PERFORMANCE EVALUATION

For our performance evaluation we use accuracy:

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{P} + \text{N}} \tag{16}$$

where TP is true positive, TN true negative, P all the images with blood and N all the images without blood.

The datasets are divided into 80% training data and 20% cross validation data for the proposed CNN approach. The CNN approach is trained on the training data and evaluated both on the training data and the cross validation data. An important problem in machine learning is overfitting. Overfitting is when the machine learning algorithm learn the quirks and noise of the training data instead of general trends. When this occurs the network achieves a much higher accuracy on the training data than on the cross validation data. Therefore, the accuracy on the cross validation data is the most important performance measurement as it shows how well the network is able to predict on data it has never seen before.

IV. ALTERNATIVE TECHNIQUES

To investigate the performance of the proposed approach, a comparison with alternative techniques is done in the experiments. The alternative techniques used are based on color thresholding.

A. COLOR THRESHOLD TECHNIQUE

An argument could be made that the classification could simply be solved with a color threshold algorithm. Two different color threshold techniques, one in RGB color space and one in HSV color space, are presented to give a robust comparison. The threshold function for each pixel using the RGB color model is

$$\sigma = \begin{cases} 1, & R \geq G + \theta_G \text{ and } R \geq B + \theta_B \\ 0, & \text{otherwise.} \end{cases} \tag{17}$$

where θ_G and θ_B are the green and blue color threshold, respectively, R is the red color value, G green and B blue. For the HSV color model the threshold function is

$$\sigma = \begin{cases} 1, & \theta_{HL} \leq H \leq \theta_{HH}, S \geq \theta_S \text{ and } V \geq \theta_V \\ 0, & \text{otherwise.} \end{cases} \tag{18}$$

where θ are the different thresholds. The number of pixels above the threshold, as well as the thresholds themselves can then be tuned in order to achieve the highest accuracy on the classification problem.

RGB is the most common color model. RGB stand for Red, Green and Blue, and a color is represented by three numbers, one for each of the base colors. An alternative to RGB is the HSV color space as it is more intuitive. HSV stands for Hue, Saturation and Value, where hue is the color, saturation decides how intense the color is (0 saturation is white), and value determines the shade of the color.

B. CNN AS COLOR THRESHOLD TECHNIQUE

The color threshold technique described in the previous section checks every single pixel independently, and the same threshold values is used on all the pixels regardless of the location of the pixel. This is similar to how a 1×1 filter of a CNN operates, with the exception that the output is continuous for the CNN instead of binary as with the threshold. Then, in the thresholding technique, all the pixels above the thresholds are summed to determine which class the image belongs to. Therefore one can argue that with the correct weight initialization, a CNN with a single layer with 1×1 filters, a max pooling layer with a kernel size that result in a 1×1 output feature map and ReLU activation could do a similar operation as the color threshold technique presented in the previous section performs. The proposed network architecture is shown in Figure 2.

V. EXPERIMENTS

A. BASELINE EXPERIMENTS USING COLOR THRESHOLD

In order to establish a baseline for performance evaluation of the proposed method, a number of tests were done with the color thresholding technique, which is an alternative technique. The best parameters of the thresholding technique were found by an exhaustive search. All the experiments on the color thresholding technique used 100% of the dataset. This is because the color thresholding technique does not have the same ability to overfit compared to a machine learning algorithm. The results of the color thresholding technique, which are shown in Table 3, will be compared to the results

TABLE 2. Network architecture of CNN as color threshold technique. Variations of the network use different number of filters.

Layer	Type	Kernel size	Stride	Filters/outputs
1	Conv	1×1	1×1	x
2	MaxPool	100×100	100×100	x
3	Output	-	-	2

TABLE 3. Accuracy using color threshold.

Accuracy	Color threshold RGB	Color threshold HSV
Dataset 1	0.985	0.989
Dataset 2	0.970	0.964
Dataset 3	0.789	0.782
Dataset 4	0.929	0.879
Dataset 5	0.887	0.843

TABLE 4. Accuracy using CNN color threshold on dataset 1.

Num filters	Weight init	Training accuracy	Cross validation accuracy
1	Xavier	0.500	0.500
1	ReLU	0.500	0.500
2	Xavier	**0.982**	**0.989**
2	ReLU	**0.982**	**0.989**
20	Xavier	0.983	0.982
20	ReLU	0.981	0.982

TABLE 5. Accuracy using CNN color threshold on dataset 2.

Num filters	Weight init	Training accuracy	Cross validation accuracy
1	Xavier	0.500	0.500
1	ReLU	0.500	0.500
2	Xavier	0.985	0.964
2	ReLU	**0.985**	**0.975**
20	Xavier	0.981	0.964
20	ReLU	0.981	0.967

TABLE 6. Accuracy using CNN color threshold on dataset 3.

Num filters	Weight init	Training accuracy	Cross validation accuracy
1	Xavier	0.500	0.500
1	ReLU	0.500	0.500
2	Xavier	0.723	0.745
2	ReLU	0.716	0.729
20	Xavier	0.911	0.927
20	ReLU	**0.914**	**0.929**

TABLE 7. Accuracy using CNN color threshold on dataset 4.

Num filters	Weight init	Training accuracy	Cross validation accuracy
1	Xavier	0.500	0.500
1	ReLU	0.500	0.500
2	Xavier	0.902	0.909
2	ReLU	**0.901**	**0.911**
5	Xavier	0.892	0.904
5	ReLU	0.896	0.898
20	Xavier	0.889	0.898
20	ReLU	0.884	0.885

from the proposed approach. It is seen from the table that the color thresholding technique performed well on dataset 1 and 2, and not as well on dataset 3, 4 and 5.

Also CNN as a color thresholding technique was evaluated as a baseline. The network is shown in Table 2. The results on the different datasets are presented in Tables 4, 5, 6, 7 and 8. Using CNN as strictly a color threshold technique achieved approximately the same accuracy as using a conventional threshold technique. The exception is dataset 3, where color threshold using CNN performed better when the number of filters were increased.

TABLE 8. Accuracy using CNN color threshold on dataset 5.

Num filters	Weight init	Training accuracy	Cross validation accuracy
1	Xavier	0.500	0.500
1	ReLU	0.500	0.500
2	Xavier	0.872	0.870
2	ReLU	0.863	0.849
5	Xavier	0.862	0.849
5	ReLU	0.877	0.865
20	Xavier	0.874	0.869
20	ReLU	**0.878**	**0.877**

The complexity of the datasets increase from dataset 1 to 5, with 1 being the least complex and 5 being the most complex. On the least complex dataset, dataset 1, the best result achieved by a color threshold technique was 98.9%. In contrast to this, the best result achieved by a color threshold technique on the most complex dataset, which is dataset 5, was 88.7%. An important observation is that the most complex dataset will be the dataset that best reflect the variations in images captured in a real world application. It is therefore an important performance indicator for a method that it has a high accuracy for dataset 5.

B. EXPERIMENTAL VALIDATION OF PROPOSED APPROACH WITHOUT IT

Next, the proposed CNN method was tested in experiments. Dataset 5 was chosen for all the experiments on the proposed approach due to its size and complexity. The experiments were performed on 10 different CNN architectures. These CNN architectures had 6-8 layers, where the two last layers were FCN. The size of the kernels and the number of neurons were changed in the different network architectures. The architectures are characterized by the parameters given in Table 9, while the hyperparameters are given in Table 10. The results for the 10 CNN networks are documented in Table 11.

C. EXPERIMENTAL VALIDATION OF PROPOSED APPROACH WITH IT

Next, we did experiments using the Image Transform (IT) technique on the training data. In the experiments without IT the best result were obtained with Architecture 1, which is seen from Table 11. Therefore this architecture was selected for the experiments with IT. Different versions of IT were tested. Each experiment on the use of IT was run three times, and the average values for the cross validation accuracy and training time measured in epochs are presented in the following. First, experiments with different values for the warp δ in the WarpTransform were performed. The rest of the settings of the IT were selected as 0.9 probability for FlipTransform 1, 0.8 for FlipTransform 2 and 0.5 probability for the WarpTransform. The cross validation accuracy is shown in Figure 8.

The next experiment was performed to investigate the optimal value of the probability of the WarpTransform. The best value for the Warp δ of the previous experiment, which was $\delta = 30$, was used in this experiment. The cross validation accuracy as a function of warp probability is shown in

TABLE 9. Various network structures tested. Ouput layer omitted.

1	Layer	Layer type	Kernel	Stride	Filters/neurons	Dropout
	1	Conv	5×5	1×1	6	0.0
	2	MaxPool	2×2	2×2	6	0.0
	3	Conv	5×5	1×1	16	0.0
	4	MaxPool	2×2	2×2	16	0.0
	5	FCN			120	0.0
	6	FCN			84	0.0

2	Layer	Layer type	Kernel	Stride	Filters/neurons	Dropout
	1	Conv	5×5	1×1	6	0.0
	2	MaxPool	2×2	2×2	6	0.0
	3	Conv	5×5	1×1	14	0.0
	4	MaxPool	2×2	2×2	14	0.0
	5	FCN			120	0.0
	6	FCN			84	0.0

3	Layer	Layer type	Kernel	Stride	Filters/neurons	Dropout
	1	Conv	5×5	1×1	8	0.0
	2	MaxPool	2×2	2×2	8	0.0
	3	Conv	5×5	1×1	20	0.0
	4	MaxPool	2×2	2×2	20	0.0
	5	FCN			120	0.0
	6	FCN			84	0.0

4	Layer	Layer type	Kernel	Stride	Filters/neurons	Dropout
	1	Conv	3×3	1×1	6	0.0
	2	MaxPool	2×2	2×2	6	0.0
	3	Conv	3×3	1×1	16	0.0
	4	MaxPool	2×2	2×2	16	0.0
	5	FCN			120	0.0
	6	FCN			84	0.0

5	Layer	Layer type	Kernel	Stride	Filters/neurons	Dropout
	1	Conv	1×1	1×1	6	0.0
	2	MaxPool	2×2	2×2	6	0.0
	3	Conv	1×1	1×1	16	0.0
	4	MaxPool	2×2	2×2	16	0.0
	5	FCN			120	0.0
	6	FCN			84	0.0

6	Layer	Layer type	Kernel	Stride	Filters/neurons	Dropout
	1	Conv	9×9	1×1	6	0.0
	2	MaxPool	2×2	2×2	6	0.0
	3	Conv	9×9	1×1	16	0.0
	4	MaxPool	2×2	2×2	16	0.0
	5	FCN			120	0.0
	6	FCN			84	0.0

7	Layer	Layer type	Kernel	Stride	Filters/neurons	Dropout
	1	Conv	5×5	1×1	6	0.0
	2	MaxPool	2×2	2×2	6	0.0
	3	Conv	5×5	1×1	16	0.0
	4	MaxPool	2×2	2×2	16	0.0
	5	FCN			120	0.5
	6	FCN			84	0.0

8	Layer	Layer type	Kernel	Stride	Filters/neurons	Dropout
	1	Conv	5×5	1×1	6	0.0
	2	MaxPool	2×2	2×2	6	0.0
	3	Conv	5×5	1×1	16	0.0
	4	MaxPool	2×2	2×2	16	0.0
	4	Conv	5×5	1×1	22	0.0
	5	MaxPool	2×2	2×2	22	0.0
	6	FCN			120	0.0
	7	FCN			84	0.0

9	Layer	Layer type	Kernel	Stride	Filters/neurons	Dropout
	1	Conv	5×5	1×1	6	0.0
	2	MaxPool	2×2	2×2	6	0.0
	3	Conv	5×5	1×1	16	0.0
	4	MaxPool	2×2	2×2	16	0.0
	4	Conv	5×5	1×1	30	0.0
	5	MaxPool	2×2	2×2	30	0.0
	6	FCN			120	0.0
	7	FCN			84	0.0

10	Layer	Layer type	Kernel	Stride	Filters/neurons	Dropout
	1	Conv	5×5	1×1	6	0.0
	2	MaxPool	2×2	2×2	6	0.0
	3	Conv	5×5	1×1	16	0.0
	4	MaxPool	2×2	2×2	16	0.0
	4	Conv	5×5	1×1	30	0.0
	5	MaxPool	2×2	2×2	30	0.0
	6	Conv	5×5	1×1	30	0.0
	7	FCN			120	0.0
	8	FCN			84	0.0

TABLE 10. Hyperparameters for the proposed CNN approach.

Hyperparameter	Type
Activation function CNN layers	ReLU
Activation function FCN layers	ReLU
Activation function output layer	SoftMax
Weight initialization	ReLU
Batch size	128
Learning rate	0.001
Updater	Learning rate only
L2 regularization	0.0001
Loss function	Cross entropy

TABLE 11. Accuracy using network architectures shown in table 9 and hyperparameters shown in table 10 on dataset 5. All networks trained without Image Transform.

Network	Training accuracy	Cross validation accuracy
1	**0.998**	**0.971**
2	0.980	0.931
3	0.500	0.500
4	0.978	0.887
5	0.817	0.758
6	0.663	0.655
7	0.836	0.813
8	0.633	0.617
9	0.982	0.965
10	0.796	0.764

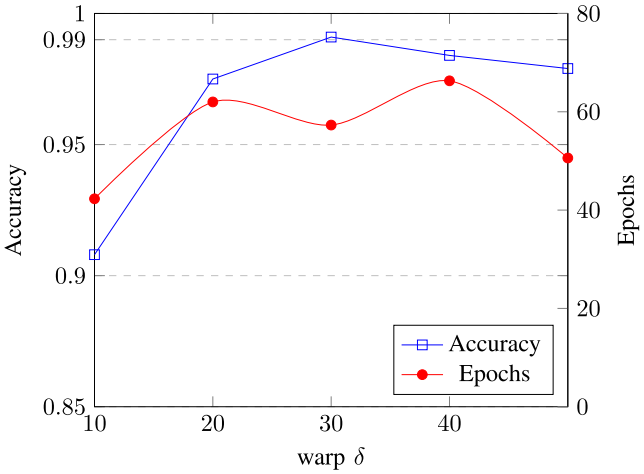


FIGURE 8. Accuracy on cross validation as a function of warp δ on architecture 1 with Image Transform. 0.5 probability for WarpTransform, 0.9 probability for FlipTransform 1 and 0.8 probability FlipTransform 2. The training time measured in epochs is also shown. Average of 3 trainings.

Figure 9. A warp probability over 0.5 seemed to reduce the cross validation accuracy.

Finally, experiments were performed to study the effect of different values of the probability of the FlipTransforms. In the first FlipTransform experiment the effect of varying the probability of one of FlipTransform 1 was studied. The remaining parameters were 0.8 probability of FlipTransform 2, 0.5 probability of WarpTransform and warp $\delta = 30$. The resulting cross validation accuracy as a function of the flip probability is shown in Figure 10. The best result was a cross validation accuracy of 99.27%. This result was obtained with 0.0 probability for FlipTransform 1. This indicated that

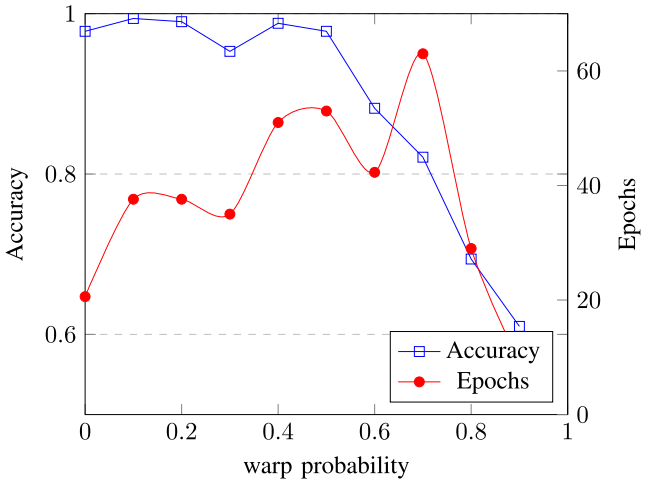


FIGURE 9. Accuracy on cross validation as a functions of percentage of warp in the training set. Warp δ set to 30. 0.9 and 0.8 probability for FlipTransform 1 and 2, respectively. The training time measured in epochs is also shown. Average of 3 trainings.

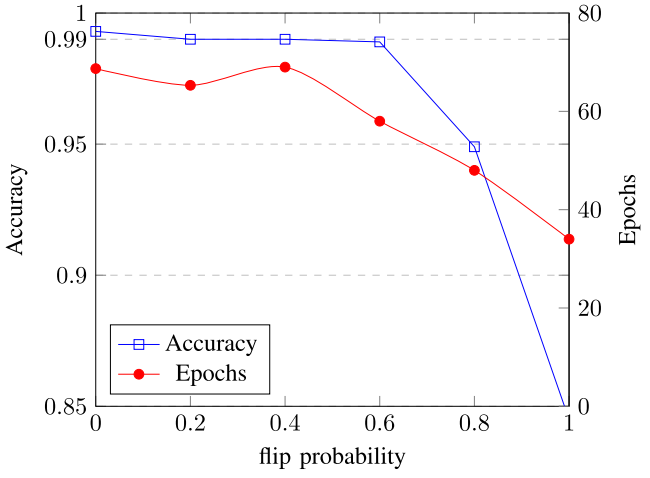


FIGURE 10. Accuracy on cross validation as a function of probability of FlipTransform 1 in the training set. 0.8 probability for FlipTransform 2, Warp δ set to 30. 0.5 probability for WarpTransform.The training time measured in epochs is also shown. Average of 3 trainings.

only one FlipTransform is required to achieve the best accuracy. Therefore, the next FlipTransform experiment used only one FlipTransform. The parameters for the WarpTransform were 0.5 probability and warp $\delta = 30$. The resulting cross validation accuracy as a function of the flip probability is shown in Figure 11. The highest cross validation accuracy was 98.93%, which was achieved with 40% probability of FlipTransform. With the same parameters as in the experiment documented in Figure 10, where the cross validation accuracy was 99.27%, this experiment achieved 98.55%.

The experiments showed that the inclusion of at least one FlipTransform gave better accuracy. For all the various network architectures the average training time for one epoch on Dataset 5 was 7143 ms with a Nvidia Geforce GTX 1070 GPU card. The total training time for the network configurations varied substantially due to early stopping.

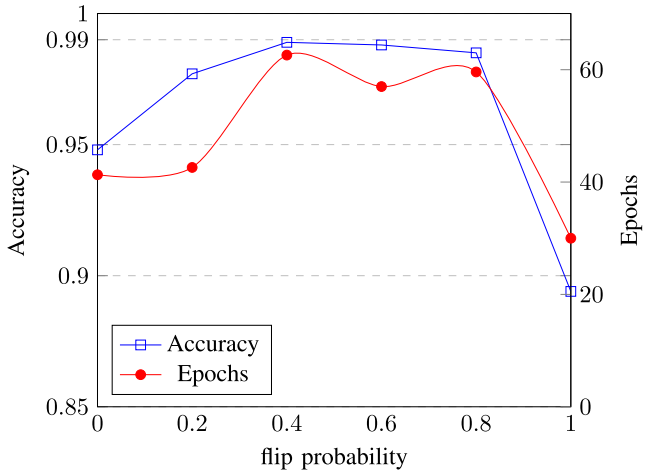


FIGURE 11. Accuracy on cross validation as a function of probability of flip in the training set. Only one FlipTransform. Warp δ set to 30. 0.5 probability for WarpTransform. The training time measured in epochs is also shown. Average of 3 trainings.

VI. CONCLUSION AND FUTURE WORK

The best result achieved by the proposed CNN approach on the most complex dataset were 99.27%. The best result of the color threshold techniques on the most complex dataset were 88.7%. This is a reduction in error rate of 93.5%. This shows that the proposed CNN approach is superior to the color threshold techniques on the most complex dataset. The most complex dataset best accounts for variations in images captured in a real world application. Therefore it is desirable to achieve a high accuracy on this dataset.

Regarding the alternative methods that were used for comparison with the proposed method, the results of using CNN as strictly a color threshold technique achieved approximately the same accuracy as using a conventional threshold technique. The exception is dataset 3, where color threshold using CNN performed better when the number of filters were increased.

Data augmentation techniques were used to increase the size and complexity of the dataset. The increased complexity became obvious when using simple color threshold techniques. The proposed CNN approach is able to learn more complex datasets, thus producing an algorithm that is more robust to blurring, variation in contrast and poor illumination. This enables the proposed approach for real world application. The increased accuracy of the proposed CNN approach compared to color threshold techniques may be due to spatial information in the blood stains. This enables CNN to better distinguish between blood and other sources of red color compared to a color threshold technique.

An important problem in machine learning is overfitting. As is evident in Table 11, the CNN solution was able to achieve very good training accuracy on dataset 5, even without IT. However, some overfitting is present, limiting the accuracy on the cross validation dataset.

The experiments with IT showed that this technique enables the CNN system to better generalize trends in the training data, instead of overfitting. In several of the experiments the accuracy on the cross validation set reached over 99% accuracy.

The best result ever achieved was 99.94%. This was with Architecture 1 using IT. The settings for the IT were one FlipTransform with 0.8 probability, and one WarpTransform with $\delta = 30$ and probability of 0.5. However, due to the randomness involved in the neural network training procedure, both related to weight initialization and to the transforms, it is important to note that the average of 3 runs with the same settings achieved 99.27%. This is shown in Figure 10. The randomness also explains why the same settings in Figure 11 results in 98.55% accuracy.

For the relatively shallow networks used in all the experiments, the difference between Xavier and ReLU weight initialization turned out to be negligible.

Future work will include the use of UV light and spectral imaging for detecting biofilm on the cleaned surfaces. This will give a more accurate classification of which of the surfaces are sufficiently clean and which are not, as the absence of blood will not necessarily exclude the presence of biofilms. Biofilms is a more true sign of whether a surface is clean or not in a food-contact context. Detection of fish debris will also be necessary to decide if the equipment is sufficiently cleaned, as fish debris may be present after conventional cleaning. The CNN and data augmentation approach will still be applicable for biofilm detection in spectral images. Some work has already looked at using CNN for spectral images [40] and some work has looked at using hyperspectral imaging for detecting microbial biofilm on food contact surfaces [41]. Combining CNN and spectral images for biofilm detection is a promising approach.

ACKNOWLEDGMENT

This research was funded by the Research Council of Norway, grant no. 245613/O30. We thank Lars Andre Giske for help with capturing the images.

CONFLICT OF INTEREST

The author declares that there is no conflict of interest regarding the publication of this article.

REFERENCES

[1] L. Gram and H. H. Huss, "Microbiological spoilage of fish and fish products," *Int. J. Food Microbiol.*, vol. 33, no. 1, pp. 121–137, Nov. 1996. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0168160596011348

[2] T. Møretrø, B. Moen, E. Heir, Å. Hansen, and S. Langsrud, "Contamination of salmon fillets and processing plants with spoilage bacteria," *Int. J. Food Microbiol.*, vol. 237, pp. 98–108, Nov. 2016.

[3] Y. Christi, "Process hygiene: Modern systems of plant cleaning," in *Encyclopedia of Food Microbiology*. London, U.K.: Academic Press, 2014, pp. 1806–1815.

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[5] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in Neural Information Processing Systems*, D. S. Touretzky, Ed. Burlington, MA, USA: Morgan-Kaufmann, 1990, pp. 396–404. [Online]. Available: http://papers.nips.cc/paper/293-handwritten-digit-recognition-with-a-back-propagation-network.pdf

[6] Y. L. Cun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[7] K. Chellapilla, S. Puri, and P. Simard, "High performance convolutional neural networks for document processing," in *Proc. 10th Int. Workshop Frontiers Handwriting Recognit.*, Oct. 2006.

[8] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[9] M. Ranzato, C. Poultney, S. Chopra, and Y. L. Cun, "Efficient learning of sparse representations with an energy-based model," in *Proc. 19th Int. Conf. Neural Inf. Process. Syst.* Cambridge, MA, USA: MIT Press, 2006, pp. 1137–1144. [Online]. Available: http://dl.acm.org/citation.cfm?id=2976456.2976599

[10] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, and U. Montreal, "Greedy layer-wise training of deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 19, Jan. 2007, pp. 153–160.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, Jan. 2012, pp. 1097–1105.

[12] J. Flusser and T. Suk, "Pattern recognition by affine moment invariants," *Pattern Recognit.*, vol. 26, no. 1, pp. 167–174, 1993.

[13] L. S. Yaeger, R. F. Lyon, and B. J. Webb, "Effective training of a neural network character classifier for word recognition," in *Proc. NIPS*, 1996, pp. 807–816.

[14] W. Tao, M. C. Leu, and Z. Yin, "American sign language alphabet recognition using convolutional neural networks with multiview augmentation and inference fusion," *Eng. Appl. Artif. Intell.*, vol. 76, pp. 202–213, Nov. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0952197618301921

[15] L. Taylor and G. S. Nitschke, "Improving deep learning using generic data augmentation," *CoRR*, vol. abs/1708.06020, 2017.

[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv 1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[18] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[19] M. V. dos Santos Ferreira, A. O. de Carvalho Filho, A. D. de Sousa, A. C. Silva, and M. Gattass, "Convolutional neural network and texture descriptor-based automatic detection and diagnosis of glaucoma," *Expert Syst. Appl.*, vol. 110, pp. 250–263, Nov. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417418303567

[20] P. Tiwari, J. Qian, Q. Li, B. Wang, D. Gupta, A. Khanna, J. J. P. C. Rodrigues, and V. C. H. de Albuquerque, "Detection of subtype blood cells using deep learning," *Cognit. Syst. Res.*, vol. 52, pp. 1036–1044, Dec. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389041718303760

[21] L. H. Vogado, R. M. S. Veras, F. H. D. Araujo, R. R. V. Silva, and K. R. T. Aires, "Leukemia diagnosis in blood slides using transfer learning in CNNs and SVM for classification," *Eng. Appl. Artif. Intell.*, vol. 72, pp. 415–422, Jun. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0952197618301039

[22] B. Penna, T. Tillo, M. Grangetto, E. Magli, and G. Olmo, "A technique for blood detection in wireless capsule endoscopy images," in *Proc. 17th Eur. Signal Process. Conf.*, Aug. 2009, pp. 1864–1868.

[23] M. W. Mackiewicz, M. Fishe, and C. Jamieson, "Bleeding detection in wireless capsule endoscopy using adaptive colour histogram model and support vector classification," *Proc. SPIE*, vol. 6914, Mar. 2008, Art. no. 691412. doi: 10.1117/12.770510.

[24] S. Benalia, S. Cubero, J. M. Prats-montalbán, B. Bernardi, G. Zimbalatti, and J. Blasco, "Computer vision for automatic quality inspection of dried figs (Ficus carica L.) in real-time," *Comput. Electron. Agricult.*, vol. 120, pp. 17–25, Jan. 2016.

[25] C. Iglesias, J. Martínez, and J. Taboada, "Automated vision system for quality inspection of slate slabs," *Comput. Ind.*, vol. 99, pp. 119–129, Aug. 2018.

[26] R. Kestur, A. Meduri, and O. Narasipura, "MangoNet: A deep semantic segmentation architecture for a method to detect and count mangoes in an open orchard," *Eng. Appl. Artif. Intell.*, vol. 77, pp. 59–69, Jan. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0952197618301970

[27] E. Bjørlykhaug and O. Egeland, "Mechanical design optimization of a 6dof serial manipulator using genetic algorithm," *IEEE Access*, vol. 6, pp. 59087–59095, 2018.

[28] A. F. M. Agarap, "Deep learning using rectified linear units (ReLU)," *CoRR*, vol. abs/1803.08375, 2018.

[29] Y. T. Zhou and R. Chellappa, "Computation of optical flow using a neural network," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 2, Jul. 1988, pp. 71–78.

[30] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012.

[31] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *J. Mach. Learn. Res.*, vol. 9, pp. 249–256, May 2010.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Washington, DC, USA, 2015, pp. 1026–1034. doi: 10.1109/ICCV.2015.123.

[33] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[34] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *Proc. 7th IEEE Int. Conf. Document Anal. Recognit.*, Edinburgh, U.K., Aug. 2003, pp. 958–963.

[35] T. Ahonen, E. Rahtu, V. Ojansivu, and J. Heikkila, "Recognition of blurred faces using local phase quantization," in *Proc. IEEE Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.

[36] L. G. Shapiro and G. Stockman, *The Visual Computer*, 1st ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2001.

[37] M. Signaevsky, M. Prastawa, K. Farrell, N. Tabish, E. Baldwin, N. Han, M. A. Iida, J. Koll, C. Bryce, D. Purohit, V. Haroutunian, A. Mckee, T. Stein, C. White, J. Walker, T. Richardson, R. Hanson, M. Donovan, C. Cordon-Cardo, and J. Crary, "Artificial intelligence in neuropathology: deep learning-based assessment of tauopathy," *Lab. Invest.*, vol. 15, no. 1, 2019.

[38] A. Shashua and T. Riklin-Raviv, "The quotient image: Class-based re-rendering and recognition with varying illuminations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 2, pp. 129–139, Feb. 2001.

[39] R. G. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 29, no. 6, pp. 1153–1160, Dec. 1981.

[40] C. Chen, F. Jiang, C. Yang, S. Rho, W. Shen, S. Liu, and Z. Liu, "Hyperspectral classification based on spectral-spatial convolutional neural networks," *Eng. Appl. Artif. Intell.*, vol. 68, pp. 165–171, Feb. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0952197617302671

[41] W. Jun, M. S. Kim, B.-K. Cho, P. D. Millner, K. Chao, and D. E. Chan, "Microbial biofilm detection on food contact surfaces by macro-scale fluorescence imaging," *J. Food Eng.*, vol. 99, no. 3, pp. 314–322, Aug. 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0260877410001147