International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:14, No:10, 2020

Automatic Product Identification Based on Deep-Learning Theory in an Assembly Line

Fidel López Saca, Carlos Avilés-Cruz, Miguel Magos-Rivera, José Antonio Lara-Chávez

Abstract—Automated object recognition and identification systems are widely used throughout the world, particularly in assembly lines, where they perform quality control and automatic part selection tasks. This article presents the design and implementation of an object recognition system in an assembly line. The proposed shapes-color recognition system is based on deep learning theory in a specially designed convolutional network architecture. The used methodology involve stages such as: image capturing, color filtering, location of object mass centers, horizontal and vertical object boundaries, and object clipping. Once the objects are cut out, they are sent to a convolutional neural network, which automatically identifies the type of figure. The identification system works in real-time. The implementation was done on a Raspberry Pi 3 system and on a Jetson-Nano device. The proposal is used in an assembly course of bachelor's degree in industrial engineering. The results presented include studying the efficiency of the recognition and processing time.

Keywords—Deep-learning, image classification, image identification, industrial engineering.

I. Introduction

IN recent decades, the constant growth of digital images, as a primary source of information representation for technical applications, has made image classification a common and challenging task. Besides, in the field of industrial engineering, particularly in assembly lines, there is a need to automate the processes of quality control and storage of the products generated. It is therefore important to propose an automatic identification of essential products to ensure process efficiency. One of the possible ways is through artificial vision, using cameras in the whole process of quality control and storage. In the assembly line, different products are produced. Thus, the proper identification of the product via digital image processing allows its proper selection to store a specific class of product.

In order to ensure a high classification rate, image processing techniques based on pattern recognition have been proposed, deep learning methods for instance, which are a main focus of study in image processing and computer vision nowadays. In deep learning method approach, the most popular architecture for the image classification tasks is convolutional neural networks (CNNs) containing multiple layers where each layer models a receptive field of the visual cortex, making it much more effective in machine vision tasks [1].

In this article, we propose an image processing system to identify colors and shapes for manufacturing assembly lines. The proposed scheme adopts an automatic method, using a digital camera and an *ad-hoc* digital video processing algorithm based on a deep-learning CNN classification. This research work aims to emulate the quality control in an automatic scheme of the production workflow. Moreover, the scheme provides the possibility of online identification of shape and color of a product. The implementation was done on a Raspberry Pi 3 system, and on a Jetson-Nano device involving a graphics processing unit GPU.

The rest of the paper is organized as follows. In Section II, the State-of-the-Art is presented. The system description and its implementation are presented in Sections III and IV. Results are shown in Section V. Finally, conclusions and future works are given in Section VI.

Fidel López-Saca, Carlos Avilés-Cruz, Miguel Magos-Rivera and José Antonio Lara-Chávez are with Departamento de Electrónica de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, Av. San Pablo 180, Ciudad de México, México (e-mail: fidel.lopez.saca@gmail.com).

II. State-of-the-Art

The current trend in industrial progress is to combine different methods and approaches from different fields of sciences to ensure reliable and efficient systems [2]. The impact of industry 4.0 has been witnessed in recent years in industrial, manufacturing, medical applications, etc. [3] [4] [5].

Computer vision has previously employed in the field of manufacturing by Jagtap et al. [6]. The authors propose an automated, real-time system that sorts potatoes and potato waste within the potato processing industry. Authors developed modern image processing and pattern recognition techniques. The potato analysis band, using a digital camera, was used to identify damaged and unusable potatoes. Later, through a CNN architecture, the type and characteristics of potato waste were determined. The proposed system achieved 99.79% accuracy in training, 94.06% accuracy in testing, and 85% accuracy validation, which is significantly better than the manual inspection and monitoring. Finally, the real-time data generated by this system helps those involved in potato production, transport, and processing to determine the various causes of waste generation and assist in the implementation of corrective actions. The second most related work is developed by Abdo et al. [7]. The authors use a set of sensors in series, synchronized to correct the workflow in a manufacturing process. The primary sensor used was a digital camera, developing specific purpose digital image processing algorithms to detect possible failures in the assembly line, as well as to decrease the response time of the sensors used in the procedure.Their proposed scheme was tested using a scale model of the assembly line in a laboratory.

Several works have been developed for manufacturing processes. Table I presents different examples of object identification in manufacturing processes.

TABLE I
EXAMPLES OF OBJECT IDENTIFICATION IN MANUFACTURING PROCESSES

Area	Work type/Reference
Lithography	Criticality of photo track monitoring for lithography defect control [8].
Metallurgy	A new automated procedure Hin line quality control system based on non-destructive evaluation for additive manufacturing of net-shape parts from particulates [9].
Ceramic	Use of an eye-tracker to assess workers in ceramic tile surface defect detection [10].
Food	Application of Digital Image Processing in Monitoring some Physical Properties of Tarkhineh during Drying [11].
Pharmaceutical	Bliss Bot for pharmaceutical inspection [12].
Cartridge production	Real-Time Visual Inspection and Rejection Machine for Bullet Production [13].
Manufacturing	Teaching optimization of manufacturing problems via code components of a Jupyter Notebook [14].

Regarding CNN, the theoretical foundations were introduced by Fukushima [15], later improved by LeCun [16] and fine-tuned by Ciresan [17]. Recently, CNN has been used to classify due to its incredible generalization capability. The CNN has the ability to train millions of parameters contained in its structure [18], and then, use the trained CNN for the classification task. Because of the hard training task and the limited number of labeled images, it has been suggested that fine-tuning a trained architecture with a robust database is better than training the same architecture from scratch [1].

In the last decade, the type of CNN networks and their paradigms for image classification have significantly grown. Among those, the ones that stand out are: Deep Recurrent Neural Network (DRNN) [19] and Long Short-Term Memory (LSTM) [20], DeepConvLSTM that combines convolutional and recurrent layers [21], and binarized bidirectional CNN [22].

Our proposal is based on an *ad-hoc* image processing task and a modern pattern recognition technique based on CNN, which are explained in the following section.

III. PROPOSAL

The proposal is used in the course of assembly of the bachelor's degree in industrial engineering. The architecture of the proposed object recognition system in an assembly line is shown in Fig. 1. The system developed consists of a conveyor belt, a webcam, two image processing systems, and a monitor to display the information of the identified objects. The image processing devices can be either a Raspberry Pi or a Jetson-Nano of Nvidia company.

- ***Camera***
  Object recognition starts with the webcam transferring image frames; the image is transferred in RGB format.
- ***Digital image processing device***
  The image is received in one of the processing units, being either a Raspberry Pi or Jetson-Nano device. Two sequential processes are carried out, as follows:
  1) The first stage of digital image processing, which allows identifying both the central position and the color of the object. Subsequently, the found object is passed to the second stage of pattern recognition.
  2) A deep CNN, designed specifically for our recognition purpose, is used.
- ***Monitor device***
  A high-resolution monitor is used to display the identified object. The monitor displays the captured and analyzed object in the camera's field of view, as well as the tag of its identification.
- ***Conveyor belt***
  The conveyor belt moves only one piece at a time with a constant speed. The parts come from an assembly line.

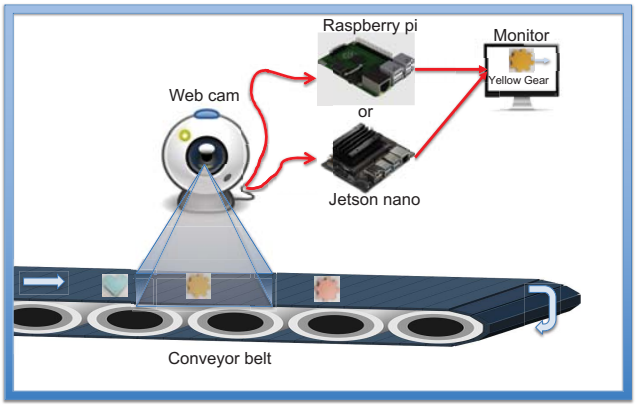


Fig. 1 General scheme of automatic figure identification

Recognition system is based on deep learning theory in a specially designed CNN architecture. The implementation was done using a Raspberry Pi 3 system, and a Jetson-Nano device involving a graphics processing GPU unit. A digital image processing stage was developed to detect the object, its shape, and its center of gravity.

IV. SYSTEM ARCHITECTURE

A. Digital Image Processing Device

The two specialized processing devices used in this research work were a Respberry Pi 3 [1] and a Jetson-Nano [2]. Fig. 2 shows both devices: a Raspberry Pi 3 (See Fig. 2(a)) and

[1]https://www.raspberrypi.org/
[2]https://developer.nvidia.com/embedded/jetson-nano-developer-kit

TABLE II
KEY FEATURES OF RASPBERRY PI 3 AND JETSON-NANO DEVICES

Raspberry PI 3	***Jetson-Nano***
CPU + GPU: Broadcom BCM2837B0,	GPU: 128-core NVIDIA Maxwell™
Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz	architecture-based GPU
RAM: 1GB LPDDR2 SDRAM	CPU: Quad-core ARM® A57
Wi-Fi + Bluetooth: 2.4GHz and	Video: 4K @ 30 fps (H.264/H.265) /
5GHz IEEE 802.11.b/g/n/ac, Bluetooth	4K @ 60 fps (H.264/H.265) encode and decode
4.2, BLE	Camera: MIPI CSI-2 DPHY lanes, 12x (Module)
Ethernet: Gigabit Ethernet sobre USB 2.0 (300 Mbps)	and 1x (Developer Kit)
GPIO de 40 pines	Memory: 4 GB 64-bit LPDDR4; 25.6 gigabytes/second
HDMI	Connectivity: Gigabit Ethernet
4 USB 2.0 ports	4 USB 2.0 ports
Power-over-Ethernet (PoE)	
CSI and DSI ports for camera and touch screen	OS Support: Linux for Tegra®
Audio and video outputs ports	
Micro-SD	

a Jetson-Nano (See Fig. 2(b)). The Raspberry Pi is running on the *Raspbian* operating system (O.S.) and the Jetson-Nano devices is running on the Ubuntu (O.S.) (See Table II for more key features). For both devices, the O.S. was flashed into a 64 GB micro memory. The necessary libraries were installed to use the Python 3 programming language. The main difference between the Raspberry Pi and Jetson-Nano is that the latter has a graphics processing unit (GPU). The GPU has 4GB of memory and is programmed with CUDA-NVIDIA libraries.

B. Digital Image Processing

In the discrete case, an image is defined as a three-dimensional function $f(x, y, b)$, for (x, y) viewed as integer coordinates, with $1 \leq x \leq M$ and $1 \leq y \leq N$, and b is the RGB band, stands $b = \{1, 2, 3\}$. Let $O(x, y, b)$ be the filtered image band. Thus, the corresponding filter band is defined as all pixels values belonging to a minimum and maximum threshold T_l and T_h, respectively (see e.q. 1).

$$O^b(x, y) = \left\{ \begin{array}{cc} f(x, y, b) & if \quad T_l^b \leq f(x, y, b) \leq T_h^b \\ 0 & otherwise \end{array} \right\} \quad (1)$$

where stand $T_l^b = [central - values^b] - 20$ and $T_h^b = [central - values^b] + 20$.

The system was implemented to filter 4 types of colors, pink, gold, blue, and orange. For each color, its central values are defined in the RGB bands. For the filtering of the pink color, the values were taken $[232, 183, 169]$, for the gold $[213, 152, 44]$, for the blue $[151, 197, 204]$, and for the orange $[233, 160, 85]$. For each central color, an interval of 20 values was taken, thus defining the minimum and maximum thresholds for each color band.

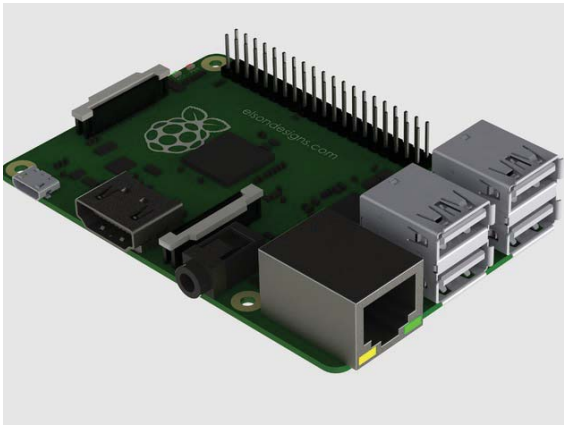
Only one figure is found in the camera's field of view on the conveyor belt. In addition, the processing is done in real-time, so each image has to be processed according to the following steps:

a) The image provided by the camera is resized to a constant value of 64×64 (see Fig. 3(a)).

b) Apply the filtering by color band, for each one of the colors sought (4 in total)(see Fig. 3(b)).

c) Identify the color band where there is the most information (see Fig. 3(c)). Once the figure is located, there are both the minimum and maximum space span on the x-axis; the same is done for the y-axis. This limits the region, obtaining the coordinates of the box containing the figure $(x_{min}, y_{min})(x_{max}, y_{max})$.

d) The zone where only the figure is located is trimmed (see



(a) Raspberry Pi 3

(b) Jetson-Nano

Fig. 2 Two specialized processing devices (mini computers) used in the project

Fig. 3(d)).
e) Calculate the center of mass or gravity of each figure (see Fig. 3(e)). From opencv the moments calculated and obtained were only the $M(0,0)$, $M(0,1)$, and $M(1,0)$. Thus, the gravity center of the shapes was obtained as $c_x = int(M(1,0)/M(0,0))$ and $c_y = int(M(0,1)/M(0,0)$ [3].

The cropped image is now sent to the second stage of processing. The isolated figure passes to the pattern recognition subsystem based on deep CNN.

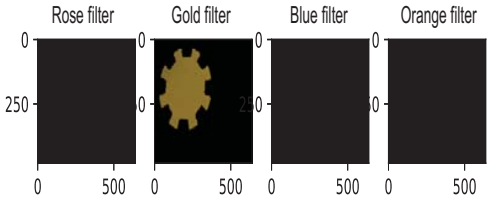
C. Deep-Learning System: Image Identification

Deep convolutional neural network was inspired by the biology of the human visual system, is implemented in the shape recognition step. In the present proposal, detailed information is represented by several stages of convolution, max-pooling, ReLU, and normalization. The overall structure of CNNs is described below:

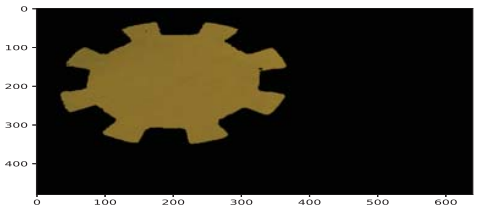
- *Convolutional layer:* A bi-dimensional convolution operation between an image $f(x,y)$ of dimension $M \times N$ and a kernel matrix $h(x,y)$ of dimension $K \times K$, in which N, M, and K are integers. In this sense, the convolution is represented by $c = f(x,y) \otimes h(x,y)$, where $\otimes$ illustrates the convolution operation. Thus, in discrete domain, the convolution is expressed as $c(x,y) = \sum_{k \in K} \sum_{l \in K} f(k,l)h[x-k, y-l]$, for $\forall\, x, y \in [1...M]$ and $[1...N]$. In other words, a reflected matrix $h(x,y)$, which is also called a convolutional filter, is sliding along image $f(x,y)$, a dot product is computed at each value of x and y.
- *Activation function:* The activation function determines the output of a neural network and it is defined by a mathematical expression which can be non-linear functions such as sigmoid, tangent, hyperbolic tangent, and ReLU. In the present application, the ReLU activation function was chosen. Rectified linear unit (ReLU) is defined as $ReLU(\rho(x,y)) = max\{0, \rho(x,y)\}$. The resulting output will be "activated" or not, depending on the input values. Thus, the output values will be only positive or zero depending upon the input matrix ρ(x, y) [23].
- *Pooling layer:* Pooling performs a sub-sampling of the input image, generating a lower resolution version of an input image. Pooling preserves the largest and most important structural elements, removing fine details from the image. In this proposal, the max-pooling function is used to perform the sub-sampling.
- *Flattening layer:* This stage aims to generate a one-dimensional vector, concatenating all the characteristics of the last feature extraction layer. The resulting vector will be used as an input to carry out the classification task.
- *Soft-max layer:* The classification task is carried out using a fully connected multilayer. At the output layer, a soft-max function is used to calculate the probability for each class.

[3]The spatial moments are computed as: $M(i,j) = \sum_{x,y} f(x,y) \cdot x^i \cdot y^j$

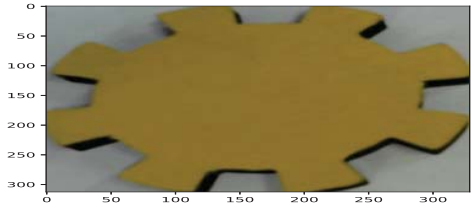
(a) Original image: Gold gear



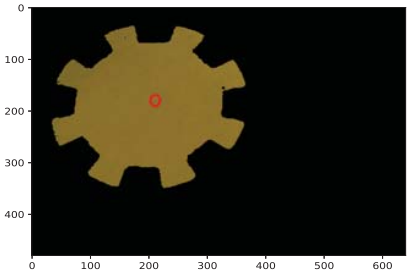
(b) Color filtering



(c) Color band identification: Gold



(d) Cropped shape



(e) Identification of shape with its central coordinates ($x = 200$, $y = 190$)

Fig. 3 Example of digital image processing for gear shape

The training of the whole network is carried out using the stochastic descending gradient algorithm in the backpropagation scheme.

The proposed CNN architecture is presented in Fig. 4 and Table III. The proposed CNN is fed by an image of 64×64

pixels which passes throughout the extraction layers of three features. Afterwards, a flattening task is applied and fully connected layer is used. Finally, a Soft-max layer is used as a classification task. The model is trained to minimize the cross-entropy loss function using back propagation algorithm and to carry out an optimization of the training parameters using stochastic gradient descent [24].

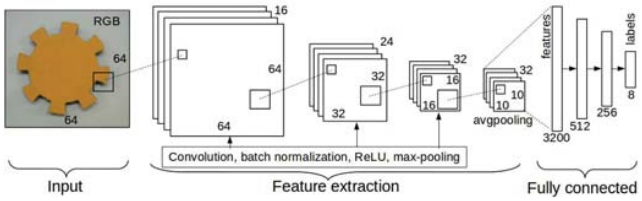


Fig. 4 The proposed CNN architecture consisting of the input, feature extraction, and fully connected layers

TABLE III
PROPOSED CNN STRUCTURE

Size of input image	64×64
First layer	Convolution with a 5×5 kernel Number of filters 16, stride= 1 Batch Normalization Activation function: ReLu Max Pooling, kernel size of 3×3, stride= 2
Second layer	Convolution with a $3x3$ kernel Number of filters 24, stride= 2 Batch Normalization Activation function: ReLu Max Pooling, kernel size of 3×3, stride= 2
Third layer	Convolution with a $3x3$ kernel Number of filters 32, stride= 1 Batch Normalization Activation function: ReLu Avg. Pooling, kernel size of 7×7, stride= 1
Full-connected layer	3200 features are extracted from the previous layer
Soft-max layer	512 input neurons and 8 output neurons, three layers.

The loss entropy function for the proposed CNN is defined by the following equation:

$$\xi_T(\Theta_1, W) = \xi_{FMC}(\Theta_1) + \xi_{CLA}(W) \tag{2}$$

where $\xi_{FMC}(\Theta_1)$ corresponds to the loss function, and $\xi_{CLA}(W)$ corresponds to the loss function of the whole classification layer (*dropout* and *soft-max* layers). Total loss function ξ_T can be rewritten as:

$$\xi_T(\Theta_1, W) = -log[\hat{p}(S_c/\Theta_1)] - log[\hat{p}(S_c/W)] \tag{3}$$

where $\Omega = \{\Theta_1\}$ are the parameter sets for the CNN. W is the parameter sets for the classification layer (*dropout* and *soft-max* layers). S_c= Shape type, where c={blue, pink, yellow, and gold} color. $\hat{p}(S_c/\Theta_i)$ expresses the conditional probability function for a given shape type conditioned to a Θ_1 parameter set, and $\hat{p}(S_c/W)$ stands for the conditional probability function for a given shape type conditioned to a W classification parameter set layer.

The parameter set for the CNN is defined as follows:

$$\begin{aligned} \Theta_1 &= \text{parameter sets for the fine-CNN} \\ W &= \text{parameter sets for the classification layer} \\ &\quad (\textit{dropout} \text{ and } \textit{soft-max} \text{ layers}) \end{aligned} \tag{4}$$

TABLE IV
CNN LEARNING SETUP

Parameter	Value/Range
Learning rate	0.0001
Weight decay	0.0005
Momentum	$0.5 - 0.99$
Probability of dropout	0.8
Size of mini-batches	32
Maximum epochs	150

D. Dataset

The total database is divided into two sets in which 70% of it is used for the training stage which consists of 640 images (see Fig. 5 and Table V). Testing dataset is conformed of 160 images.

TABLE V
SHAPES DISTRIBUTION OVER THE TRAINING AND TESTING SETS

	# images	
Shape-Color	Training	Testing
Blue Gear	80	20
Pink Gear	80	20
Yellow Gear	80	20
Gold Gear	80	20
Blue Heart	80	20
Pink Heart	80	20
Yellow Heart	80	20
Gold Heart	80	20
Total	640	160

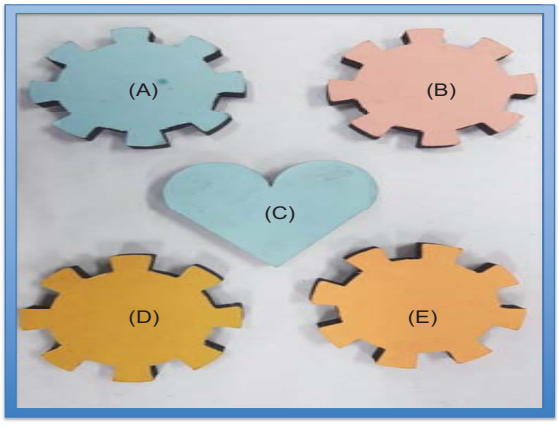


Fig. 5 Example of shapes to be recognized: (A) Blue gear, (B) Pink gear, (C) Blue heart, (D) Yellow gear, and (E) Gold gear

V. SYSTEM IMPLEMENTATION AND RESULTS

The implementation of the proposed figure identification system was accomplished on a Raspberry Pi 3 along with a Jetson-Nano device operating in a work environment of Ubuntu 18.1 Operating System (O.S.). The O.S. was flashed on

a 64 mini USB memory. A Webcam Pc Microsoft 6ch-00001 was used and connected by USB port. For the configuration propose a standard USB keyboard and a standard USB mouse were connected to the Raspberry Pi and Jetson-Nano. The output of the system is displayed on a high resolution HDMI monitor. The whole system was implemented on a python language programing.

The basic Python software which is required for this work is as follows: Open Computer Vision library **opencv**, numpy, matplotlib, TensorFlow, keras, os, and time. The same libraries were installed in the Raspberry Pi and Jetson-Nano devices.

The proposed figure recognition system was gauged in two different dimensions including its correct operational implementation and its usefulness which are described as follows.

A. Learning Evaluation

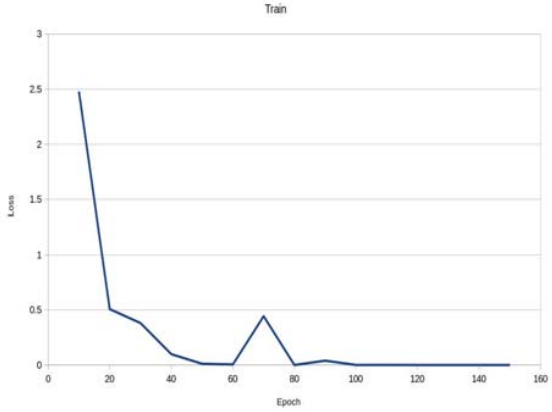
The system is evaluated in two parts; firstly, the training and testing. The parameters which are used for the training of the CNN (see Table IV). The accuracy and loss are parameters to measure the CNN performance. The size of the convolution filter is the most important among all the different parameters in CNN. Experiments were carried out with different filter sizes, retaining the size of the filter that gave the best results. Fig. 6 shows the training loss and accuracy curves and it can be seen that the accuracy reached 100% with an error of 0% after 100 epochs.

B. On-line Object Recognition

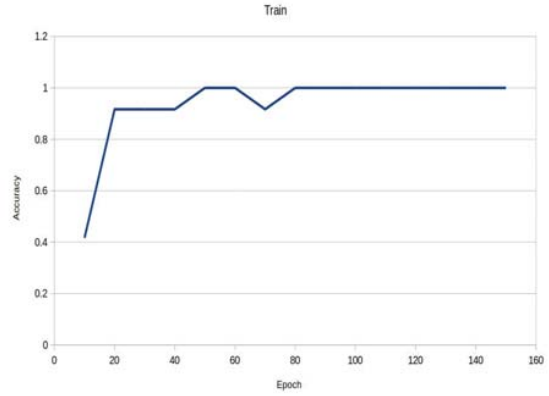
After the accomplishment of the learning optimization process; the authors, in the next step, continued assembling the system as described in section 3 including camera, digital image processing unit, monitor, and conveyor belt. Figs. 7 (a) and (b) present the figure identification system based on Raspberry Pi 3 image processing unit and Jetson-Nano processing unit, respectively.

The Raspberry Pi 3 based system is relatively slower as compared to Jetson-Nano based system processing only one image per second; whereas the letter one was able to process 20 images per second while identifying in the real-time for the maximum speed of the conveyor belt. It can be noted that the figure is correctly identified in both systems in which the center od mass of the figure is outlined with a circle because the figure happened to exist in the field of camera's vision.

The identification of figures along with their color was made by placing it on the conveyor belt. For each figure, its identification was counted (over 20), and its results were expressed as a percentage. Afterwards, a confusion matrix is made consisting of eight classified shapes and colors which is given in Table VI. The identified shapes and colors are: Blue Gear 100%, Pink Gear 100%, Yellow Gear 100%, Gold Gear 100%, Blue Heart 100%, Pink Heart 100%, Yellow Heart 100%, Gold Heart 100%, yielding an average of 100%.



(a) Training loss



(b) Training accuracy

Fig. 6 Training results for the proposed CNN architecture

VI. Conclusion and Further Research

The proposed automatic figure identification system worked as it was expected. The developed unsupervised method, to identify color-shape figures from a camera on a conveyor belt, has an identification efficiency of 100%. The identification system was implemented on two processing unit, Raspberry Pi 3 and Jetson-Nano.

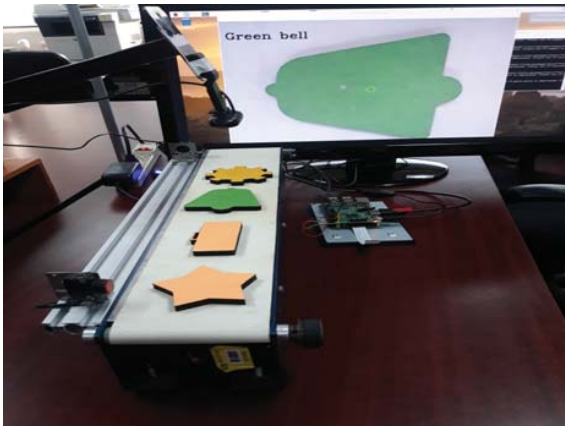
The proposed system works successfully in its two stages. In the first one, the color and center of mass of each figure are detected using digital image processing. In the next stage, the module of the proposed convolutional network was successfully implemented which is employed for the identification of the type of the figure/shape.

The Raspberry Pi 3-based system is relatively slower as compared to Jetson-Nano based system processing only one image per second; whereas the latter one was able to process 20 images per second while identifying in the real-time for the maximum speed of the conveyor belt. It can be noted that the figure is correctly identified in both systems in which the center of mass of the figure is outlined with a circle because the figure happened to exist in the field of camera's vision.

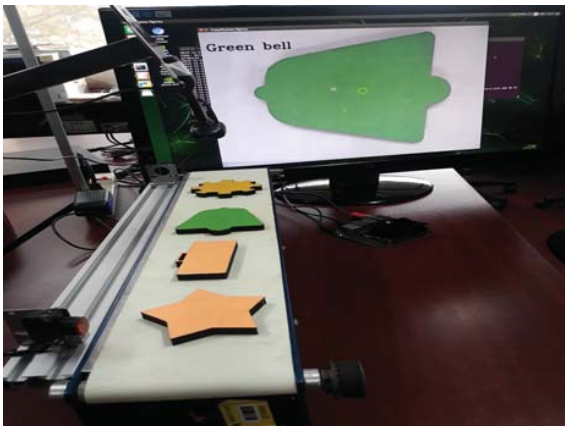
In future work, the authors intend to improve the system to include an extended variety of colors and shapes by

TABLE VI
CONFUSION MATRIX OF THE EIGHT CLASSIFIED SHAPES AND COLORS

	Blue gear	Pink gear	Yellow gear	Gold gear	Blue heart	Pink heart	Yellow heart	Gold heart
Blue gear	100	0	0	0	0	0	0	0
Pink gear	0	100	0	0	0	0	0	0
Yellow gear	0	0	100	0	0	0	0	0
Gold gear	0	0	0	100	0	0	0	0
Blue heart	0	0	0	0	100	0	0	0
Pink heart	0	0	0	0	0	100	0	0
Yellow heart	0	0	0	0	0	0	100	0
Gold heart	0	0	0	0	0	0	0	100



(a) Raspberry Pi-based system identification.



(b) Jetson-Nano-based system identification.

Fig. 7 System identification based on a Raspberry Pi and Jetson-Nano processing devices

introducing enhanced self-training. The authors also intend to integrate it with smartphones to display the

REFERENCES

[1] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3320–3328.

[2] K. Schwab, *The Fourth Industrial Revolution*, C. Business., Ed. Crown Business., 2017, vol. 1, no. 1.

[3] S. F. Kurniawan, I. K. G. D. Putra, and A. A. K. O. Sudana, "Bone fracture detection using opencv," *Journal of Theoretical and Applied Information Technology*, vol. 64, no. 1, pp. 249–254, June 2014.

[4] D. Jacobsen and P. Ott, "Cloud architecture for industrial image processing: Platform for realtime inline quality assurance," in *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, July 2017, pp. 72–74.

[5] S. Lee and C. Yang, "A real time object recognition and counting system for smart industrial camera sensor," *IEEE Sensors Journal*, vol. 17, no. 8, pp. 2516–2523, April 2017.

[6] S. Jagtap, C. Bhatt, J. Thik, and S. Rahimifard, "Monitoring potato waste in food manufacturing using image processing and internet of things approach," *Sustainability (Switzerland)*, vol. 11, no. 11, 2019.

[7] A. Abdo, J. Siam, B. Salah, and M. Krid, "Multiple-sensor fault detection and isolation using video processing in production lines," *International Journal of Computer Integrated Manufacturing*, 2019.

[8] N. Mowell, B. Sheumaker, T. Han, J. Chaung, S. Sanghavi, Y. Khopkar, F. Levitov, B. Bielec, D. Salvador, K. Naguib, and V. Nguyen, "Criticality of photo track monitoring for lithography defect control," vol. 2019-May, 2019.

[9] I. Szabo, J. Sun, C. Selcuk, and T.-H. Gan, "A new automated in line quality control system based on non-destructive evaluation for additive manufacturing of net-shape parts from particulates," *World PM2016 Proceedings,* 2016.

[10] F. Ozkan and B. Ulutas, "Use of an eye-tracker to assess workers in ceramic tile surface defect detection," *2016 International Conference on Control, Decision and Information Technologies (CoDIT), St. Julian's,* 2016, pp. 088-091, doi: 10.1109/CoDIT.2016.7593540.

[11] A. Ghaitaranpour, A. Rastegar, F. Tabatabaei Yazdi, M. Mohebbi, and B. Alizadeh Behbahani, "Application of digital image processing in monitoring some physical properties of tarkhineh during drying," *Journal of Food Processing and Preservation*, vol. 41, no. 2, 2017.

[12] G. Reddy, T. Jahnavi, D. Rushali, and B. Kumar, "Bliss bot for pharmaceutical inspection," *2017 International Conference on Trends in Electronics and Informatics (ICEI), Tirunelveli,* 2017, pp. 354-359, doi: 10.1109/ICOEI.2017.8300947.

[13] A. Thamna, P. Srisungsitthisunti, and S. Dechjarem, "Real-time visual inspection and rejection machine for bullet production," *2018 2nd International Conference on Engineering Innovation (ICEI 2018),* 2018, pp. 13–17.

[14] A. Suárez, M. A. Alvarez-Feijoo, R. Fernández González, and E. Arce, "Teaching optimization of manufacturing problems via code components of a jupyter notebook," *Computer Applications in Engineering Education*, vol. 26, no. 5, pp. 1102–1110, 2018.

[15] K. Fukushima and S. Miyake, "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position," *Pattern Recognition*, vol. 15, no. 6, pp. 455 – 469, 1982. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0031320382900243

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

[17] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, ser. IJCAI'11. AAAI Press, 2011, pp. 1237–1242.

[18] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 248–255.

[19] M. Inoue, S. Inoue, and T. Nishida, "Deep recurrent neural network for mobile human activity recognition with high throughput," *Artif. Life Robot.*, vol. 23, no. 2, pp. 173–185, Jun. 2018.
[20] "An analysis of convolutional long short-term memory recurrent neural networks for gesture recognition," *Neurocomputing*, vol. 268, pp. 76 – 86, 2017, advances in artificial neural networks, machine learning and computational intelligence.
[21] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, 2016.
[22] M. Edel and E. Köppe, "Binarized-blstm-rnn based human activity recognition," in *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Oct 2016, pp. 1–7.
[23] C. Avilés-Cruz, A. Ferreyra-Ramírez, A. Zúñiga-López, and J. Villegas-Cortéz, "Coarse-fine convolutional deep-learning strategy for human activity recognition," *Sensors*, vol. 19, no. 7, 2019.
[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

Fidel López-Saca MS. Fidel López-Saca was born in Puebla-Mexico. He is graduated in Applied Mathematics (1989) from the *Universidad Nacional Autónoma de México*-Mexico, He received the master degree in computer science from the *Universidad Autónoma Metropolitana*-Mexico. His research interests are: computer vision, digital signal and image processing, pattern recognition and neural networks.

Carlos Avilés-Cruz Dr. Carlos Aviles-Cruz was born in Mexico City (March 13, 1966). He is graduated in Electronics Engineering (1989) from the *Universidad Autónoma Metropolitana*-Mexico, He received the master degree in signal and image processing from the National Institute Polytechnic of Grenoble-FRANCE (1993). He received the doctorate in signal and image processing from the National Institute Polytechnic of Grenoble-FRANCE (1997). Currently, he is a full-time professor at the *Universidad Autónoma Metropolitana*-Mexico City. His research interests are: computer vision, digital signal and image processing, High Order Statistics and neural networks. Dr. Aviles-Cruz is author and co-author of about 100 national and international articles and also co-author of two books.

Miguel Magos-Rivera Dr. Miguel Magos-Rivera was born in Mexico. He is graduated in Electronics Engineering from the *Universidad Autónoma Metropolitana*-Mexico, He received the master and doctor degree in automatic control from Claude Bernard University-FRANCE. Currently, he is a full-time professor at the *Universidad Autónoma Metropolitana*-Mexico City. His research interests are: automatic control, instrumentation, and sensors.

José Antonio Lara-Chávez Engr. José Antonio Lara-Chávez was born in Mexico. He is graduated in Electronics Engineering from the *Universidad Autónoma Metropolitana*-Mexico. Currently, he is a full-time assistant professor at the *Universidad Autónoma Metropolitana*-Mexico City. His research interests are: automatic control, instrumentation, and sensors.