electronics

Article

Steel Bar Counting from Images with Machine Learning

Ana Caren Hernández-Ruiz *, Javier Alejandro Martínez-Nieto and Julio David Buldain-Pérez

Department of Electronic Engineering and Communications, University of Zaragoza, 50018 Zaragoza, Spain; almartinez@unizar.es (J.A.M.-N.); buldain@unizar.es (J.D.B.-P.)

* Correspondence: anaacaren@unizar.es; Tel.: +34-602-803-025

Abstract: Counting has become a fundamental task for data processing in areas such as microbiology, medicine, agriculture and astrophysics. The proposed SA-CNN-DC (Scale Adaptive—Convolutional Neural Network—Distance Clustering) methodology in this paper is designed for automated counting of steel bars from images. Its design consists of two Machine Learning techniques: Neural Networks and Clustering. The system has been trained to count round and squared steel bars, obtaining an average detection accuracy of 98.81% and 98.57%, respectively. In the steel industry, counting steel bars is a time consuming task which highly relies on human labour and is prone to errors. Reduction of counting time and resources, safety and productivity of employees and high confidence of the inventory are some of the advantages of the proposed methodology in a steel warehouse.

Keywords: convolutional neural networks; steel bars; counting; clustering; machine learning

check for updates

Citation: Hernández-Ruiz, A.C.; Martínez-Nieto, J.A.; Buldain-Pérez, J.D. Steel Bar Counting from Images with Machine Learning. *Electronics* **2021**, *10*, 402. https://doi.org/10.3390/electronics10040402

Academic Editor: Gwanggil Jeon

Received: 18 December 2020

Accepted: 3 February 2021

Published: 7 February 2021

1. Introduction

Counting is a time-consuming task and a key factor in keeping track of the inventory of any material. When talking about objects with different shapes and sizes, the task becomes more challenging. In the steel industry, the steel bar is one of the most widely used product in the world for building construction and forge. During the manufacturing process, the bars are usually counted by using images, thus allowing for distance, lighting and angle control. However, once they leave the factory, these heavy and large materials must be stacked and stored in warehouses or retails, where a hostile environment prevails in order to track a smooth and reliable inventory.

Traditional steel bar counting is based on human calculation; however, due to the shifting conditions and low manoeuvrability, the manual counting is quite slow and labour-intensive with low accuracy rate. Therefore, an automatic system capable to count these materials regardless of the physical conditions is required in order to improve the effectiveness and reliability of the steel bars counting process.

Basically, it is possible to distinguish two main approaches for counting tasks. Image processing techniques implement algorithms based on mathematical functions to transform an image. Filters, threshold segmentation, edge detection and matching are commonly used techniques [1–4]. Although these techniques are highly accurate, they are bounded to specific conditions such as constant lightning and background, or special camera requirements. Moreover, they are limited to round steel bars with fixed shape and size, assuming their shape is quasicircular, thus lacking robustness [5]. Note that some of these methods are bounded to the production line in steel fabrics where physical separation of the materials is viable [6,7].

Similarly, other steel bar counting algorithms based on image processing are mainly based on area and template algorithms. Both of these methods are feasible, but there are some disadvantages. The results of the first method cannot directly locate the steel bar in the counting result, so it makes great inconvenience in error analysis of the algorithm [8,9].

Template matching method heavily depends on the shape of the template and target object, so the adaptive ability is limited [10,11].

In the case of machine learning (ML) techniques, convolutional neural networks (CNN) have demonstrated to be highly accurate and fast enough for image processing. Well-known CNN architectures such as Feature Pyramid Network, Visual Geometric Group and Inception-ResNet are used as classifiers and regressors in order to obtain the total number of elements [12–16]. As for industrial applications, neural networks have proven to achieve high accuracy and provide a fast-track solution for problems in hostile environments [17,18].

A methodology that resembles how humans count is presented in [19]. A CNN implemented as a Binary Classifier is used to detect each bar in the image and mark it as a candidate centre. Once every candidate centre is detected, a clustering technique is applied in order to extract the actual geometric centre of the bars. The final count is the number of centres detected. This methodology, referred to as CNN-DC, although it achieves 99.26% of accuracy in 3.58 s, it is restricted to a constant background among the images and a fixed patch size, thus implying that the distance from the camera is constant.

On the other hand, a deep learning fusion model for detecting objects is proposed in [20]. Localisation and segmentation of steel bars is done by using a combined model. The model achieves a 98.17% in F1 score (harmonic mean value of precision and recall) in object detection in 0.03 s; however, the proposed Inception-RFB-FPN architecture is quite complex with many layers and requires high computational resources for its deployment, thus making it unaffordable for an embedded and portable system.

Most of the implementations focus on counting a single object kind with specific characteristics and constant background. Although some machine learning based works have a high performance, a portable system might not be viable due to the required amount of computational resources. By considering these issues, a CNN-based counting methodology approach, namely Scale Adaptive Convolutional Neural Network Distance Clustering (SA-CNN-DC), is presented in this paper in order to count steel bars regardless of their size and shape by adopting a compact design with a minimum number of parameters. The proposed counting methodology is described in Section 2. Section 3 summarises the image processing techniques applied to generate the dataset for each network in the system. A description of the implemented neural network architectures and distance clustering algorithm is shown in Section 4, where a training performance of each one is also presented. The most relevant results and the SA-CNN-DC overall performance validation are presented in Section 5. A comparison with similar implementations is also made in this section. The desktop app implementation is described in Section 6. Finally, conclusions are drawn in Section 7.

2. Proposed SA-CNN-DC Counting Approach

A variety of steel bar shapes can be found in the steel industry, and each bar kind usually is manufactured with different dimensions. The diversity of sizes and shapes is a challenge that the proposed methodology addresses by adding an extra input called *density*. This parameter is a fundamental factor in order to obtain an effective bar classification as well as its accurate localisation.

The proposed methodology attempts to improve the CNN-DC framework presented in [19] which requires specific conditions of the images considered, as well as a fixed size and shape bar. In addition, the proposed method is designed to be portable, replicable and robust to natural variability of the conditions in the warehouse, such as noise, light and scale.

Basically, the proposed SA-CNN-DC methodology automatically classifies the steel bar type, locates each bar centre and provides an output with the total amount of bars. Both the material image and the density are the inputs required for the system. It is worth mentioning that based on an analysis of the storing material physical conditions within the warehouse, three types of density were defined (low, medium and high) in order to ensure robustness to the variability of the steel bar dimensions. Figure 1 shows an example of each of the possible steel bar densities.

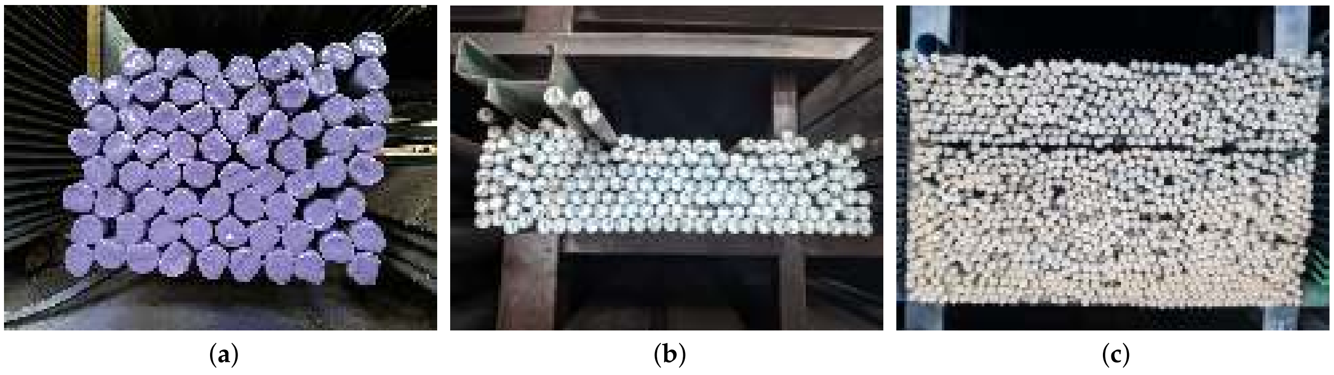


Figure 1. Possible steel bar densities: (**a**) Low, (**b**) medium and (**c**) high.

The SA-CNN-DC main core consists of three neural networks and a clustering technique. Each network solves an specific task: bar classification, image resizing and centre localisation. The modular design not only allows portability but also it is replicable, thus providing the capability to add new materials by using the same methodology. Bearing this in mind, round and squared bars were considered for counting purposes; however, angled and rectangular bars were also used for training.

A graphical representation of the proposed methodology is shown in Figure 2. Their main stages are described as follows:

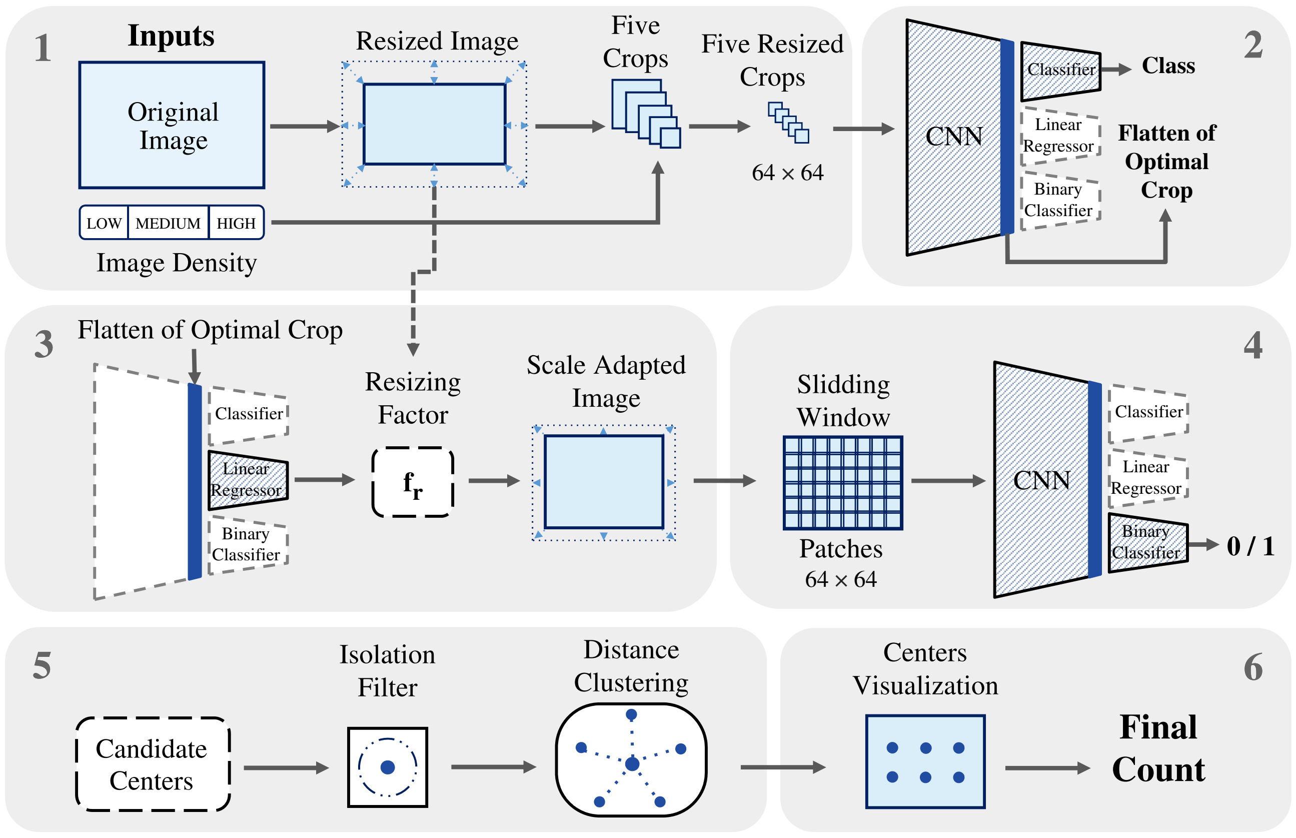


Figure 2. SA-CNN-DC methodology block diagram.

1. *Preprocessing:* The input image is resized to a size limit between 3500 and 1700 pixels. According to its density type, five crops of different sizes are extracted from the centre

of the image and resized to the CNN input size. For high density, the crops extracted are smaller and vice-versa for the low density.

2. *Classification:* These five crops are feed-forward into the classifier to obtain the softmax output. The class is chosen according to the five predictions obtained through voting, thus the class with the highest number of predictions is selected. From the correct predictions the one with the highest probability is selected. A higher probability means a more reliable prediction and it implies the network is able to detect features with high certainty. The flatten vector of the response with the highest probability, which is generated after the convolutional layers and contains all the information of the image condensed, is stored as the *optimal flatten* for the next step.
3. *Linear Regressor:* This network outputs a *resizing factor*, f_R, which is used to resize the image. This factor determines how much an image must be resized so that a single bar completely fits in a patch of a fixed size, as seen in Figure 3. Then, the scale of the resulting image is adapted for the next stage. The convolutional layers of the classifier are used to train a simple multilayer perceptron with a linear output that acts as regressor for the resizing factor. This process, called transfer learning, helps to drastically reduce the training data and time of the regressor network.
4. *Binary Classification:* A fixed-size sliding window moves through the scale adapted image with a small stride. A binary-output network classifies whenever the resulting patch contains a bar or not. If a positive detection is made, the centre coordinate of the patch is stored as a candidate centre.
5. *Distance Clustering:* Candidate centres are filtered according to their horizontal and vertical proximity with other candidate centres. If a candidate centre is completely isolated, it is deleted. The distance clustering algorithm measures the Euclidean distance between candidate centres and groups them within a threshold distance. For each cluster, a centre coordinate is stored, ideally this is the geometric centre of the material [19].
6. *Output:* The final count is the total number of centre coordinates. For visualisation purposes, the centres are marked with a colour dot in the image.

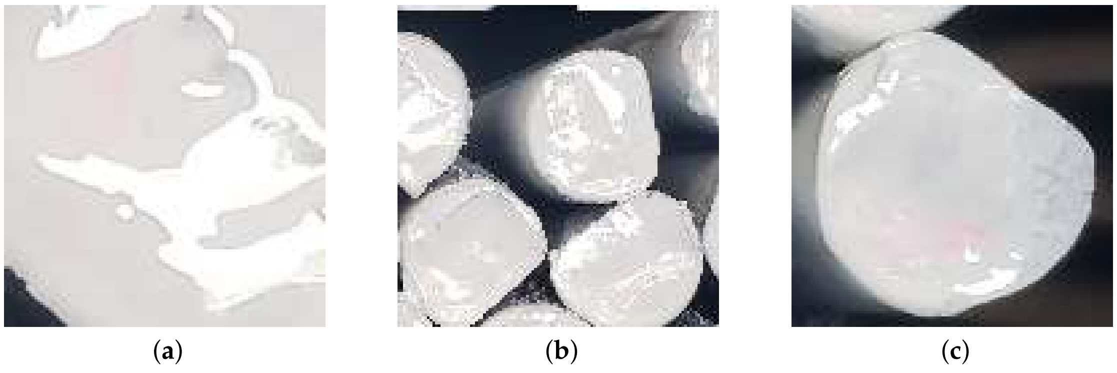


Figure 3. Resized image: (**a**,**b**) show a wrong resizing, while (**c**) a correctly resized image.

3. Dataset Acquisition

Since each neural network has an specific task within the proposed system, three different datasets were built for each network: Classifier-Dataset, Regressor-Dataset and Binary-Dataset. It is worth mentioning that a single input size was fixed for the networks, thus obtaining a modular design. This input size determines how much GPU memory will be required for the kernel weights in the network layers, as well as the batch size and the training time. When an image with large dimensions and high resolution is resized to a considerably smaller size, it ends up distorted and the steel bars lose their shape. In consequence, the CNN is unable to learn characteristics of the materials, instead it learns to extract information from the noise in the image. By considering a trade-off between the required GPU memory and image distortion, the input size was set to 64×64 pixels. Moreover, since the RGB channels do not contain relevant information about the shape and they also require a larger memory consumption, the images were converted to grayscale.

3.1. Classifier-Dataset

The classifier was built to distinguish between steel bar types. For this network it is important that the samples contain the relevant shape characteristics of the bars. Note that an angled, rectangular, round and squared steel bars with different size and colour, as shown in Figure 4, were considered as possible classes. It is important to emphasise that only the round and squared bars were used for counting, while the angled and rectangular categories were introduced as rejection classes, but they will be considered for future research.

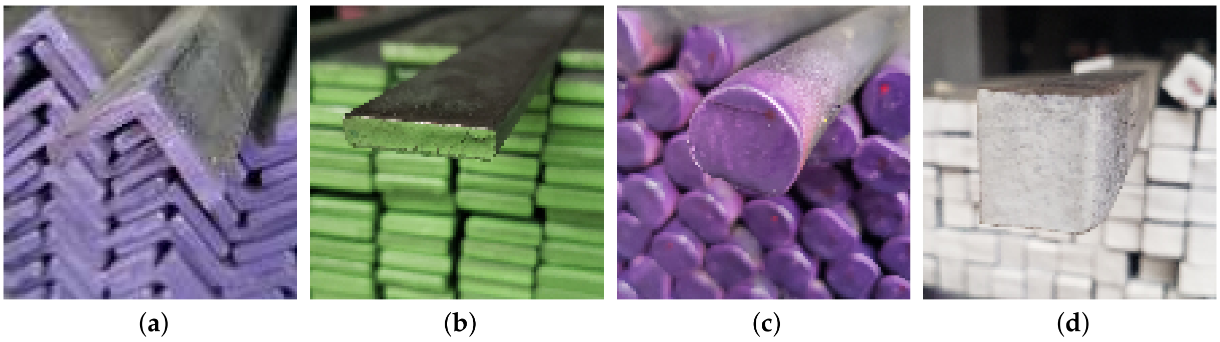


Figure 4. Steel bar classes for the classifier: (**a**) angled, (**b**) rectangular, (**c**) round and (**d**) squared.

First, steel bar photographs with different dimensions were manually collected in an ordinary warehouse. To ensure an appropriate representation of the possible conditions in the place, it was required to collect several images with variations in the light, frame, angle and position. In this way, around 400 photographs were taken from each single material pile, thus obtaining a small set of 12,793 images.

From these collected images and by considering the aforementioned *density* parameter, variable size crops (with a random increase or decrease) were extracted from different coordinates within the images. Once extracted, they were resized to 64×64 pixels and converted to grayscale. This technique helps to increase the amount of data and prevents distortion due to resizing. Figure 5 shows an example of the most appropriate resized crop (Figure 5d) obtained from the original image (Figure 5a). The crop size was selected according to both the density and the image dimensions. Table 1 shows the dataset created for the CNN. The categories are quite balanced and the amount of data is enough to avoid overfitting when using a fitted CNN size.

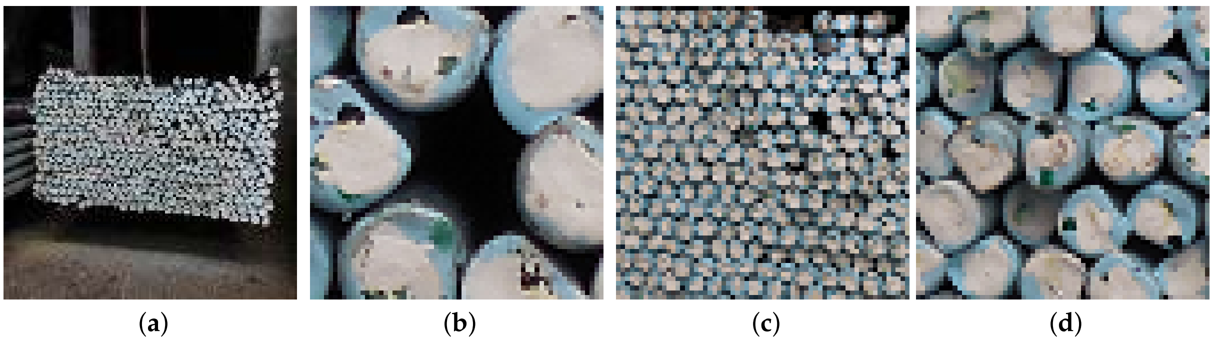


Figure 5. Squared steel bars crops with different dimensions: (**a**) Original image with large dimensions and high resolution image, (**b**) resized small crop, (**c**) resized big crop and (**d**) resized correct crop.

Table 1. Classifier-Dataset sample distribution.

Steel Bar	Number of Samples	Percent (%)
Angled	15,204	25.14
Rectangular	14,995	24.80
Round	15,329	25.35
Squared	14,945	24.71
Total	60,473	100

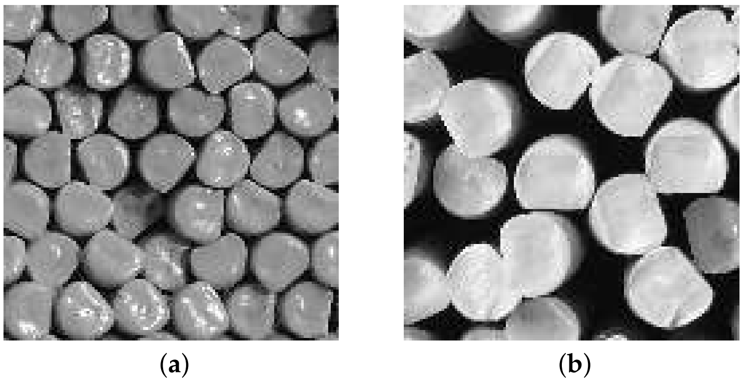
3.2. *Regressor-Dataset*

As mentioned before, only round and squared bars were considered for counting. For the Linear Regressor, new photographs with variable dimensions were taken horizontally in front of the material piles where the steel bars were uniformly painted. The resulting 263 images were manually labelled with the bar size in *pixels* and their corresponding density. The bar size is determined by the height and diameter or width of the bar's section and it is a single number. In this way, the resizing factor could be computed with the equation shown in Equation (1). Moreover, rotations and shifts were used to considerably increase the amount of data. The resulting sample distribution is summarised in Table 2 and some samples are presented in Figure 6.

$$f_R = \frac{bar_{size}}{64} \tag{1}$$

Table 2. Regressor-Dataset sample distribution for round and squared steel bars.

Steel Bar	Number of Samples
Round	5318
Squared	2114



(**a**) (**b**)

Figure 6. Samples of the Regressor-Dataset for round steel bars with (**a**) $f_R = 1.18$ and (**b**) $f_R = 3.37$.

3.3. *Binary-Dataset*

The Binary Classifier classifies each bar in the image. Input patches generated by a sliding window were classified into two classes: *zeros* and *ones*. The first one refers to images containing background, incomplete elements or joints between them, as shown in Figure 7a, and the group labelled as *ones* contains images with centred and complete elements, as shown in Figure 7b.

An image of each size of the round and squared steel bars were considered for the dataset building (12 images in total). Bar centres were manually marked by using an image editor with a 10% brush size of the bar size. Note that this size determines how many pixels will be considered as centre. Next, a sliding window of the steel bar sizes pass through the image with a stride of 5% in order to extract the required patches. If the centre of the patch matches a centre of an element, which is recognisable by its marker colour, the patch is saved as *one* in grayscale. On the contrary, if the patch centre is not a steel bar centre, the patch is saved as *zero*.

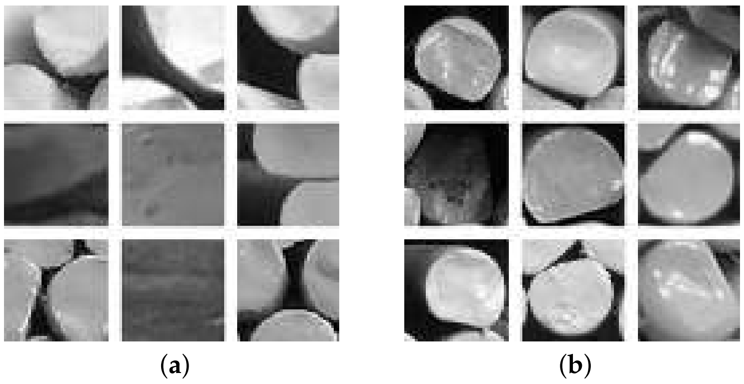


Figure 7. Samples from the Binary Classifier Dataset for round steel bars: (**a**) Zeros and (**b**) Ones.

The sample distribution for the Binary-Dataset is presented in Table 3. It is worth mentioning that the unbalanced data shown for the round bars is not an issue for this network because the data can be chosen randomly in order to balance both classes. More importantly, the addition of new materials is easily done because the two required datasets are automatically processed with the implemented algorithms. Labelling the images is also a simple task just by changing the filename and using a simple raster graphics editor, such as *Microsoft Paint*. This ensures the replicability of the methodology and provides a fast-track addition for different steel bar types.

Table 3. Binary-Dataset sample distribution for round and squared steel bars.

Steel Bar	Ones Samples	Zeros Samples
Round	25,397	45,446
Squared	26,399	24,398

4. Neural Networks Performance

The proposed networks were efficiently designed with the least number of neurons and layers in order to create a modular and portable architecture. The more convolution layers are added, the more abstract information is extracted. However, if the number of convolutional layers exceeds the one required, features with new information are not created because there is no further information to learn. Therefore, it is not recommended to add a large number of convolutional layers. In 1987, Lippmann demonstrated that a multilayer perceptron with two hidden layers is enough to form arbitrary decision regions [21]. These simple guidelines were taken into consideration for the architectures design.

The classifier provides the convolutional layers for the other two networks: the Linear Regressor and the Binary Classifier, which were trained for each single steel bar type with two new datasets. These two networks are smaller and require less data and training time. This characteristic is an advantage if new steel bar types need to be added. Instead of training the whole model, only training of two new multilayer perceptrons is required. This methodology of reusing the feature-extraction part of a trained model with a particular goal to be used in other model with different goal is commonly known as *transfer learning* [22,23]. Figure 8 shows how this process is carried out: the classifier convolutional layers remains the same and only the small multilayer perceptrons corresponding to both, the Regressor and the Binary Classifier, are trained.

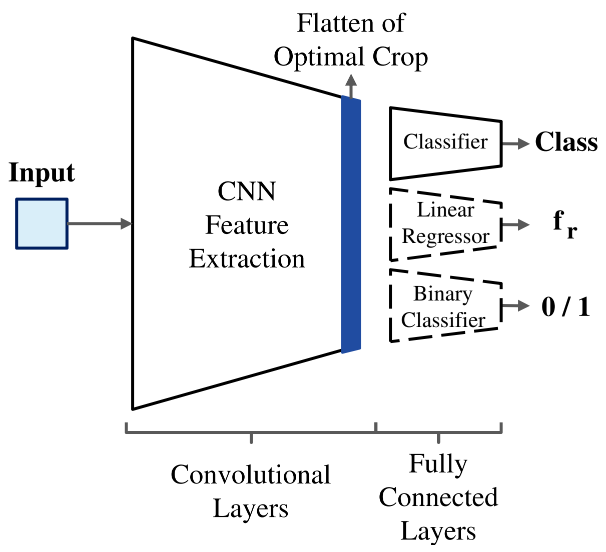


Figure 8. Block diagram of the implemented Neural Networks architectures. First, the classifier is trained to classify angled, rectangular, round and squared steel bars. Then, transfer learning is applied to train the Linear Regressor and the Binary Classifier for round and squared bars. Note that the CNN feature extraction part (convolutional layers) of the classifier, is frozen and reused to train the previously mentioned networks.

For each network, the data was divided into 70% for training, 20% for validation and 10% for test in order to cross-validate results. Several simulations were carried out to determine the best architecture and the hyperparameters for each network by using a NVIDIA GeForce GTX 1050 GPU with Keras [24,25].

4.1. Classifier

The classifier was designed and trained to classify the four different steel bar types. This network also works as a feature extractor or encoder, which means it reduces the information of a larger input into a compact (*flatten*) vector.

It was trained for 12 epochs with a batch size of 108 samples and its architecture consists of three convolutional layers and two fully connected layers, as shown in Figure 9. The validation and test accuracies are 99.28% and 99.35%, respectively, and the training time was barely 84 s.

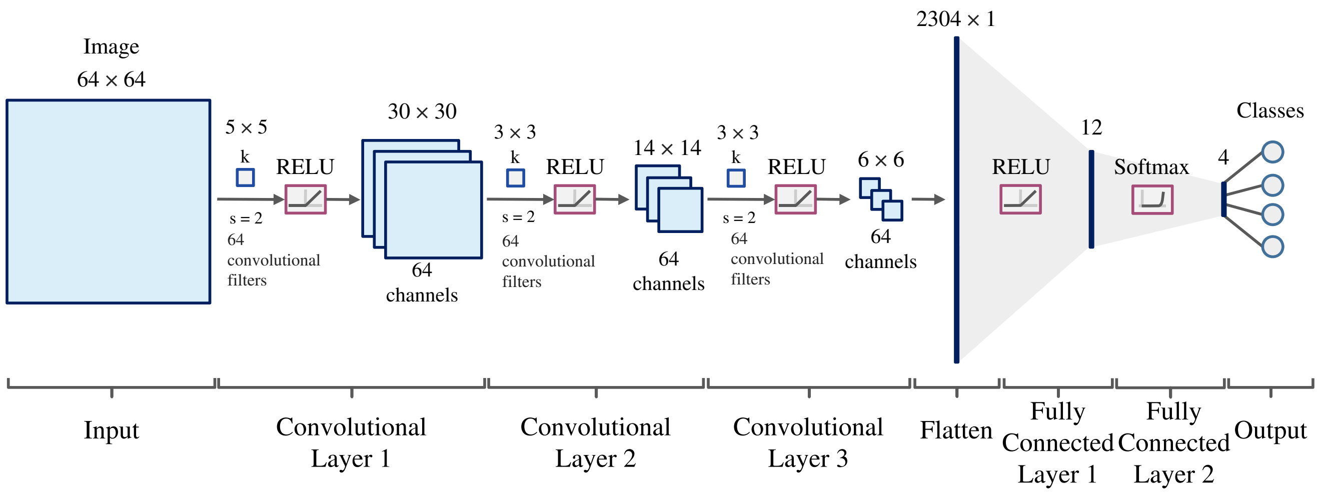


Figure 9. Classifier architecture. Three convolutional layers and two fully connected layers, with 12 and 4 neurons, are used to classify the four steel bar types.

The confusion matrix is used as performance metric for the classification network. The class predictions made by the network with the test data are compared with the actual class in Table 4. Note that round and squared bars are the classes most likely to be confused.

Table 4. Confusion matrix for the classifier.

		Predicted Class			
		Angled	**Squared**	**Round**	**Rectangular**
Actual Class	Angled	1554	1	3	2
	Squared	1	1554	8	3
	Round	0	9	1544	0
	Rectangular	9	4	0	1477

4.2. Linear Regressor

The selected architecture for the Linear Regressor network consists of one hidden layer with four neurons and an output layer with a single neuron, as shown in Figure 10. The input corresponds to the *flatten vector* generated by the convolutional encoder of the classifier and the linear output is a numerical value which represents the resizing factor. The loss is calculated by using the Mean Squared Error (MAE), while the Mean Absolute Error (MAE) is the metric considered for this network.

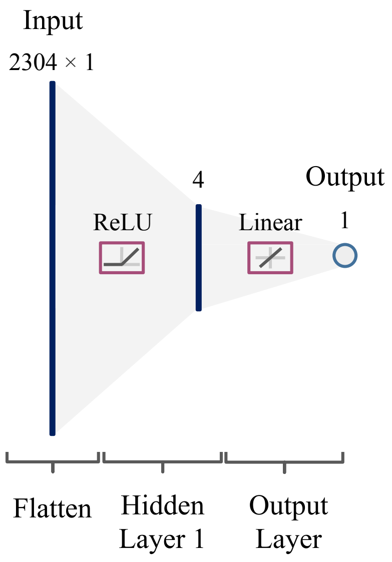
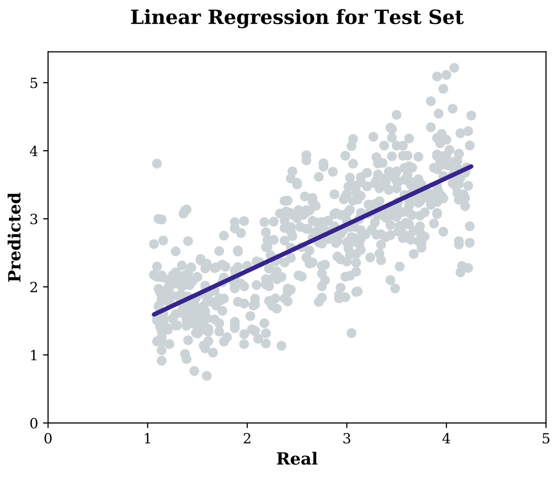


Figure 10. Linear Regressor modular architecture.

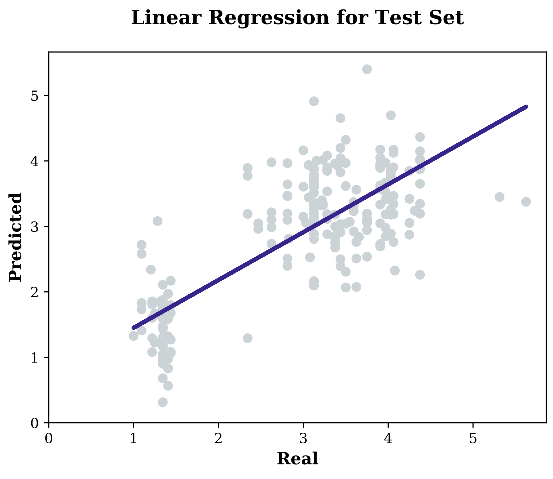
A batch size of 256 round and squared bar samples was considered, resulting in a training time of 80 and 100 epochs, respectively. Once trained, predictions for the validation and test sets were computed by using the Scikit-Learn's linear regression algorithm [26], thus obtaining both, the linear regression (W) and determination (R^2) coefficients. The best possible score for the R^2 coefficient is 1, which means that the predictions are equal to the real values and is calculated by:

$$R^2 = 1 - \frac{\sum(y_{true} - y_{pred})^2}{\sum(y_{true} - \overline{y_{true}})^2} \tag{2}$$

where y_{true} corresponds to the real values and y_{pred} are the values predicted by the network. Figure 11 shows the real and predicted values of the test set for both material types, while Table 5 summarises their resulting metrics and coefficients.



(**a**)



(**b**)

Figure 11. Linear Regressor test set plots for (**a**) round and (**b**) squared bars. The real and predicted values are contrasted and a line which has the best fit for the data is drawn.

Table 5. Resulting Linear Regressor metrics for round and squared bars.

	Round				**Squared**			
	MSE	**MAE**	**W**	**R^2**	**MSE**	**MAE**	**W**	**R^2**
Validation	0.4115	0.5042	0.7977	0.5906	0.5540	0.5796	0.8313	0.8635
Test	0.4030	0.5028	0.6829	0.5906	0.5246	0.5749	0.7303	0.5861

4.3. Binary Classifier

Finally, the Binary Classifier consists of a dense hidden layer with six neurons and an output layer with a single neuron and sigmoid activation function, as shown in Figure 12. This kind of output separates the two classes by a threshold, i.e., an output value smaller than 0.5 corresponds to one class while an equal or greater value is another. If required, the threshold can work as a variable hyperparameter, thus adding flexibility to the material detection.

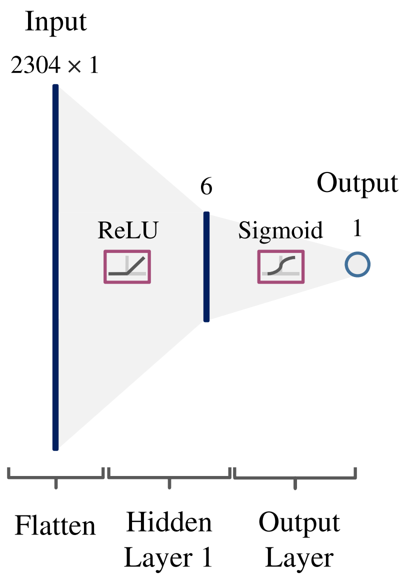


Figure 12. Binary Classifier modular architecture.

The model was trained for 30 s in 20 epochs with a batch size of 128 round-bar samples. The obtained maximum validation and test accuracy are 98.92% and 98.93%, respectively. On other hand, a batch size of 100 squared-bar samples were used for training during 48 epochs (55 s). For this kind of material, the validation and test accuracy are 97.75% and 97.82%.

The Receiver Operating Characteristics curve (ROC) shown in Figure 13 represents the metric considered to validate the round-bar network performance. The area under the curve (AUC) defines the separability degree of the classes [27]. Note that the trained model shows a remarkable performance, thus achieving an AUC of 0.9977.

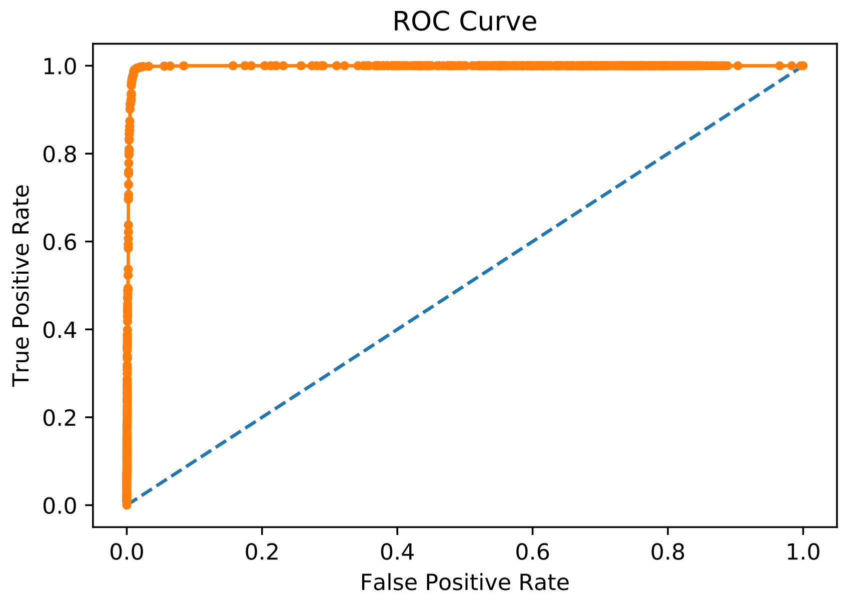


Figure 13. Receiver Operating Characteristics (ROC) curve for the round-bar Binary Classifier when different threshold values are used.

The confusion matrix for the round-bar Binary Classifier network is shown in Table 6. Each class in the Table is defined as follows:

- TP = True positive "1" (correctly identified)
- TN = True negative "0" (correctly rejected)
- FP = False positive "1" (incorrectly identified)
- FN = False negative "0" (incorrectly rejected)

Table 6. Confusion matrix for the Binary Classifier network.

		Predicted Class	
		0	**1**
Actual Class	0	4553 TN	54 FP
	1	23 FN	2597 TP

Portability and processing time highly depends on the number of network parameters. The deep learning networks tendency is to have millions of parameters; however, the proposed methodology just has a total of 149,356. Table 7 summarises the number of parameters for each network conforming the system. It is worth mentioning that for each material addition the number of parameters increases as it is a modular design. However, if counting four different materials was required, the number of parameters would be barely 195,480 because the classifier design actually considers the four material classes.

Table 7. Network parameters in the SA-CNN-DC methodology.

Network	Layers	Parameters
Classifier	5	103,232
Linear Regressor	2	18,450
Binary Classifier	2	27,674
	Total	149,356

4.4. Distance Clustering

The implemented distance clustering algorithm is used to group the candidate center coordinates. The algorithm, described in [19], is summarized in the next steps:

1. The Euclidean distance between coordinates of every candidate centre is calculated.
2. The group for each candidate centre point is created by using a distance threshold as a reference.
3. Groups that contain close elements are merged.
4. The mean point of each group of candidate centres is calculated by averaging the maximum and minimum coordinate values of the group.

Once the clusters are found, the sum of every centre point found is the total count of materials. It is worth mentioning that this technique is the same regardless of the material type.

5. Proposed Methodology Validation

Final tests of the whole SA-CNN-DC were made in order to verify its proper performance. In the first test, round and squared bar counting efficiency was measured and analysed. Then, the proposed methodology was compared with the existing frameworks and, finally, a particular comparison with the top-performance methodology was made.

5.1. SA-CNN-DC Performance

A new set of 21 different dimensions images of round bars was collected from two steel warehouses for testing purposes. In addition, the only test image available in [19] also was included. On the other hand, 10 images of squared bars were also collected for test. Note that these 32 images are unknown for the network during the training phase and the bar numbers range from 26 to 1108.

Precision, recall, F1-score and relative accuracy were the metrics evaluated to verify the performance during the test. Precision corresponds to the correct instances ratio between the retrieved instances, and recall is the true positive rate or sensitivity. As shown in Equations (3) and (4), this two metrics are based on the TP, FP and FN values. On other hand, the F1-score, defined in Equation (5), is calculated based on precision and recall values, while the relative accuracy simply considers the final count (FC), which is the output of the methodology, and the ground truth (GT), as shown in Equation (6).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4}$$

$$\text{F1} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{5}$$

$$\text{Accuracy} = 1 - \left| \frac{\text{FC} - \text{GT}}{\text{GT}} \right| \tag{6}$$

The results of the methodology by considering round and squared test images are shown in Table 8. After testing, a general analysis of the results was made and common errors were detected. For neural networks, noisy background with cardboards or tags may lead to false positives. Similarly, when elements suffer variations in painting and lightning, it becomes more challenging to detect them. Stacking disorder (cluttering) is the main cause for false negatives because the elements appear partially hidden. Finally, if the image has a low resolution, the results become uncertain due to the blurry patches. For the clustering technique, scattered candidate centres for the same bar lead to a cluster division. In contrast, merged clusters with rectangled shape are formed by centre groups of two bars which are too close or joined by a point.

Table 8. Proposed SA-CNN-DC results for round and squared bars.

Bar Type	Samples	Accuracy (%)	Precision	Recall	F1	Inference Time (s)
Round	22	98.81	0.9926	0.9888	0.9906	25
Squared	10	98.57	0.9956	0.9786	0.9870	8

Results also show that the bar detection is independent of the material shape and dimension. Even when the round bars have deformations in their tips, the proposed SA-CNN-DC is able to detect them. Moreover, the detection is correct although the image shows variations in the bar orientation. It is worth mentioning that some of the partially hidden or cluttered elements are correctly counted, and the clustering technique successfully works considering that, sometimes, the material size is not constant throughout the image.

Some test images for round and squared bars are presented in Tables 9 and 10, respectively. The first column shows the output obtained by applying the clustering technique, while the other two show the system output and the main remarks. The resulting mistakes are marked in bounding boxes and some of them are described in the last column. Note that the density type, final count, ground truth and inference time are also shown in each image. It is worth mentioning that the inference time value includes not only the network inference time, but every step of the methodology, from preprocessing to the output.

5.2. General Comparison

By considering round steel bars, the proposed methodology was compared with other frameworks, and their main performance features are summarised in Table 11. As mentioned before, computational resources were limited during the development and testing of the proposed methodology, thus resulting in higher processing time. However, the obtained results are pretty competitive by achieving an accuracy of 98.72% with precision of 0.9926. In addition, the SA-CNN-DC offers robustness to variations in image dimensions, bar size and conditions such as lightning, background noise, etcetera.

5.3. Specific Comparison

Finally, the proposed methodology was compared with the framework described in [19], which shows the best metric results. The same images and computational resources (CPU and GPU) were used for the comparison. It is worth mentioning that the trained model was considered and the test images were fitted to the patch size used in [19]. From the results presented in Table 12, the accuracy of the proposed methodology is 7% higher than Fan's work, and the average processing time is almost four times less. It is worth mentioning that the proposed methodology processing time considers the performance of three networks, while the CNN-DC [19] just a single network but with 14 times more parameters.

Table 9. Test images samples for round bars.

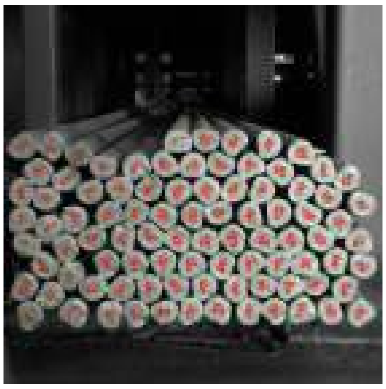
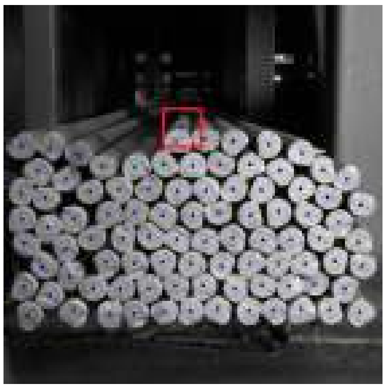
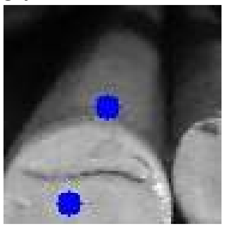
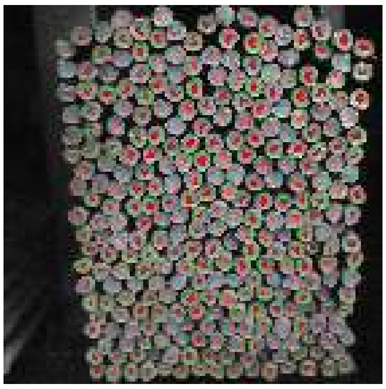
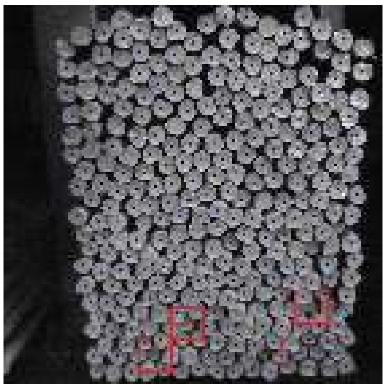
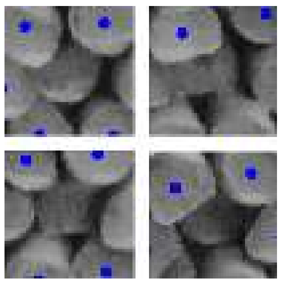
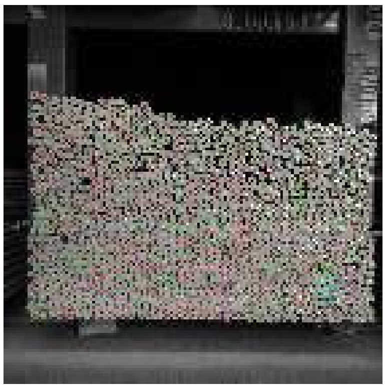
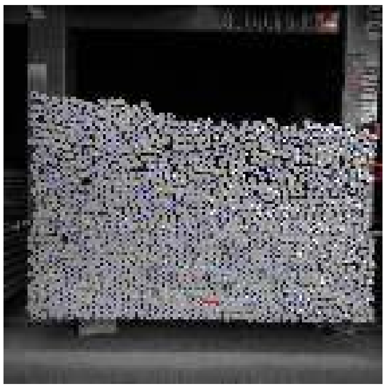
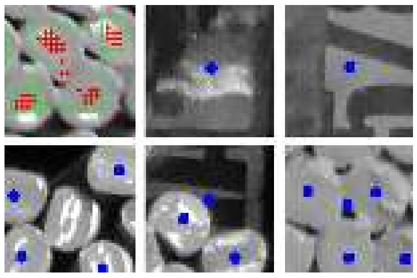
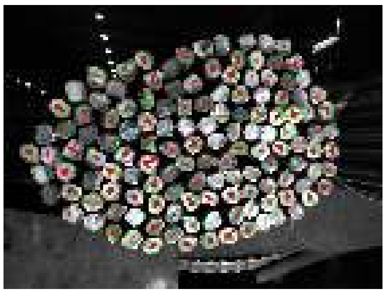
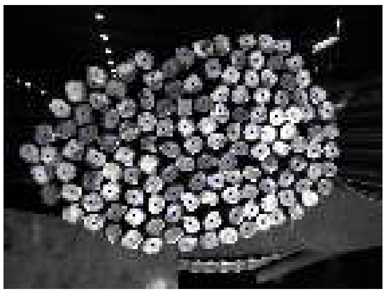
Clustering Output	Final Output	Errors
	Low density sample (*FC: 97, GT: 96, Time: 6 s*)	
		A bar is wrongly detected. 
	Medium density sample (*FC: 298, GT: 302, Time: 20 s*)	
		Four bars are not detected due to cluttering. 
	High density sample (*FC: 1091, GT: 1088, Time: 70 s*)	
		Paint caused two bars to fuse together generating a detection and clustering problem. Background noise generated other false positives. 
	Fan's sample [19] (*FC: 135, GT: 135, Time: 8 s*)	
		Every bar is detected correctly .

Table 10. Test images samples for squared bars.

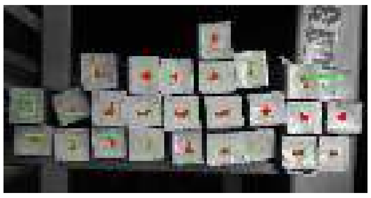
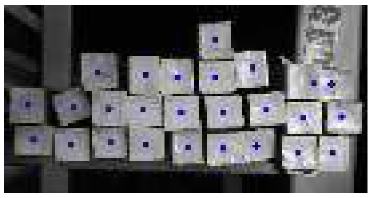
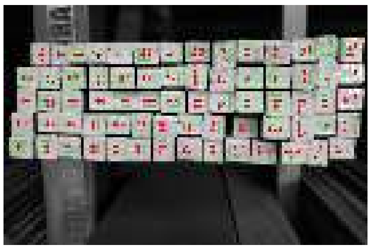
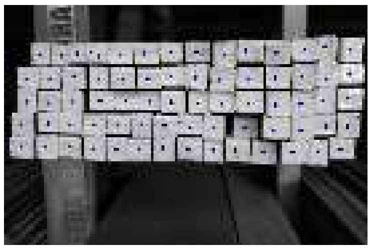
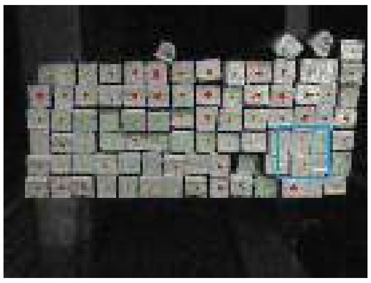
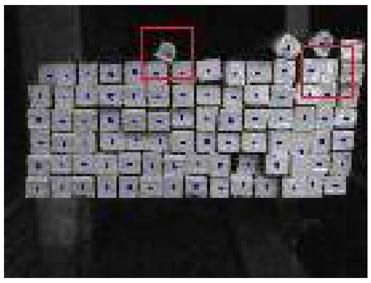
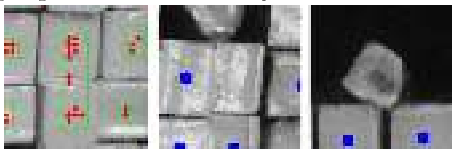
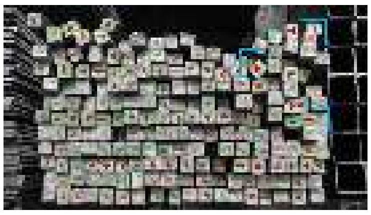
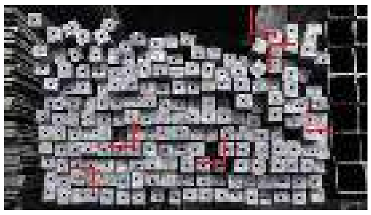
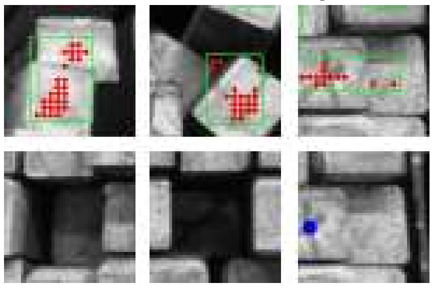
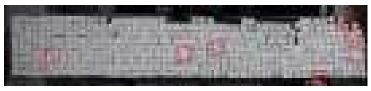
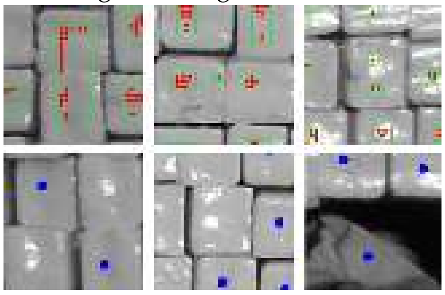
Clustering Output	Centers Output	Observations
Low density sample (*FC: 26, GT: 26, Time: 3 s*)		
		All bars are correctly detected.
Low density sample (*FC: 70, GT: 70, Time: 6 s*)		
		All bars are correctly detected.
Medium density sample (*FC: 85, GT: 88, Time: 7 s*)		
		Three bars are not recognized because their shape, paint and cluttering. 
Medium density sample (*FC: 189, GT: 196, Time: 9 s*)		
		Occlusion results in 8 FN and background in a FP. 
High density sample (*FC: 449, GT: 452, Time: 23 s*)		
		Paint, cluttering and background noise cause errors. 

Table 11. Comparison with other implementations by considering different round bars datasets.

Method	Precision	Recall	F1	Accuracy (%)	Inference Time (s)
Zhang et.al. [1]	0.9360	0.8864	0.9103	94.69	0.3023
Ying et.al. [28]	0.8417	0.9617	0.8975	85.68	0.2404
Liu et.al. [11]	0.6833	0.8123	0.7420	80.99	0.0313
Fan et.al. [19]	0.9976	0.9951	0.9963	99.26	3.5862
Zhu et.al. [20]	0.9753	0.9881	0.9817	98.72	0.0306
Proposed *	0.9926	0.9888	0.9906	98.81	25

* NVIDIA GeForce GTX 1050 GPU.

Table 12. Comparison between the proposed methodology and the CNN-DC presented in [19] by considering both, the same computational resources and same images.

Method	Parameters	Accuracy (%)	Inference Time (s)
Fan [19] *	2,899,138	91.15	96
Proposed *	149,356	98.81	25

* NVIDIA GeForce GTX 1050 GPU.

6. Desktop App

A user friendly desktop app was developed in order to implement the proposed methodology. Minor improvements were considered within the app to create a practical user interface. The possibility to define the element size as well as the area of interest within the raw image was added. The patch size used for the binary network can be defined just by selecting the element size with a bounding box in the image (Figure 14a). While reducing the area of interest (Figure 14b), the image is cropped so that the possible noise in the original image frames can be reduced.

On the other hand, once the processing is completed and the final count is obtained, the app shows the possible counting errors highlighted in red colour (Figure 14c), so the user can easily verify if those errors are related with cluttering or clustering problems.

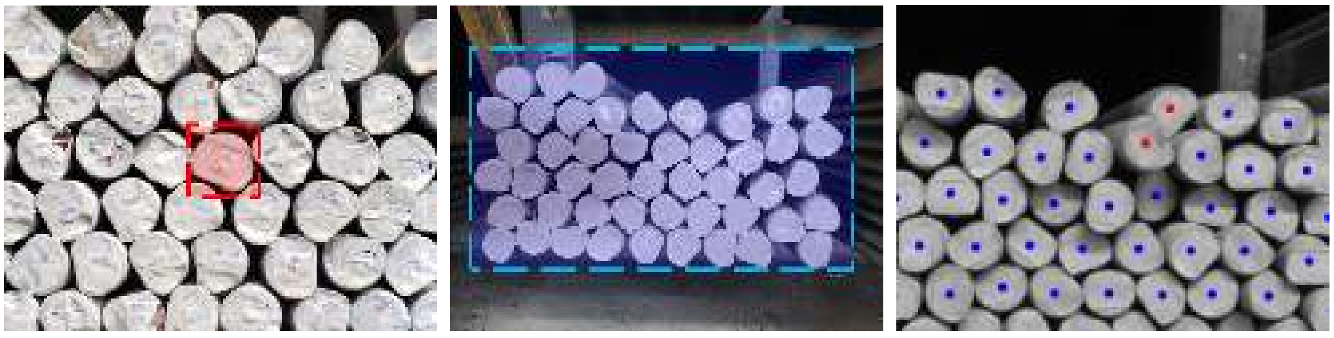

(**a**) (**b**) (**c**)

Figure 14. Screen-shots of the app: (**a**) element selection, (**b**) area of interest selection and (**c**) possible errors highlighted.

Figure 15 shows the flow diagram of the desktop app implementation. First, the app asks for the username and password to login. The user can access the file directory and selects the image. The image is displayed in the density selection window where the user must choose the image density. At this stage, the user has the option to either select a bounding box for an element or for the stack or pile of bars, or both. Once the processing is completed, an output window provides the image with marked centres (where the possible errors also appear), the material type, the final count and the processing time, as shown in Figure 16.

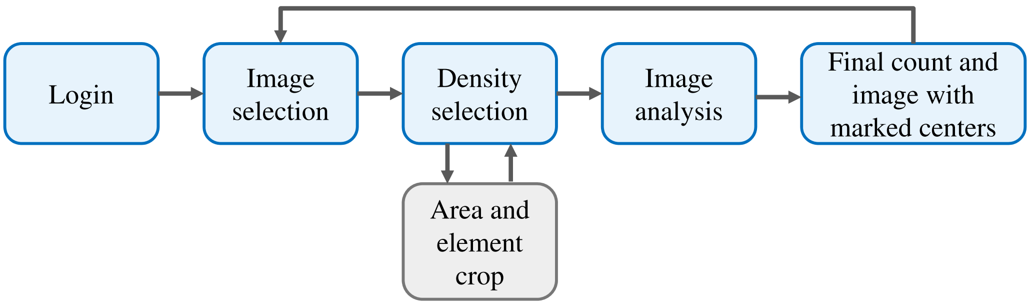


Figure 15. Flow diagram of the developed desktop app where the grey step is optional.

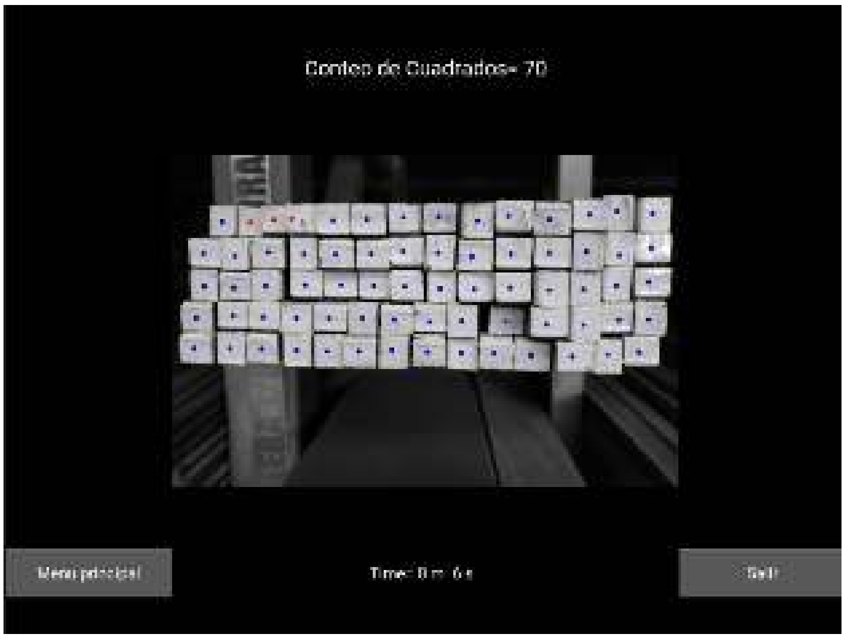


Figure 16. Desktop app output window. The text on top indicates the bar type and the final count while the processing time is shown at the bottom. Red markers are displayed due to material cluttering.

7. Conclusions

Bar counting is a time-consuming and tedious task for the workers in the steel warehouse and it becomes more complicated due to the different sizes of each bar type. Convolutional neural networks have become suitable for this task because they are able to perform as feature extractors, just exploiting their inherent capability to learn abstract concepts and detecting different shape objects. Therefore, the proposed work is a machine learning based methodology capable to identify the bar type and count the number of elements from an image, which has been used in a real steel warehouse with high level of user satisfaction. The addition of challenging steel bars shapes, such as angled and rectangular bars, has been considered for future work. Collecting more images, general improvements and inference time reduction have been also taken into account. More importantly, the implementation of the current networks in an embedded system is a high priority goal.

The SA-CNN-DC methodology shows a modular and portable design implemented by three multilayer networks, processing the responses of a common convolutional encoder and a clustering technique. A compact design approach was adopted with a minimum number of parameters in order to reduce the computational resources but without compromising its performance and accuracy.

Test simulations were carried out in order to validate its proper performance, so that different warehouses images were considered to verify the generalisation capability. Simulation results showed a good performance by considering round and squared bars counting, with an accuracy of 98.81% and 98.57% respectively. Compared with the implementations found in literature, the proposed methodology is capable of achieving competitive results with the minimum computational resources. Moreover, its modular design allows the addition of new bar types in a simple way, thus fulfilling the real warehouses expectations.

The usage of the desktop app in the steel warehouse has drastically reduced the counting time for round and squared bars. Moreover, there is a higher confidence in the inventory due to the low error rate and the possible errors marks. Painting materials for manual counting is no longer necessary which results in a significant reduction of resources and environmental damage. Finally, productivity is improved since employees can focus on other activities, while avoiding the exhaustive activity of counting a large amount of elements in a hostile environment.

Author Contributions: Conceptualisation, A.C.H.-R. and J.D.B.-P.; Methodology, A.C.H.-R. and J.D.B.-P.; Software, A.C.H.-R.; Validation, A.C.H.-R.; Formal analysis, A.C.H.-R. and J.D.B.-P.; Investigation, A.C.H.-R. and J.D.B.-P.; Resources, A.C.H.-R. and J.D.B.-P.; Data Curation, A.C.H.-R.; Writing—original draft preparation, A.C.H.-R., J.D.B.-P. and J.A.M.-N.; Writing—review and editing, A.C.H.-R., J.D.B.-P. and J.A.M.-N.; Visualisation, A.C.H.-R., J.D.B.-P. and J.A.M.-N.; and Supervision, A.C.H.-R. and J.D.B.-P.; Funding Acquisition, A.C.H.-R. and J.D.B.-P. All authors have read and agreed to the published version of the manuscript.

Funding: This research work has been partially supported by the National Council of Science and Technology (CONACYT) by the M.Sc. Grant with CVU No. 855821.

Acknowledgments: The authors would like to thank *DIACO* and *TEX* steel warehouses for their support during this research, especially to the employees who helped take photographs when needed.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Zhang, D.; Xie, Z.; Wang, C. Bar section image enhancement and positioning method in on-line steel bar counting and automatic separating system. In Proceedings of the 2008 Congress on Image and Signal Processing, Sanya, China, 27–30 May 2008; pp. 319–323. [CrossRef]
2. Zhao, J.; Xia, X.; Wang, H.; Kong, S. Design of real-time steel bars recognition system based on machine vision. In Proceedings of the 2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, 27–28 August 2016; pp. 505–509. [CrossRef]
3. Otsu, N. A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66. [CrossRef]
4. Wu, Y.; Zhou, X.; Zhang, Y. Steel bars counting and splitting method based on machine vision. In Proceedings of the 2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), Shenyang, China, 8–12 June 2015; pp. 420–425. [CrossRef]
5. Wang, J.; Hao, C.; Xiaoqing, X. Pattern recognition for counting of bounded bar steel. In Proceedings of the Fourth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2011), Stevens Point, WI, USA, 4–6 August 2011; pp. 173–176. [CrossRef]
6. Su, Z.; Fang, K.; Peng, Z.; Feng, Z. Rebar automatically counting on the product line. In Proceedings of the 2010 IEEE International Conference on Progress in Informatics and Computing, Shanghai, China, 10–12 December 2010; pp. 756–760. [CrossRef]
7. Nie, Z.; Hung, M.-H.; Huang, J. A novel algorithm of rebar counting on conveyor belt based on machine vision. *J. Inf. Hiding Multimed. Sign. Process* **2016**, *7*, 425–437.
8. Youlian, Z. Research of image recognition arithmetic in automatic counting system of steel bars. *J. Wuhan Eng. Inst.* **2008**, *20*, 31–34.
9. Chi, L. A counting method for bundled steel bars based on image processing. *J. Shenyang Univ. Technol.* **2016**, *38*, 551–554. [CrossRef]
10. Dahou, Z.; Sbartaï, Z.M.; Castel, A.; Ghomari, F. Artificial neural network model for steel–concrete bond prediction. *Eng. Struct.* **2009**, *31*, 1724–1733. [CrossRef]
11. Xiaohu, L.; Jineng, O. Research on steel bar detection and counting method based on contours. In Proceedings of the 2018 International Conference on Electronics Technology (ICET), Chengdu, China, 23–27 May 2018; pp. 294–297. [CrossRef]
12. Cohen, J.P.; Boucher, G.; Glastonbury, C.A.; Lo, H.Z.; Bengio, Y. Count-ception: Counting by Fully Convolutional Redundant Counting. In Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), Venice, Italy, 22–29 October 2017; pp. 18–26. [CrossRef]

13. Rahnemoonfar, M.; Sheppard, C. Deep Count: Fruit Counting Based on Deep Simulated Learning. *Sensors* **2017**, *17*, 905. [CrossRef] [PubMed]
14. Sam, D.B.; Surya, S.; Babu, R.V. Switching Convolutional Neural Network for Crowd Counting. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 4031–4039. [CrossRef]
15. Zeng, L.; Xu, X.; Cai, B.; Qiu, S.; Zhang, T. Multi-scale convolutional neural networks for crowd counting. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 465–469. [CrossRef]
16. Seguí, S.; Pujol, O.; Vitrià, J. Learning to count with deep object features. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Boston, MA, USA, 7–12 June 2015; pp. 90–96. [CrossRef]
17. Li, X.; Jia, X.; Wang, Y.; Yang, S.; Zhao, H.; Lee, J. Industrial Remaining Useful Life Prediction by Partial Observation Using Deep Learning With Supervised Attention. *IEEE/ASME Trans. Mechatron.* **2020**, *25*, 2241–2251. [CrossRef]
18. Li, X.; Jia, X.; Yang, Q.; Lee, J. Quality analysis in metal additive manufacturing with deep learning. *J. Intell. Manuf.* **2020**, *31*, 2003–2017. [CrossRef]
19. Fan, Z.; Lu, J.; Qiu, B.; Jiang, T.; An, K.; Josephraj, A.N.; Wei, C. Automated steel bar counting and center localization with convolutional neural networks. *arXiv* **2019**, arXiv:1906.00891.
20. Zhu, Y.; Tang, C.; Liu, H.; Huang, P. End-Face Localization and Segmentation of Steel Bar Based on Convolution Neural Network. *IEEE Access* **2020**, *8*, 74679–74690. [CrossRef]
21. Lippmann, R. An introduction to computing with neural nets. *IEEE ASSP Mag.* **1987**, *4*, 4–22. [CrossRef]
22. Weiss, K.; Khoshgoftaar, T.M.; Wang, D. A survey of transfer learning. *J. Big Data* **2016**, *3*, 9. [CrossRef]
23. Alom, M.Z.; Taha, T.M.; Yakopcic, C.; Westberg, S.; Sidike, P.; Nasrin, M.S.; Hasan, M.; Van Essen, B.C.; Awwal, A.A.S.; Asari, V.K. A State-of-the-Art Survey on Deep Learning Theory and Architectures. *Electronics* **2019**, *8*, 292. [CrossRef]
24. Bell, N.; Garland, M. *Efficient Sparse Matrix-Vector Multiplication on CUDA*; NVIDIA Technical Report NVR-2008-004; NVIDIA Corporation: Santa Clara, CA, USA, December 2008.
25. Keras: Simple. Flexible. Powerful. Available online: https://keras.io (accessed on 12 October 2020).
26. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
27. Sarang Narkhede. Understanding AUC-ROC Curve. 2018. Available online: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5 (accessed on 12 October 2020).
28. Ying, X.; Wei, X.; Pei-xin, Y.; Qing-da, H.; Chang-hai, C. Research on an Automatic Counting Method for Steel Bars' Image. In Proceedings of the 2010 International Conference on Electrical and Control Engineering, Wuhan, China, 25–27 June 2010; pp. 1644–1647. [CrossRef]