information

MDPI

Article

Recognition of Industrial Spare Parts Using an Optimized Convolutional Neural Network Model

Chandralekha Mohan [1,*], Takfarinas Saber [2] and Priyadharshini Jayadurga Nallathambi [1]

[1] Department of Computer Science and Engineering, Amrita School of Computing, Amrita Vishwa Vidyapeetham, Chennai 601103, India; np_jayadurga@ch.students.amrita.edu
[2] Lero—The Irish Software Research Centre, School of Computer Science, University of Galway, H91 TK33 Galway, Ireland; takfarinas.saber@universityofgalway.ie
* Correspondence: m_chandralekha@ch.amrita.edu

Abstract: Spare parts search and retrieval processes are of paramount importance in manufacturing and supply chains. Image recognition using 2D and 3D image properties plays an important part in the success of such processes, as it facilitates the identification of the types and components associated with spare parts, a step that is crucial for their success. In this article, a novel Deep Learning-based object recognition model based on a convolutional neural network architecture is proposed and constructed using stacked convolutional layers to extract and learn features of the spare parts efficiently with the goal of improving the effectiveness of the spare part image recognition process. The proposed model is assessed using industrial spare parts datasets, and its performance is compared against different transfer learning models using precision, accuracy, recall, and F1 score. The proposed model demonstrated efficiency in spare parts recognition and achieved the highest accuracy compared to state-of-the-art image recognition models.

Keywords: deep learning; convolutional neural network; industrial spare parts; stacked convolutional layer; accuracy and loss; confusion matrix

Citation: Mohan, C.; Saber, T.; Nallathambi, P.J. Recognition of Industrial Spare Parts Using an Optimized Convolutional Neural Network Model. *Information* **2024**, *15*, 793. https://doi.org/10.3390/info15120793

Academic Editor: Marco Leo

Received: 1 October 2024
Revised: 10 November 2024
Accepted: 12 November 2024
Published: 10 December 2024

1. Introduction

In the manufacturing industry, there has been an increasing demand for efficient and sophisticated methods for searching for and retrieving spare parts with a similar appearance. Searching for and retrieving spare parts requires a large amount of work, and it is time-consuming to match spare parts manually. Traditional methods for searching for spare parts often rely on large and complex feature-based catalogs, whereas spare part retrieval is often based on annotations.

With the advent of the machine learning (ML) approach, spare parts recognition has been explored in manufacturing industries with respect to various contexts, such as production, process planning, and inventory management [1]. Common ML techniques include approaches that group similar products and match parts that share common features [2]. However, industrial spare parts can be identical while performing different functions and can perform the same function while having different shapes. Therefore, spare parts recognition requires more than an understanding of shape features. When parts are attached to a specific functional set, utilizing functional properties to recognize a spare part becomes more effective in providing information to distinguish it from other components or spare parts [3].

Spare parts recognition systems that are based on image recognition use features from the image to precisely recognize the particular spare part it depicts. Various image characteristics, such as texture features, shape features, color features, spatial features, statistical features, and local features, are utilized in object recognition [4].

In computer vision, deep learning (DL) techniques are widely used for product classification and object recognition in various fields, such as medical imaging, remote sensing,

security, manufacturing, logistics, and supply chains. In the manufacturing industry, object recognition is mainly used for fault diagnosis, product assembly, spare parts identification, and inventory management [5]. In the manufacturing industry, spare parts recognition is widely used for inventory management and control, process management, fault replacement, and after-sales service support.

The classification of spare parts involves categorizing spare parts into product category, function, utilization, manufacturer, and supplier, as well as retrieval from its corresponding category [6]. In the manufacturing industry, recognition of spare parts can be difficult because of the nature of the spare parts, which varies in quality, function, raw material, and suitability to certain components, brands, or models. Managing thousands of spare parts required for a particular industry is difficult, and spare parts management requires a sophisticated system to recognize and retrieve spare parts efficiently. Thus, the classification of spare parts becomes essential for recognizing and retrieving spare parts.

A convolutional neural network (CNN)-driven classification model is described in this article to reliably and precisely recognize industrial spare parts while addressing issues in spare parts search and retrieval. The proposed model has several advantages, including automatic feature learning, the ability to accommodate large volumes of data, accurate predictions, and a hierarchical learning ability [7]. The proposed model classifies the input color image into the corresponding product category through feature recognition in RGB layers. The recommended model's performance is evaluated by comparing it to state-of-the-art approaches employing various industry datasets.

The present research work focuses on matching the correct spare part and its corresponding product category with respect to the spare part's function. The present work contributes to ensuring that the likelihood information about a spare part is accurate and its suitability to a specific function, which is important for improving the accuracy of spare parts recognition and retrieval. Fine-tuning the hyperparameters of a CNN is crucial to model performance. Model performance is greatly impacted by the choice of activation function, the number of convolutional layers, the kernel size, the pooling size, the number of dense layers, the regularization of weights, the dropout layer, batch size, and the learning rate. The optimization of parameters has a direct effect on feature learning and model performance; thus, the objective of the proposed CNN model is to achieve the right balance of hyperparameters, which improves the model's performance with high accuracy in object classification for industrial spare parts recognition.

This paper is organized into five sections. An introduction to computer vision in object recognition is discussed in Section 1. Section 2 discusses related works. Section 3 introduces the proposed CNN model. Section 4 describes the experiments and analysis. Section 5 provides the results and discussion. Finally, Section 6 concludes this paper.

2. Related Works

Image classification is improving through the development of different models. In computer vision, CNN models are effectively deployed to discriminate features in object classification. In this section, various works that use deep learning models for object recognition are discussed. The productivity of any machine learning model depends on its ability to learn features, and deep learning models are especially adept at acquiring intricate characteristics. High-level features include objects, scenes, and events, while low-level features include properties such as edges, colors, angles, and so on.

Ref. [8] proposed a deep learning model named MelNet. The model uses a pyramid scheme to extract features, deep convolutional layers are stacked with ReLU activation, and the L2 regularization technique is employed to avoid overfitting. The model was trained on the KITTI dataset with nine classes. The nine classes included different objects such as car, don't care, van, pedestrian, cyclist, truck, person sitting, misc, and tram. Compared to MobileNetV3 and YOLOv5, the proposed model outperformed other models with a precision rate of 73%, while EfficientNet achieved 77% precision. The study concluded that the effectiveness of a model depends on the broader dataset used for training. Ref. [9]

proposed a transfer learning-based deep learning model using multi-scale fusion to address the local information loss while learning. The MFIL-FCOS model was built using ResNet-50 and was based on the idea of extracting low-level features (shape and color) and fusing them with high-level features. The fusion-based method was tested on the COCO and DIOR datasets using the SGD optimizer. The COCO dataset consists of 80 object classes (remote, book, mouse, chair, cup, carrot, etc.), and the DIOR dataset consists of 20 categories, including ship, airport, tennis court, bridge, etc. The proposed model achieved precision scores of 72.2% on the DIOR dataset and 41.6% on the COCO dataset. Ref. [10] proposed a neural network model to overcome the limitation of multi-frame technology. Using one-shot object detection and a multi-object detection method, the approach eliminates the limitations of existing models. The model's performance was compared with that of Slowfast and Openpose on the COCO dataset. The network was constructed using the YOLO network with the SGD optimizer to find the minimal loss value and consisted of 12.1 million trainable parameters. The proposed model achieved an accuracy of 74.95%, outperforming Openpose (70.08%) and Slowfast (70.16%). The SGD optimizer is good at regularization but weak in generalization and converges slowly. Ref. [11] proposed a deep learning model for object recognition using the ResNet architecture. The transfer learning model was trained on 1000 classes using the ImageNet dataset and used the SGD optimizer. The performance of the model was compared against that of SVM+SIFT and LeNet. Utilizing features extracted from the ResNet model, the proposed model exhibited an accuracy of 80%, while SVM+SIFT achieved an accuracy of 50% and LeNet achieved an accuracy of 65%. Ref. [12] proposed a deep learning model to improve image recognition. The model was optimized for noise reduction and improved feature extraction. The method employs a modified VGG network and was compared against AlexNet and LeNet. The proposed model was tested on the Flower dataset, which contains 17 classes. The modified VGG model achieved an accuracy of 72%, outperforming LeNet (58%) and AlexNet (41%). Ref. [13] proposed an intelligent machine vision model for defect prediction in products using a CNN. The proposed method was compared with three pre-trained models (VGG-16, Inception v3, and EfficientNetB0) using defect and defect-free product images. The proposed method achieved an accuracy of 90.04%, while VGG16 achieved an accuracy of 95.6%, Inceptionv3 achieved an accuracy of 93.2%, and EfficientNetB0 achieved an accuracy of 96.8%. The proposed method's performance was low compared to that of the pre-trained models. The study recommended Inceptionv3 for defect classification and to improve the manufacturing process. Ref. [14] proposed a deep learning method to identify faulty images captured by industrial cameras. In the manufacturing sector, industrial camera images become faulty for various reasons. To improve the quality of automatic identification of faulty images, the authors introduced a deep learning artificial intelligence system. The proposed method was compared with KNN, SVM, MLP, and NN (BP), and the proposed CNN-based model outperformed other methods with 91% accuracy. Ref. [4] proposed a deep learning model for object recognition. The proposed model initially identifies features that are extracted using two pre-trained models (Vgg16 and Inception V3). Using a covariance method, the features are fused together, and the best features are selected using the multi-logistic regression method. The performance of the proposed method was tested using four different datasets, and the proposed method achieved accuracies of 98% on the Butterflies dataset, 95.55% on the Calctech101 dataset, 69.76% on the CIFAR-100 dataset, and 100% on the Birds dataset. Calctech101 consists of 101 object categories, including animals, instruments, helicopters, chairs, etc., while the CIFAR-100 dataset consists of 100 object classes. The selection of features through the MRcEV method influenced the classifier to achieve better results. Ref. [15] investigated the suitability of deep learning networks on satellite images since these types of radar images have six layers. The six-channel image data were applied to four layers of a CNN. Radar images have random speckle noise, which affects the feature learning of the models. The proposed model was evaluated on the Flevoland dataset, achieving an accuracy of 92.46%. By using spatial features in the image, the network model learns the distinguishing

features for 15 classes. Ref. [16] proposed an activation function called exponential linear units. This activation function aims to minimize the vanishing gradient problem that arises in other activation functions. The performance of the proposed activation function with a seven-layer CNN model was evaluated with other models on the CIFAR dataset. The ELU improved the learning rate and generalization. The ELU achieved good performance with a 24% error rate on the CIFAR-100 dataset. Deep learning has become a popular method for achieving automated learning and classification. Although deep learning models are efficient in terms of feature learning and classification as a single step, achieving novel performance with respect to different tasks in different applications is still under development. Ref. [9] used a feature fusion method to detect objects in remote sensing images, but the model achieved a precision of only 72%. Ref. [10] eliminated the problem of multi-frame analysis, which not only detects objects but also the actions associated with the objects. The DL model achieved an accuracy of 75%. Ref. [12] modified the VGG network to improve noise reduction and feature extraction, and the model achieved an accuracy of 72%. Ref. [13] constructed and optimized a four-layer CNN model for defect detection. The proposed model achieved an accuracy of 90.04%. Ref. [14] developed a CNN model to detect faulty images and improve the quality of automatic identification of faulty images. The model achieved an accuracy of 91% using the tanh activation function.

3. Proposed Approach

This section describes the proposed approach in two parts. The first part details the CNN architecture, and the second part describes the proposed model for addressing spare parts recognition and classification.

3.1. Convolutional Neural Network

In image classification, the image is represented as a fixed-length vector, and classifiers using these vector representations learn to classify the image, as shown in Figure 1. A convolutional neural network (CNN) consists of an input layer, a convolutional layer, an activation layer, a pooling layer, a fully connected layer, and an output layer. The convolutional layer is composed of mathematical linear operations referred to as convolutions, which output a new function from a set of functions i and j. The input vector $\mathbf{X}$ represents the image input (height, width, and color channels), and the CNN learns from the set of functions $f(h, w)$ that maps the input $\mathbf{X}$ to the predicted class C. The output image $O(h, w)$ is given by

$$O(h, w) = g(h, w) * f(h, w) \tag{1}$$

A two-dimensional convolution is defined as

$$g[x, y] = f(h, w)[x, y] = \sum_i \sum_j [i, j] \cdot w[x - i, y - j] \tag{2}$$

where h and w are the input image height and width, and x and y are the rows and columns represented in matrix form. This matrix is the kernel that moves along the horizontal and vertical directions of the image. For a digital image, the image is represented in pixels, and for a single matrix, it denotes a grayscale image with a single channel. Color images have three channels (RGB), which are represented in three matrices. For a 2D CNN, the kernel moves horizontally and vertically to generate a feature map, and the sliding of the kernel across the input vector is termed a convolution operation [17]. To limit the number of outputs from the convolutional layer, a non-linearity is introduced into the network. ReLU, sigmoid, and tanh are the three non-linearities commonly used in CNNs, but ReLU (Rectified Linear Unit) converges faster and better than sigmoid and tanh. The non-linearity is represented as [16]

$$\mathrm{ReLU}(x) = \max(0, x) \tag{3}$$

$$\begin{aligned} \text{ReLU}(x) &= 1 \quad \text{for } x > 0 \\ &\text{and} \\ \text{ReLU}(x) &= 0 \quad \text{for } x \leq 0 \end{aligned} \tag{4}$$

Pooling layers aim to reduce the dimensions of the input vector while retaining higher-level features. In the input feature, a window is passed through the pooling layers, and the elements inside the window are selected to pass through a pooling function to downsample the input features. Average pooling and max-pooling techniques are widely used in classification problems. Max-pooling returns an output vector with maximum values, while average pooling returns average values [15]. Downsampling aims to reduce the number of training parameters and improve learning performance. Max-pooling can be expressed as [18]

$$h_{xy}^{l} = \max_{i=0\ldots n, j=0\ldots n} h_{(x+i)(y+j)}^{(l-1)} \tag{5}$$

where s is the stride and h^l is the activation function at layer l. The fully connected layer represents the fully connected nodes, where the feature vector is flattened by converting it into a 1D vector. From the input layers to the output layers, all the layers are fully connected, and a dot product of the weight vector and input vector is executed to yield the final output. The weights and bias are represented as an output function of the network (N_0):

$$N_0 = b + w_1 X_1 + w_2 X_2 + \ldots + w_n X_n \tag{6}$$

where w represents the weights, X is the input vector, and n is the number of features in the input vector. Finally, a softmax function is applied to convert the real values of the output vectors to prediction probabilities. The prediction probabilities are in the range between 0 and 1. This function returns the probabilities of each class, and softmax requires the same number of nodes as that of the output layer. The softmax function is given by

$$\text{softmax}(x) = \frac{e^x}{\sum_{j=1}^{k} e^x} \tag{7}$$

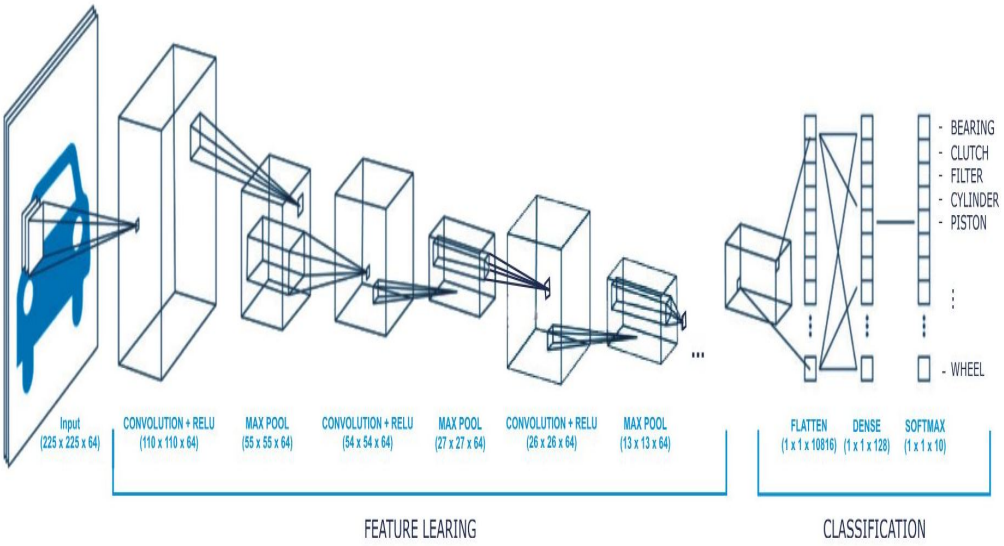


Figure 1. Architecture of the proposed CNN model.

3.2. Proposed CNN Model

In object recognition and computer vision, the performance of any CNN model depends on the right balance of different hyperparameter settings and fine-tuning. The performance of a CNN model changes with respect to the problem and the network size. In object recognition and classification, the model performance is influenced by the hyperparameter settings, the number of training samples, and computing resources. To classify industrial objects correctly, a 2D CNN network is constructed using four convolutional layers as stated in Algorithm 1. A two-dimensional network refers to the kernel that slides in two dimensions. The proposed 2D CNN consists of an input layer, three convolutional layers, and an output layer. The input layer has a size of 225 × 225 × 32 with a batch size of 64. The first layer contains 64 neurons with a kernel size of 3 × 3, a ReLU activation function, and a max-pooling layer of size 2 × 2. The second convolutional layer contains 64 neurons with a kernel size of 3 × 3 and a ReLU activation function, followed by a max-pooling layer of size 2 × 2. A dropout layer is added to the first and second convolutional layers to preserve feature information and avoid overfitting. The third convolutional layer contains 64 neurons with a kernel size of 2 × 2 and a ReLU activation function, followed by a max-pooling layer of size 2 × 2. The fully connected layer contains 128 dense neurons with a dropout layer and ReLU activation function, and the final output layer contains 10 dense neurons with softmax activation. The proposed CNN model's process flow architecture is given in Figure 1. The total number of parameters is given in Table 1, and the hyperparameter settings are given in Table 2.

Algorithm 1 Image classification using the proposed CNN

Input: image dataset X_i with input size of 256 × 256 × 32
Output: class prediction y_i (256 × 256 × 32)

1: Split dataset into train and test sets
2: Construct CNN architecture:
- Input layer = 1 (3 × 3)
- Convolutional layer = 3 (2 × 2)
- Output layer = 1 Activation = ReLU and activation = softmax
- Max-pooling size = 2 × 2
- Dropout = 0.5

3: Add fully connected dense layer with 128 neurons
4: Add activation = softmax with dense for prediction on n classes
5: Compile the model with optimizer = Adam, learning rate = 0.001, and activation = 'softmax'
6: Define early stopping;
7: **for** each epoch do patience = p;
8: **if** val_loss minimizes **then**
9: Save as best model;
10: **else**
11: Repeat step 7;
12: **end if**
13: End
14: Stop early stopping
15: Evaluate the model on test data

Table 1. Proposed CNN model architecture with the total number of parameters.

Model	Layer	Output Shape	Parameters
2D CNN	Conv2d	(None, 223, 223, 64)	1792
	Max-pooling	(None, 111, 111, 64)	0
	Conv2d	(None, 110, 110, 64)	16,448

Table 1. *Cont.*

Model	Layer	Output Shape	Parameters
	Max-pooling	(None, 55, 55, 64)	0
	Dropout	(None, 55, 55, 64)	0
	Conv2d	(None, 54, 54, 64)	16,448
	Max pooling	(None, 27, 27, 64)	0
	Dropout	(None, 27, 27, 64)	0
	Conv2d	(None, 26, 26, 64)	16,448
	Max-pooling	(None, 13, 13, 64)	0
	Flatten	(None, 10,816)	0
	Dense	(None, 128)	1,384,576
	Dropout	(None, 128)	0
	Dense	(None, 10)	1290
		Total Params	1,437,002
		Trainable Params	1,437,002
		Non-trainable Params	0

Table 2. Hyperparameter settings.

Hyperparameter	Value
Dropout rate	0.5
Learning rate	0.0001
Batch size	64
Epoch	10
Loss function	Categorical Cross-Entropy

3.3. Framework for Real-World Application and Validation of the Proposed Model

To ensure the proposed method's capability to address real-world industrial spare parts problems, a framework is established that involves the following:

1. Real-World Dataset Integration: Adding large, generalized data entries retrieved directly from industry inventories will improve model performance by preparing it to operate on diverse datasets of different manufacturers and sectors.
2. Robustness Testing: The tested robustness of the model addresses aspects such as lighting, occlusions, and the issue of overlapping parts, which are often seen in industrial business environments. These tests help ensure that the accuracy of the model can be sustained, even in field-type conditions.
3. Deployment and Continuous Learning: A deployment strategy for industrial environments is proposed with the integration of the model into an inventory or retrieval system. The aforementioned framework includes provisions for learning with the possibility of enhancement as newer data are generated.
4. Performance Metrics for Industrial Applications: Apart from sheer accuracy and precision, the framework introduces new metrics, like retrieval time, individual error rate under working conditions, and the ability to advance to new spare parts classes. These metrics enable a detailed evaluation of the model in real-life situations.
5. Collaborative Pilot Program: Industry partners and stakeholders may be involved in the implementation of a pilot project to trial the model. This program will provide a tangible understanding of the model and ascertain its readiness to be implemented commercially.

This framework shows the feasibility and versatility of the proposed model in addressing existing industrial spare parts recognition issues.

4. Experiment and Analysis

This section presents the experimental setup, dataset employed, state-of-the-art techniques, and evaluation metrics.

4.1. Dataset

The performance of the proposed 2D CNN for image classification was evaluated using the car spare parts dataset and the automobile parts dataset. The car spare parts dataset contains 50 car spare parts images with a total size of 150 MB. The dataset was already split into a training set, test set, and validation set. The training set contains a total of 8739 images across 50 classes, the test set contains 250 images across 50 classes, and the validation set contains 250 images across 50 classes. The car spare parts dataset was sourced from www.kaggle.com, available at https://www.kaggle.com/datasets/gpiosenka/car-parts-40-classes, accessed on 10 February 2024. It is suitable for multiclass classification and image recognition tasks. The second dataset was the automobile parts dataset, which was sourced from www.roboflow.com and is available at https://www.kaggle.com/datasets/mdwaquarazam/automobilepartsindentification, accessed on 10 February 2024. The automobile parts dataset contains 688 images across 14 classes with a total size of 40 MB. The automobile parts dataset is split into training, test, and validation sets. Utilizing a complete dataset may require high computing resources and result in long training times. To address this, only a small dataset was utilized in this study. The automobile parts dataset was split into a training set, a test set, and a validation set with 14 classes. The training set of the automobile parts dataset contains 412 images, the testing set contains 138 images, and the validation set contains 138 images. For the car spare parts dataset, out of 50 classes, only 10 classes were utilized for this study, with the training set containing 1744 images, the test set containing 50 images, and the validation set containing 50 images.

4.2. State-of-the-Art Techniques

4.2.1. Inceptionv3

The Inceptionv3 model is an improved version of Inceptionv2 by [19] and was introduced by Google Inc. The Inceptionv3 network is made up of 42 layers and does not contain deep layers. Instead of deep layers, the model has parallel layers. Inceptionv2 is an optimized version of its predecessor with asymmetric convolutions, auxiliary classifiers, a reduced grid size, and factorized smaller convolutions. The network of Inceptionv3 consists of three major parts. In the first part lies the convolutional layer, where the features are extracted from the input image. The inception part consists of parallel convolutions that are stacked horizontally. The last part is the classifier; with auxiliary classifiers, the Inceptionv3 model improves classification accuracy by reducing overfitting. Smaller convolutions reduce the computational cost and the number of parameters to be trained. The convolution part, inception part, and classifiers are successfully stacked in such a way as to follow the convolutional neural network architecture [20].

4.2.2. Xception

Xception is an inspired version of the inception model developed by [21], which contains a linear stack of depthwise convolutions followed by pointwise convolutions. The Xception network is made up of entry flow, middle flow, exit flow, and 36 convolutional layers. The entry flow has single-layer convolution and pooling layers, while the middle flow contains only separable convolutional layers with 3×3 filters. The main difference between Inception and Xception is that Xception is made up of separable convolutions without a non-linearity function, whereas Inception has a ReLU function. The entry flow size for Xception is $299 \times 299 \times 3$, the middle flow size is $19 \times 19 \times 728$, and the exit flow has two modules with 728 and 1024 filters, while the last module has 1536×2048 filters. The Xception model was trained on the ImageNet dataset and has been utilized for various image classification problems.

4.2.3. VGG19

The VGG19 model is an improved version of VGG16 developed by [22]. VGG16 and VGG19 refer to the number of convolutional layers. VGG16 contains a total of 16 convolutional layers and 3 fully connected layers. A max-pooling layer does not follow each of the convolutional layers; instead, five pooling layers are applied across the convolutional layers. In VGG, the size of the kernels is reduced, and the number of convolutional layers is increased in order to reduce the total number of parameters. The advantage of the VGG model is that it converges faster due to the depth of the network and the SGD optimizer. ReLU is used to add non-linearity to the decision function. The VGG19 network was also trained on the ImageNet Dataset.

4.2.4. ResNet50

ResNet50 was developed by [23], and the network is made up of convolutional blocks, identity blocks, and fully connected blocks. ResNet50 has 50 layers in the network. The features are extracted by the convolutional blocks, while the identity blocks transform the extracted features and pass them to the fully connected layer. To counter the vanishing and exploding gradients, the ResNet50 network uses a method called skip connections, which bypass a few layers and connect directly to the flatten layer. For non-linearity, the model uses the ReLU and softmax functions in the fully connected layer. ResNet50 was trained on the ImageNet dataset.

4.2.5. EfficientNet

EfficientNet was developed by [24]. EfficientNetB0 is made up of depthwise separable convolutions and inverted residual blocks. The performance of the network is influenced by the squeeze-and-excitation optimization method. The EfficientNet model scales the depth, width, and resolution of the network in the initial layers. Pointwise convolutional and depthwise convolutional layers reduce the feature map. The skip connections are used for wider layers, and the network uses a Swish activation function instead of ReLU. Batch normalization and the Swish activation function collectively improve the performance of the network. EfficientNetB0 uses the compound model scaling method to improve feature learning by adapting the tuning of dimensions, as scaling increases the dimension.

4.2.6. MobileNetV2

Ref. [25] introduced MobileNet, which uses depthwise separable convolutions and few parameters. The depthwise convolutions use a single filter on the input channels, and the pointwise convolution aggregates the output of the depthwise convolutions using two separate layers. The aggregation of the feature map by pointwise convolutions reduces the feature dimensions. The first layer is a residual block (depthwise convolutions), and the other layer is used for downsizing (pointwise convolutions). MobileNetV2 is a 53-layer deep CNN network, and the network is lightweight with fewer parameters. Three types of convolutional layers are used in EfficieintNetB0. The first layer consists of 1×1 convolutions with non-linearity, which restructure new features; the second layer is a depthwise convolution; and the third layer consists of 1×1 convolutions without non-linearity.

4.3. Evaluation Metrics

The performance of the proposed method was evaluated using a confusion matrix. The confusion matrix helps visualize the actual and predicted classes, as shown in Table 3.

Table 3. Confusion matrix.

Actual	**Predicted**	
	Positive	**Negative**
Positive	True Positive (TP)	False Positive (FP)
Negative	False Negative (FN)	True Negative (TN)

TP are the true positives, where the true positives are correctly classified as positive classes; TN are the true negatives, where the true negatives are correctly classified as negative classes; FP are the false positives, where the actual negative classes are incorrectly classified as positive classes; and FN are the false negatives, where the actual positive classes are incorrectly classified as negative classes.

Classification accuracy determines the model's performance and describes the learning ability of the classifier. The accuracy, sensitivity, specificity, and F1 score are given in Equations (8)–(11).

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{8}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{9}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{10}$$

$$\text{F1 score} = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{11}$$

5. Results and Discussion

This section presents the results and discussion of the proposed model, as well as its performance comparison against state-of-the-art models on the automobile parts and car spare parts datasets.

5.1. Training of the Proposed Model

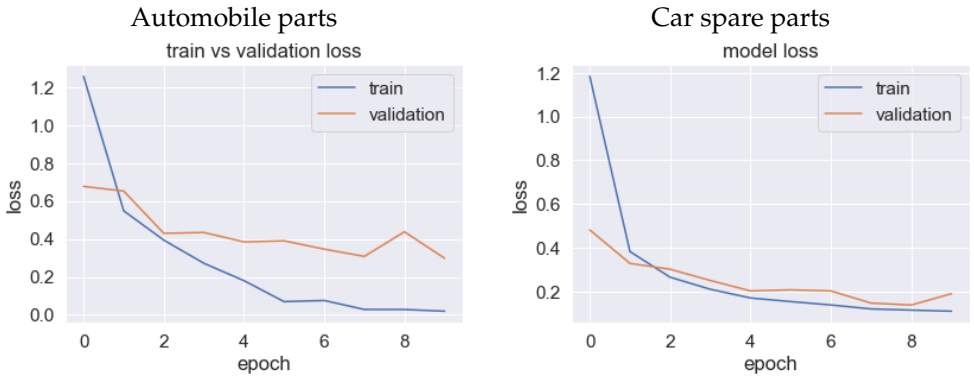
The performance of the proposed model was evaluated on two industrial spare parts datasets. The model was trained and tested on both datasets. Transfer learning models, such as Inceptionv3, Xception, VGG19, ResNet50, EfficientNetB0, and MobileNetV2, were also trained alongside the proposed model. The performance of the proposed model was evaluated using the performance metrics described above. In the proposed model, the network is constructed using three convolutional layers. Each convolutional layer has 64 neurons with a kernel size of 3×3. A non-linearity layer with the ReLU activation function and a max-pooling layer of size 2×2 are used in the network. The fully connected layer contains 128 neurons with the Adam optimizer. On the automobile parts dataset, the Inceptionv3 model achieved a testing accuracy of 91.30%, Xception achieved 86.23%, VGG19 achieved 91.16%, ResNet50 achieved 90.58%, EfficientNetB0 achieved 91.3%, and MobileNetV2 achieved 87.68%. The precision score of EfficientNetB0 was the highest at 92.15%, while IncpetionV3 achieved a precision score of 91.93%. ResNet50 achieved an accuracy of 90.58%, a precision of 91.83%, a recall of 91.71%, and an F1 score of 91.48%. The precision scores of Xception (87.7%), VGG19 (81.16%), and MobileNetV2 (87.68%) were less than 90%. EfficientNetB0 and Inceptionv3 achieved the highest recall values of 92.08% and 92.70%, while Xception, VGG19, and MobileNetV2 achieved values of 87.95%, 83.33%, and 89.45%. The F1 scores for Inceptionv3 (92.10%), ResNet50 (91.48%), and EfficientNetB0 (91.42%) were above 91%. The accuracy on the training set demonstrated the learning ability of the model, and the ability to classify objects in the automobile parts dataset remained higher than 90% on the testing set for Inceptionv3, ResNet50, and EfficientNetB0. However, the distribution of training samples with different color patterns and textures would help the model extract significant features corresponding to each category, thereby increasing its learning ability. Precision refers to the ratio of total positive instances classified as positive, and recall is the ratio of positive cases classified as positive to the total number of positive cases.

On the car spare parts dataset, the Inceptionv3 model achieved a testing accuracy of 92%, Xception achieved 88%, VGG19 achieved 82%, ResNet50 achieved 92%, EfficientNetB0 achieved 92%, and MobileNetV2 achieved 88%. The precision score for Inceptionv3 was 93.80%, while the precision scores for ResNet50 (93.47%), Xception (90.14%), and EfficientNetB0 (93%) were above 90%. VGG19 (85.71%) and MobileNetV2 (89%) achieved

precision scores of less than 90%. Inceptionv3, ResNet50, and EfficientNetB0 achieved precision scores of 91.99%. VGG19 achieved a recall of 82%, Xception achieved 88%, and MobileNetV2 achieved 88%. The F1 score for Inceptionv3 was 92.07%, while the F1 scores for ResNet50 (92.09%) and EfficientNetB0 (91.66%) were above 91%. The F1 scores for Xception (87.69%), VGG19 (81.89%), and MobileNetV2 (87.66%) were less than 90%.

On the car spare parts dataset, outstanding results were achieved by Inceptionv3 and ResNet50, both with accuracies of 92.00% and F1 scores of 92.89%. These models showed a good balance between precision and recall, making them well suited for classification within this dataset. Compared to other models, Xception and EfficientNetB0 demonstrated relatively high performance, with Xception achieving an accuracy of 88.00% and an F1 score of 88.87%, and EfficientNetB0 achieving an accuracy of 91.30% and an F1 score of 91.66%. This shows that these models are also accurate, although not as much as the Inceptionv3 and ResNet50 models. Comparing these results, VGG19 demonstrated the worst performance on this dataset, with an accuracy of 82.00% and an F1 score of 81.89%, indicating that it is a weaker model with regard to classification tasks on this dataset.

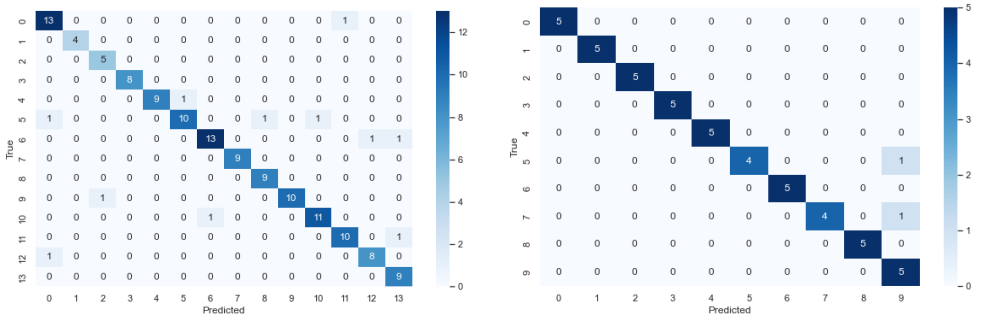
The loss curve, accuracy, and confusion matrix for the proposed model on the automobile parts and car spare parts datasets are given in Figure 2a–c.



(**a**) Training vs. validation Loss

(**b**) Accuracy

Figure 2. *Cont.*



(**c**) Confusion matrix

Figure 2. Loss, accuracy, and confusion matrix of the proposed model on the automobile parts (left) and the car spare parts (right) datasets.

5.2. Performance of the Proposed Model

On the automobile parts dataset, the proposed model achieved a testing accuracy of 92.08%, and on the car spare parts dataset, it achieved a testing accuracy of 96%, as shown in Table 4. The ability of the proposed model to classify objects on the two datasets was higher than 90% on the test set. The precision of 92.64% on the automobile parts dataset shows that the model correctly classified 92.64% of positive instances as positive. The recall of 93.48% indicates that the model correctly classified positive instances out of the total number of positive instances. On the car spare parts dataset, the proposed model achieved an accuracy of 96%, which indicates that its learning ability on the car spare parts dataset was higher compared to the automobile parts dataset. The proposed model exhibited a precision of 92.64% and a recall of 93.48% on the automobile parts dataset, showing that the model correctly classified 92.64% of positive cases, whereas on the car spare parts dataset, the model's precision was higher (4.5%) than its precision on the automobile parts dataset. The model's F1 score on the automobile parts dataset was 92.83%, and on the car spare parts dataset, it was 96.11%. The F1 score of more than 90% indicates that the model can effectively classify positive cases with low false negatives and false positives, and it also indicates that a good balance exists between precision and recall.

Table 4. Performance of the transfer learning models on the automobile parts and car spare parts datasets.

Dataset	Models	Accuracy	Precision	Recall	F1 Score
Automobile Parts	Inceptionv3	91.30%	91.93%	92.70%	92.10%
	Xception	86.23%	87.70%	87.95%	86.45%
	VGG19	81.16%	82.18%	83.33%	81.01%
	ResNet50	90.58%	91.83%	91.71%	91.48%
	EfficientNetB0	91.30%	92.15%	92.08%	91.92%
	MobileNetV2	87.68%	87.62%	89.45%	88.01%
	Proposed	**92.08%**	**92.64%**	**93.48%**	**92.83%**
Car Spare Parts	Inceptionv3	92.00%	93.80%	91.99%	92.07%
	Xception	88.00%	90.14%	88.00%	87.69%
	VGG19	82.00%	85.71%	82.00%	81.89%
	ResNet50	92.00%	93.47%	91.99%	92.09%
	EfficientNetB0	92.00%	93.00%	91.99%	91.66%
	MobileNetV2	88.00%	89.00%	88.00%	87.66%
	Proposed	**96.00%**	**97.14%**	**96.00%**	**96.11%**

From the evaluation, it is evident that the performance of the proposed model on the automobile parts and car spare parts datasets was superior to that of Inceptionv3, Xception, VGG19, ResNet50, EfficientNetB0, and MobileNetV2. On the car spare parts dataset, the proposed model achieved the best performance in terms of accuracy and overall balance across the evaluation metrics. In the case of the proposed model, the overall

accuracy was 96.00%, the precision was 97.14%, the recall was 96.00%, and the F1 score was 96.11%. These metrics demonstrate the high accuracy of the model as a classifier of car spare parts, and the ratio of true positives to false positives and false negatives is optimal. It is particularly significant to bear in mind other models that are expected to offer similar performance. For example, the ResNet50 and EfficientNetB0 models achieved accuracies of 92.00% and F1 scores of 92.09% and 91.66% on the car spare parts dataset. Although these models achieved satisfactory results, the proposed model achieved better generalization and seems to be more effective for complex industrial datasets compared to the other models in terms of precision and recall efficiency.

The proposed model has some hyperparameter settings that enhance its feature extraction and learning capabilities. These include multiple convolutional layers, dropout parameters, and an optimized learning rate. Other models used, like Inceptionv3, yielded a satisfactory accuracy of 92.00% and performed well; yet, the efforts made in refining the precision and F1 score make the proposed model more useful for the classification tasks in this dataset. Such performance highlights the fact that the proposed model is highly relevant for real-time recognition of objects in complex industrial settings. This efficient architecture for learning relevant features is thus a useful tool for deploying deep learning in functional industries. Also, the architecture of the proposed model emphasizes the fact that industries need deep learning models specific to particular domains. Performance could be further improved with data augmentation, collecting more data for training the model, and optimizing the methods used to train models for better results in different industrial settings.

6. Conclusions

Retrieving spare parts manually is time-consuming, and traditionally, spare parts are retrieved using annotations that do not match the functional context of the spare parts. Retrieval of spare parts across thousands of products should ensure that the retrieved product is accurate and suitable in the functional context. To correctly classify objects, a 2D CNN network was constructed using four convolutional layers. This paper proposed a deep learning model for industrial spare parts recognition using a CNN. The performance of the proposed model was evaluated using accuracy, precision, recall, and F1 score on two image datasets, namely the automobile parts and car spare parts datasets. The proposed model achieved an accuracy of 92.08%, a precision of 92.64%, a recall of 93.48%, and an F1 score of 92.83% on the automobile parts dataset, and an accuracy of 96%, a precision of 97.14%, a recall of 96.00%, and an F1 score of 96.11% on the car spare parts dataset. Spare parts retrieval in industrial settings is challenging due to the presence of additional objects, poor lighting, and spare parts embedded into components. The presence of additional objects may interfere with feature learning, leading to diminished performance, as features from two objects may overlap each other. The depth of the network and the non-linearity improved feature learning in the proposed model while Xception, VGG19, ResNet50, and MobileNetV2 suffered from vanishing gradients on both the automobile parts and car spare parts datasets. The ReLU activation function and dropout layer effectively avoided overfitting in the proposed CNN model. Transfer learning in some models did not aid object recognition due to the limited availability of training images in industrial settings. The experimental study demonstrated that appropriate training with desired objects enhances object recognition in industrial settings. The effectiveness of the model in recognizing objects can be further improved through better feature learning and optimization techniques. In future work, feature-fusion techniques will be incorporated along with transfer learning to improve the object recognition rate.

Author Contributions: In this paper, C.M., T.S., and P.J.N. contributed significantly and equally to the research and manuscript preparation. C.M., as a corresponding author, not only coordinated the research project but was integral to the research design, execution, and manuscript preparation, ensuring that all aspects of the work were seamlessly integrated and effectively communicated. T.S. played a crucial role in formulating the research question, developing the methodology, conducting

experiments, and analyzing the results, contributing to the thorough exploration and documentation of the findings. P.J.N., also serving as a corresponding author, managed the manuscript submissions and communications with the journal during the peer review process, in addition to her substantial involvement in the research activities. Their collaborative efforts encompassed hypothesis development, experimental design, data interpretation, and writing, reflecting a deep commitment to advancing knowledge in the field of industrial spare parts recognition using deep learning technologies. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used in this study were obtained from Kaggle, a publicly accessible online data repository. The dataset can be accessed and downloaded directly from the Kaggle website at https://www.kaggle.com/datasets/gpiosenka/car-parts-40-classes, accessed on 10 February 2024. All data used in this research are freely available to the public, and proper citation of the dataset source is provided in accordance with the terms of use specified by Kaggle.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wuest, T.; Weimer, D.; Irgens, C.; Thoben, K.-D. Machine learning in manufacturing: Advantages, challenges, and applications. *Prod. Manuf. Res.* **2016**, *4*, 23–45. [CrossRef]
2. Sizemore, N.E.; Nogueira, M.L.; Greis, N.P.; Davies, M.A. Application of Machine Learning to the Prediction of Surface Roughness in Diamond Machining. *Procedia Manuf.* **2020**, *48*, 1029–1040. [CrossRef]
3. Zhao, C.; Dinar, M.; Melkote, S.N. Automated Classification of Manufacturing Process Capability Utilizing Part Shape, Material, and Quality Attributes. *J. Comput. Inf. Sci. Eng.* **2020**, *20*, 021011. [CrossRef]
4. Rashid, M.; Khan, M.A.; Alhaisoni, M.; Wang, S.-H.; Naqvi, S.R.; Rehman, A.; Saba, T. A Sustainable Deep Learning Framework for Object Recognition Using Multi-Layers Deep Features Fusion and Selection. *Sustainability* **2020**, *12*, 5037. [CrossRef]
5. Bansal, M.; Kumar, M.; Kumar, M. 2D object recognition: A comparative analysis of SIFT, SURF and ORB feature descriptors. *Multimed. Tools Appl.* **2021**, *80*, 18839–18857. [CrossRef]
6. Mtasher, A.K.; Msad, J.J. Similar image retrieval using convolutional neural networks: A study of feature extraction techniques. *AIP Conf. Proc.* **2024**, *3092*, 040015. [CrossRef]
7. Desai, P.; Pujari, J. Artificial Intelligence Framework for Content-Based Image Retrieval: Performance Analysis. In *Congress on Intelligent Systems*; Saraswat, M., Sharma, H., Balachandran, K., Kim, J.H., Bansal, J.C., Eds.; Springer Nature: Singapore, 2022; pp. 535–547.
8. Azadvatan, Y.; Kurt, M. MelNet: A Real-Time Deep Learning Algorithm for Object Detection. *arXiv* **2024**, arXiv:2401.17972.
9. Zhang, G.; Yu, W.; Hou, R. MFIL-FCOS: A Multi-Scale Fusion and Interactive Learning Method for 2D Object Detection and Remote Sensing Image Detection. *Remote Sens.* **2024**, *16*, 936. [CrossRef]
10. Yoo, H.; Lee, S.-E.; Chung, K. Deep Learning-Based Action Classification Using One-Shot Object Detection. *Comput. Mater. Contin.* **2023**, *76*, 1343–1359. [CrossRef]
11. Liu, B.; Yu, L.; Che, C.; Lin, Q.; Hu, H.; Zhao, X. Integration and Performance Analysis of Artificial Intelligence and Computer Vision Based on Deep Learning Algorithms. *arXiv* **2023**, arXiv:2312.12872. [CrossRef]
12. Lv, Q.; Zhang, S.; Wang, Y. Deep Learning Model of Image Classification Using Machine Learning. *Adv. Multimed.* **2022**, *2022*, 3351256. [CrossRef]
13. Benbarrad, T.; Salhaoui, M.; Kenitar, S.B.; Arioua, M. Intelligent Machine Vision Model for Defective Product Inspection Based on Machine Learning. *J. Sens. Actuator Netw.* **2021**, *10*, 7. [CrossRef]
14. Wu, H.; Zhou, Z. Using Convolution Neural Network for Defective Image Classification of Industrial Components. *Mob. Inf. Syst.* **2021**, *2021*, 9092589. [CrossRef]
15. Zhou, Y.; Wang, H.; Xu, F.; Jin, Y.-Q. Polarimetric SAR Image Classification Using Deep Convolutional Neural Networks. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 1935–1939. [CrossRef]
16. Clevert, D.-A.; Unterthiner, T.; Hochreiter, S. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). *arXiv* **2016**, arXiv:1511.07289.
17. Lee, K.B.; Cheon, S.; Kim, C.O. A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes. *IEEE Trans. Semicond. Manuf.* **2017**, *30*, 135–142. [CrossRef]
18. Alajrami, E.; Ashqar, B.A.M.; Abu-Nasser, B.S.; Khalil, A.J.; Musleh, M.M.; Barhoom, A.M.; Abu-Naser, S.S. Handwritten Signature Verification Using Deep Learning. 2020. Available online: https://philarchive.org/rec/ALAHSV (accessed on 10 February 2024).
19. Meena, G.; Mohbey, K.K.; Kumar, S. Sentiment analysis on images using convolutional neural networks based Inception-V3 transfer learning approach. *Int. J. Inf. Manag. Data Insights* **2023**, *3*, 100174. [CrossRef]

20. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826. [CrossRef]
21. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1800–1807. [CrossRef]
22. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:1409.1556.
23. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]
24. Tan, M.; Le, Q.V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *arXiv* **2020**, arXiv:1905.11946.
25. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.