Al-Furat Journal of Innovations in Electronics and Computer Engineering (FJIECE)
ISSN -2708-3985

Detecting and Classification of Complex Roads Defects Using a Deep Learning Approach: A survey

Amal Mohammed jaber[1], Farah Abbas Obaid Sari[2]*
[1]University of Kufa, Faculty of Computer Science and Mathematics, Iraq. E-mail: Amalm.alkreiti@student.uokufa.edu.iq
[2]University of Kufa; Faculty of Computer Science and Mathematics, Iraq.
E-mail: faraha.altaee@uokufa.edu.iq

https://doi.org/10.46649/fjiece.v3.2.25a.4.6.2024

Abstract. *Road safety is one of the most important conditions for public driving safety around the world. This is done by monitoring the roads periodically to detect defects that can appear due to several factors affecting the asphalt, such as weather and accidents. Traditional road inspection methods are often time-consuming, labor-intensive and expensive due to the length of the methods. Deep learning (DL) has emerged as a promising technology for automated detection and classification of road defects, offering the potential for faster, more accurate, and less expensive inspections. This review paper provides a comprehensive overview of the state-of-the-art methods working on the principle of artificial intelligence for detecting defects on road surfaces. We will also discuss the different types of defect monitoring methods that differ from one method to another and furthermore, we will show the different datasets from oldest to newest with all their details that can be useful in DL. A review is then performed of various blind learning architectures and algorithms that have been used for defect detection and classification, incorporating generative adversarial networks (GANs), recurrent neural networks (RNNs), and convolutional neural networks (CNNs). Finally, we will provide a comparative analysis of the reviewed algorithms and methods by presenting the metrics used to choose the best technique, which is what these paper aims to do.*

Keywords: Deep Learning, Road monitoring, Detection algorithms and classification algorithms, CNN

1. INTRODUCTION

Road monitoring involves the crucial task of enhancing their efficiency to reduce accidents, traffic congestion and arrival time. To begin any maintenance work, road problems must be evaluated by forming committees to detect defects or reporting them by some people. The previous methods are considered traditional and expensive in terms of human and financial terms, in addition to the time taken [1]. Visual defects indicate defects in roads, and recognizing these defects can lead to practical solutions. However, more than visual inspection is needed to accurately and quickly evaluate all methods, which may lead to errors in evaluation due to human error. Recently, high-resolution and high-speed cameras have been used to capture images that can be mounted on cars or drones in order to survey entire roads within a short period of time. Thus, automated systems are used for image analysis and defect detection, with the aim of reducing errors and enhancing the evaluation process. Damage to roads can disrupt essential services and put people at risk [2]. Detecting defects on the asphalt surface is an essential process to ensure its safety and longevity. However, it represents a challenge due to the diversity of defects and the difficulty of identifying them [3]. Defects on the asphalt surface can be detected using machine learning methods. These methods rely on learning patterns from data, allowing them to identify defects that a human might not be able to detect. Deep learning methods have become a viable method for identifying infrastructure flaws in recent years. [4].

Large data sets of images, video, or other data can be analyzed using deep learning to accurately identify defects. [5].
Although previous studies have been successful in detecting defects for roads using deep learning, they often rely on public, open-source datasets that may not represent the types of defects, materials, or specific environmental conditions that roads encounter. Furthermore, there are few publicly accessible datasets created specifically for defect detection, and there are several obstacles linked to data, including harsh settings, equipment constraints, a lack of data, issues in labelling, and distinct defect attributes. The effectiveness of road surface flaw detection equipment may be considerably impacted by these difficulties [6].

2. LITERATURE REVIEW

Researchers have begun to use intelligence methods, exploring machine learning (ML), a subfield of artificial intelligence that bypasses explicit programming in favor of algorithms that learn from data. By training a neural network to automatically learn and extract high-level abstract features from raw data without the need for manual feature engineering, deep learning is a machine learning approach that focuses on representation learning. These learned representations enable the network to detect, classify, and predict effectively [7]. we will exhibit some of previous study, Y. J. Cha et al. (2017) introduced a method based on Convolutional Neural Networks (CNN) was used for structure crack detection. They tested their approach on images containing cracks with variations in width, lighting conditions, and noise levels. They compared the performance of their method with the Sobel and Canny edge detection methods. The findings demonstrate that the suggested approach performs significantly superior and can detect tangible fractures in practical scenarios. [8].

Similarly, Y. Z. Lin et al. (2017) proposed a CNN-based method to extract features from time-domain data for automatic damage detection. They compared the performance of their approach with a wavelet-based method. The discovery reveals that the acquired attributes progress as we delve deeper, transitioning from basic filters to grasping the notion of vibration mode. Its outstanding performance is due to its ability to grasp the essential characteristics naturally present in the data [9]. G. Huang (2017) put forth a network for classification, known as DenseNet, to enhance the propagation of features and mitigate the issue of vanishing gradients. DenseNet creates an implicit kind of deep supervision by giving every layer direct access to the gradients that come from the loss function and the original input signal. Through the dense interconnections of feature maps, DenseNet offers a feature propagation that is quite notable; the proposed study addresses the challenges in crack detection by introducing a fully convolutional neural network based on densely connected and intensely supervised networks. It tackles issues related to the distribution of semantic features in cracks and the imbalanced foreground-background ratio. By incorporating the deeply-supervised nets (DSN) method, the network supervises feature extraction using DenseNet. The focus is on pixel-level crack detection, considering the thin shape of cracks and the distribution of semantic features. [10].

R. X. Li et al, proposed (2018) a Faster R-CNN-based framework for detecting and localizing multiple defects in various scenarios. They incorporated multi-scale training, data augmentation, and negative mining strategies to enhance the system's ability to detect multiple defects. They introduced and improved a location block within the framework for defect localization. The extensive experiments demonstrate that the suggested approach achieves a detection rate of 80.7% and a localization rate of 86% within a time frame of 0.41 seconds per image (using a scale of 1,200 pixels in the real-world field test). This level of accuracy makes it suitable for seamless integration into intelligent autonomous inspection systems, offering valuable support for various practical applications. [11].

X. W. Ye et al, (2019), formulated a U-Net-derived fully convolutional network (FCN) to identify fractures on concrete surfaces autonomously. In order to assist with training and verifying the accuracy, a collection of crack images with detailed annotations at the pixel level was obtained. Net's ability to identify structural damage is superior to that of the edge detection techniques [12].

I. A. Kanaeva (2021), proposed crack segmentation in road pavement images through synthetic training data generation, that allowing for creating crack training datasets of any size for instance segmentation. State-of-the-art segmentation models like Mask R-CNN and U-Net can be trained on this synthetic data, producing acceptable results with over 47% IoU metrics on authentic images containing cracks. The suggested method is less susceptible to changes in light levels, road markers, and shadows [13].

P. Jing et al., in 2022, designed a model in order to obtain accurate results for extracting road cracks. Architecture of residual attention U-net (AR-UNet), a new network model, utilizes convolutional block of a convolutional neural network (CBAM) in the U-Net's encoder and decoder. The feature transmission path is increased by connecting the input and output CBAM features. Basic Block replaces the original network's convolutional layer to prevent network degradation. Tests on Deep Crack, Crack Forest Dataset, and our Roof Information Dataset (RID) dataset demonstrate improved crack extraction [14].

M. M. Mustakim (2023) depends on deep learning techniques to reduce vehicular accidents significantly, developed a road damage detection model and its potential implications on the economy and society, specifically the YOLOv7 model, has made significant contributions to the enhancement of detection accuracy, reduction of computational complexity, and overall effectiveness. This investigation encompassed the execution of multiple experiments to compare the performance of YOLOv5 and YOLOv7 models. The findings accentuate the exceptional performance of the YOLOv7 architecture in precisely identifying impaired regions and forecasting object classifications [15].

3. DEEP LEARNING

Deep learning is a type of machine learning that uses deep neural networks to gradually extract features, where the architecture determines the most important features [16]. The basic component of an artificial neural network, known as a neuron, has remained relatively unchanged since its inception, with the first model of a neuron called the M.P. The model was introduced by McCulloch and Bates in 1943. To illustrate work of deep learning will use the example of a neuron with three input elements (x1, x2, and x3) multiplied by the corresponding weights (w1, w2, and w3) and added together by a bias term (b) to demonstrate how the neural network functions. The activation function, represented by f(x), applies a nonlinear transformation to generate the output as shown in Figure 1 [17]. In 1958, Rosenblatt proposed a single-layer perceptron, which consists of multiple neurons capable of learning through perceptual convergence algorithms to enhance classification capabilities. Later, in 1986, Rumelhart et al. introduced the back-propagation algorithm, enabling the training of multi-class neural networks and this advance in turn allowed hidden layers to learn and generate meaningful features for classification tasks [18].

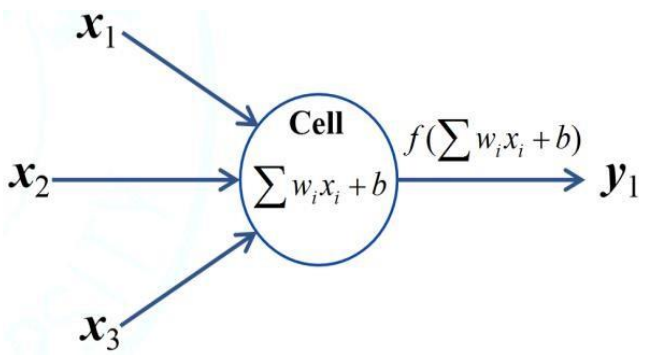

Fig. 1 The MP model

Later, LeCun et al. In 1998, LeNet-5 was introduced as a proposal to recognize handwritten characters, achieving a remarkable accuracy of over 99.65%, as visually shown in Figure 2. A recurrent neural network (RNN) is an important deep neural network (DNN) which is used to process string data [19].

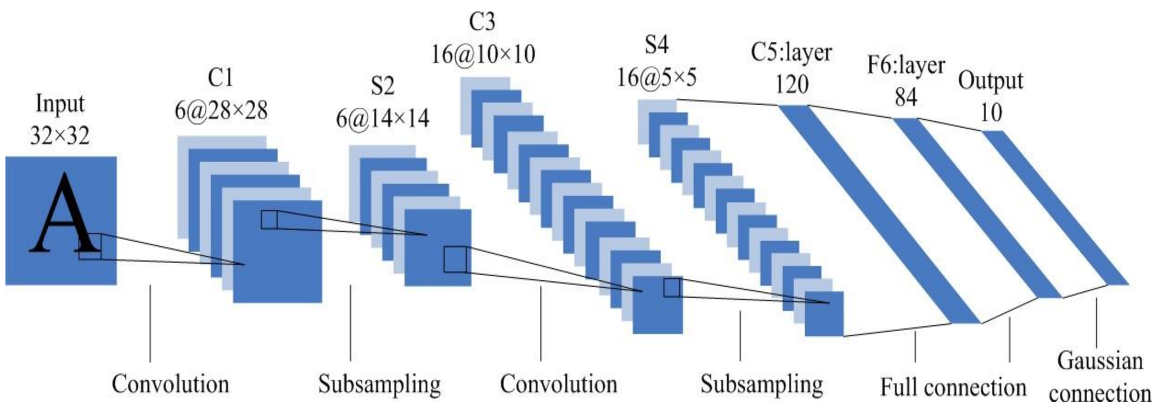

Fig. 2 The architecture of LeNet-5

The remarkable achievement of AlexNet has greatly surprised intellectuals and professionals worldwide, leading to increased interest in deep learning research, where many deep neural networks (DNNs) have been proposed for various application purposes. To summarize the stages of development of deep learning, we will illustrate the historical development series of deep learning, as shown in Figure 3. [20].

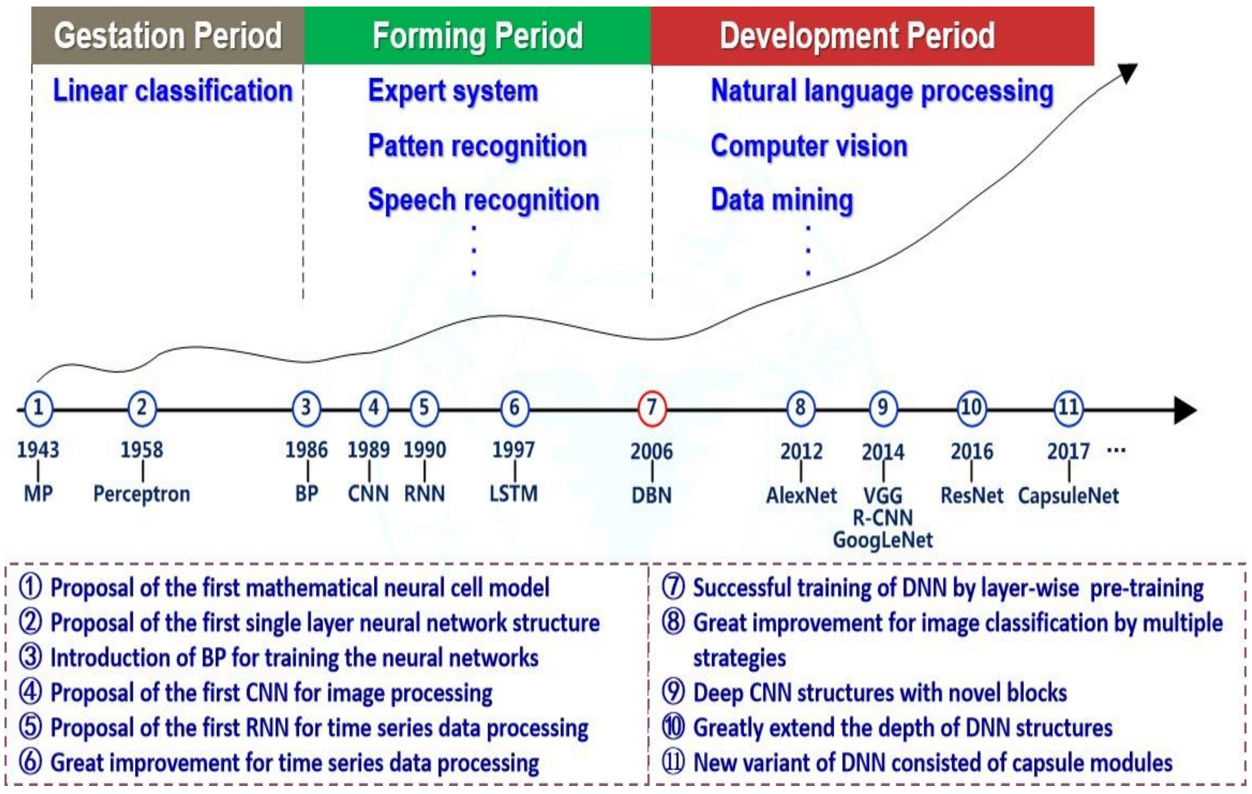
Gestation Period
Forming Period
Development Period
Linear classification
Expert system
Patten recognition
Speech recognition
Natural language processing
Computer vision
Data mining
1943 MP
1958 Perceptron
1986 BP
1989 CNN
1990 RNN
1997 LSTM
2006 DBN
2012 AlexNet
2014 VGG R-CNN GoogLeNet
2016 ResNet
2017 CapsuleNet
① Proposal of the first mathematical neural cell model
② Proposal of the first single layer neural network structure
③ Introduction of BP for training the neural networks
④ Proposal of the first CNN for image processing
⑤ Proposal of the first RNN for time series data processing
⑥ Great improvement for time series data processing
⑦ Successful training of DNN by layer-wise pre-training
⑧ Great improvement for image classification by multiple strategies
⑨ Deep CNN structures with novel blocks
⑩ Greatly extend the depth of DNN structures
⑪ New variant of DNN consisted of capsule modules

Fig. 3 Historical development of deep learning

4. ALGORITHMS FOR MONITORING ROAD

4.1 Dataset

It is worth noting that all the algorithms targeted in the study will be applied to a diverse data set containing various types of road damage.

The RDD2022 dataset contains road images from six countries (India, Japan, Czech Republic, Norway, China, and USA). However, in the presented dataset, four damage types were considered. There are many road defects in road networks. The edited and updated dataset is called N-RDD2024. 10 different types of defects were considered in this dataset. The defect classes labelled are longitudinal cracks (D00), transverse cracks (D10), alligator cracks (D20), repaired cracks (D30), potholes (D40), pedestrian crossing blurs (D50), lane line blurs (D60), manhole covers (D70), patchy road sections (D80) and rutting (D90), respectively [21].

4.2 ALGORITHMS FOR MONITORING ROAD CONDITIONS BASED ON NEURAL NETWORKS

Since neural networks are now widely used in computer vision, many issues, including road profile analysis, can be solved with convolutional neural networks and big data sets (images). *Convolutional Neural Networks* (CNN) - It is a system that accurately classifies images into categories by learning from hundreds of millions of tagged images. CNNs are judged accurate based on how well they classify images in a test dataset. One of the first investigations into the use of neural networks for pavement defect classification was the 1994 study [22]. The National Cooperative Highway Research Program (NCHRP) provided the photos used in this investigation [23]. Although the technique employed to get these pictures is not known to the general public, it is likely that a strategy akin to the ones previously mentioned was employed.

The scientists classified the pavement cracks using a moment-based technique after obtaining a binary image of the cracks using fuzzy thresholding [24]. Even though the authors only employed a tiny quantity of data for training and testing, their results were remarkably accurate when compared to their own test data set. Because the test set included augmented images from the training set, there is a chance of overfitting, which is a source of error in neural networks where the network becomes extremely good at analyzing images on which it was trained but fails to propagate this performance on images she had never seen. This work was carried out in [24], which is a fuzzy logic and neural network based system. However, this approach is not fully applicable to all crack detection scenarios due to the relatively small dataset (hundreds of images compared to thousands needed to train a truly accurate convolutional neural network). The accuracy of trained neural networks has dramatically grown in recent years because to the considerably increased availability of data storage and the processing capacity of graphics processing units (GPUs). The field of pavement condition analysis reflects this as well. For instance, in [25], a network was trained to identify flaws in road surfaces using a method called transfer learning.

A deep learning concept known as "transfer learning" allows a network that has been trained for one task to be reconfigured to do another activity. The novelty of the algorithm lies in the creation of training labels by applying the algorithm, trained on a common set of images to classify airborne images. In addition, a new texture descriptor is proposed based on the fusion of learned colour planes to obtain maximum uniformity across road sections.

Another study on CNN structures for road surface detection is [26], which detects road surfaces with 87% accuracy on the KITTI dataset by using Convolutional Patch Networks.

Pavement flaws were detected on a short dataset (78 photos) by S. Milhomem et al. (2018) using a weightless neural network based on pattern recognition in conjunction with transfer learning [27]. The authors report accuracy rates (86% accuracy on their dataset); however, the relatively small size of the data set does not give us the right to consider this system truly proven and suitable for widespread use. The problem of testing on small data sets also does not allow us to consider the system [28] ready for use on a large scale. In it, an accuracy of ~85% was obtained for a data set used to test a method for detecting defects in road surfaces. Study [29], which has a fairly large testing dataset (450 data sets, more than 40,000 images), in which detection accuracy of ~93% was obtained using a semi-supervised learning approach to neural network training. Since CNN designs on embedded computers are so simple to set up, it is possible to modify these implementations for use on inspection trucks without the need for big, cumbersome monitoring systems.

4.3 Algorithms based on threshold segmentation

Thresholding is a variation of the old and simple similarity-based image segmentation algorithms. Most defect detection studies use binarization to detect defects. To date, thousands of algorithms for determining threshold values have been published [30]. The task of binarization (thresholding) is to significantly reduce the information content Images. This method is the simplest and most common in machine vision for detecting road surface defects. Binarization is based on the choice of threshold, which allows you to build the most effective method of working with the image.

Existing binarization methods [31-32] which can be divided into two groups:

A. *global (threshold).*
B. *local (adaptive).*

The global threshold is characterized by the establishment of a certain value, which acts as a general threshold T applied to the entire image. Any point (x, y) at which the condition $f(x, y)>T$ is satisfied is an object point; if the condition is not satisfied, it is a background point. Checking whether a pixel belongs to someone range as in eq (1).

$$T = T(x, y, p(x, y), f) \tag{1}$$

Where f is the image, p(x,y) is the local data of the point, (x,y) is the image.

An example of the application of the binarization method as eq (2) point on the image g(x,y), obtained as a result of processing with a certain threshold:

$$g(x, y) = \begin{cases} 1, if\ (x, y) > T \\ 0, if\ f(x, y) \le T \end{cases} \tag{2}$$

Where $f(x, y)$ is a certain function that processes the image, T is a certain threshold.

If the threshold T is dynamic (local), then it depends on the x and y coordinates.

If the threshold T depends on the local characteristics of the point p(x, y), then it is adaptive.

In the task of detecting cracks in asphalt pavement, it is important value is determined by an object with a small area compared to the background; for this task, the optimal threshold is usually chosen as the half-sum of the minimum and maximum brightness values. Below is Figure. 4 applying a threshold to detect cracks in the image.

The main feature of using a local threshold is that, compared to the background and noise, the crack becomes darker and more continuous. The original road surface images are first divided into non-overlapping blocks. A local optimal threshold is then implemented in each block image. Below is Figure. 4 binarization of a crack simple example.

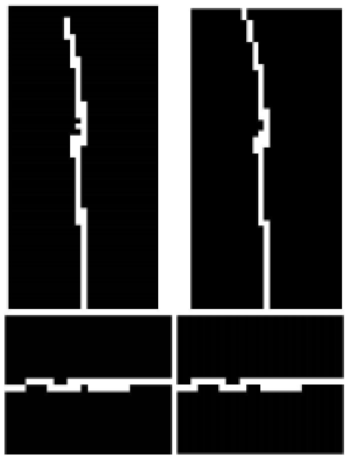

Fig .4. Example of defect binarization

The image is broken into smaller portions using local thresholding, and the threshold for each image fragment is determined by the location or local characteristics of the point. However, in computer vision applications in particular, global threshold setting is better than local, however this approach has limitations due to noise, object and background variations, average difference, contrast, and other factors.

The most common binarization method is the Otsu method [33], which is used as the basis for all pavement defect segmentation algorithms. There are also other local threshold algorithms. The objective of the suggested approach [34] is to determine a locally ideal threshold value by utilizing the crack density distribution. Compared to the background and noise, the crack becomes darker and more continuous. This is necessary to eliminate problems associated with image processing associated with noise, illumination, and reflectivity of the road surface. Original images Blocks that do not overlap the road surface are first created. After that, it gives each block picture a local optimal threshold. Its result is better than the global threshold. Compared to the background and noise, the crack becomes darker and more continuous. Noises appear more independently. An example is shown in Figure 5.

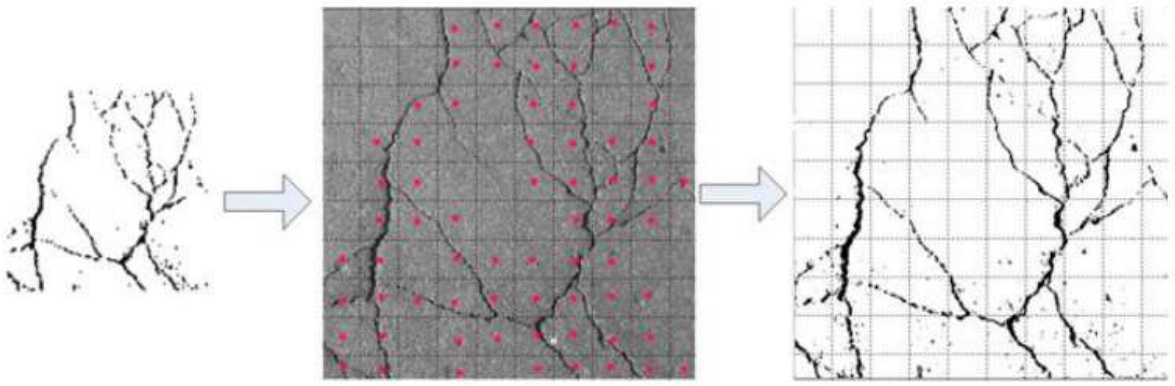

Fig. 5. Image example of processing local thresholds

4.2 Algorithms Based on Morphological Methods

These methods are based on the use of threshold binarization algorithms for image processing [35-36] as shown in Figure 6.

The mathematical morphology method is a fairly old and frequently used method for preprocessing and detecting road surface defects. Besides binarization algorithms based on morphology are used in defect detection because of their sophistication and versatility. Morphological methods are used to improve the quality of image binarization and segmentation. Image enhancement is the process of making an image's background noise smaller. Road surface photos in particular have a lot of noise because of their uneven illumination and rough surface. The morphological operator operates on two images: a structuring element that functions similarly to the convolution operation's core and source data for analysis. Dilation and erosion are the fundamental mathematical morphological operators; additional morphological operations are a combination of these two fundamental operations. This type of algorithm is based on the assumption that defects have certain properties, from which it follows:

- Property 1. Cracks are dark objects on a lighter background. Cracks are darker than the object than the background such that the pixels belonging to the cracks are local 1 minimum.
- Property 2: A crack is a saddle point. A crack is a thin groove such that the cross-section of the crack is saddle-shaped.
- Property 3: The crack is linear in nature or direction. A crack is a coherent subset of elements in an image such that there is a preferred direction for the development of the defect.

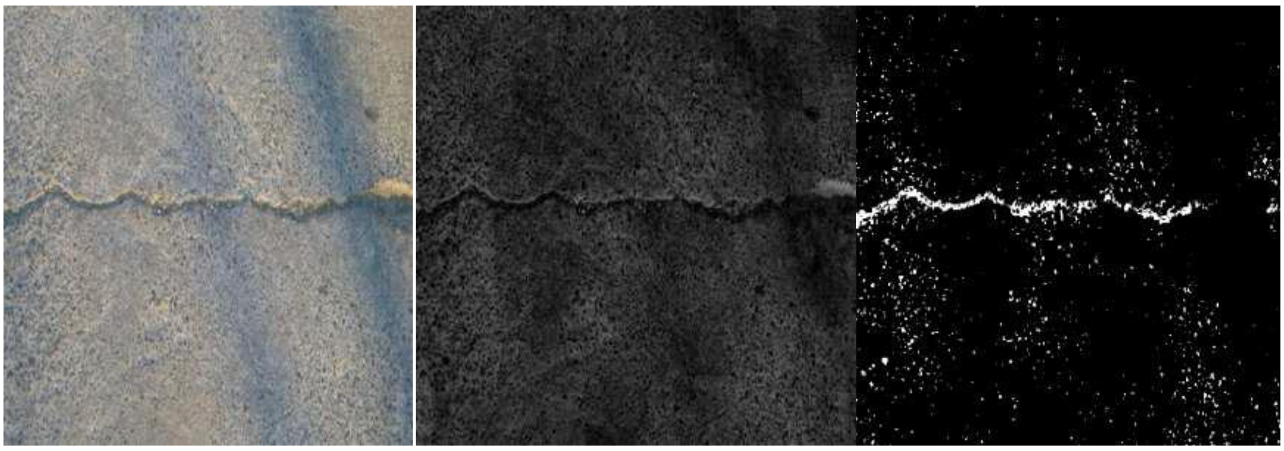
Fig. 7. Example of processing with morphological operations

Some works using mathematical morphology for defect segmentation [36] use two Top-Hat operations; Black-Hat operations extract small details from the processed images.

4.4 Neural network algorithms for stereo image processing

Convolutional neural networks (CNNs) are a subclass of neural networks created to solve computer vision problems, and have de facto become dominant in solving such problems. The basis of convolutional networks are convolution layers, directly carrying out the convolution procedure.

The purpose of the convolution algorithm is to extract various features from the picture in order to simplify further analysis of the depicted objects.

This is done using so-called convolution kernels - small (usually no more than 9 by 9 elements) two-dimensional arrays, the values of which are element-by-element multiplied by the pixel values of the image. Thus, each convolution kernel extracts a specific feature map from the original image or another feature map, achieving better extraction of image features of the initial image as shown in Figure (8).

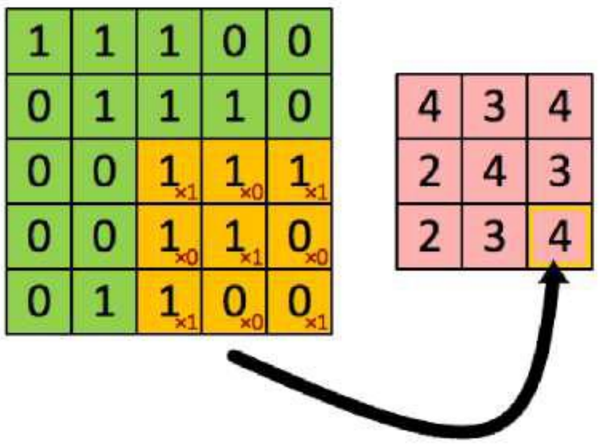

Fig. 8. Convolution procedure

In stereo algorithms, convolutional neural networks typically perform the task of pixel matching cost regression. The architecture of the MC-CNN-fast (Matching cost convolutional neural network) neural network [37] as shown in figure 9, is based on a similar mechanism, where convolutional layers are used as input. Using a machine learning model in this case with proper network training improves the quality of the resulting disparity map.

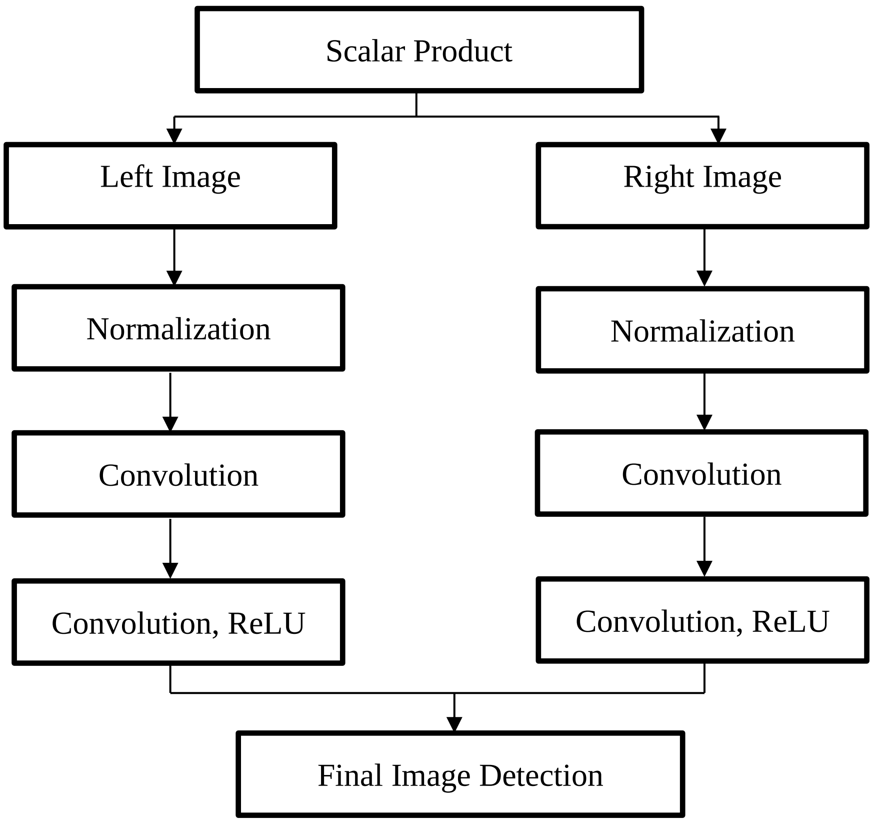

Fig .9. Neural network architecture MC-CNN-fast

The advantage of this architecture is the complete absence of fully connected layers (the architecture is a completely convolutional network), which significantly speeds up calculations. The activation function becomes important in deep and convolutional neural networks. The activation functions themselves have biological similarities, something similar happens in biological neural networks and is called a spike. In addition to its similarity to biological neurons, the activation function helps keep the output value within a certain limit. Modern neural networks use a rectification activation function. The rectification activation function makes it easier to train neural networks where the model behaviour is close to linear. The use of the ReLU function is important in hidden and convolutional layers as in eq (2).

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

Where x are combinations of the input vector and weight matrix

ReLU is differentiable at all points except x=0.

Linear rectification makes it easier to optimize functions due to its similarity to linear functions. An important difference between the linear and rectification function is that zero is given in half of the part areas. The ReLU function creates large and constant gradients.

The second derivative of the operation is equal to 0 almost everywhere, and the first derivative is equal to 1 everywhere where the neuron is active. One of the disadvantages of the ReLU activation function is that they are unable to learn using gradient methods from examples for which their activation is zero.

5. RESEARCH OF NEURAL NETWORK DETECTION ALGORITHM

When using neural network algorithms to detect road defects, it is necessary to have models to train. Well-known machine learning problems usually have public datasets available for research and training. We have prepared some examples of the process to illustrate the working and structure of the algorithms.

5.1 Study of the SegStereo Algorithm

A fundamental problem in computer vision, which is especially acute when using stereo images, is disparity estimation. it can play an important role in forecasting, scene understanding, automatic vehicle control and many other fields. Finding the comparable pixels from a stereo image and using the offset between matching pixels to calculate the distance to an object is the primary goal of disparity estimation. By utilizing the advantages of semantic segmentation and introducing a SoftMax loss function, SegStereo [38] enhances the precision of disparity map predictions. This model has shown good results in both supervised and unsupervised learning. The architecture of this network is shown in Figure 10.

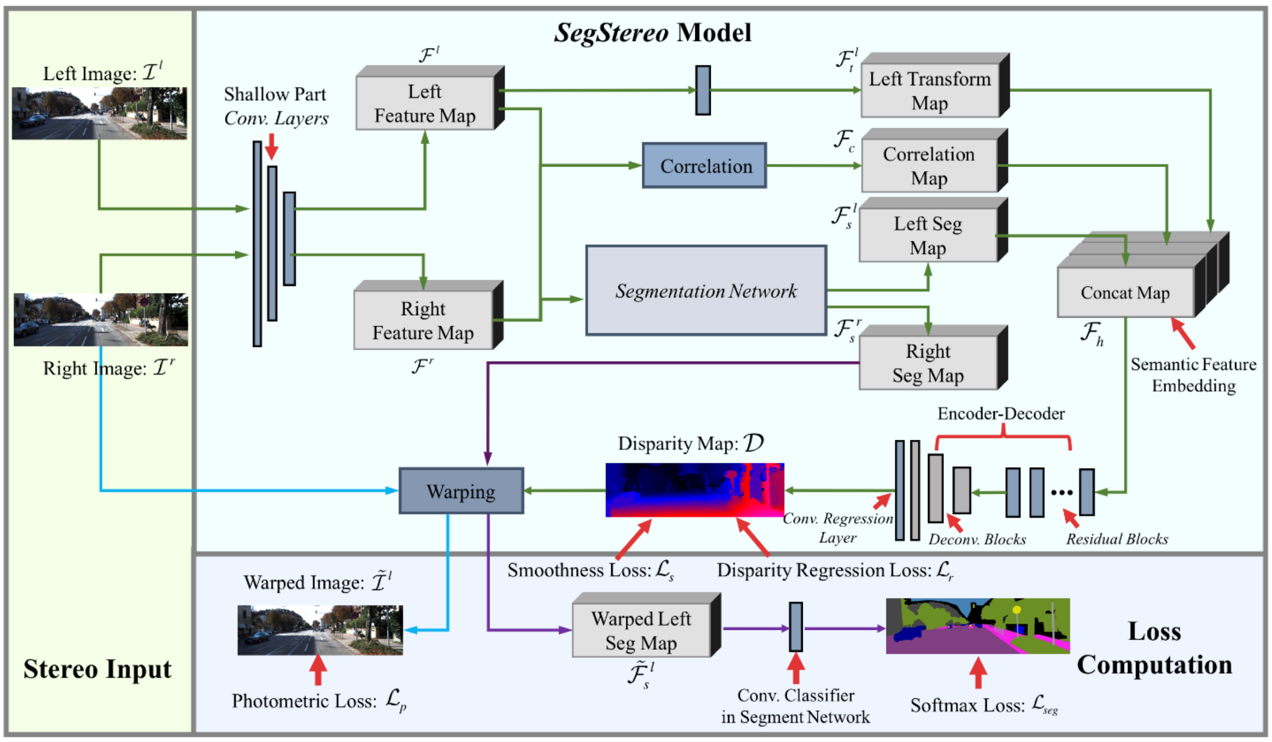

Fig. 10. SegStereo Architecture

Information Processing Process

1. The intermediate characteristics $f_{\square}^{1}$ and $f_{r}^{\square}$ are extracted from the stereo input;
2. The volume of costs $f_{c}^{\square}$ is calculated using the correlation operator;
3. The left segmentation feature map f_{s}^{1} is aggregated into a branch inconsistency as embedding of semantic features.
4. The f_{s}^{r} feature map of the right segmentation is warped into a left view for pixel-by-pixel semantic prediction with softmax loss regularization.

It was chosen to train the model across a various number of epochs and compare the results to find the ideal training time in order to prevent undertraining of the model. Using TensorFlow, the training results were compared at 20, 40 and 60 epochs, which presented in Figure 11. In the graph, orange indicates training at 20 epochs, blue at 40, and red at 60.

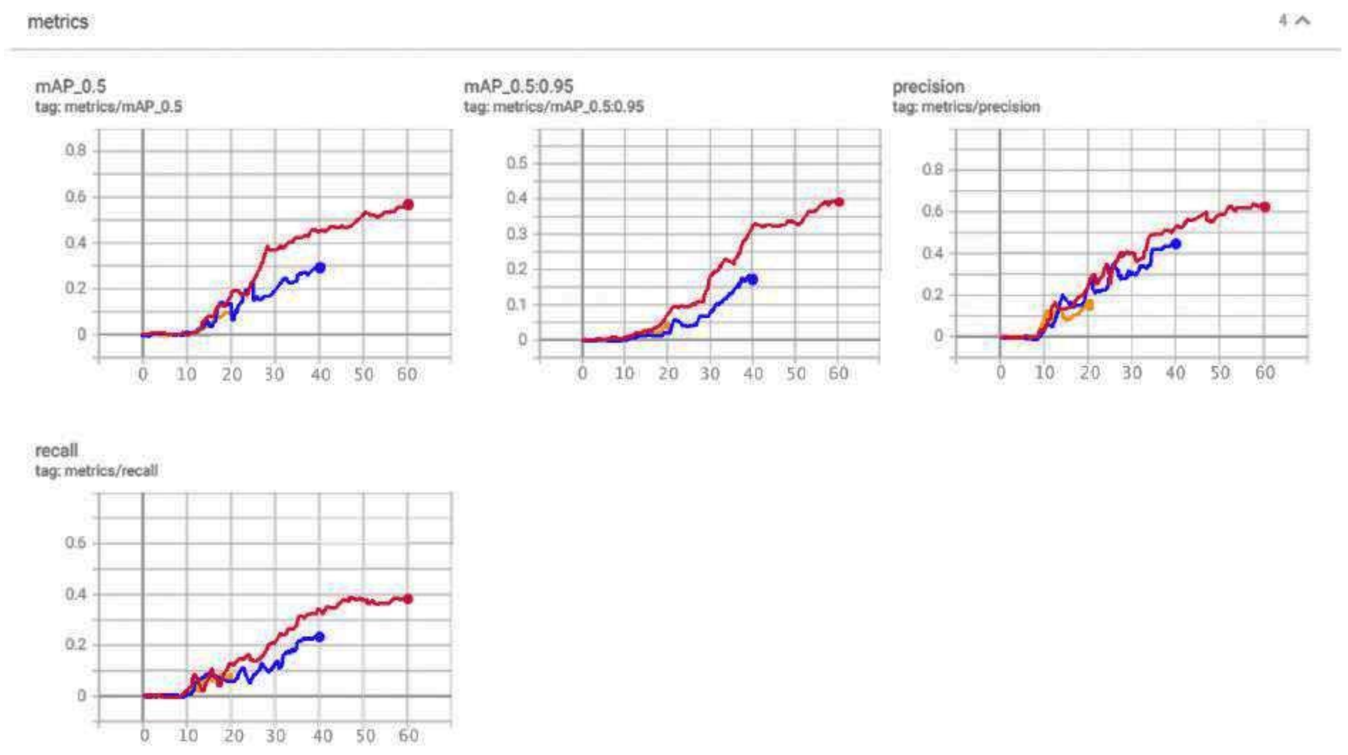

Fig 11. Comparison of training at 20, 40 and 60 epochs

Based on the graphs comparing the training results for 20, 40 and 60 epochs, we can conclude that the best results are achieved when training for 60 epochs. This is because this model detects more objects with greater accuracy and also has less error compared to other models.

5.2 Study of the GC-Net Algorithm

GC-Net [39] (Global Context Network) provides a simple, fast and efficient approach to global context modelling. This network is obtained by combining a local network (NLNet) and compression excitation networks (SENet). As a result of their combination, a three-stage global context modelling framework. The global context network contains the following blocks:

- Simplified non-local block: A condensed form of the non-local block divides the map attention for each request item and computes a global (query-independent) attention map. This change is made after observing similar attention maps generated at different query positions. A comparison of nonlocal and simplified nonlocal blocks is presented in Figure 12.

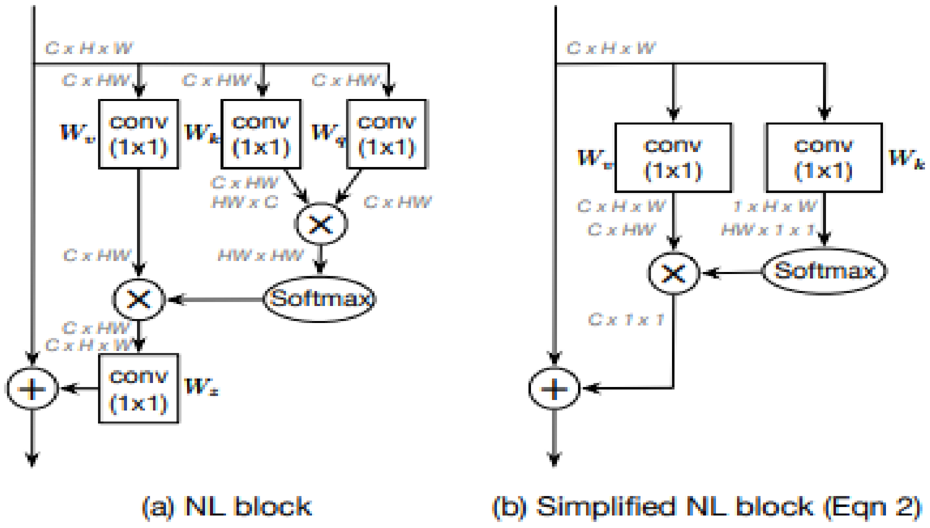

Fig. 12. Comparison of non-local and simplified non-local blocks

- The global context modelling framework is the basic building block used in the global context network, it can be divided into three procedures: The attention weights are first obtained via global attention pooling, which requires 1x1 convolution and the SoftMax function. The global context features are then obtained by applying attention pooling. After applying attention pooling, Over the weekend, 1x1 convolution is used to modify the features, and they are then combined to add global contextual characteristics to each position's features. Figure 13 displays this block's diagram.

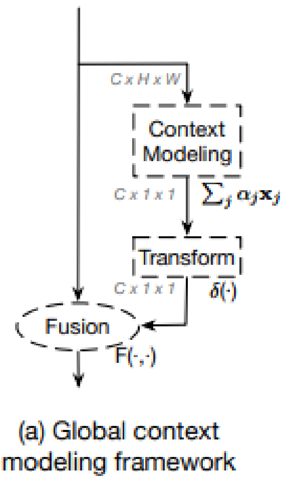

Fig. 13. Global Context Modelling Block

- A streamlined non-local block and a compression-excitation block are used by the global context block. To lower the number of parameters, the model underwent a few modifications. Modelling global context is made easy with the help of the lightweight Global Context (GC) block. Generally, it outperforms both lightweight NLNet and SENet. Its lightweight architecture enables it to be implemented at numerous layers of a feature extraction network to produce a global context network (GCNet). basic reference values for various recognition tasks. This block is presented in Figure 14.

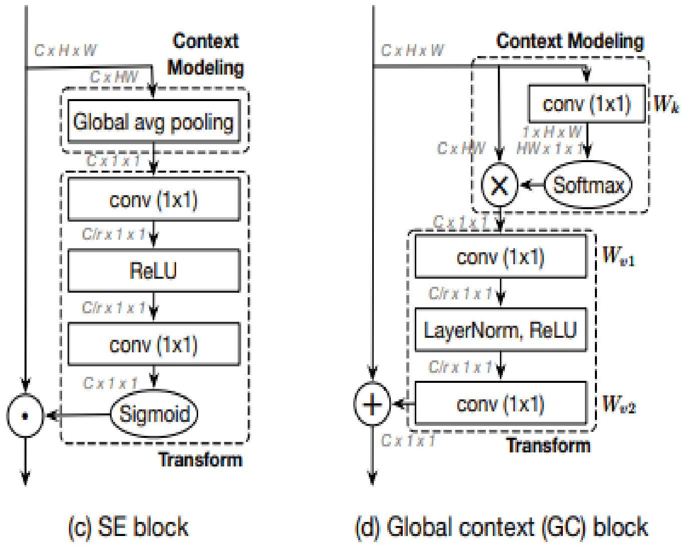

Fig. 14. Global context block

In order to keep the model from becoming undertrained, it was found that it was best to train it over a variety of epochs and compare the results to ascertain the appropriate training time. TensorFlow was used to compare training at 25, 50, and 75 epochs, and the results were visualized in Figure 15. The graph with the orange line represents training at 25 epochs, the blue graph at 50 epochs, and the red graph at 75 epochs.

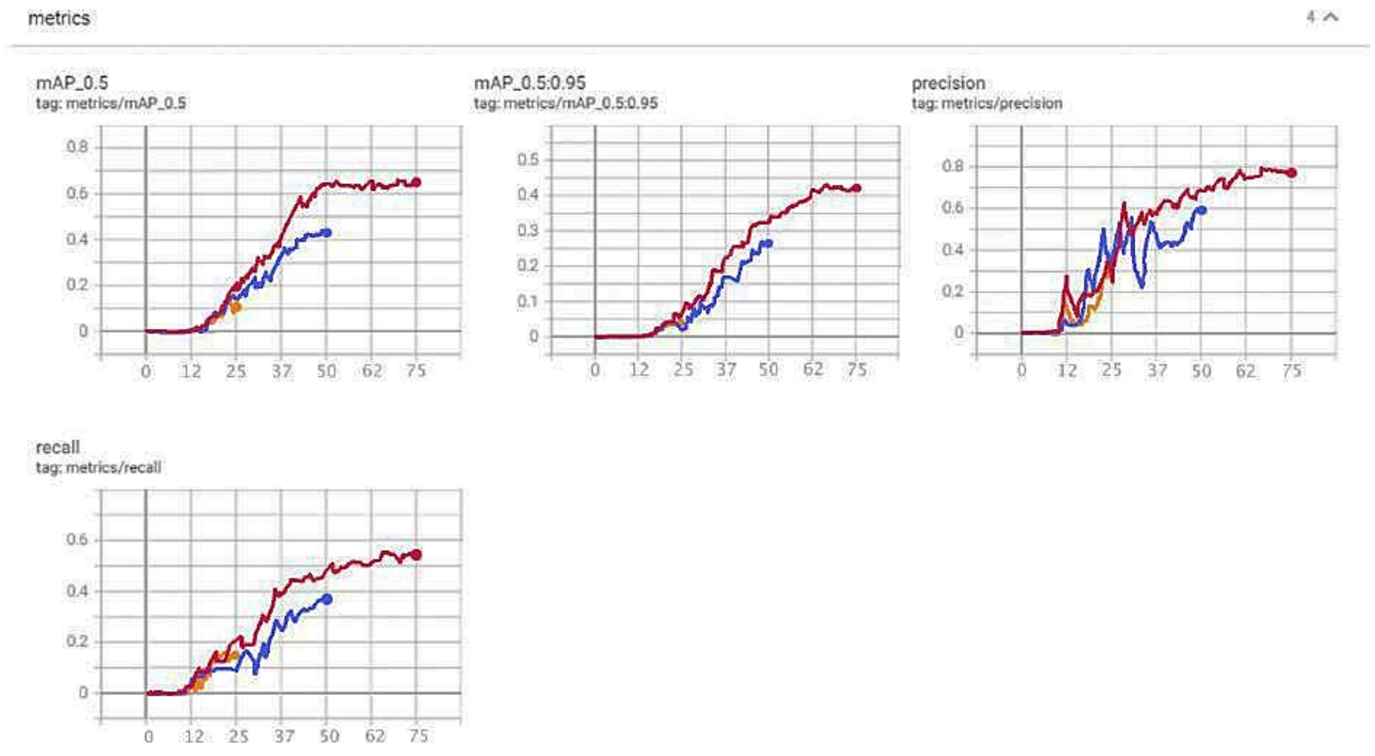

Fig. 15. Comparison of training at 25, 50 and 75 epochs

The graphs comparing training at 25, 50, and 75 epochs show that the best outcomes were seen at 75 epochs. This is due to the fact that this model's degree and accuracy of detection turned out to be much higher and its error was far lower than those of the other models.

5.3 Study of the SGM-Nets Algorithm

SGM [40] (Semi-Global Matching) is a deep neural network for predicting an accurate difference map with semi-global consistency. SGM has become a popular regularization method for real-world scenes due to its high accuracy and speed of calculations. SGM is capable of producing accurate results using penalty parameters that adjust smoothness and choppiness discrepancy maps. Despite the accuracy of the results, this approach is quite difficult, so it was proposed to use empirical methods. SGM-Nets is a learning-based penalty estimation method using convolutional neural networks. A new loss function is introduced to train networks. A new SGM parameterization is also used, which applies different penalties depending on whether changes in the legibility of the representation of object structures are positive or negative as shown in figure 16.

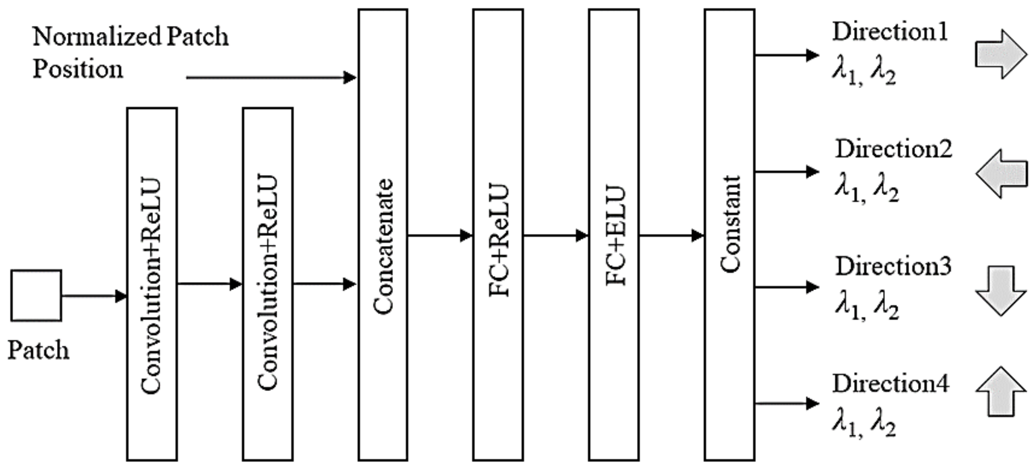
Normalized Patch Position
Patch
Convolution+ReLU
Convolution+ReLU
Concatenate
FC+ReLU
FC+ELU
Constant
Direction1 λ_1, λ_2
Direction2 λ_1, λ_2
Direction3 λ_1, λ_2
Direction4 λ_1, λ_2

Fig 16. Architecture of SGM-Nets

A layer of linear units (ReLU) after each convolution layer is combined into a layer to combine two kinds of information, two fully connected (FC) layers of 128x128 pixels each and a ReLU after the first FC layer. in addition, the exponential linear unit (RELU) is used with $\alpha = 1$. To keep SGM penalties positive, ReLU has zero gradients for negative input values, however ELU eliminates the vanishing gradient problem. This means that ELU speeds up the training of the neural network and results in increased accuracy.

The preprocessing step subtracts the average value from the image region and divides it by the maximum intensity of the image. By dividing the reward's position by the image's height or breadth, it is normalized. To avoid the model being undertrained, it was decided to train the model over a variety of epochs and compare the outcomes to determine the best training period. Using TensorFlow, a comparison was made of training at 30, 60 and 90 epochs, the results of which are presented in Figure 17. The orange graph corresponds to 30 epochs, the blue graph corresponds to 60 epochs, and the red graph corresponds to 90 epochs.

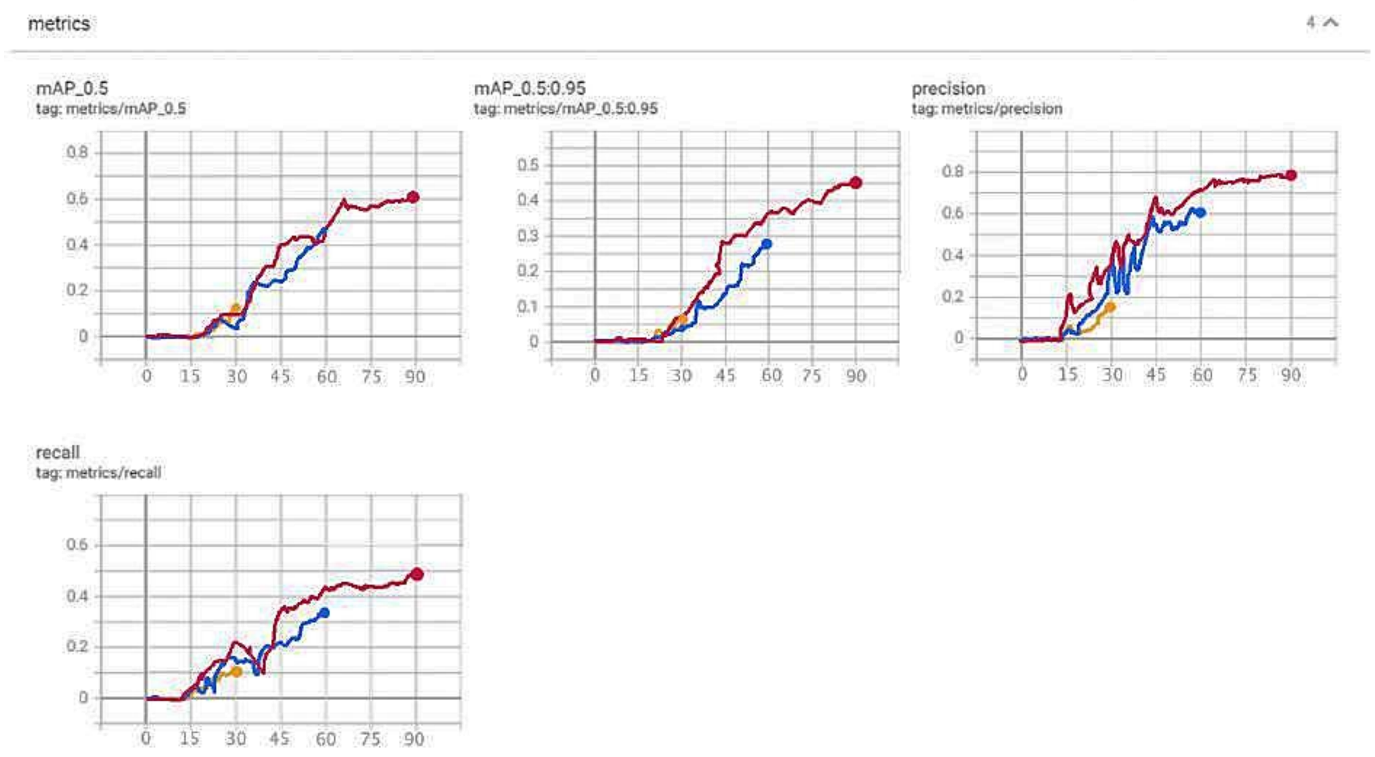

Fig. 17. Comparison of training at 30, 60 and 90 epochs

The graphs comparing training at 30, 60, and 90 epochs show that the best outcomes were seen at 90 epochs. This model's degree of correctness of detection proved to be substantially greater than the others, and its error was significantly reduced.

5.4 Study of the MC-CNN algorithm

A multi-channel network is a type of convolutional neural network. [37] The difference is that instead of a single input channel, a multichannel network has multiple input channels. There is one core for each input channel, each of which is unique. To prioritize particular input channels over others, some cores may have a higher weight than others. After that, the channel's processed versions are added together to create a single channel. One version of each channel is produced by each filter core, while one common output channel is produced by the filter as a whole. [41] Figure 18 shows the architecture of a multi-channel convolutional network.

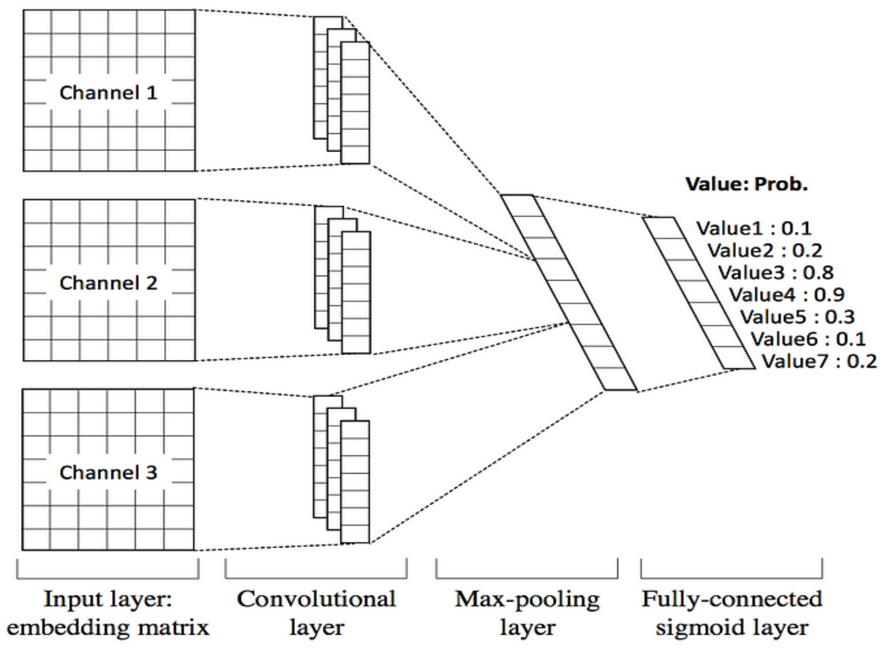

Fig .18. Multi-channel convolutional network architecture

A typical stereo algorithm starts by calculating the corresponding cost at each position *p* for all differences *d* under consideration. One method for calculating the cost of comparison is the sum of absolute differences as in eq (4).

$$C_{SAD}(p,d) = \sum_{q\epsilon N_p} |I^L(q) - I^R(q-d)| \quad (4)$$

Where $I^L(p)$ and $I^R(p)$ – image intensity at position p on the left and right images.

N_p- A set of locations in a fixed rectangular window centered at *p*,

p and *q* – designation of the location of images, [37].

d – mismatch, d reduced to a vector, that is, d = (d, 0).

To find the best training duration and avoid the model being over- or under-trained, it was decided to train the model over a variety of epochs and compare the results.

TensorFlow was used to visualize the training data across 11, 22, 33, and 44 epochs, and the results are shown in Figure 19. There are 11 epochs in the orange graph, 22 in the blue graph, 33 in the red graph, and 44 in the purple graph.

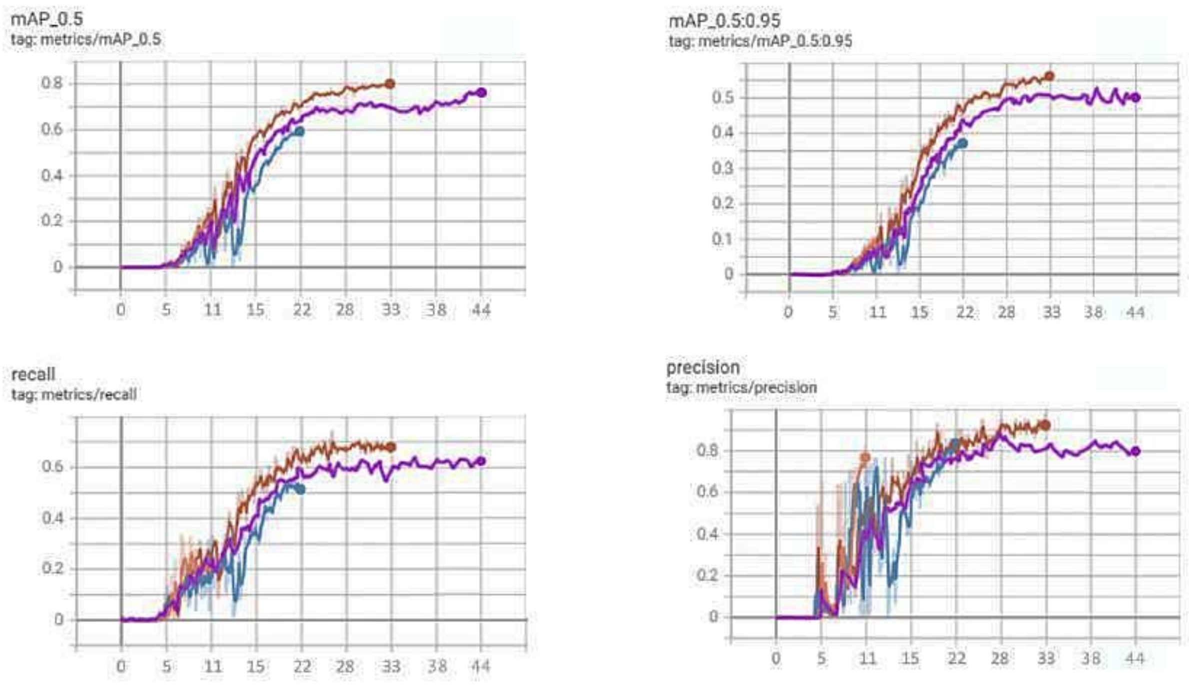

Fig. 19. Comparison of training in epochs 11, 22, 33 and 44

The best results were displayed at epoch 33, as indicated by the training comparison graphs at epochs 11, 22, 33, and 44. This is because, in comparison to the previous epochs, the degree and accuracy of detection using this model turned out to be substantially greater, and the error was significantly lower.

Based on the study of neural network detection algorithms, namely: SegStereo, GC-Net (Geometry and Context Network), SGM-Nets (Semi-Global Matching Networks) and MC-CNN (Multi-channel Convolutional Neural Network), a comparison of their performance can be made (Table 2.1). The comparison took place based on three metrics - Inference Time (processing time), Inference Frames (frame rate), IoU (Intersection over Union - ratio of bounding frame areas).

Table 1. Comparison of neural network detection algorithms

Algorithm	Inference Time, MC	Inference Frames	IoU
SegStereo	62,31	37,19	0,799
GC-Net	53,19	41,54	0,887
SGM-Nets	46,73	47,25	0,776
MC-CNN	**39,81**	**56,29**	**0,96**

According to the results of the study, the MC-CNN detection algorithm showed the best accuracy - no less than 96%. The result of the algorithm is presented in Fig. 20.

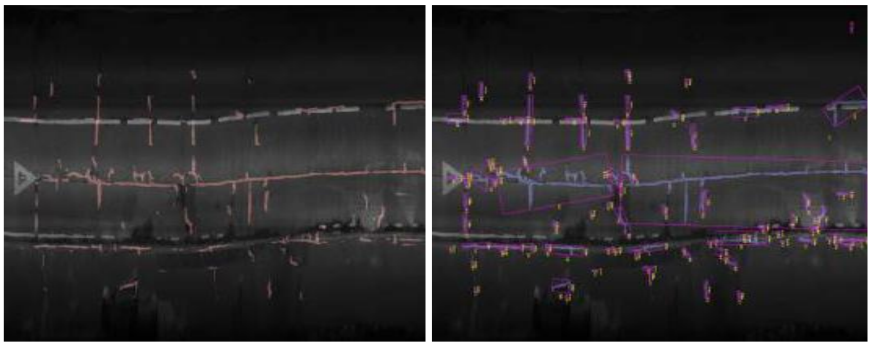

Fig 20. Result of crack detection

6. STUDY OF NEURAL NETWORK ALGORITHMS FOR CLASSIFYING ROAD DEFECTS

6.1 Study of the YOLO algorithm

YOLO (You only look once) [42] is a deep machine learning algorithm that has gained wide popularity in solving problems of finding objects in an image. The YOLO algorithm demonstrates high performance compared to other algorithms, as well as higher frame rates when used in real time. Regression is the foundation of the YOLO algorithm. In a single algorithm run, it predicts classes and bounding boxes for the full image rather than just the important portions. Regression is the foundation of the YOLO algorithm. In a single algorithm run, the full image's classes and bounding boxes are predicted, as opposed to just the interesting portions. Predicting the object's class and the bounding box that denotes its location is our goal while utilizing YOLO. Four descriptors can be used to characterize each bounding box:

- Field Center (bx, by);
- Width (bw);
- Height (bh);
- The value of c corresponding to the class of the object.

Eventually, the probability that the object is inside the bounding box is predicted as a real number, or pc. Instead of searching for potential object-containing regions of interest in the input image, YOLO divides the image into cells, usually forming a 19x19 grid. Next, it is up to each cell to make a forecast. Only if the coordinates of the anchor field's center are found in a certain cell is it considered that an object is there. Because of this characteristic, the height and width of an image are determined in relation to its total size, while the center coordinates are always calculated in relation to the cell. In one forward propagation step, YOLO calculates the likelihood that a given class is present in a given cell. The equation for calculating the probability of an object existing in a cell as in eq (5):

$$score_{c,i} = P_c * c_i \tag{5}$$

The class allocated to that particular grid cell is the one with the highest probability. For every grid cell in the image, the procedure is the same. Non-Maximum Suppression is utilized in order to address the issue of superfluous binding fields [42]. As seen in Figure 21, it initially forms an IoU (intersection over union) using the bounding box that has the highest-class probability among them, eliminating very near bounding boxes.

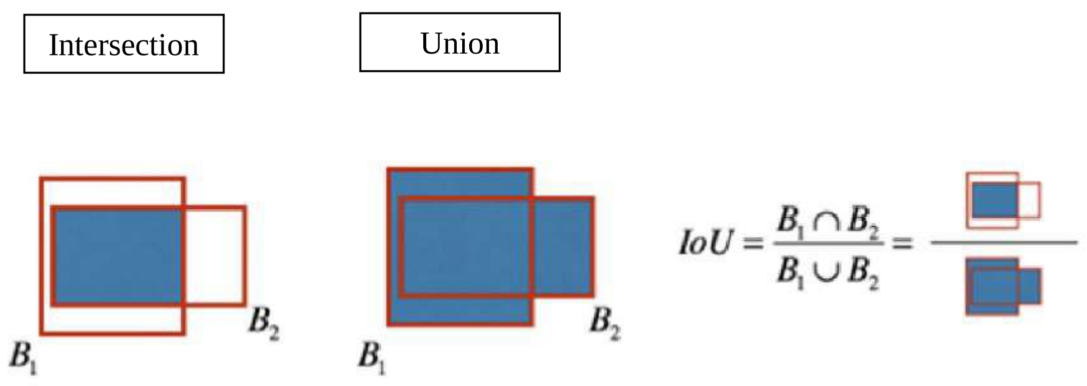

Fig. 21. Intersection divided by Union

At the end, the algorithm outputs the required vector showing the bounding box details of the corresponding class. The general architecture of the algorithm is shown in Figure 22.

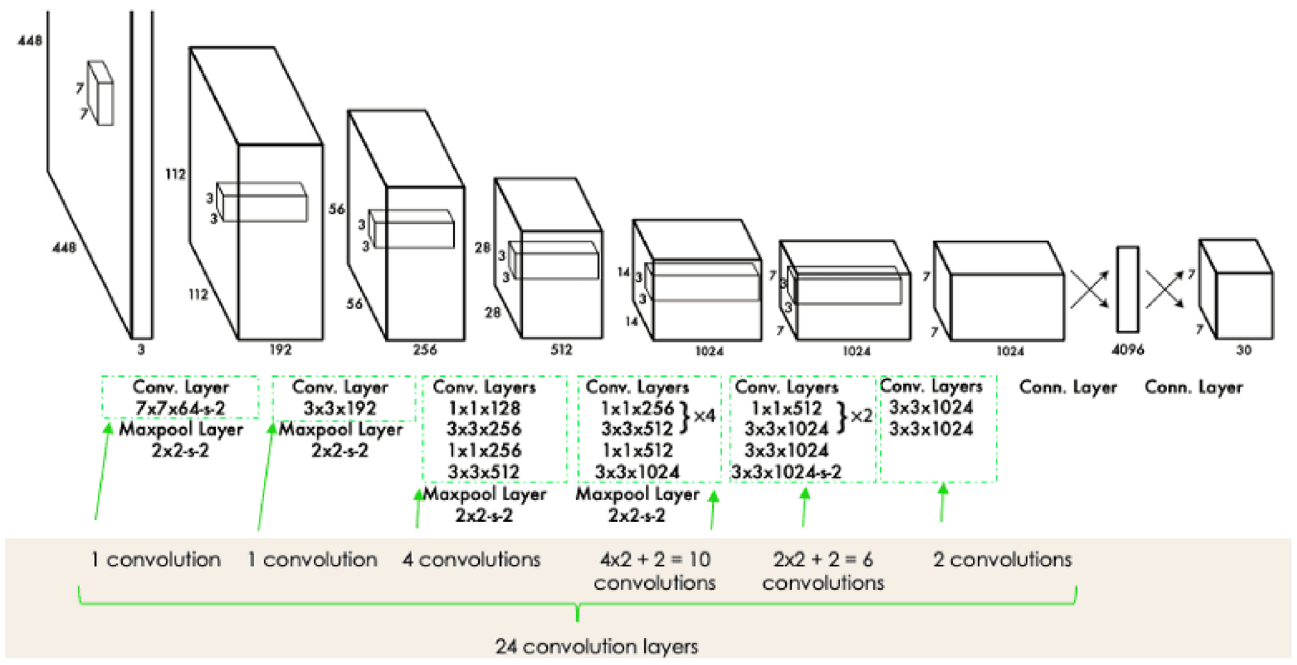

Fig. 22 YOLO Algorithm Architecture

It was chosen to train the model across a various number of epochs and compare the results to determine the ideal training time in order to prevent undertraining or overtraining of the model. Using TensorFlow, we carried out training on 25, 50 and 75 epochs and the results are visualized in Figure 23, where the red graph corresponds to 25 epochs, blue to 50 epochs, and purple to 75 epochs.

ATU-FJIECE, Volume: 3, Issue: 2, Jun 4, 2024, © 2020 FJIECE, All Rights Reserved

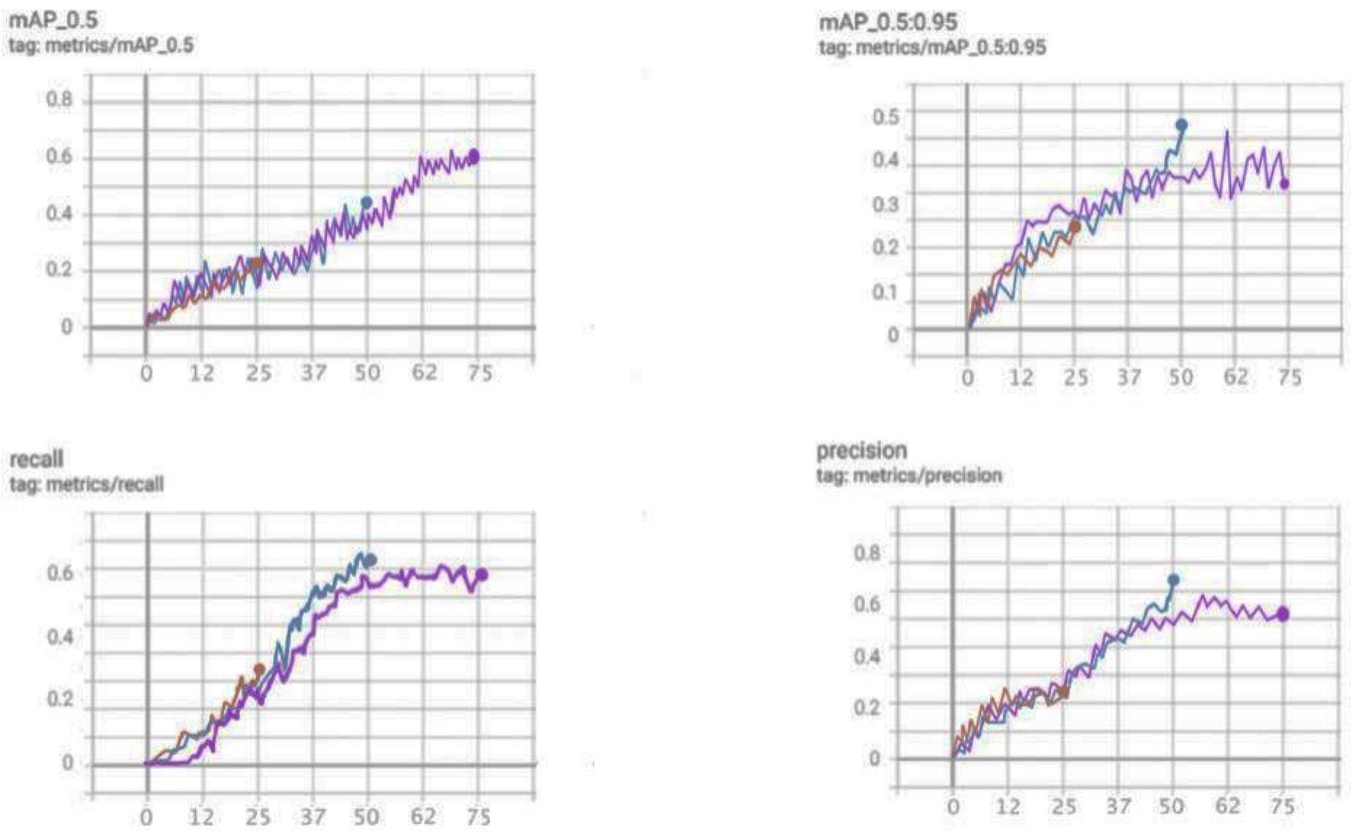

Fig. 23. Comparison of training at 25, 50 and 75 epochs

The best results were displayed at 75 epochs, as indicated by the graphs comparing training at 25, 50, and 75 epochs. This is because the degree and correctness of detection with this model proved to be substantially greater, and the error was significantly smaller than with the other models.

6.2 Study of the Mask R-CNN algorithm

Mask R-CNN [44, 45] is a convolutional neural network (CNN) that achieves high performance in image segmentation tasks. This variant of a deep neural network detects objects on image and generates a high-quality segmentation mask for each instance.

Mask R-CNN was built using Faster R-CNN [46]. While Faster R-CNN has 2 outputs for each candidate object, class label and bounding box offset, Mask R-CNN is about adding a third branch that outputs the object mask. Additional mask output differs from class and block inference, requiring the extraction of a much finer spatial layout of the object.

Mask R-CNN is an extension of Faster R-CNN and works by adding a branch to predict an object mask (region of interest). The key element of Mask R-CNN is the alignment between pixels, which is the main missing element of Fast/Faster RCNN.

Mask R-CNN uses the same two-stage procedure with an identical first stage (which is RPN). In the second stage, in parallel with class and field bias prediction, Mask R-CNN also outputs the binary mask for each area of interest. This is different from most recent systems, where classification depends on mask predictions.

In addition, Mask R-CNN is easy to implement and train, thanks to the Faster R-CNN framework, which facilitates a wide range of flexible architectural designs. Additionally, the mask branch only adds a small amount of computational load, ensuring a fast system and fast experiments.

Faster R-CNN, works in two stages:

Stage 1: Region Proposal Network (RPN) - proposal of regions. This step generates a set of rectangular regions that can contain objects. RPN uses a convolutional neural network to quickly find rectangular regions that may contain objects. This is done by scanning an image and generating rectangular regions that can contain objects.

Stage 2: Classification and Localization - classification and localization. This stage uses a convolutional neural network to determine the class of objects in each region, as well as to determine the coordinates of the boundaries of objects in each area.

This helps determine exactly where an object is in an image and what kind of object it is shown in Figure 24.

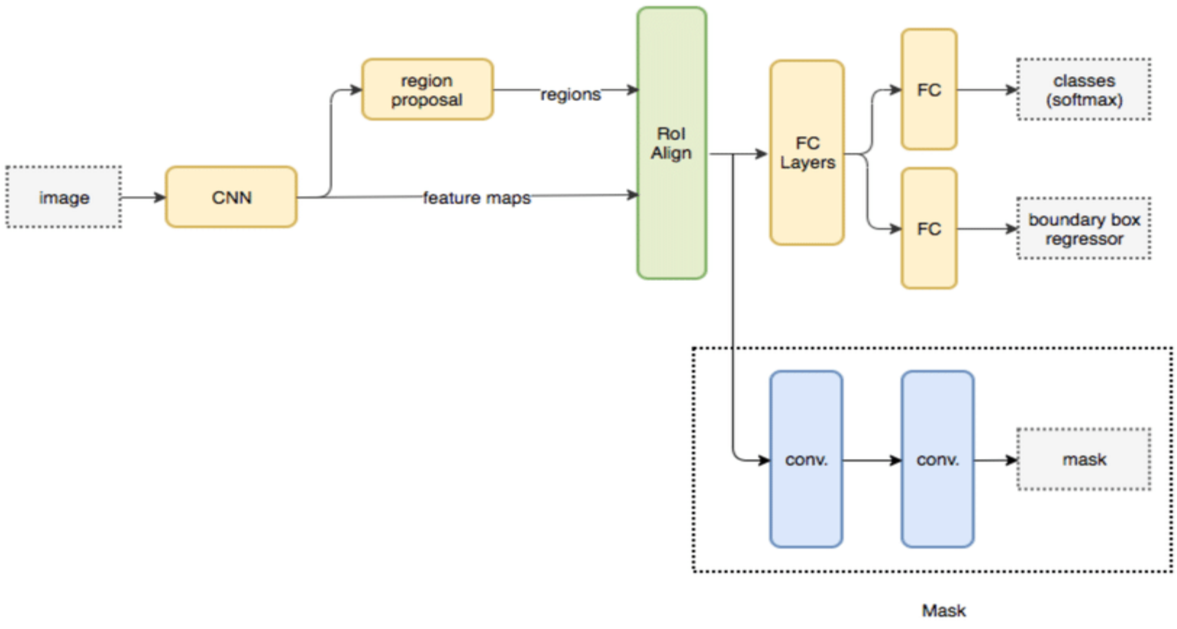

Fig. 24. Architecture of Mask R-CNN

To avoid undertraining or overtraining of the model when choosing the optimal training period, it was decided to conduct training on a different number of epochs and compare the results. TensorFlow was used to compare training at 20, 40 and 60 epochs, and the results were presented in Figure 25, where the red graph corresponds to 20 epochs, blue to 40, and purple to 60.

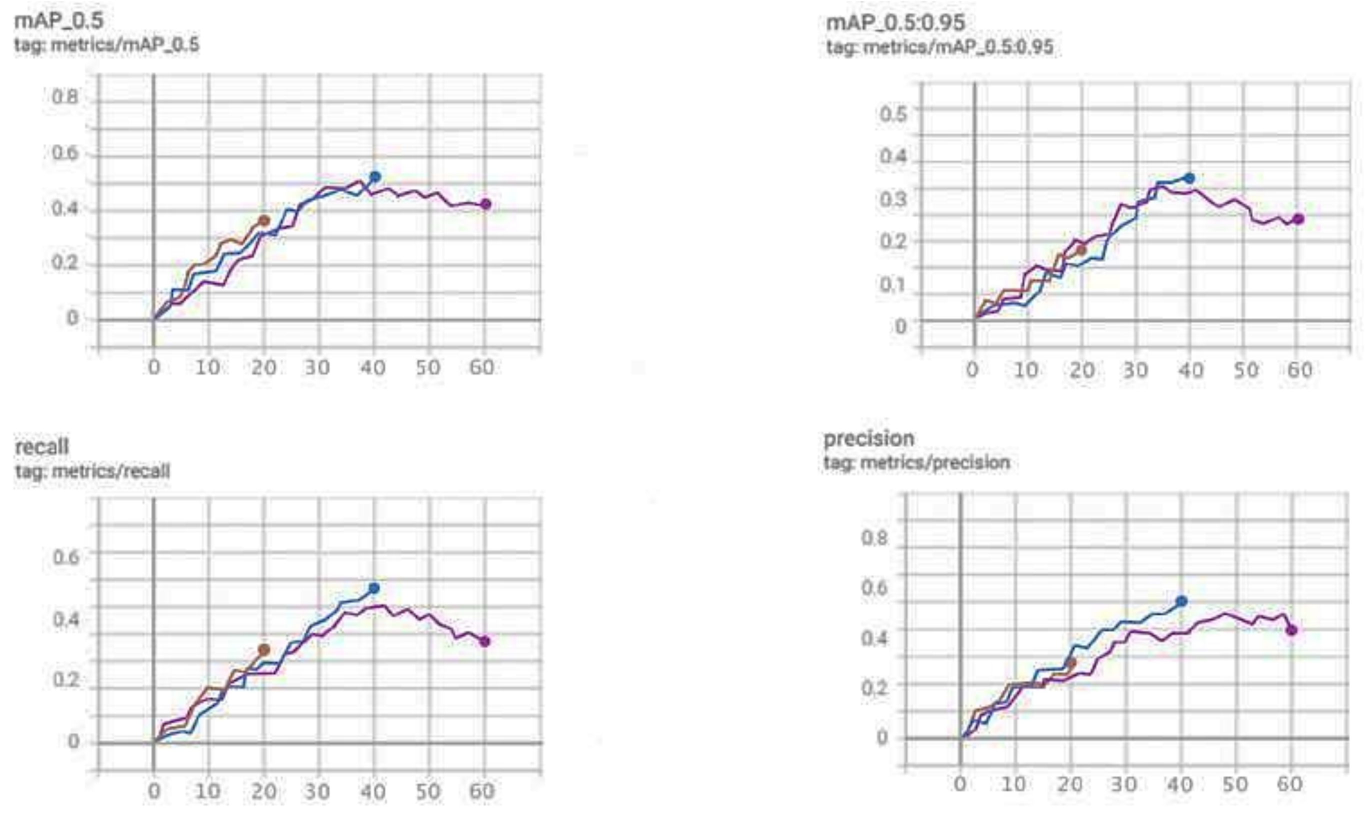

Fig. 25. Comparison of training at 20, 40 and 60 epochs

The best results were displayed at 60 epochs, as indicated by the graphs comparing training at 20, 40, and 60 epochs. This model's degree and correctness of detection proved to be much greater, and its error was significantly smaller than the other models.

6.3 Study of the ResNeXt algorithm

The ResNeXt architecture [47] is an extension of the deep residual network that replaces the standard residual block with a block that uses the split-transform-merge strategy (i.e., branched paths within a cell) used in the initial models. Simply put, rather than performing convolutions across the entire input feature map, the block's inputs are projected into a series of lower (channel) dimensional representations to which several convolutional filters are separately applied before merging the results.

You need to know that a simple neuron, as indicated in Figure 26, the output is the sum of wi multiplied by xi. The described operation can be converted into a combination of partitioning, transformation, and aggregation.

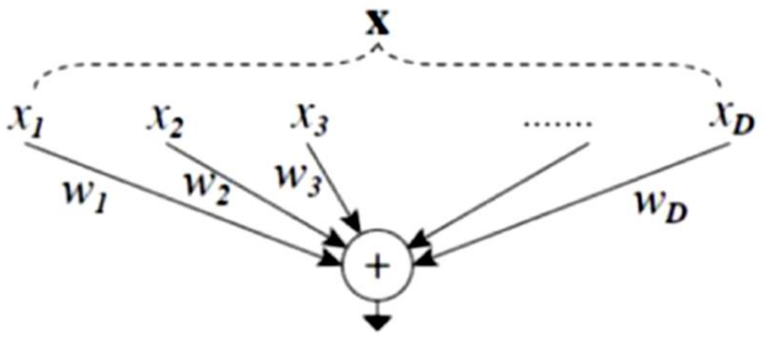

Fig. 26. Simple neuron

Partition: The vector x is divided into a one-dimensional subspace called x_i in the example above. This low-dimensional embedding is called partitioning.

Transform: the higher dimensional representation (w_i x_i) is just scaled while the low dimensional representation is altered.

Aggregation: Summation is used to aggregate transformations over all nestings.

The ResNeXt block with cardinality = 32 and its general equation are presented in Figure 27.

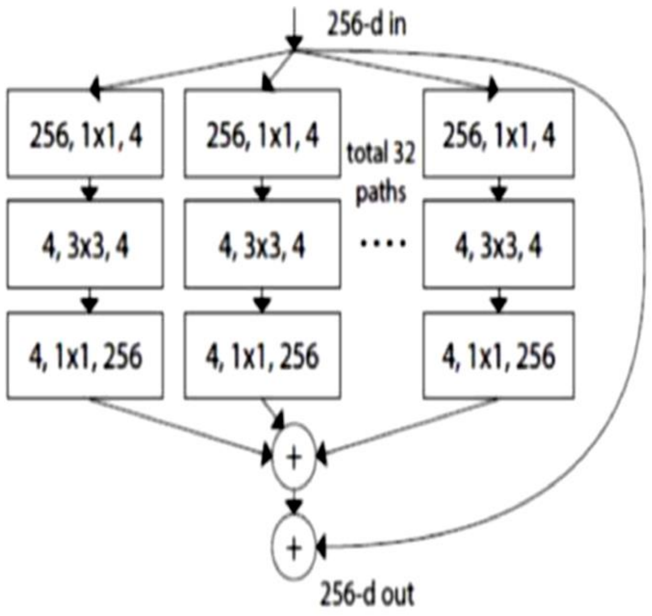

Fig. 26. ResNeXt block with cardinality = 32

The general equation of a ResNet block with cardinality = 32 is:

$$F(x) = \sum_{i=1}^{C} T_i(x) \qquad (6)$$

This Network-in-Neuron opens up into a new dimension, unlike Network-in-Network. Instead of performing a linear function where w_i is multiplied by x_i, each route in a simple neuron performs a non-linear function. A new dimension of C called "power" is introduced. The quantity of more intricate transformations is determined by the cardinality dimension.

- Convolutions 1×1–3×3–1×1 is carried out on every path. The ResNet block's bottleneck is this. For every path, the internal dimension is represented by d (d = 4). The cardinality C (C = 32) equals the number of pathways. When the dimensions of every Conv3×3 are added together (d×C = 4×32), dimension 128 will also result.

- The dimension is added by omitting the link path and raised straight from 4 to 256 before being summed.
- Compared to Inception-ResNet, which requires increasing the dimensionality from 4 to 128 to 256, ResNeXt requires minimal additional effort in designing each path.
- Unlike ResNet, in ResNeXt neurons on one path will not be connected to neurons on other paths.

The ResNeXt architecture is shown in Figure 28.

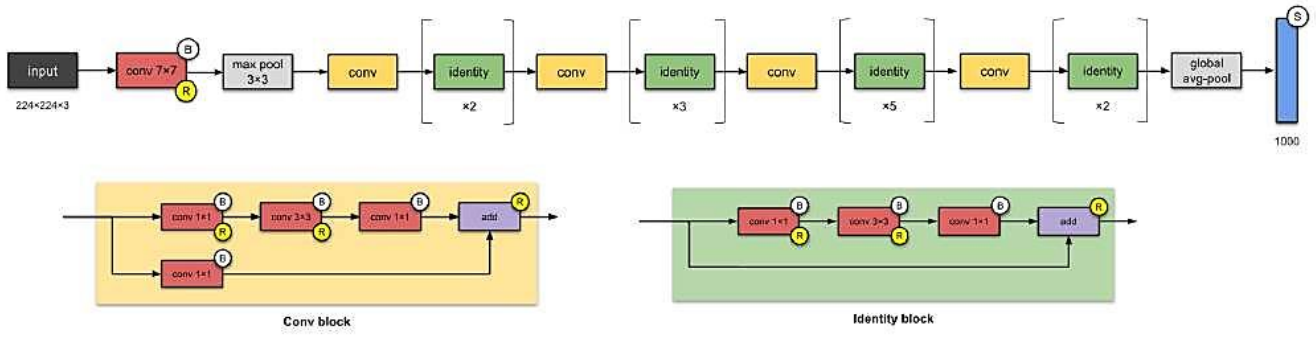

Fig. 28. ResNeXt architecture

To avoid undertraining or overtraining of the model and choosing the optimal training period, it was decided to conduct training on different numbers of epochs and compare the results. Using TensorFlow, training was carried out on 60, 75 and 90 epochs, and their results were presented on graph 3.9, where the red graph corresponds to 60 epochs, blue to 75, and purple to 90.

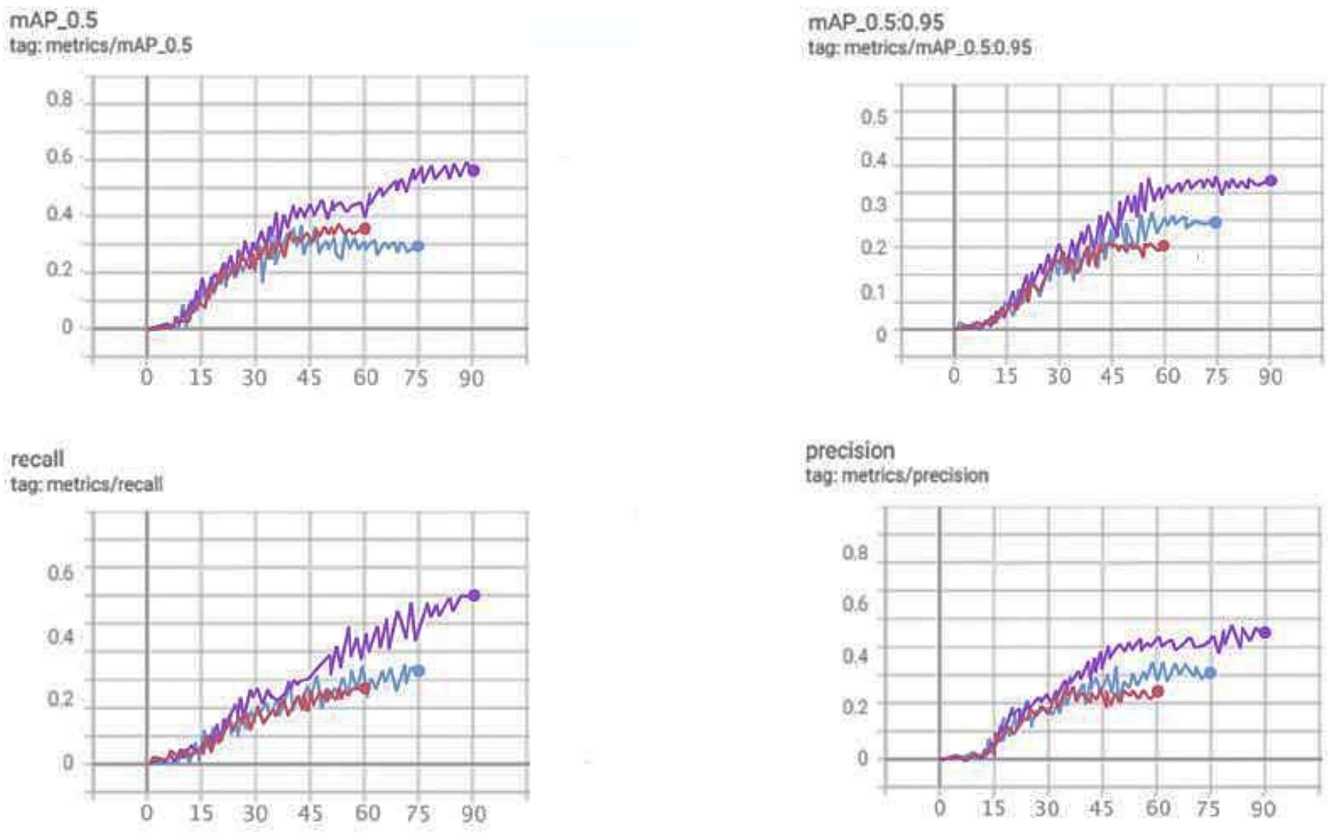

Fig. 29. Comparison of training in 60, 75 and 90 epochs

The best results were displayed at 90 epochs, as indicated by the graphs comparing training at 60, 75, and 90 epochs. This model's degree and correctness of detection proved to be substantially greater than the others, and its error was significantly reduced.

6.4 Study of the VGG algorithm

The VGG16 (Visual Geometry Group) convolutional neural network model was created by K. Simonyan and A. Zisserman [48]. When testing against the top 5 on ImageNet, a dataset comprising over 14 million images across 1000 classes, the model attains 92.7% testing accuracy. By substituting numerous 3x3 kernels for the huge kernel sizes (11 and 5, respectively, in the first and second convolutional layers) in AlexNet, VGG16 is an upgraded version of the network. Figure 30 shows the construction of the VGG16.

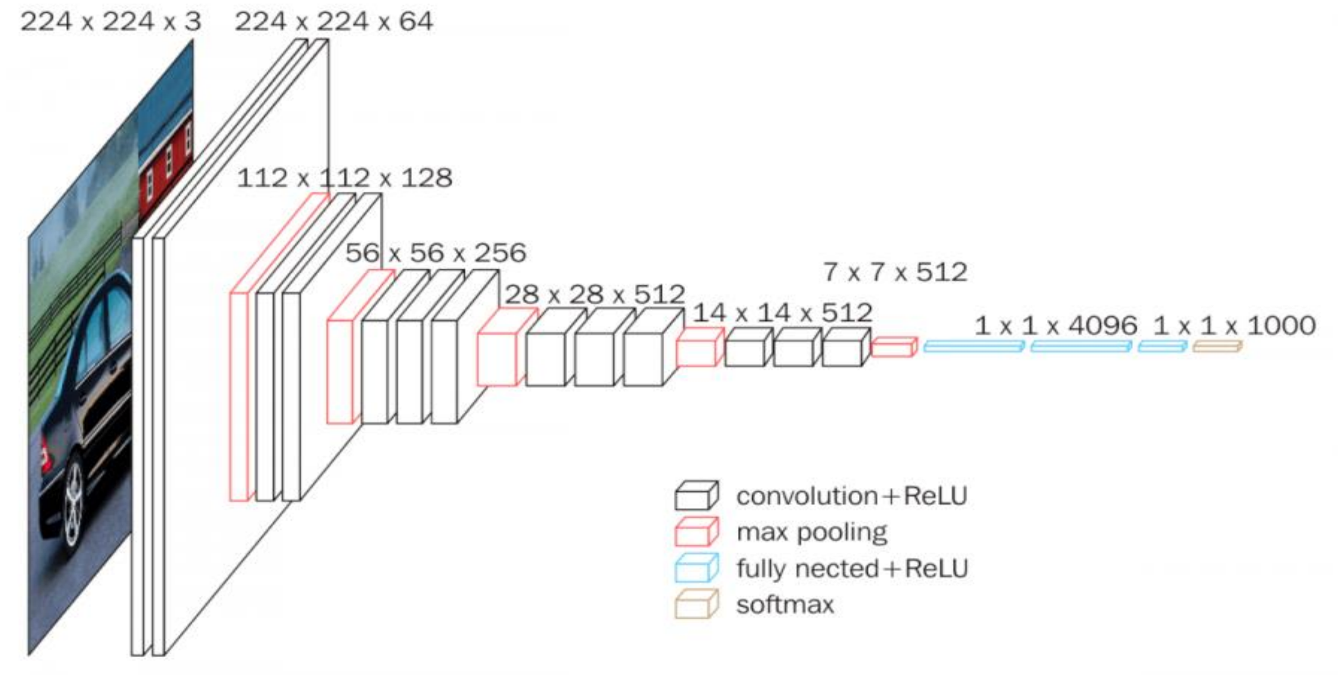

Fig. 30. VGG16 architecture

The VGG16 architecture includes the following layers:

1- Input layer - accepts an image of 224 x 224 pixels.
2- Convolutional layer - 64 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 64 feature maps measuring 224 x 224 pixels.
3- Convolutional layer - 64 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 64 feature maps measuring 224 x 224 pixels.
4- Subsampling layer - selects the maximum element from each 2 x 2 pixel region in each feature map. The result is 64 feature maps measuring 112 x 112 pixels.
5- Convolutional layer - 128 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 128 feature maps measuring 112 x 112 pixels.
6- Convolutional layer - 128 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 128 feature maps measuring 112 x 112 pixels.
7- Subsampling layer - selects the maximum element from each 2 x 2 pixel region in each feature map. Result - 128 cards features measuring 56 x 56 pixels.
8- Convolutional layer - 256 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 256 feature maps measuring 56 x 56 pixels.
9- Convolutional layer - 256 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 256 feature maps measuring 56 x 56 pixels.
10. Convolutional layer - 256 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 256 feature maps measuring 56 x 56 pixels.
11. Subsampling layer - selects the maximum element from each region 2 x 2 pixels in size on each feature map. Result - 256 cards
features measuring 28 x 28 pixels.
12. Convolutional layer - 512 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 512 feature maps measuring 28 x 28 pixels.
13. Convolutional layer - 512 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 512 feature maps measuring 28 x 28 pixels.

14. Convolutional layer - 512 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 512 feature maps measuring 28 x 28 pixels.
15. Subsampling layer - selects the maximum element from each region 2 x 2 pixels in size on each feature map. Result - 512 cards
features measuring 14 x 14 pixels.
16. Convolutional layer - 512 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 512 feature maps measuring 14 x 14 pixels.
17. Convolutional layer - 512 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 512 feature maps measuring 14 x 14 pixels.
18. Convolutional layer - 512 filters of 3 x 3 pixels, with a step of 1 pixel. The result is 512 feature maps measuring 14 x 14 pixels.
19. Subsampling layer - selects the maximum element from each area 2 x 2 pixels in size on each feature map. Result - 512 cards
features measuring 7 x 7 pixels.
20. Fully connected layer - 4096 neurons.
21. Fully connected layer - 4096 neurons.
22. Output layer - 1000 neurons corresponding to 1000 classes images in the ImageNet dataset.

VGG16 is used for image classification and can be further trained for a specific task. It demonstrates high accuracy in image recognition, but requires a large amount of computing resources for training.

It was chosen to train the model across a various number of epochs and compare the results to determine the ideal training time in order to prevent undertraining or overtraining of the model. Using TensorFlow it was. The training at 25, 50, and 75 epochs was compared; the findings are shown in Figure 31. In the graph, training on 25 epochs is marked in red, training on 50 epochs is marked in blue, and training on 75 epochs is marked in purple.

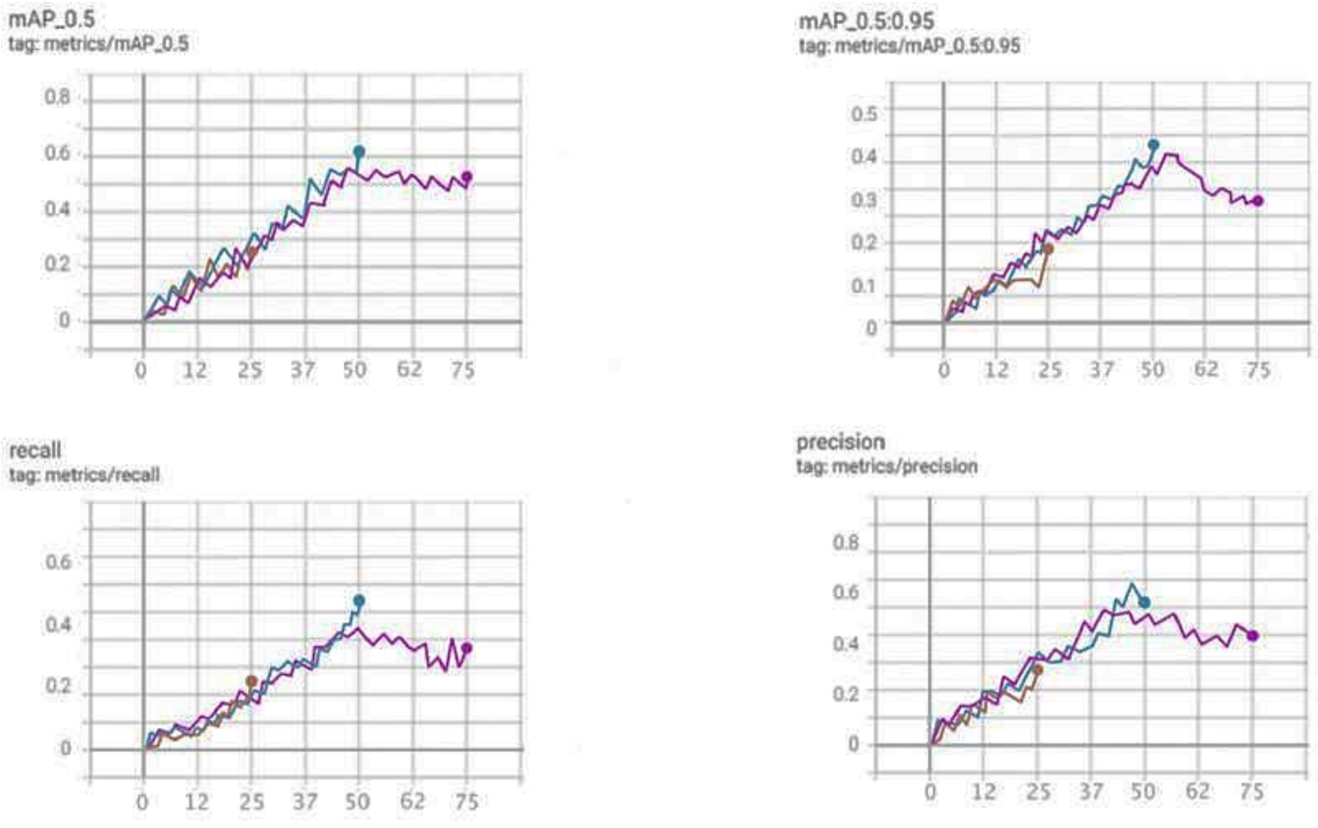

Fig. 31. Comparison of training in 25, 50, 75 epochs

The graphs comparing training at 25, 50, and 75 epochs show that the best outcomes were seen at 75 epochs. This is because the degree and correctness of detection with this model proved to be substantially greater, and the error was significantly smaller than with the other models.

6.4 Study of an algorithm based on Mask R-CNN with the addition of batch normalization layers

As the study and analysis of four neural network models revealed that Mask R-CNN produced the best results, we will attempt a method to remedy the issue that, among other things, adds layers of batch normalization as seen in Figure 32 [49].

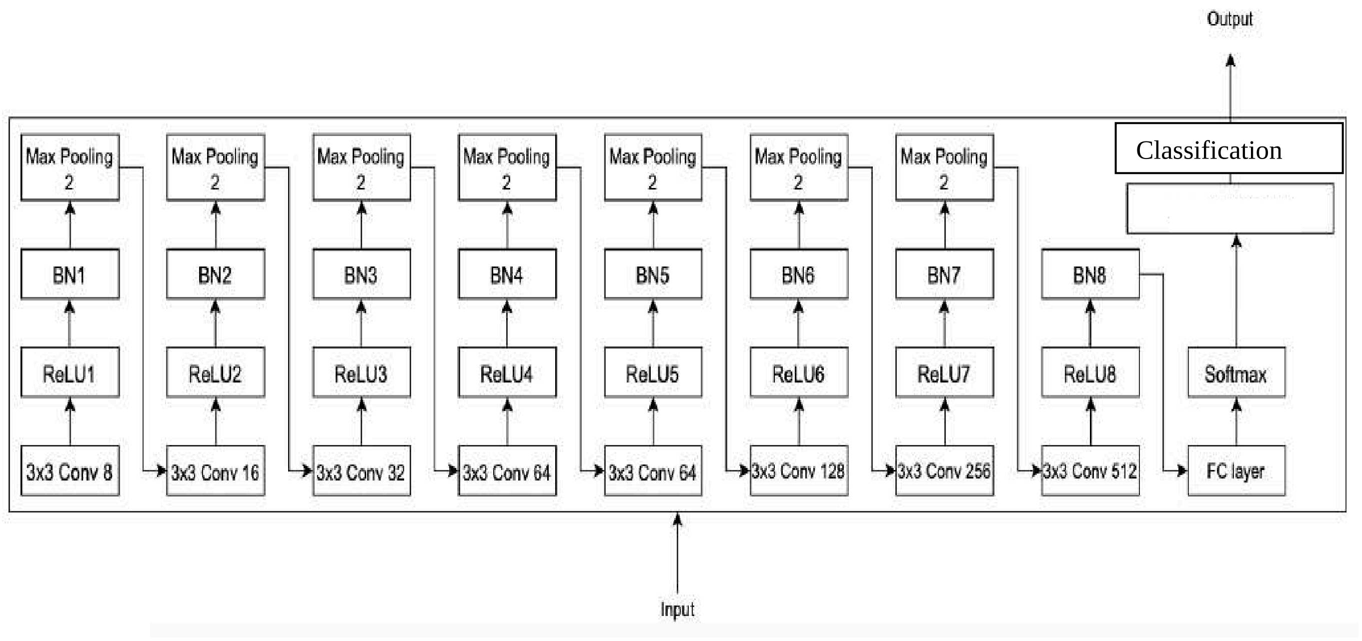

Fig. 32. Neural network algorithm based on Mask R-CNN with the addition of batch normalization layers

An additional layer introduced into the existing architecture is represented by the packet normalization (BN) layer [50, 51]. Reducing the number of layers in the architecture and increasing the classification results' accuracy over the original design were the two main objectives of the modifications. There are 35 layers in the architecture. The layers are as follows: eight convolutional layers with a 3x3 kernel and sizes ranging from 8 to 512, eight batch normalization layers, eight ReLU layers, seven max-pooling layers with stride 2, an input layer, a fully connected layer (FC), a SoftMax layer, and a classification layer [49].

It was determined to train the model across a variety of epochs and compare the outcomes to determine the ideal training period in order to prevent the model from being under- or overtrained. Using TensorFlow, training was compared at 20, 40 and 60 epochs, the results of which are presented in Figure 33. The red graph corresponds to 20 epochs, the blue graph corresponds to 40 epochs, and the purple graph corresponds to 60 epochs.

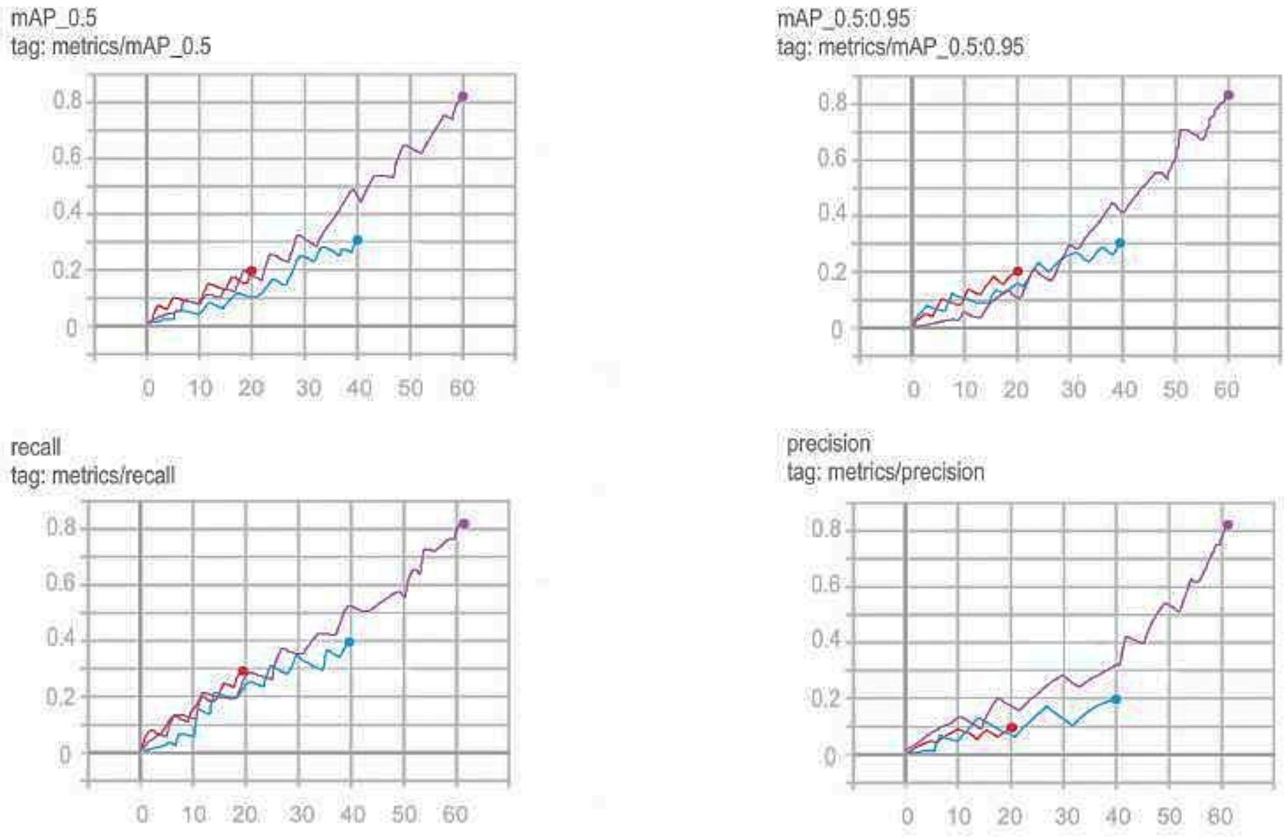

Fig. 33. Comparison of training in 20, 40, 60 epochs

The graphs comparing training at 20, 40, and 60 epochs show that the best results were shown at 60 epochs. This model's degree and correctness of detection proved to be much greater, and its error was significantly smaller than the other models.

Based on a study of neural network classification algorithms, namely: YOLOv8 (You only look once version 8), Mask R-CNN (Mask Region-Based Convolutional Neural Network), ResNeXt (Residual Networks), VGG16 (Visual Geometry Group 16), as well as a neural network algorithm based on Mask R-CNN with the addition of batch normalization (BN) layers, you can compare their work (Table 2). The comparison took place according to three metrics - Precision (accuracy), Recall (completeness), F-Score (measure of accuracy).

Table 2. Comparison of neural network classification algorithms

Metrics	Algorithm	Cracks	Potholes	Breaks	Medium
Precision	YOLO 8	0,733%	0,818%	0,874%	0,8%
	Mask R-CNN	0,786%	0,842%	0,882%	0,84%
	ResNeXt	0,67%	0,487%	0,76%	0,639%
	VGG16	0,709%	0,547%	0,66%	0,638%
	Mask R-CNN c BN	0,812%	0,788%	0,945%	0,848%
Recall	YOLO 8	0,614%	0,806%	0,85%	0,757%
	Mask R-CNN	0,574%	0,788%	0,882%	0,748%
	ResNeXt	0,379%	0,368%	0,618%	0,455%
	VGG16	0,352%	0,299%	0,688%	0,446%
	Mask R-CNN c BN	0,752%	**0,927%**	0,94%	0,873%
F-Score	YOLO 8	0,668%	0,812%	0,862%	0,78%
	Mask R-CNN	0,663%	0,814%	0,882%	0,786%
	ResNeXt	0,484%	0,419%	0,682%	0,528%

	VGG16	0,471%	0,386%	0,674%	0,51%
	Mask R-CNN c BN	0,781%	0,843%	0,942%	0,855%

According to the results of the study, the neural network classification algorithm Mask R-CNN with BN showed the best accuracy. The result of the best neural network classification algorithm as shown in figure 34.

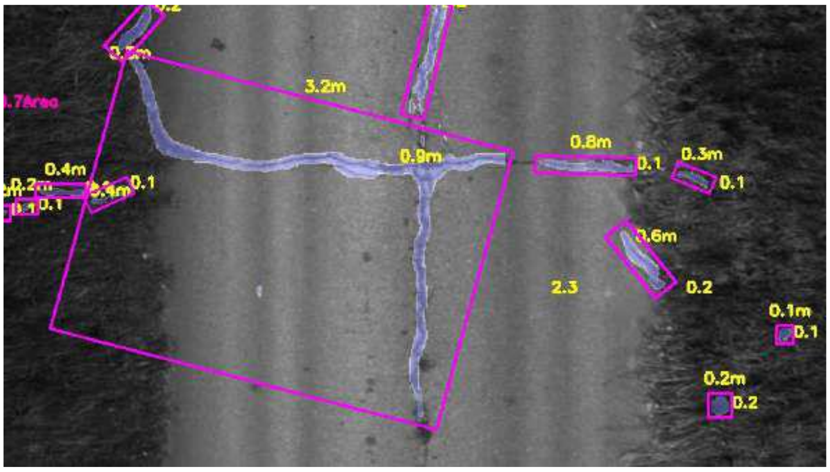

Fig. 34. Result of Mask R-CNN c BN classification algorithm

7. CONCLUSION

This study aims to introduce deep learning and its use in discovering road defects by reviewing the most important previous works, the aim of which is to discover and classify road defects on several types of data sets and the level of accuracy that has been achieved. In this study, a group of the most important algorithms for detecting road defects were reviewed: SegStereo; GC-Net; SGM-Nets; MC-CNN; algorithm based on MC-CNN using dilated convolution. All considered algorithms were trained on different numbers of epochs due to the characteristics of each of them; the best results were shown at 99 the following parameters: SegStereo – 60 epochs, GC-Net – 75 epochs, SGM-Nets – 90 epochs, MC-CNN – 33 epochs.

On the other hand, a comparative description of the algorithms is presented: YOLO, Mask RCNN, ResNeXt, VGG, a neural network algorithm based on Mask R-CNN with the addition of batch normalization layers. All considered algorithms were trained on different numbers of epochs due to the characteristics of each of them; the best results were shown with the following parameters: YOLOv8 – 75 epochs, Mask R-CNN – 60 epochs, ResNeXt – 90 epochs, VGG16 – 75 epochs, Mask R- CNN with added batch normalization layers 124 – 60 epochs. When training over a larger number of epochs, accuracy began to stagnate, and therefore training stopped. The regularization method was used to control overfitting. This study contributes greatly to helping researchers choose the best algorithm. Moreover, it will be a tool for developing other algorithms to detect and classify other road defects.

References

[1] R. Kirthiga and S. Elavenil, "A survey on crack detection in the concrete surface using image processing and machine learning," Journal of Building Pathology and Rehabilitation, vol. 9, 2023. doi: 10.1007/s41024-023-00371-6.
[2] H. Zakeri, F. M. Nejad, and A. H. Gandomi, "Automatic Detection and Its Applications in Infrastructure," Journal Name, 2022.
[3] C. R. Farrar and K. Worden, "Structural Health Monitoring: A Machine Learning Perspective," John Wiley & Sons, 2012.
[4] V. Peansupap and T. Jewbunchu, "Development of System for Detecting Railway Surface Defects by Using Deep Learning Technique," in Proceedings of the International Conference on Computational Science and Its Applications, 2023, pp. 37-46, doi: 10.1007/978-981-99-4049-3_37.
[5] F. Yang et al., "Feature Pyramid and Hierarchical Boosting Network for Pavement Crack Detection," IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 4, pp. 1525-1535, Apr. 2020, doi: 10.1109/TITS.2019.2913673.
[6] J. Guo et al., "Surface defect detection of civil structures using images: Review from a data perspective," Automation in Construction, vol. 158, p. 105186, May 2024, doi: 10.1016/j.autcon.2021.10518.
[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436-444, May 2015, doi: 10.1038/nature14539.
[8] Y. J. Cha, W. Choi, and O. Buyukozturk, "Deep learning-based crack damage detection using convolutional neural networks," Computer-Aided Civil and Infrastructure Engineering, vol. 32, no. 5, pp. 361-378, May 2017, doi: 10.1111/mice.12263.
[9] Y. Z. Lin, Z. H. Nie, and H. W. Ma, "Structural damage detection with automatic feature-extraction through deep learning," Computer-Aided Civil and Infrastructure Engineering, vol. 32, no. 12, pp. 1025-1046, Dec. 2017, doi: 10.1111/mice.12313.
[10] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, ``Densely connected convolutional networks," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Piscataway, NJ, USA, Jul. 2017, pp. 22612269.
[11] R. X. Li et al., "Unified vision-based methodology for simultaneous concrete defect detection and geolocalization," Computer-Aided Civil and Infrastructure Engineering, vol. 33, no. 7, pp. 527-544, Jul. 2018, doi: 10.1111/mice.12351.
[12] X. W. Ye, T. Jin, and P. Y. Chen, "Structural crack detection using deep learning-based fully convolutional networks," Advances in Structural Engineering, Mar. 2019, doi: 10.1177/1369433219836292.
[13] I. A. Kanaeva and J. A. Ivanova, "Road pavement crack detection using deep learning with synthetic data," IOP Conference Series: Materials Science and Engineering, vol. 1019, no. 1, p. 012036, 2021.
[14] P. Jing et al., "Road Crack Detection Using Deep Neural Network Based on Attention Mechanism and Residual Structure," IEEE Access, vol. 11, pp. 919-929, 2022.
[15] M. M. Mustakim, "Road Damage Detection Based on Deep Learning," 2023, doi 10.13140/RG.2.2.22523.28967.
[16] Goodfellow, Y. Bengio, and A. Courville, "Deep Learning," in Deep Learning, vol. 1, MIT Press, Cambridge, MA, USA, 2016.
[17] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," Bulletin of Mathematical Biophysics, vol. 5, pp. 115-133, 1943. doi: 10.1007/BF02478259.
[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998. doi: 10.1109/5.726791.
[19] X. W. Ye, T. Jin, and C. B. Yun, "A review on deep learning-based structural health monitoring of civil infrastructures," Smart Structures and Systems, vol. 24, no. 5, pp. 567-585, 2019.
[20] A. Sharawi and L. Taha, "Deep learning Artificial Neural Networks (ANN)," 2020.
[21] KAYA, Ömer; Çodur, Muhammed Yasin (2024), "N-RDD2024:Road damage and defects", Mendeley Data, V4, doi: 10.17632/27c8pwsd6v.4.
[22] J. C. Chou, W. A. O'Neill, and H. D. Cheng, "Pavement distress classification using neural networks," in Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 1994, vol. 1, pp. 397–401, doi: 10.1109/icsmc.1994.399871.
[23] N. Academies, " Transportation Research Board," NCHRP, 17 11 2021. [Online]. Available: https://www.trb.org/NCHRP/NCHRP.aspx.
[24] Kimiko O. Bowman and L. R. Shenton, "Estimator: Method of Moments", pp 2092–2098, Encyclopedia of statistical sciences, Wiley (1998).
[25] H.-D. Cheng, "Automated real-time pavement distress detection using fuzzy logic and neural network," in Nondestructive Evaluation of Bridges and Highways, 1996, vol. 2946, pp. 140–151, doi: 10.1117/12.259131.
[26] J. M. Alvarez, T. Gevers, Y. LeCun, and A. M. Lopez, "Road scene segmentation from a single image," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2012, vol. 7578 LNCS, no. PART 7, pp. 376–389, doi: 10.1007/978-3-642-33786- 4_28.
[27] C. A. Brust, S. Sickert, M. Simon, E. Rodner, and J. Denzler, "Convolutional Patch Networks with Spatial Prior for Road Detection and Urban Scene Understanding," Feb. 2015.

[28] S. Milhomem, T. da S. Almeida, W. G. da Silva, E. M. da Silva, and R. L. de Carvalho, “Weightless Neural Network with Transfer Learning to Detect Distress in Asphalt,” Int. J. Adv. Eng. Res. Sci., vol. 5, no. 12, pp. 294–299, 2018, doi: 10.22161/ijaers.5.12.40.
[29] Chun and S. K. Ryu, “Road surface damage detection using fully convolutional neural networks and semi-supervised learning,” Sensors (Switzerland), vol. 19, no. 24, p. 5501, Dec. 2019, doi: 10.3390/s19245501.
[30] Chun and S. K. Ryu, “Road surface damage detection using fully convolutional neural networks and semi-supervised learning,” Sensors (Switzerland), vol. 19, no. 24, p. 5501, Dec. 2019, doi: 10.3390/s19245501.
[31] Henrique Oliveira, Paulo Lobato Correia Automatic Road Crack Segmentation Using Entropy And Image Dynamic Thresholding // 17th European Signal Processing Conference (EUSIPCO 2009) Glasgow, Scotland, August 24- 28, 200.
[32] G Gunawan, Heri Nuriyanto, S Sriadhi, Achmad Fauzi, Ari Usman, F Fadlina, Haida Dafitri, Janner Simarmata, Andysah Putera Utama Siahaan, and Robbi Rahim Mobile Application Detection of Road Damage sing Canny Algorithm // 1st International Conference on Green and Sustainable Computing (ICoGeS) 2017.
[33] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," in IEEE Transactions on Systems, Man, and Cybernetics, vol. 9, no. 1, pp. 62-66, Jan. 1979, doi: 10.1109/TSMC.1979.4310076.
[34] Shengchun Wang and Wensheng Tang /Pavement Crack Segmentation Algorithm Based on Local Optimal Threshold of Cracks Density Distribution /
International Conference on Intelligent Computing ICIC 2011: Advanced Intelligent Computing pp 298-302.
[35] Naoki Tanak , Kenji Uernatsu A Crack Detection Method in Road Surface Images Using Morphology. //IAPR Workshop on Machine Vision Applications, Nov. 17-19, 1998.
[36] Wei, Na, Xiangmo Zhao, Tao Wang, and Hongxun Song. "Mathematical morphology based asphalt pavement crack detection." In International Conference on Transportation Engineering 2009, pp. 3883-3887. 2009.
[37] Zbontar, Jure, and Yann LeCun. "Stereo matching by training a convolutional neural network to compare image patches." J. Mach. Learn. Res. 17.1 (2016): 2287-2318.
[38] A. Shaked and L. Wolf, "Improved Stereo Matching with Constant Highway Networks and Reflective Confidence Learning," 2017 IEEE Conference on
Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6901-6910, doi: 10.1109/CVPR.2017.730.
[39] Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T., 2016. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation nikolaus. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition. pp. 4040–4048. https://doi.org/10.1109/CVPR.2016.438.
[40] Close Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A., 2017. End-to-end Learning of Geometry and Context for Deep Stereo Regression. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 66–75. https://doi.org/10.1109/ICCV.2017.17.
[41] H. Shi, T. Ushio, M. Endo, K. Yamagami, and N. Horii, "A multichannel convolutional neural network for cross-language dialog state tracking," in 2016 IEEE Spoken Language Technology Workshop (SLT), 2016, pp. 559-564: IEEE.
[42] Redmon, Joseph et al. “You Only Look Once: Unified, Real-Time Object Detection.” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 779-788.
[43] S. Jiang, T. Xu, J. Li, B. Huang, J. Guo and Z. Bian, "IdentifyNet for Non- Maximum Suppression," in IEEE Access, vol. 7, pp. 148245-148253, 2019, doi: 10.1109/ACCESS.2019.2944671.
[44] Zhuanzhi. Yan Lecun, 09.12.2021. [Online]. Available. – URL: https://www.zhuanzhi.ai/document/bbf11e032d71fc226a3c3373f438ca2d.
[45] He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask rcnn." In Proceedings of the IEEE international conference on computer vision, pp. 2961-2969. 2017.
[46] Ren, Shaoqing et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.” IEEE Transactions on Pattern Analysis and
Machine Intelligence 39 (2015): 1137-1149.
[47] Xie, Saining et al. “Aggregated Residual Transformations for Deep Neural Networks.” 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017): 5987-5995.
[48] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).
[49] Setiawan, Wahyudi & Damayanti, Fitri. (2020). Layers Modification of Convolutional Neural Network for Pneumonia Detection. Journal of Physics: Conference Series. 1477. 052055. 10.1088/1742-6596/1477/5/052055.
[50] F. Schilling, The Effect of Batch Normalization on Deep Convolutional Neural Networks. Stockholm, Sweden: KTH Royal Institue of Technology, School of Computer Science and Communication, 2016.
[51] Ioffe, Sergey and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.” ArXiv abs/1502.03167 (2015).