infrastructures

MDPI

Article

A Model Classifying Four Classes of Defects in Reinforced Concrete Bridge Elements Using Convolutional Neural Networks

Roman Trach [1,2]

[1] Institute of Civil Engineering, Warsaw University of Life Sciences, 02-776 Warsaw, Poland; roman_trach@sggw.edu.pl
[2] Institute of Civil Engineering and Architecture, National University of Water and Environmental Engineering, 33028 Rivne, Ukraine

Abstract: Recently, the bridge infrastructure in Ukraine has faced the problem of having a significant number of damaged bridges. It is obvious that the repair and restoration of bridges should be preceded by a procedure consisting of visual inspection and evaluation of the technical condition. The problem of fast and high-quality collection, processing and storing large datasets is gaining more and more relevance. An effective way to solve this problem is to use various machine learning methods in bridge infrastructure management. The purpose of this study was to create a model based on convolutional neural networks (CNNs) for classifying images of concrete bridge elements into four classes: "defect free", "crack", "spalling" and "popout". The eight CNN models were created and used to conduct its training, validation and testing. In general, it can be stated that all CNN models showed high performance. The analysis of loss function (categorical cross-entropy) and quality measure (accuracy) showed that the model on the MobileNet architecture has optimal values (loss, 0.0264, and accuracy, 94.61%). This model can be used further without retraining, and it can classify images on datasets that it has not yet "seen". Practical use of such a model allows for the identification of three damage types.

Keywords: bridge infrastructures; defects; reinforced concrete; convolutional neural network; machine learning

Citation: Trach, R. A Model Classifying Four Classes of Defects in Reinforced Concrete Bridge Elements Using Convolutional Neural Networks. *Infrastructures* **2023**, *8*, 123. https://doi.org/10.3390/infrastructures8080123

Academic Editor: Alessandro Zona

Received: 10 July 2023
Revised: 2 August 2023
Accepted: 3 August 2023
Published: 11 August 2023

1. Introduction

Bridges are one of the most important types of transport infrastructure and require the attention of state institutions and constant funding for proper maintenance.

According to the Ministry of Regional Development of Ukraine, at the end of 2020, 12,097 road bridges were in operation and their total length was 746.8 km [1]. According to the Ukrainian National Standard "Guidelines regarding the inspection of building objects to determine and assess their technical condition", the expected service life of bridges ranges from 70 to 100 years, depending on the type of construction [2]. However, the average age of bridges is 55 years, with approximately 12% of them being over 80 years old. Consequently, the structural integrity of these bridges is approaching exhaustion.

Another global and urgent problem of the bridge industry in Ukraine is the need to develop a single management system that would unify the knowledge base regarding the condition and characteristics of all existing bridges [3]. The role of bridge management systems consists of monitoring, diagnosing and forecasting the technical condition of structures and planning optimal maintenance of bridges [4]. The collected data are analyzed, verified and integrated into a centralized database [5]. This enables efficient tracking and management of information on bridge inspections, maintenance and structural evaluation [6]. In addition, bridge management systems are important when making decisions about the possible retrofit of bridges [7]. For instance, Rokneddin et al. studied the present-day seismic reliability of aging bridges in highway networks, evaluated through a time-dependent seismic fragility analysis of typical bridge classes [8].

As a rule, data are collected at various stages during the life cycle of a bridge (Figure 1).

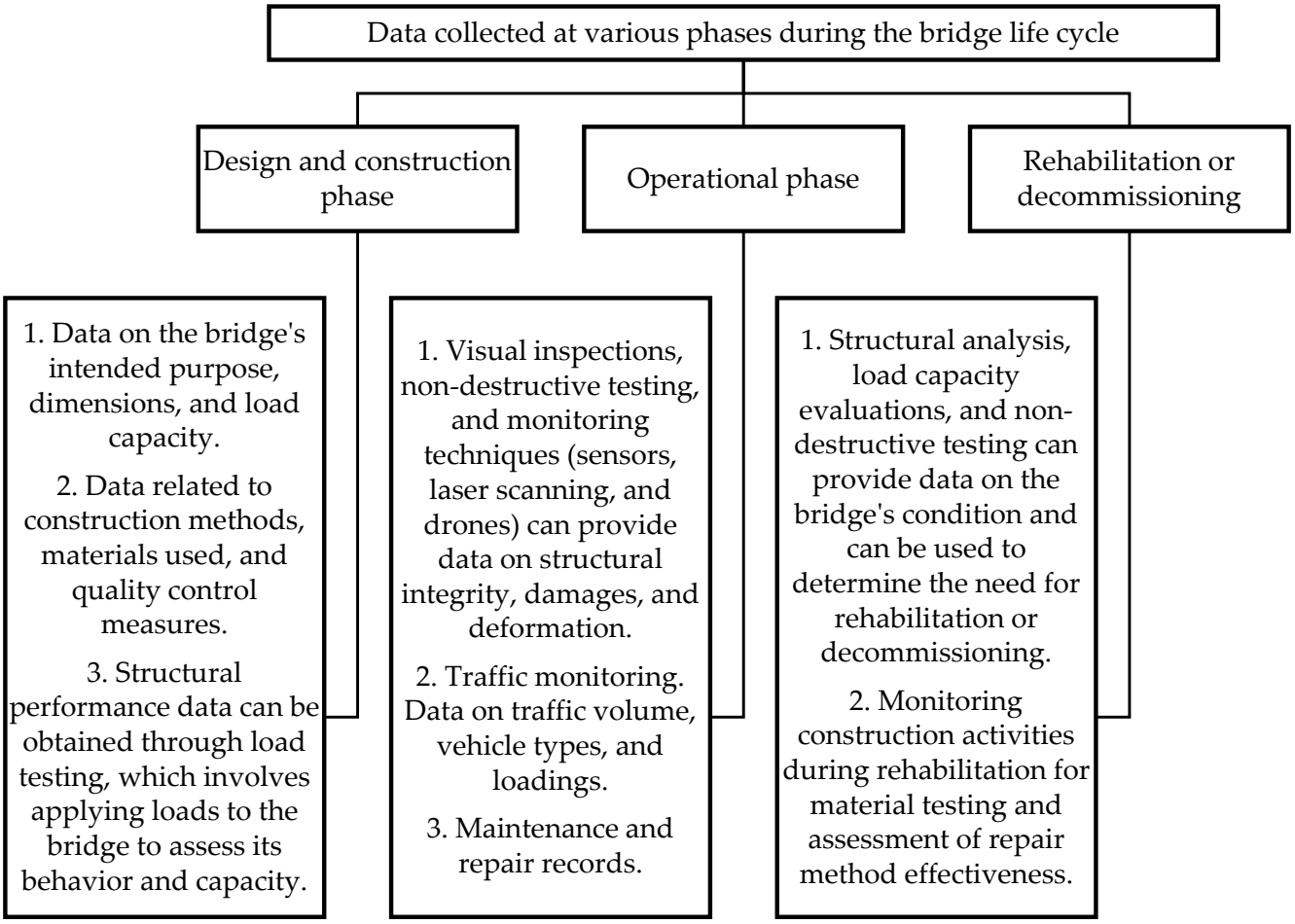

Figure 1. Data collected at various phases during the bridge life cycle (source: own design).

Considering the fairly significant lifetime of bridges, the largest amount of information and data is collected during the operation phase. Accordingly, the problem of fast and high-quality collection, processing and storage of large datasets is gaining more and more relevance. An effective way to solve this problem is to use various machine learning methods in bridge infrastructure management.

This article is the third in a series of studies focused on the use of machine learning tools that can be applied to make management decisions at the stage of bridge operation. In the first article, the object of research was the method of quantitative assessment of the condition of bridge components. A tool based on artificial neural networks (ANNs) was developed to quantitatively assess the technical condition of bridge components [9]. Based on the classification tables of the operating conditions of the bridge components, five datasets were formed: bridge span, bridge deck, pier cap beam, piers and abutments and approaches. This approach allowed for the minimization of the uncertainties associated with the subjective assessment of experts and for an increase in accuracy.

The aim of the second study was to determine the possible causes of defects in reinforced concrete elements of bridges based on the identification of external indicators using ML (machine learning) tools [10]. That study created and compared the performance of four ML models, namely support vector regression (SVR), decision trees (DTs), random forest (RF) and artificial neural networks (ANNs). The practical use of such models allows for the identification of some causes of defects during a visual inspection of the structures.

2. Background

In general, ML methods can be used in bridge infrastructure in many ways [11]. ML models can assist in analyzing various risk factors that contribute to bridge management, such as traffic patterns, environmental conditions and structural characteristics [12,13]. ML models can analyze historical traffic data, including vehicle types, volumes and loadings, to predict future traffic patterns and load demands on bridges [14]. ML algorithms can be used to optimize bridge design by analyzing historical design data, performance data from existing bridges and structural simulations [15]. One of the most important directions of using ML in the bridge industry is damage identification, assessment and forecasting of the technical condition [16,17].

This direction can be divided into two large groups depending on the type of input data. The first type is numeric input. Most often, such problems are considered as classification or regression problems and are solved using algorithms such as support vector regression (SVR), decision trees (DTs), artificial neural networks (ANNs), etc. ML algorithms can analyze historical data on bridge conditions, maintenance records and environmental factors to develop predictive maintenance models. This approach helps to optimize maintenance schedules, extend the lifespan of bridges and minimize disruptions to operations.

The second type is input data in the form of photo and video files. Most often, such problems are considered as classification or segmentation problems and are solved using algorithms of convolutional neural networks (CNNs): U-Net, RCNN, YOLO, etc. ML algorithms can analyze data collected from sensors, laser scanning, drones and monitoring systems. By automatically detecting and classifying visual defects, cracks or corrosion, the models help in the early identification of deterioration, ensuring timely maintenance and reducing the risk of failure.

It is worth noting that the successful implementation of ML in bridge infrastructure requires access to high-quality and diverse data, as well as careful model training, validation and interpretation.

Within the second direction, ML models can be used in the following scope:

- Detecting and classifying various types of defects on bridge structures, such as cracks, corrosion, spalling or deformations [18]. By training the network on labeled images of different defect categories, it can learn to identify and locate these defects accurately [19].
- CNNs can learn the normal patterns and visual characteristics of damaged bridge components [20]. By analyzing images or videos of bridges, CNNs can identify anomalies that deviate from the expected patterns, highlighting potential areas of concern that require further inspection or assessment.
- CNNs can perform image segmentation to separate bridge components or regions of interest from the background [21]. This can be useful in identifying specific areas or elements for further analysis, such as detecting cracks or corrosion within a specific part of the bridge. For example, Geng et al. used CNNs for detecting, locating and quantifying corrosion damage in a truss-type bridge through the autocorrelation of vibration signals [22].
- By comparing images or videos captured during different inspection cycles, CNNs can identify changes in the condition of bridge elements over time [23]. This enables the detection of progressive deterioration or changes that might require attention or further investigation.

Summarizing the above, this study was motivated by two problems. The first one is urgent and related to the presence of a significant number of damaged and destroyed bridges in Ukraine which require an urgent and high-quality inventory and assessment of their technical condition. To solve it, the CNN model was proposed, which is capable of classifying elements of reinforced concrete structures according to four classes: “defect free”, “crack”, “spalling” and “popout”.

The second problem is strategic and is related to the development of a bridge management system in Ukraine which covers a large part of them with information. Accordingly, this study, along with the previous two, is aimed at creating effective tools based on ML which can receive, process and store various data and generate the knowledge necessary to make management decisions.

The novelty of this study consists of the development of the model based on convolutional neural networks, which allows for the classification of reinforced concrete damage into four classes. Moreover, the model can be used further without retraining, and it can classify images it has not yet "seen".

The next part of the article has the following structure: Section 3 contains a description of the whole process of defect classification, a description of the CNN architecture used in image classification tasks, the model parameters and evaluation metrics used in the study.

Section 4 presents the results of a productivity comparison of eight CNN models and the examples of image classification. Section 5 presents the discussion; Section 6 presents the conclusions of the study.

3. Methodology

3.1. General Description of the Process

As mentioned earlier, this study used the images' division into four classes: "defect free", "crack", "spalling" and "popout". The classification of defects was selected based on the following criteria:

- defects that are most often found in various types of reinforced concrete structures [24];
- the ability of computer vision to clearly separate one class of defects from another. The division into classes was motivated by the Guide by the ACI Committee, "Guide for Making a Condition Survey of Concrete in Service" [25]. Its purpose was to standardize the reporting of the condition of the concrete in a structure. The Guide collected a large database on the external indications of damage to concrete, illustrated with photographs.

The classification proposed in this study has the following structure:

- "crack". This class includes all the defects from group A.1 (various type of cracks);
- "spalling". This class includes two groups of defects—A.2.20 (scaling) and A.2.21 (spalling);
- "popuot". This class includes defects from group A.2.19 (popout—the breaking away of small portions of a concrete surface that leaves shallow, typically conical, depressions).

The Guide separately distinguishes two classes of damage—scaling (A.2.20) and spalling (A.2.21)—which differ in the reasons of damage. However, in the case of using computer vision, it is quite difficult to separate these two types of defects, so they were combined into one set of "spalling".

Figure 2 shows the schematic diagram of the complete methodology used in this study.

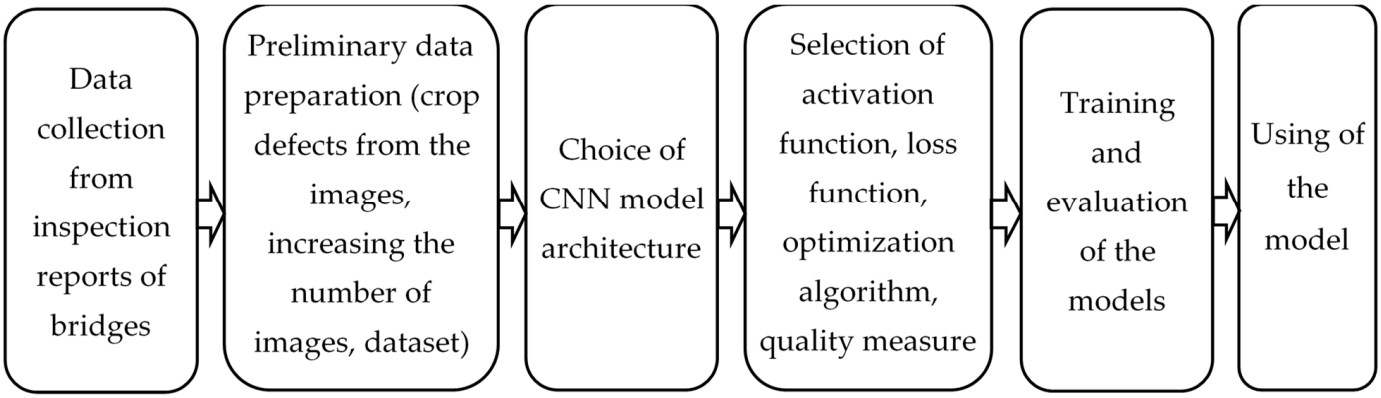

Figure 2. The schematic diagram of the complete methodology.

The whole process of classification of defects had the following sequence.

1. Data collection. It was proven that the presence of a large and high-quality dataset is the key to the success of the CNN modeling process. A large amount is used to mean a dataset that contains from several thousand to several tens of thousands of images [26,27]. The quality of the dataset is based on the fact that the images should be as close to real conditions as possible: images with background noise and different shades of color, including surface roughness, different lighting conditions, background debris, etc. [28,29].

The images were collected from real inspection reports of bridge structures, which were taken for over 15 years in different regions of Ukraine. Sample images of elements of bridge structures obtained from the survey report of a beam-type road bridge made of precast concrete are shown in Figure 3. The survey of the bridge was carried out in September-October 2019 in Rivne city, Ukraine. Figure 4 shows the cross-section of the bridge as viewed from the abutment No. 0.

Figure 3. Some structural elements of the bridge with damages (Source: own photos).

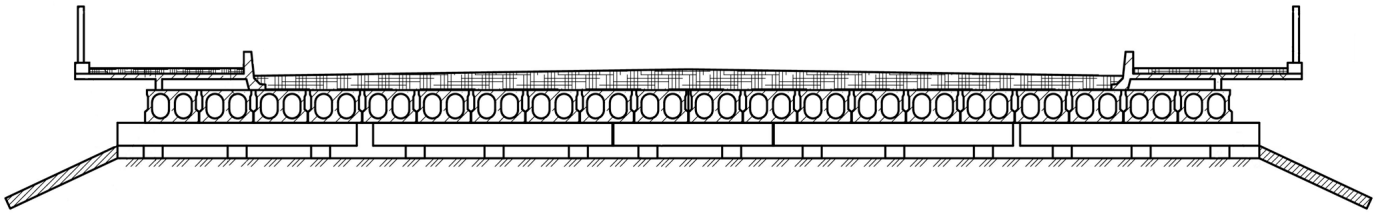

Figure 4. The cross-section of the bridge (Source: own design).

2. Preliminary data preparation. The next step was to manually crop 256 by 256 pixel defects from the images. This enabled us to obtain a dataset containing 1300 "defect free" images, "crack", "spalling" and 800 "popout" images. An example of four types of images is shown in Figure 5. At the next stage of the study, "accuracy" metric was applied to evaluate the quality of the model, which is quite sensitive for unbalanced datasets. Therefore, it was decided to equalize the number of images in each class by increasing the number of images in the "popout" set. For this purpose, part of the images was rotated by 90° and saved in files with a new name. The entire dataset was divided into three parts—training, testing and validation in the proportion of 80%/10%/10%, respectively.

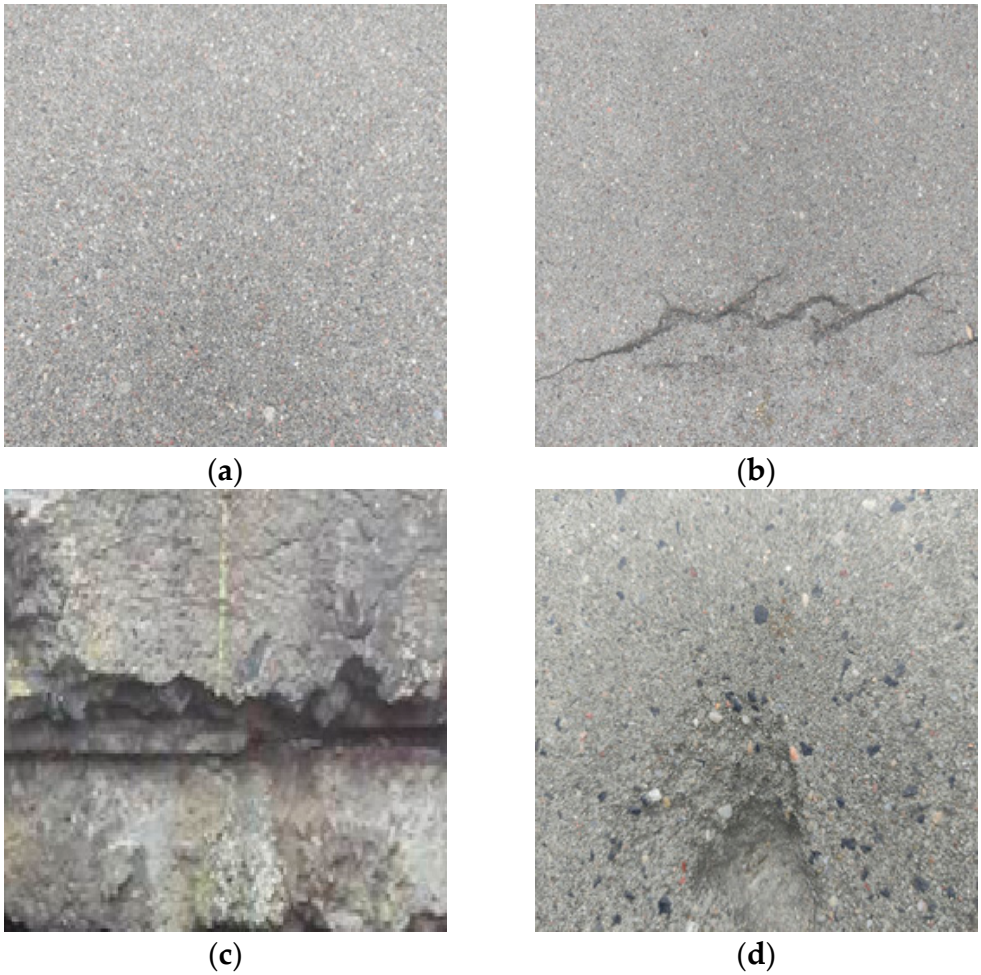

Figure 5. Example of images from the dataset (**a**)—defect free, (**b**)—crack, (**c**)—spalling, (**d**)—popout. (Source: own photos).

3. Choice of CNN model architecture (described in Section 3.2).
4. Selection of the main parameters of the CNN model: activation function, loss function, optimization algorithm and quality measure. This process is described in Section 3.3.
5. Training of the models. The selected models were trained using supervised learning methods. Since one of the research objectives was to determine the training time of several types of models over 20 epochs, the callback function was not used.
6. Evaluation of the models. Pre-trained models were evaluated on a separate validation dataset to assess their performance and identify any potential problems.
7. Using of the model. After the evaluation, a model with the highest performance was used to classify the images it "has not seen".

3.2. CNN Architecture Used in Image Classification Task

VGG16 (Visual Geometry Group) (2014) [26] is the type of CNN that is often used in classification tasks due to its simplicity and high performance [30]. It consists of 16 convolutional and fully connected layers with a fixed-size input image. VGG16 employs small convolutional filters (3×3) stacked on top of each other, enabling deeper representations and increased model capacity. InceptionV3 (2015) [31] is a popular CNN architecture developed by Google. It introduced the concept of "inception modules", which employ parallel convolutions with different filter sizes to capture local and global features of an image. This architecture significantly improved the accuracy of image classification models and has been influential in subsequent CNN designs. ResNet50 (2015) [32] is a recurrent-neural-network (RNN)-based architecture primarily used for image recognition

tasks. Unlike traditional CNNs, which operate on fixed-sized windows, ReNet50 processes images using recurrent connections, allowing it to capture spatial dependencies across different image regions. This unique design enables ReNet50 to effectively model complex patterns and relationships within images. Xception (2016) [33] utilizes depth-wise separable convolutions extensively, separating the spatial and channel-wise operations, which reduces the model's computational complexity to a large scale. Xception achieves state-of-the-art performance on various image classification benchmarks while maintaining a relatively compact model size. DenseNet201 (2017) [34] is a CNN architecture designed for image classification tasks. It employs a deep network structure with 201 layers, utilizing a combination of convolutional, pooling and fully connected layers. DeneNet201 is known for its exceptional performance in accurately classifying images across various datasets. MobileNet (2017) [35] is a lightweight CNN architecture designed for mobile and embedded devices with limited computational resources. It focuses on reducing the number of parameters and computations while maintaining competitive accuracy. MobileNet utilizes depth-wise separable convolutions which separate the spatial and channel-wise convolutions, resulting in a smaller model size and faster inference speed compared with traditional CNNs. NASNetLarge (2017) [36] is a neural architecture specifically developed for the task of object detection. It combines feature extraction and object-detection modules, enabling accurate localization and classification of objects in images. NASNetLarge leverages a "neck-and-anchor" design, which effectively captures multi-scale contextual information and improves the detection performance across different object sizes and aspect ratios. EfficientNetV2 (2019) [37] is an advanced CNN architecture that focuses on achieving high performance while maintaining computational efficiency. By utilizing a compound scaling technique, EfficientNetV2 achieves excellent accuracy with fewer parameters compared with traditional CNN models, making it computationally efficient and suitable for deployment on resource-constrained devices.

3.3. Model Parameters and Evaluation Metric

The softmax activation function was used, which is often applied for multiclass classification tasks [38]. In multi-class classification tasks, the output of the softmax function is often interpreted as the predicted probability distribution over the classes. The class with the highest probability is typically chosen as the predicted class label. The function transforms the input vector into an output vector of the same shape, where each element represents the probability of the corresponding class. The output values range between 0 and 1, and the sum of all probabilities is equal to 1, making it suitable for representing class probabilities. The softmax calculates the probability p_i for each class i using the following formula [39]:

$$p_i = \frac{e^{z_i}}{\sum_j^k e^{z_i}} \quad (1)$$

where $z_1, z_1, ..., z_n$ ¯input vector and i—classes.

The categorical cross-entropy loss function is a widely used loss function in multi-class classification problems, where each input sample belongs to exactly one class and the goal is to assign the correct class label [40,41]. It measures the dissimilarity between the predicted probability distribution and the true probability distribution of the classes. Given a predicted probability distribution (obtained from the output of a softmax activation function) and the true labels in the form of a one-hot encoded vector, the categorical cross-entropy loss is computed by taking the negative logarithm of the predicted probability corresponding to the true class label. The categorical cross-entropy loss function is determined with the formula [42]:

$$loss_{CCE} = -\left[y_1 \cdot \log(\hat{y_1}) + y_2 \cdot \log(\hat{y_2}) + y_3 \cdot \log(\hat{y_3}) + y_4 \cdot \log(\hat{y_4})\right] \quad (2)$$

where y_1, y_2, y_3, y_4—true values and $\hat{y_1}, \hat{y_2}, \hat{y_3}, \hat{y_4}$— predicted values.

The loss function aims to minimize the difference between the predicted probabilities and the ground truth labels. It penalizes larger deviations between the predicted and true probabilities, encouraging the model to assign higher probabilities to the correct class. Minimizing the categorical cross-entropy loss during training helps the model learn appropriate weights and biases to make accurate predictions for each class.

The optimization algorithm ADAM (Adaptive Moment Estimation) was used, which combines ideas from both the AdaGrad and RMSProp algorithms [43]. The algorithm has gained popularity due to its effectiveness and efficiency in optimizing deep neural networks. It has been successfully applied to various tasks, including image recognition, natural language processing and reinforcement learning. ADAM offers several advantages, including fast convergence, robustness to different types of neural network architectures and the ability to handle sparse gradients effectively. ADAM adapts the learning rate for each parameter individually based on estimates of the first and second moments of the gradients. It maintains a learning rate per parameter, allowing it to adapt the learning rate on a per-parameter basis. The update rule for this algorithm involves calculating exponential moving averages of the gradients and their squared values. The updated parameters are computed using a combination of the moving averages and the current gradients. ADAM performs bias correction to address the initialization bias issue. This correction helps in the early stages of training, when the parameter estimates are biased due to initialization.

The accuracy was used as evaluation metric [44], and it was calculated by dividing the number of correctly predicted samples by the total number of samples in the dataset. It represents the proportion of correctly classified instances. Accuracy values are expressed as a percentage, ranging from 0% to 100%. A higher value indicates a better-performing classifier, while a lower one suggests more errors in the predictions. This metric is easy to understand, interpret and communicate. It provides a simple and intuitive measure of overall model performance, making it widely used in classification tasks. Accuracy may not be the most appropriate metric in situations where class imbalances exist within the dataset. When the classes are imbalanced, accuracy can be misleading because a classifier that always predicts the majority class may achieve a high accuracy even if it performs poorly on the minority class.

4. Results

The modeling was performed in Python using the numpy, matplotlib and tensorflow libraries. PC characteristics: processor AMD Ryzen 5 5600X 6-Core 3.70 GHz, RAM 32.0 GB, NVIDIA GeForce RTX 3070 8.0 GB, Windows 11 64-bit. Table 1 shows the results of model performance testing.

Table 1. Comparison of ANN models' performance.

	DenseNet 201	EfficientNet V2	Inception V3	Mobile Net	NASNet Large	ResNet 50	VGG16	Xception
Total times, s	2405	5949	1252	1250	4996	1588	2026	1522
Mean times, s	120	297	63	63	250	79	101	76
Categorical cross-entropy	0.078	0.2343	0.1414	0.0264	0.065	0.0422	0.0525	0.0313
Accuracy, %	91.82	86.21	89.54	94.61	92.28	93.1	92.73	93.49

In general, it can be stated that all CNN models showed high performance. The model built on the MobileNet architecture has the highest value for accuracy and, at the same time, the lowest value for categorical cross-entropy (loss, 0.0264, and accuracy, 94.61%). The model built on the EfficientNetV2 architecture has the lowest accuracy value and, at the same time, the highest categorical cross-entropy value (loss, 0.2343, and accuracy, 86.21%). The situation is similar with the training time—the lowest values for the total training

time (20 epochs) and the average time of one epoch are in the MobileNet model. The EfficientNetV2 model shows the longest time indicators. It is worth noting that the models based on ResNet50 and Xception also show indicators close to the MobileNet model.

Figures 6 and 7 show a comparison of the loss function and accuracy for the training and validation datasets for the best (MobileNet) and worst (EfficientNetV2) CNN models.

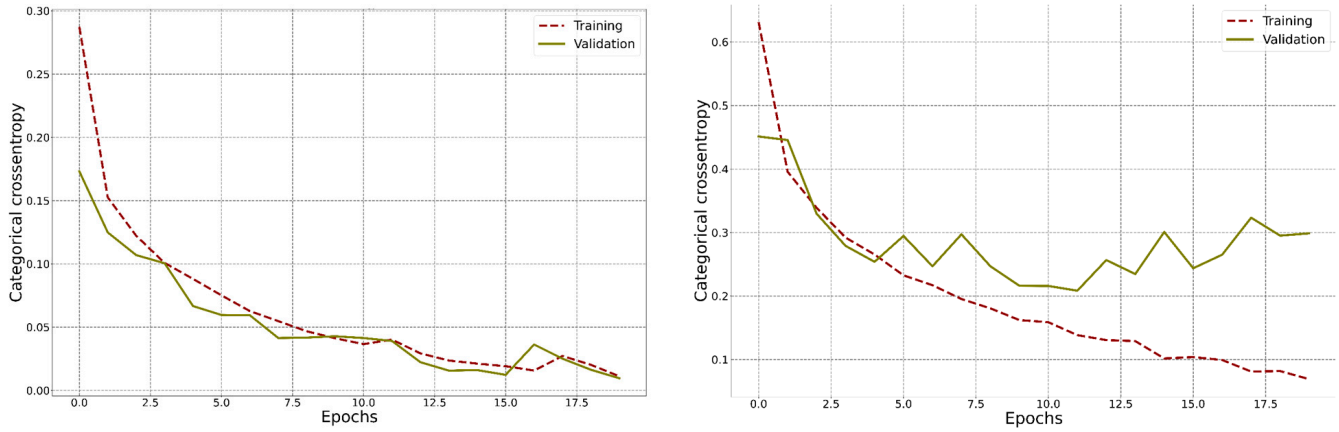

Figure 6. The results of loss function for training and validation datasets of MobileNet and EfficientNetV2: (**a**) loss function results, MobileNet; (**b**) loss function results, EfficientNetV2.

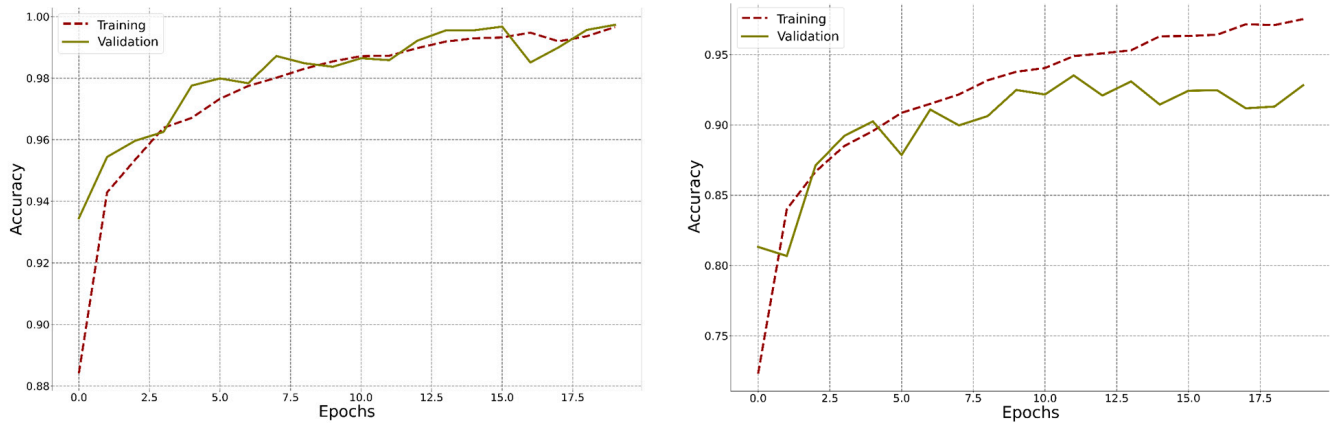

Figure 7. The results of accuracy for training and validation datasets of MobileNet and EfficientNetV2: (**a**) accuracy, MobileNet; (**b**) accuracy, EfficientNetV2.

Figure 6a shows the comparison of the results of loss function for the training and validation datasets of the MobileNet model. The maximum loss function of the training set was in the first epoch—0.287; that of the validation set was in the first epoch—0.173. The minimum loss function of the training set was in the 20th epoch—0.011—and that of the validation set was in the 20th epoch—0.009. Figure 6b shows the comparison results of the loss function for the training and validation datasets of the EfficientNetV2 model. The maximum loss function of the training set was in the first epoch—0.631; that of the validation set was in the first epoch—0.451. The minimum loss function of the training set was in the 20th epoch—0.069—and that of the validation set was in the 12th epoch—0.208.

Figure 7a shows the comparison of the results of accuracy for the training and validation datasets of the MobileNet model. The minimum accuracy of the training set was in the first epoch—0.884; that of the validation set was in the first epoch—0.934. The maximum accuracy of the training set was in the 20th epoch—0.996 and that of the validation set was in the 20th epoch—0.997. Figure 7b shows the comparison of the results of accuracy for the

training and validation datasets of the EfficientNetV2 model. The minimum accuracy of the training set was in the first epoch—0.558; that of the validation set was in the second epoch—0.806. The maximum accuracy of the training set was in the 20th epoch—0.975 and that of the validation set was in the 12th epoch—0.935.

The next step was to check how the model would classify images that it "has not seen". After loading from the previously saved model files, it was "shown" four images corresponding to the four classes. Table 2 shows example images and classification results.

Table 2. Examples of images and classification results (predicted classes are indicated in Italic).

	Defect Free (Class 0)	**Crack (Class 1)**	**Spalling (Class 2)**	**Popout (Class 3)**	**Predicted Class**
	*9.994* $\times$ *10*$^{-1}$	2.474 $\times$ 10^{-8}	5.386 $\times$ 10^{-4}	9.499 $\times$ 10^{-7}	(Class 0)
	7.336 $\times$ 10^{-6}	*9.999* $\times$ *10*$^{-1}$	8.023 $\times$ 10^{-8}	2.186 $\times$ 10^{-16}	(Class 1)
	8.645 $\times$ 10^{-3}	9.883 $\times$ 10^{-7}	*9.913* $\times$ *10*$^{-1}$	2.673 $\times$ 10^{-11}	(Class 2)
	9.499 $\times$ 10^{-11}	1.737 $\times$ 10^{-7}	3.734 $\times$ 10^{-1}	*6.265* $\times$ *10*$^{-1}$	(Class 3)

The analysis of Table 2 shows that the CNN model classified "defect free", "crack" and "spalling" images at a high level (probability values very close to 1.0). In the case of "popout" image classification, a result of 6.265 $\times$ 10^{-1} was obtained, which was the largest value. The probability of classification into the "spalling" class (3.734 $\times$ 10^{-1}) was also quite high.

5. Discussion

The possibility of using the proposed CNN model for image classification was analyzed and compared in terms of the quality of its work with models from similar studies. Li and Zhao proposed a convolutional encoder–decoder network to detect cracks in the images (two-class classification) [45]. The validation results showed 98.90% accuracy. Yang et al. used three neural networks (AlexNet, VGGNet13, ResNet18) to detect and recognize structural cracks. First, a training dataset of a model was built [46]. The validation showed that the ResNet18 model worked with most satisfactory results. Wu et al. presented a crack detection technology based on a convolutional neural network, GoogLeNet Inception V3 [47]. The dataset included images with and without cracks. The accuracy of the final trained model on the test set could reach 98.50%. Shin et al. developed a convolution-based concrete multi-damage recognition neural network. The image datasets consisted of different types of concrete surface damages, including surface cracks, rebar exposure and delamination. The trained model demonstrated a damage-detection accuracy of 98.9% [48]. Zoubir et al. used the deep convolutional neural networks model to classify three types of images; the best proposed approach achieved a high testing accuracy (97.13%) [49].

This study has several limitations. First, the model classifies four types of images, three of which are concrete damage types. At the same time, as noted earlier, two different types of damage were assigned to the "spalling" class—scaling and spalling—which differ in the reasons that caused the damage. Second, this study was focused on solving the problem of image classification and did not concern the solution of segmentation problems. The goal of classification is to determine which objects are present in an image and potentially label them with an appropriate class. Object classification algorithms analyze images to make predictions about the objects present. Image segmentation involves dividing an image into several segments based on certain criteria. The goal is to assign a label or category to each pixel in the image, effectively creating a pixel-level mask that identifies different objects within the image. The segmentation of different types of concrete damage is the next direction of research.

6. Conclusions

Recently, the bridge infrastructure in Ukraine has faced the problem of a significant number of damaged bridges. It is obvious that the repair and restoration of bridges should be preceded by the procedure of their visual inspection and evaluation of their technical condition. The purpose of this study was to create a model based on CNNs for classifying images of concrete bridge elements into four classes: "defect free", "crack", "spalling" and "popout". For this purpose, at the first stage of the research, a dataset was created that included 1300 images of each class. At the next stage, eight CNN models were created, trained, validated and tested. In general, it can be stated that all CNN models showed high performances. The analysis of loss function (categorical cross-entropy) and quality measure (accuracy) showed that the MobileNet model architecture has the optimal values (loss, 0.0264, and accuracy, 94.61%). This model can be used further without retraining, and it can classify images on datasets it "has not yet seen". The practical use of such a model allows for the identification of three damage types. Further research may be aimed at expanding the number of damage classes, increasing the level of model performance and expanding the model by identifying more damage classes.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets generated and analyzed during the current study are available from the author upon reasonable request.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Moshynskyi, V.; Striletskyi, P.; Trach, R. Application of the Building Information Modelling (BIM) for Bridge Structures. *Acta Sci. Pol.-Archit. Bud.* **2022**, *20*, 3–9. [CrossRef]
2. *DSTU-N B V. 1.2-18:2016*; Guidelines Regarding the Inspection of Building Objects to Determine and Assess Their Technical Condition. Ministry of Regional Development and Construction of Ukraine: Kyiv, Ukraine, 2016.
3. World Bank Group. *Ukraine Rapid Damage and Needs Assessment: February 2022–February 2023*; World Bank Group: Washington, DC, USA, 2023.
4. Szymanek, S. Construction Production Trends and Industry Optimism in EU Countries after the COVID-19 Pandemic. *Acta Sci. Pol. Archit.* **2022**, *21*, 21. [CrossRef]
5. Bodnar, L.; Koval, P.; Stepanov, S.; Panibratets, L. Operational state of bridges of Ukraine. *Avtošljachovyk Ukr.* **2019**, *2*, 57–68. [CrossRef]
6. Rashidi, M.; Samali, B.; Sharafi, P. A New Model for Bridge Management: Part A: Condition Assessment and Priority Ranking of Bridges. *Aust. J. Civ. Eng.* **2016**, *14*, 35–45. [CrossRef]
7. Frangopol, D.M.; Kong, J.S.; Gharaibeh, E.S. Reliability-Based Life-Cycle Management of Highway Bridges. *J. Comput. Civ. Eng.* **2001**, *15*, 27–34. [CrossRef]
8. Wakjira, T.G.; Nehdi, M.L.; Ebead, U. Fractional Factorial Design Model for Seismic Performance of RC Bridge Piers Retrofitted with Steel-Reinforced Polymer Composites. *Eng. Struct.* **2020**, *221*, 111100. [CrossRef]
9. Rokneddin, K.; Ghosh, J.; Dueñas-Osorio, L.; Padgett, J.E. Bridge Retrofit Prioritisation for Ageing Transportation Networks Subject to Seismic Hazards. *Struct. Infrastruct. Eng.* **2013**, *9*, 1050–1066. [CrossRef]
10. Forcellini, D.; Alzabeebee, S. Seismic Fragility Assessment of Geotechnical Seismic Isolation (GSI) for Bridge Configuration. *Bull. Earthq. Eng.* **2023**, *21*, 3969–3990. [CrossRef]
11. Mackie, K.R.; Lu, J.; Elgamal, A. Performance-Based Earthquake Assessment of Bridge Systems Including Ground-Foundation Interaction. *Soil Dyn. Earthq. Eng.* **2012**, *42*, 184–196. [CrossRef]
12. Trach, R.; Moshynskyi, V.; Chernyshev, D.; Borysyuk, O.; Trach, Y.; Striletskyi, P.; Tyvoniuk, V. Modeling the Quantitative Assessment of the Condition of Bridge Components Made of Reinforced Concrete Using ANN. *Sustainability* **2022**, *14*, 15779. [CrossRef]
13. Trach, R.; Ryzhakova, G.; Trach, Y.; Shpakov, A.; Tyvoniuk, V. Modeling the Cause-and-Effect Relationships between the Causes of Damage and External Indicators of RC Elements Using ML Tools. *Sustainability* **2023**, *15*, 5250. [CrossRef]
14. Assaad, R.; El-adaway, I.H. Bridge Infrastructure Asset Management System: Comparative Computational Machine Learning Approach for Evaluating and Predicting Deck Deterioration Conditions. *J. Infrastruct. Syst.* **2020**, *26*, 04020032. [CrossRef]
15. Yang, D.Y. Deep Reinforcement Learning–Enabled Bridge Management Considering Asset and Network Risks. *J. Infrastruct. Syst.* **2022**, *28*, 04022023. [CrossRef]
16. Pena-Caballero, C.; Kim, D.; Gonzalez, A.; Castellanos, O.; Cantu, A.; Ho, J. Real-Time Road Hazard Information System. *Infrastructures* **2020**, *5*, 75. [CrossRef]
17. Zhang, Y.; Li, Y.; Zhou, X.; Luo, J. cST-ML: Continuous spatial-temporal meta-learning for traffic dynamics prediction. In Proceedings of the 2020 IEEE International Conference on Data Mining (ICDM), Sorrento, Italy, 17–20 November 2020; pp. 1418–1423.
18. Gomez-Cabrera, A.; Escamilla-Ambrosio, P.J. Review of Machine-Learning Techniques Applied to Structural Health Monitoring Systems for Building and Bridge Structures. *Appl. Sci.* **2022**, *12*, 10754. [CrossRef]
19. Hajializadeh, D. Deep Learning-Based Indirect Bridge Damage Identification System. *Struct. Health Monit.* **2023**, *22*, 897–912. [CrossRef]
20. Hammouch, W.; Chouiekh, C.; Khaissidi, G.; Mrabti, M. Crack Detection and Classification in Moroccan Pavement Using Convolutional Neural Network. *Infrastructures* **2022**, *7*, 152. [CrossRef]
21. Dzięcioł, J.; Sas, W. Perspective on the Application of Machine Learning Algorithms for Flow Parameter Estimation in Recycled Concrete Aggregate. *Materials* **2023**, *16*, 1500. [CrossRef]
22. Nguyen, T.; Nguyen, T.; Sidorov, D.N.; Dreglea, A. Machine Learning Algorithms Application to Road Defects Classification. *Intell. Decis. Technol.* **2018**, *12*, 59–66. [CrossRef]
23. Duan, Y.; Chen, Q.; Zhang, H.; Yun, C.B.; Wu, S.; Zhu, Q. CNN-Based Damage Identification Method of Tied-Arch Bridge Using Spatial-Spectral Information. *Smart Struct. Syst.* **2019**, *23*, 507–520.
24. Geng, Q.; Zhou, Z.; Cao, X. Survey of Recent Progress in Semantic Image Segmentation with CNNs. *Sci. China Inf. Sci.* **2018**, *61*, 051101. [CrossRef]
25. Yanez-Borjas, J.J.; Valtierra-Rodriguez, M.; Machorro-Lopez, J.M.; Camarena-Martinez, D.; Amezquita-Sanchez, J.P. Convolutional Neural Network-Based Methodology for Detecting, Locating and Quantifying Corrosion Damage in a Truss-Type Bridge Through the Autocorrelation of Vibration Signals. *Arab. J. Sci. Eng.* **2023**, *48*, 1119–1141. [CrossRef]
26. Yessoufou, F.; Zhu, J. One-Class Convolutional Neural Network (OC-CNN) Model for Rapid Bridge Damage Detection Using Bridge Response Data. *KSCE J. Civ. Eng.* **2023**, *27*, 1640–1660. [CrossRef]
27. Żółtowski, B.; Castañeda, L.F.; Żółtowski, M.; Wierzbicki, T. Research Condition of Complex Technical Objects. *Acta Sci. Pol. Archit.* **2022**, *21*, 21. [CrossRef]
28. *ACI 201.1 R-92*; ACI Committee 201 Guide for Making a Condition Survey of Concrete in Service. American Concrete Institute: Farmington Hills, MI, USA, 1997.

29. Fujita, Y.; Hamamoto, Y. A Robust Automatic Crack Detection Method from Noisy Concrete Surfaces. *Mach. Vis. Appl.* **2011**, *22*, 245–254. [CrossRef]
30. Nishikawa, T.; Yoshida, J.; Sugiyama, T.; Fujino, Y. Concrete Crack Detection by Multiple Sequential Image Filtering: Concrete Crack Detection by Image Processing. *Comput.-Aided Civ. Infrastruct. Eng.* **2012**, *27*, 29–47. [CrossRef]
31. Luo, H.; Xiong, C.; Fang, W.; Love, P.E.D.; Zhang, B.; Ouyang, X. Convolutional Neural Networks: Computer Vision-Based Workforce Activity Assessment in Construction. *Autom. Constr.* **2018**, *94*, 282–289. [CrossRef]
32. Zhang, K.; Zhang, Y.; Cheng, H.-D. CrackGAN: Pavement Crack Detection Using Partially Accurate Ground Truths Based on Generative Adversarial Learning. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 1306–1319. [CrossRef]
33. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**. [CrossRef]
34. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9. [CrossRef]
35. 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.
36. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. *arXiv* **2016**. [CrossRef]
37. Huang, G.; Liu, Z.; van der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. *arXiv* **2016**. [CrossRef]
38. Michele, A.; Colin, V.; Santika, D.D. MobileNet Convolutional Neural Networks and Support Vector Machines for Palmprint Recognition. *Procedia Comput. Sci.* **2019**, *157*, 110–117. [CrossRef]
39. Zoph, B.; Vasudevan, V.; Shlens, J.; Le, Q.V. Learning Transferable Architectures for Scalable Image Recognition. *arXiv* **2017**. [CrossRef]
40. Tan, M.; Le, Q.V. EfficientNetV2: Smaller Models and Faster Training. *arXiv* **2021**. [CrossRef]
41. Kouretas, I.; Paliouras, V. Simplified hardware implementation of the softmax activation function. In Proceedings of the 2019 8th International Conference on Modern Circuits and Systems Technologies (MOCAST), Thessaloniki, Greece, 13–15 May 2019; pp. 1–4.
42. Kowalski, J.; Połoński, M.; Lendo-Siwicka, M.; Trach, R.; Wrzesiński, G. Method of Assessing the Risk of Implementing Railway Investments in Terms of the Cost of Their Implementation. *Sustainability* **2021**, *13*, 13085. [CrossRef]
43. Rusiecki, A. Trimmed Categorical Cross-entropy for Deep Learning with Label Noise. *Electron. Lett.* **2019**, *55*, 319–320. [CrossRef]
44. Alay, N.; Al-Baity, H.H. Deep Learning Approach for Multimodal Biometric Recognition System Based on Fusion of Iris, Face, and Finger Vein Traits. *Sensors* **2020**, *20*, 5523. [CrossRef]
45. Trach, Y.; Melnychuk, V.; Michel, M.M.; Reczek, L.; Siwiec, T.; Trach, R. The Characterization of Ukrainian Volcanic Tuffs from the Khmelnytsky Region with the Theoretical Analysis of Their Application in Construction and Environmental Technologies. *Materials* **2021**, *14*, 7723. [CrossRef]
46. Trach, R.; Lendo-Siwicka, M.; Pawluk, K.; Bilous, N. Assessment of the Effect of Integration Realisation in Construction Projects. *Teh. Glas.* **2019**, *13*, 254–259. [CrossRef]
47. Gunawardana, A.; Shani, G. A Survey of Accuracy Evaluation Metrics of Recommendation Tasks. *J. Mach. Learn. Res.* **2009**, *10*, 2935–2962.
48. Li, S.; Zhao, X. Automatic Crack Detection and Measurement of Concrete Structure Using Convolutional Encoder-Decoder Network. *IEEE Access* **2020**, *8*, 134602–134618. [CrossRef]
49. Yang, C.; Chen, J.; Li, Z.; Huang, Y. Structural Crack Detection and Recognition Based on Deep Learning. *Appl. Sci.* **2021**, *11*, 2868. [CrossRef]