Arabian Journal for Science and Engineering
https://doi.org/10.1007/s13369-025-10279-7

RESEARCH ARTICLE-CIVIL ENGINEERING

A structural damage ranking using ConvNeXt for post-earthquake image classification

O. Tugrul Turan[1] · Huseyin Kaya[2] · Gulsen Taskin[3] · Tolga Cinar[1] · Alper Ilki[1]

Received: 20 January 2025 / Accepted: 24 April 2025

Abstract
A structural damage classifier utilizing ConvNeXt, a state-of-the-art deep convolutional and residual neural network, is proposed for the automatic evaluation of post-earthquake images. The classifier distinguishes between two types of damage: structural and nonstructural. To achieve high accuracy and reliability, transfer learning was used to fine-tune the ConvNeXt model with our dataset. The model was trained on 9,645 labeled images from 1,789 reinforced concrete buildings affected by the Elazığ earthquake on January 24, 2020. To enhance detection performance while minimizing training costs, various transfer learning strategies, data augmentation, and regularization techniques were implemented. The final model was tested on images of reinforced concrete buildings taken after the Kahramanmaraş Earthquake on February 6, 2023. The results demonstrated consistent classification performance aligned with domain knowledge, along with strong generalization on the Kahramanmaraş dataset, highlighting the effectiveness of ConvNeXt in post-earthquake damage assessment.

Keywords Damage assessment · Earthquake · Deep learning · Transfer learning

1 Introduction

Damage assessment after natural hazards is crucial to determine the post-disaster condition and safety of the structures and to make informed decisions regarding their future. These assessments are conducted manually by dedicating one or more inspectors, and the damage's significance and the structural members' current state are determined. According to German et al. [1], the assessment procedures rely on the inspector's knowledge, experience, and visual inspection capability, which are time-consuming. Advanced sensing technologies can better detect the current situation of the structures with higher accuracy and resolution. Thus, to increase the speed of structural inspection and damage evaluation, it is necessary to automate the current practices with advanced sensing technologies to collect data and data processing techniques.

This study aims to design a deep learning architecture to prioritize and classify post-earthquake damages in reinforced concrete buildings by distinguishing damages of structural members (i.e., columns, beams, shear walls, etc.) and nonstructural members (i.e., partition and infill walls). In other words, after an earthquake, distinguishing between structures that have incurred structural damage, requiring detailed and time-consuming inspections, and those that have sustained nonstructural damage, which do not need detailed examination, enables efficient utilization of time and resources. ConvNeXt, a new-generation deep learning architecture, was utilized and carefully fine-tuned in this study to detect damage automatically. Additionally, this study is novel in its use of images captured by most likely nonexperts during post-earthquake damage assessment campaigns in the reinforced concrete units that remained untouched after the earthquake.

✉ O. Tugrul Turan
tturan@itu.edu.tr

Huseyin Kaya
huseyin.kaya@payten.com

Gulsen Taskin
tasking@itu.edu.tr

Tolga Cinar
cinarto@itu.edu.tr

Alper Ilki
ailki@itu.edu.tr

[1] Civil Engineering Department, Istanbul Technical University, Maslak, Istanbul 34469, Turkey

[2] Payten Teknoloji A.S., ITU ARI Teknokent, Istanbul 34469, Turkey

[3] Disaster Management Institute, Istanbul Technical University, Maslak, Istanbul 34469, Turkey

Published online: 23 May 2025

These images include surrounding household objects, such as furniture, appliances, household items, and personal belongings, which may pose more challenges for computer vision but are more realistic. This study differs from previous ones that focused on using deep convolutional neural networks to detect damage while ignoring irrelevant objects. Instead, it aims to address the more challenging task of identifying damage in post-earthquake images that are cluttered and diverse. Therefore, the proposed model in this study is better suited for real-world scenarios compared to previous models.

2 Related Works

Vision-based structural surveillance and evaluation are rapidly developing, and various structural engineering applications are reaping the benefits of this emerging technology. Several researchers developed and used image processing techniques, such as edge detection algorithms (e.g., Sobel and Canny), Haar and fast Fourier transforms, filter-based crack detectors, and cluster formation algorithms, to evaluate buildings and infrastructures and provide an automated crack detection system. On concrete bridges, Tung et al. [2] developed a stereo camera imaging system to automate bridge crack inspection. Abdel-Qader et al. [3] applied the Haar, fast Fourier, and well-known Fourier transforms edge detection algorithms, Sobel, and Canny, on concrete bridges to detect cracks. Likewise, Chen et al. [4] applied edge detection methods to multitemporal images from cracked concrete surfaces. Sinha and Fieguth [5] presented a filter-based crack detector for buried concrete pipes. . Abdel-Qader et al. [6] utilized principal component analysis to detect cracks in concrete decks. Yu et al. [7] proposed an integrated crack detection system for a reinforced concrete tunnel, while Oh et al. [8] developed an automated bridge crack inspection system. Yamaguchi et al. [9] proposed a crack detection algorithm established on cluster formation with the neighborhood connectivity of pixels. Zhu and Brilakis [10] detected concrete columns with edge detection and Hough transformation, while in a separate study, they utilized machine vision to investigate concrete surfaces for air pockets and discoloration [11]. Furthermore, they identified cracks and measured crack properties on images collected from the Haiti earthquake on January 12, 2010, using a percolation-based algorithm [12]. In another study, Gul and Catbas [13] used the Sobel edge detection algorithm to assess the lubrication level of an open gear of a movable bridge. Additionally, Higgins and Turan [14] used image processing and close-range photogrammetry techniques to quickly collect gusset plate geometry used in connection evaluation and rating steel gusset plates. Kim et al. [15] developed a stereo vision system utilizing cameras with two different focal lengths, enabling effective simultaneous crack localization and characterization, which was field-validated on an in-service bridge. Wang et al. [16] proposed a framework procedure based on the automatic extraction of visible geometry using computer vision and unmanned aerial vehicles, combining the extracted data to perform seismic risk assessment of bridges. The sliding window approach was used by Yeum and Dyke [17] to identify fractures close to bolts on steel bridge members. Despite promising results in all these efforts using traditional image processing techniques such as pixel-level edge detection, filtering, and other operations, these techniques rely heavily on pre-defined rules and mathematical models to process and analyze images. This reliance can result in limitations in handling complex and diverse datasets, making them less effective in real-world applications such as crack detection on structural elements. Zhang et al. [18–20] propose noncontact methods for structural health monitoring, including impact load identification using optical flow, automated vision-based displacement tracking for bridges and moving force identification via visual perception, offering efficient and cost-effective solutions without manual calibration or traditional sensors.

On the other hand, machine learning algorithms have an advantage over traditional image processing techniques in their ability to automatically extract many low-level features by leveraging large datasets, making them ideal for handling complex and diverse image datasets.

As the number of datasets and images in these datasets increase, the machine learning-based approaches for feature detection on images have also increased remarkably. Sensing technologies combined with machine learning algorithms are becoming invaluable, robust detection tools for inspection and evaluation. Several researchers used shallow machine learning methods, such as support vector machines (SVM), k-nearest neighbors (KNN), and artificial neural networks (ANN), to improve the performance of image processing techniques in the inspection and evaluation process. Furuta et al. [21] established a four-layer neural network, TAM network (Topographic Attentive Mapping network), to determine the damage state of reinforced concrete bridge decks from their images, whereas Nishikawa et al. [22] utilized genetic programming to generate image filters to detect cracks on concrete surfaces. Similarly, O'Byrne et al. [23] employed SVM classification models to identify the damage to infrastructures by including textural features on the images as additional features, while Lee et al. [24] used ANN to measure cracks' width, length, and orientation on concrete surfaces, and Jahanshahi et al. [25] introduced a pattern recognition concept to detect and quantify cracks on concrete surfaces using SVM and NN. Additionally, O'Byrne et al. [26] presented a new regionally enhanced multiphase segmentation technique and employed SVM to classify corroded surfaces of infrastructures. Similarly, Gul et al. [27] assessed the lubricant level of open gears of movable bridges

by using a fuzzy NN and Sobel edge detection, and Cha et al. [28] utilized the Hough transform along with SVM to inspect the bolts. Nonetheless, since all of these studies rely on shallow machine learning algorithms, their capacity to extract more informative features is limited, resulting in poorer performance than deep learning methods.

Currently, deep learning methods such as convolutional neural networks (CNN) have revolutionized the field of computer vision. Unlike shallow networks, which typically have only one or two hidden layers, CNNs have many hidden layers that allow them to learn more abstract and high-level features from images, such as edges, shapes, and objects, making them more robust and accurate predictions. Thus, CNN became one of the first choices in image recognition and processing due to its capability to extract discriminative features, making them well suited for detecting thin cracks that may be difficult for humans or traditional image processing techniques to identify. Cha et al. [29] proposed a CNN and compared its performance to those utilizing Sobel and Canny edge detection on concrete cracks in this context. They observed that CNN is superior to Sobel and Canny, especially when the cracks are thin and under poor lighting conditions. Traditional CNN-based damage detection methods are limited in that they cannot provide the precise location of the damage with an image. The emergence of the region-based CNN (R-CNN) approach has provided a solution to this limitation for detecting damage, as it divides the image into several regions and analyzes each region to identify the target object. This approach has gained popularity in recent years, as evidenced by its successful application in detecting various types of damage, such as concrete cracks, medium- and low-level corrosion on steel sections, corrosion on bolts, and steel delamination, as demonstrated by Cha et al. [30] through the use of Faster R-CNN. Likewise, Xue and Li [31] used a fully convolutional network to detect defects in concrete tunnel linings and a region-based fully convolutional network to locate the cracks. Dung et al. [32], Alipour et al.[33], and Yuan et al. [34] employed deep convolutional networks for crack detection, and several other studies in the literature have utilized different deep learning architectures, including AlexNet ([35]), VGGNet ([36]), and SegNet ([37]), for detecting cracks. For example, Yeum et al. [38] focused on classifying images of collapsed buildings and detecting concrete spalling in the post-disaster images with Alexnet, while Gao and Mosalam [39] utilized VGGNet to classify images into component types (*beam/column or wall*), spalling condition (*spalling or no spalling*), damage level (*no damage, minor damage, and moderate to heavy damage*), and damage type (*no damage, flexural damage, shear damage, and combined damage*). To effectively leverage the semantic information hidden in the bridge images, Liang [40] implemented SegNet along with VGG-16 and R-CNN to see bridge system-level failure, bridge components, and local-level component damages. Likewise, Zhang et al. [41] used a deep convolution semantic segmentation network to detect cracks on concrete surfaces and compared the proposed method with Sobel, Canny, Roberts, and Prewitt edge detection algorithms. Using the deep CNN, Gao and Mosalam [42] gathered 36, 413 images and classified them in scene level (*pixel level, object level, and structural level*), damage state (*damaged and undamaged*), spalling condition (*spalling and no spalling*), material type (*steel and other materials*), collapse mode (*noncollapse, partial collapse, and global collapse*), component type (*beam, column, wall, and others*), damage level (*minor, moderate, and heavy*), and damage type (*flexural, shear*, and combined). Moreover, Karaaslan et al. [43] introduced a methodology for concrete defect assessment in infrastructures, while a VGG-16 architecture was used for damage detection, and a modified SegNet was used for damage segmentation. Meanwhile, Dais et al. [44] investigated the detection and segmentation of cracks on masonry walls, comparing the performance of multiple CNNs for this task. Chen et al. [45] investigated the utilization of deep CNNs for categorizing and identifying images of damaged bridge structures. Dogan et al. [46] employed VGGNet to distinguish corrosion damages from earthquake-induced damages for reinforced concrete structures.

Previous studies on damage detection using machine learning have typically focused on structural images that only contain damages without any other unrelated objects in the background, such as furniture or household items, as detecting damages in such images using machine learning might be relatively easier. However, it is worth noting that images including unrelated objects on the images are more realistic, as they may be captured by nonexperts following earthquakes. Additionally, using such images poses a more significant challenge as the types of objects and their positions can vary widely between different images. Despite these challenges, this study intentionally uses such images as input to train the deep learning model to generate the classifier model, which also localizes the damages in the structures using gradient-based sensitivity methodologies. Moreover, evaluating the safety of buildings by engineers, particularly after a large-scale earthquake, is time-consuming. Consequently, there is a need for machine learning-based auxiliary technologies that can reduce the time and number of engineers required for post-earthquake damage evaluations. This study automatically classifies, localizes, and prioritizes post-earthquake damage to save valuable time after an earthquake.

3 Post-earthquake Damage Surveys and Categories

After an earthquake, post-earthquake site investigation and damage assessment have always been challenging for engineers, authorities, and decision-makers. With regard to decreasing unnecessary financial losses associated with the repair, replacement, and retrofit efforts, disruption of production and services, and occupant relocation, a damage assessment has a crucial role [47]. Therefore, considering the number of structures needed to be inspected and to increase the resilience of the communities in earthquake-vulnerable regions, the assessment procedures used in a post-earthquake survey need to be easy to execute, rapidly applicable, and straightforward. Various parameters such as structural systems, design philosophy, construction practice, and fault characteristics play a role in the damage created in an earthquake. Thus, various countries, including Japan, Italy, the USA, and Turkey, have developed and widely employed distinct damage scales to quantitatively assess earthquake damage. In this paper, the Turkish post-earthquake damage assessment protocol is employed.

After the 1999 Kocaeli Earthquake, the Turkish government adopted a new policy that made it mandatory for homeowners to have earthquake insurance. Turkish Catastrophe Insurance Pool (TCIP) was founded in 2000, and the first version of an assessment methodology was established in 2002 [47]. Since its initial development and with the experience of several earthquakes, a few adjustments and improvements have been adopted, and the latest version of the TCIP damage assessment protocol has been developed and also adopted for post-earthquake damage evaluation recently. Ilki et al. [47] proposed two inspection procedures, a rapid and a detailed one, that determine the safety of the building based on the residual energy dissipation capacity of the structural components. According to the evaluation, six outcomes are obtained: *undamaged, slightly damaged, moderately damaged, heavily damaged, building to be urgently demolished*, and *collapsed*. The damage evaluation aims to determine the long-term use of the building, suggest repairing, retrofitting, or demolishing it, and estimate the needed financial contribution.

As per the Turkish post-earthquake damage assessment procedure, in this paper the six categories mentioned earlier are reorganized into two groups: *nonstructural damage* and *structural damage*. This reorganization aims to give priority to structurally damaged buildings and efficiently allocate limited manpower. Undamaged and some slightly damaged classes are grouped under the nonstructural damage label, while moderately damaged, heavily damaged, building to be urgently demolished, and some slightly damaged classes are assembled under the structural damaged label.

In the nonstructural damage labeled images, vertical and horizontal structural members, beams, and columns exhibit no cracks. Any damage caused by the earthquake occurs in infill and partition walls, such as shear cracks, flexural cracks, separation of infill and frame, corner crushing, and plaster spalling. Nonstructural damage labeled buildings, where all of the images from that building are nonstructural damage labeled, lateral load-bearing capacity is close to the pre-earthquake lateral load-bearing capacity, and occupancy is permitted.

In the structural damage tagged images, earthquake damage is observed in beams and columns, such as flexural cracks, shear cracks, spalling of concrete cover, longitudinal rebar buckling, and core crushing. In structural damage-marked buildings, where at least one image from the building is structural damage labeled, lateral load-bearing capacity might significantly decrease; therefore, structural engineers should assess its post-earthquake lateral load-bearing performance. These buildings should be prioritized in the pre-assessment to speed up the evaluation procedure. Earthquake damage observed in beams, columns, and walls is depicted in Fig. 1, and some samples of *structural damage* and *nonstructural damage* labeled images are shown in Fig. 2.

4 Dataset

On January 24, 2020, at 8 : 55 pm local time (5 : 55 pm UTC), an earthquake with a moment magnitude of 6.8 struck Sivrice, district of Elazığ, Turkey, with a focal depth and epicenter coordinates of 8.06 km, 38.3593° North, and 39.0630° East, respectively [48]. After the earthquake, government officers conducted building-by-building damage assessments. Based on the available records from the CSB, a total of 61, 152 buildings were inspected. Of these, 263 buildings collapsed, 7, 698 were severely damaged, and 1, 540 were classified as moderately damaged [49]. All of these buildings are either masonry or reinforced concrete buildings. During the assessment process, the inspectors took several images with consumer-grade cameras, tablet computers, or cellphone cameras, i.e., off-shelf cameras were used to take pictures. These images from concrete structures, except those from collapsed or undamaged units, were used as a database in this study.

As this study aims to classify nonstructural and structural damages, only those images that fit into these two categories were utilized to create a dataset for the experiments. The dataset consists of 9, 645 images taken from 1, 789 reinforced concrete buildings, which include 4, 235 separate units. All images were used as-is and were not subjected to adjustments such as cropping, stretching, sharpening, or smoothing. Some pictures included a cluttered front or back-

Fig. 1 a) Flexural crack. b) Shear cracks. c) Longitudinal rebar buckling. d) Core crushing. e) Cover Spalling. f) Plaster spalling. g) Separation of infill wall and frame. h) Frame and wall separation and shear cracks on infill walls. i) Sliding shear crack. j) Shear cracks and plaster spalling on infill walls

Table 1 Number of structural and nonstructural images for training, validation, and testing categories

Category	Structural	Nonstructural	Total
Training	2,151	5,529	7,680
Validation	253	707	960
Testing	277	728	1,005
Total	2,681	6,964	9,645

ground (e.g., shadows, house owners, pieces of furniture, household goods, etc.) with a broadly varying field of view, camera angle, distance from objects, and emphasized targets. The resolutions of the images range from 448×448 to 1920×1080, with aspect ratios ranging from 1.00 to 1.15. The images were labeled through a labor-intensive procedure by structural engineers with experience in post-earthquake assessment campaigns, with 6,964 images labeled as nonstructural damage and 2,681 images as structural damage. The dataset was divided into training, validation, and testing sets using an 80/10/10 ratio, with 7, 680/960/1, 005 samples for each set, as shown in Table 1.

Several sample images from our dataset are presented in Fig. 2. Regarding a computer vision problem, the difficulty in separating structural damage from nonstructural is that the distinction lies in the whereabouts of the cracks. For instance, in both Fig. 2(a) and (e), there are diagonal cracks, whereas, in the latter, they occur on the beams, making it structural-type damage. Another challenge is the existence of unrelated objects appearing in the images, such as various stored items, including plastic-wrapped goods and boxes in Fig. 2(h), a bed in (b), a pipe in (f), a picture frame in (c), etc. Because of the moderate size of the dataset, the features generated due to the unrelated objects can potentially suppress the relevant features required to discriminate the classes. Even worse, under some circumstances, the model can rely on only these secondary objects to make a decision about the damage level, which is an undesirable behavior because such household objects do not obviously play a role in structural damage classification. Additionally, the images taken from the field might come in two different forms: *object level* and *pixel level* [39]. The majority of the images in our dataset belong to the *object* category at the scene level. For example, in Fig. 2, all the images, except for (g), fall under the category of *object-level* scenes due to their inclusion of other unrelated objects. Fig. 2(g) falls into the category of *pixel level* as it is a close-up shot of a target object that needs to be classified. In this study, it is decided to include pixel-level images in the dataset as well, considering that in real-world scenarios, engineers would probably capture scenes at both the object and pixel level when utilizing the suggested machine learning model.

5 Proposed Structural Damage Classifier

In this study, the state-of-the-art ConvNeXt model family, dubbed as the next generation of classical convolutional neural networks [50], was used to classify post-earthquake images into structural and nonstructural classes. Transfer learning was performed to adapt the existing ConvNext mod-

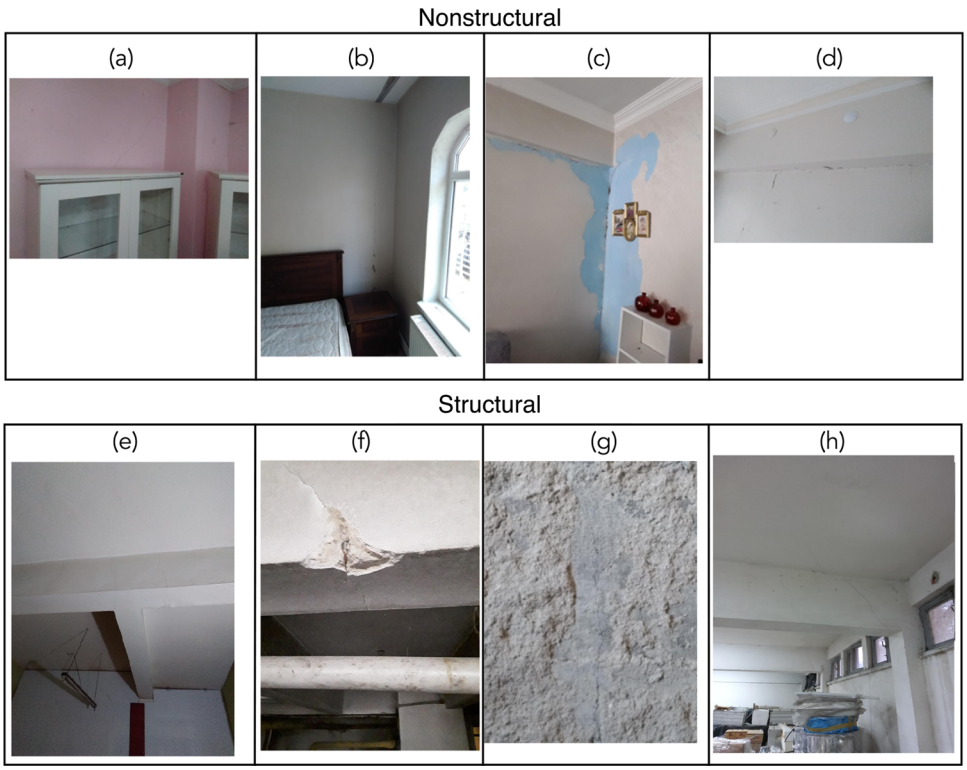

Fig. 2 Multiple image samples in our dataset for structural damage type and nonstructural damage type

els to the dataset, therefore achieving an accurate and reliable structural damage classifier. The reliability of the predicted results was evaluated using GradCAM [51], an abbreviation for gradient-weighted class activation mapping, which is a technique for making CNNs more transparent by visualizing the parts of a given input image that are important for predictions made by the network. This approach helps to interpret the decision-making process of CNNs, particularly in tasks such as image classification. With this approach, the location of the damages was also identified.

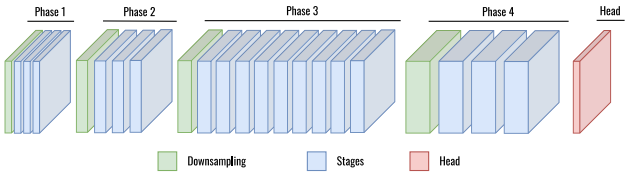

Fig. 3 Multiphase design of ConvNeXt architecture. In each phase, there is a downsampling step followed by a number of stages

6 Deep Learning Architecture: ConvNeXt

The ConvNeXt model is a contemporary and streamlined adaptation of the well-known ResNet architectures developed by He et al. [52]. It employs a multiphase design approach, illustrated in Fig. 3, with each phase comprising a downsampling step (green) followed by the repetition of identical stages (blue). The model is finalized with a *head* layer.

Each phase in the ConvNeXt model begins with a downsampling block that reduces the spatial dimensions of the input images and increases the spectral dimension, i.e., the number of channels/features. In the first phase, downsampling reduces the spatial dimension by a factor of four, while in the second, third, and fourth phases, a twofold downscaling is applied. The number of spectral channels in the ConvNeXt model is determined by the downsampling layers. For example, in ConvNeXt Tiny, a variant of the ConvNeXt family, the downsampling layers generate 96, 192, 384, and 768 spectral features in each phase, respectively.

Upon completion of all phases, the network generates an image with a low resolution in the spatial domain but a high resolution in the spectral domain. The resulting image is classified in the head block by mapping all the features to corresponding class labels. The mapping begins with a global average pooling, which takes each channel's average and eliminates the spatial information. Subsequently, a normalization step is performed, followed by the utilization of a dense layer to obtain the final outputs of the network. It is worth mentioning that when dealing with multiple classes, it is common to apply a softmax activation to the outputs to generate class probabilities. However, in this particular study with two classes, sigmoid activation was chosen instead of softmax.

The ConvNeXt models' depth and complexity are determined by the number of stages and the number of features generated in each stage, resulting in various ConvNeXt variants, as indicated in Table 2. In all ConvNeXt variants, the feature dimension doubles at each phase, leading to an increase in the number of trainable parameters as the model progresses toward the final stages. For instance, in the ConvNeXt Tiny model, the final stage produces 768 features, resulting in a total of 28 million trainable weights throughout the entire model. A detailed breakdown of the trainable parameters and the number of spatial/spectral features in each phase is given in Table 3.

7 Training Strategy

A successful deep learning application requires careful planning and a well-defined training strategy. In this study, we followed an incremental development strategy which can be divided into four phases: (§8.1) baseline selection (§8.2) model selection (§8.3) transfer learning and (§8.4) data augmentation and regularization. The choices in each phase are made according to the best practices in the literature as well as the authors' expertise. The performance in each phase is monitored by analyzing and changing one configuration aspect at a time, similar to the ablation studies [53]. In baseline selection (§8.1), we selected a small pre-trained ConvNeXt model for starters. In the next phase (§8.2), we tested several other deep learning models under the same conditions to select the "best" model. In the transfer learning phase (§8.3), we tested different fine-tuning approaches to choose the optimal number of layers to be frozen in fine-tuning. Finally, in data augmentation and regularization phase (§8.4), we tested several different approaches including label smoothing, color jittering [54], random augmentation and erasing [55], MixUp [56], and CutMix [57].

8 Experimental Results

Separate experiments were conducted to gradually improve the accuracy and performance of our proposed approach in each step of the incremental development process, including baseline model, model selection, transfer learning, regularization, and data augmentation. The experiments and their outcomes are explained in the forthcoming sections and summarized in Table 6 and Fig. 7. Due to class imbalance in our dataset, the *balanced accuracy* metric was selected as the primary measure of accuracy. Additionally, other performance metrics such as precision, recall, f-score, and receiver operating characteristic (ROC) curves were provided for the initial experiment only. All experiments were repeated ten times, employing different random realizations of training and validation datasets to ensure more reliable results, and averaged metrics were reported. Hyperparameter tuning and model selection were performed on the validation dataset. The average balanced accuracies of each experiment, evaluated on the training, validation, and testing sets, are also presented in Table 6 and Fig. 4.

8.1 Baseline Model

As the first step in our development strategy, the pre-trained ConvNeXt Tiny model was chosen as the baseline for our structural damage classifier due to its relatively small size and moderate performance. This model allows us to conduct numerous experiments quickly compared to larger models without significantly sacrificing accuracy. For the initial weights and biases, pre-trained weights provided by the ConvNeXt developers were used. These weights were fully utilized during the fine-tuning process. The original hyperparameters used to generate these pre-trained weights are detailed in Table 5 of Liu et al. [50]. Because the number of classes in our study differs from those in the ImageNet classification, we did not transfer the weights in the head layer of the model. Instead, we used the Xavier initialization technique to generate random weights for the nontransferred layers [58].

For training the ConvNeXt Tiny model, the Elazığ dataset was used. The images in the dataset were scaled to 256 by 256 via bicubic interpolation, center cropped to 224 by 224, then normalized using ImageNet statistics. 80% of the dataset is used for fine-tuning, 10% for validation, and 10% for the testing test leaving 7680/960/1005 samples in each set. The number of epochs is set to 3 with a batch size of 64. Training and validation splitting is repeated ten times with different random realizations. The model was fine-tuned on the training dataset and evaluated on each fold's training, validation, and testing datasets. The classification performance of the starting base is provided in Fig. 4 as a collection of receiver operating characteristic (ROC) curves. The results demonstrate that the baseline model is effectively trained on the training set, as evidenced by the training ROC curves tending toward the upper left region, with an average area under the curve (AUC) of approximately 0.99. On the other hand, the average AUC drops to 0.93 levels for validation and testing sets, which is an expected behavior for supervised learning. The similarity of ROC curves on validation and testing is a good indication that the validation/test split is performed properly and that both datasets exhibit similar characteristics.

Two confusion matrices were computed for each fold, one for training and one for validation. Table 4 presents an example confusion matrix where positive and negative classes represent structural and nonstructural damages, respectively. The overall accuracy, defined as the ratio of correct predictions to total predictions, is a common metric for evaluating

Table 2 Name of model variant, number of stages and features in each phase as well as trainable parameters (millions) of different ConvNeXt variants

ConvNeXt Variant	#Stages in phase	#Filters in phase	#Parameters (millions)
Tiny	[3,3,9,3]	[96,192,384,768]	28
Small	[3,3,27,3]	[96,192,384,768]	50
Base	[3,3,27,3]	[128,256,512,1024]	89
Large	[3,3,27,3]	[192,384,768,1536]	198

Table 3 Breakdown of ConvNeXt Tiny: the shapes of features are represented as $W \times H \times C$, where W, H, and C correspond to width, height, and channel size, respectively. The number of trainable parameters of each processing unit is given in the Weights+Bias column. The Repeat column represents how many times stages are stacked on top of each other. Note that there is no repetition for the downsampling and head layers

	Preprocessing	Input $W \times H \times 3$	Output $224 \times 224 \times 3$	Weights+Bias 0	Repeat $\times 1$	Total 0
Phase 1	Downsampling	$224 \times 224 \times 3$	$56 \times 56 \times 96$	4,896	$\times 1$	4,996
	Stage	$56 \times 56 \times 96$	$56 \times 56 \times 96$	79,296	$\times 3$	237,888
Phase 2	Downsampling	$56 \times 56 \times 192$	$28 \times 28 \times 192$	74,112	$\times 1$	74,112
	Stage	$28 \times 28 \times 192$	$28 \times 28 \times 192$	306,048	$\times 3$	918,144
Phase 3	Downsampling	$28 \times 28 \times 192$	$14 \times 14 \times 384$	295,680	$\times 1$	295,680
	Stage	$14 \times 14 \times 384$	$14 \times 14 \times 384$	1,201,920	$\times 9$	10,817,280
Phase4	Downsampling	$14 \times 14 \times 384$	$7 \times 7 \times 768$	1,181,184	$\times 1$	1,181,184
	Stage	$7 \times 7 \times 768$	$7 \times 7 \times 768$	4,763,136	$\times 3$	14,289,408
	Head	$7 \times 7 \times 768$	1	2,305	$\times 1$	2,305
					Total	27, 820, 897

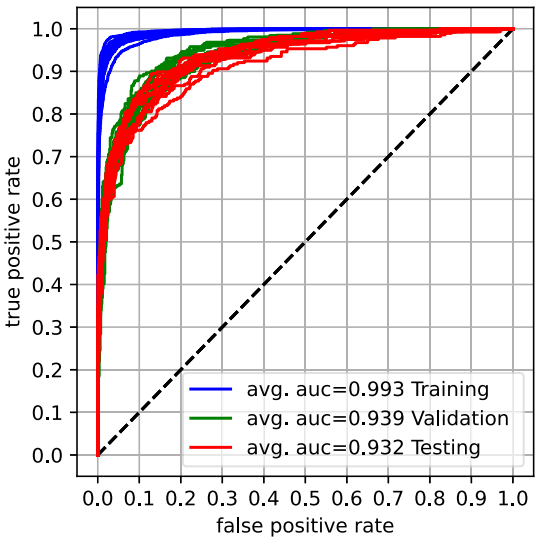

Fig. 4 ROC curves obtained by executing the baseline model and parameters specified in §8.1. The experiment was repeated 10 times, and ROC curves were computed for each training, validation, and testing dataset. The average area under curves is displayed in the legend

confusion matrices. In this example, the overall accuracy is 89%.

Additionally, the model's class-wise performance can be assessed by calculating the true positive rate (sensitivity) and true negative rate (specificity). The true positive rate indicates the model's ability to correctly identify structural damage, while the true negative rate measures its ability to correctly identify nonstructural cases. For the confusion matrix in Table 4, the true negative rate (93%) is higher than the true negative rate (78%), suggesting that the model is more effective at detecting nonstructural cases.

Table 4 A sample confusion matrix from one validation fold for the Elazığ dataset. Positive and negative classes represent structural and nonstructural damage, respectively

Actual / Predicted	Negative (Nonstructural)	Positive (Structural)
Negative (N)	661 (TN)	46 (FP)
Positive (P)	56 (FN)	197 (TP)

Other performance metrics, such as the F1-score, balanced accuracy, the area under the curve (AUC), and many others are also provided in Table 5. Given the class imbalance of the dataset, special emphasis is placed on balanced accuracy, which is the average of the true positive rate and true negative rate. In the example confusion matrix, the balanced accuracy is 86%, slightly lower than the overall accuracy of 89%. Throughout the remainder of the manuscript, balanced accuracy will be used as the primary performance metric for evaluation.

Table 5 Performance result of tenfold cross-validation obtained by executing starting base configuration. Each row corresponds to a different metric, whereas the columns provide some statistics, where **tpr:** true positive rate; sensitivity, recall, hit rate; **tnr:** true negative rate, specificity, selectivity, **fnr:** false negative rate, miss rate; **fpr:** false positive rate, fallout; **fdr:** false discovery rate; **ppv:** positive predictive value, precision; **npv:** negative predictive value; **for:** false omission rate; **lr+:** positive likelihood ratio; **lr-:** negative likelihood ratio; **pt:** prevalence threshold; **ts:** threat score, critical success index; **ba:** balanced accuracy; **acc:** accuracy; **pre:** prevalence; **auc:** area under the curve; **f1:** F1 score;

	# folds	mean	std	min	25%	50%	75%	max
tpr	10	0.710	0.110	0.504	0.658	0.692	0.775	0.892
tnr	10	0.942	0.034	0.863	0.931	0.942	0.962	0.987
fnr	10	0.290	0.110	0.108	0.225	0.308	0.342	0.496
fpr	10	0.058	0.034	0.013	0.038	0.058	0.069	0.137
fdr	10	0.165	0.063	0.060	0.128	0.173	0.191	0.293
ppv	10	0.835	0.063	0.707	0.809	0.827	0.872	0.940
npv	10	0.894	0.038	0.830	0.871	0.890	0.920	0.956
for	10	0.106	0.038	0.044	0.080	0.110	0.129	0.170
lr+	10	15.751	8.964	6.506	11.294	12.362	16.891	38.161
lr-	10	0.305	0.108	0.125	0.241	0.326	0.354	0.503
pt	10	0.213	0.038	0.139	0.196	0.221	0.229	0.282
ts	10	0.613	0.056	0.488	0.600	0.609	0.649	0.690
ba	10	0.826	0.040	0.745	0.810	0.820	0.854	0.878
acc	10	0.876	0.016	0.847	0.867	0.877	0.884	0.899
pre	10	0.280	0.014	0.264	0.268	0.279	0.289	0.306
auc	10	0.939	0.010	0.927	0.931	0.938	0.947	0.954
f1	10	0.758	0.044	0.656	0.750	0.757	0.787	0.817

8.2 Model Selection

The primary objective of the baseline model is to establish a reference accuracy for the simplest form of the proposed approach. In the subsequent step, alternative deep learning models are explored. As part of this exploration, several pre-trained convolutional neural networks, including all variants of ConvNeXt available in the *TensorFlow* framework, were tested. These models were evaluated under identical hyperparameters and preprocessing settings to ensure a fair comparison. All pre-trained weights were transferred and fine-tuned, and multiple performance metrics and training duration were assessed for each model. The performance of the models was compared using balanced accuracy and training speed, presented in a scatter plot in Fig. 5.

Based on the results from Fig. 5, it can be seen that all ConvNeXt variants, except for the small variant, exhibit superior performance in terms of balanced accuracy compared to other models. Notably, the ConvNeXt models achieve over 80% balanced accuracy in just three epochs of fine-tuning. While accuracy is an important aspect, the inference speed of neural networks is also crucial, especially in real-time production environments. For instance, in live camera feed applications, the model's processing speed should ideally exceed 24 frames per second (fps). Considering both accuracy and speed, ConvNeXt Tiny emerges as the optimal neural network architecture for the remainder of this study. This choice is particularly significant for ensuring efficient inference when deploying the model as a smartphone application.

8.3 Transfer Learning

In this study, the parameters of ConvNeXt models pre-trained on millions of images in ImageNet were transferred. The decision to select ImageNet as the source domain was based on the intuition that ImageNet encompasses thousands of household objects, as well as structural and nonstructural components that are also present in our structural image dataset. As a result, it is anticipated that the transferred weights will include a sufficient number of low-level features that can aid in distinguishing these components.

Transfer learning comes with its dilemma regarding the number of layers that should be transferred, the number that should be fine-tuned, and the ones that should be frozen. Fine-tuning all the transferred weights requires a high computational cost with the benefit of higher model flexibility. On the other hand, freezing all the transferred weights reduces the computational cost of fine-tuning but results in a more rigid model. This trade-off must be considered to maintain a balance between the model's performance and cost. An experimental analysis of transfer learning with three strategies, namely *default initialization*, *transfer weights*, and *transfer and freeze weights* was conducted to determine this trade-off optimally. The results of this analysis are depicted in Fig. 6, where the strategies are represented by gray, blue, and light blue colors, respectively. Figure 6(a) illustrates the different combinations of transfer learning strategies for ConvNeXt. The *default initialization* strategy represents the absence of transfer learning, where the weights are randomly initialized before training. The *transfer weights* strategy indicates the transfer of weights that are

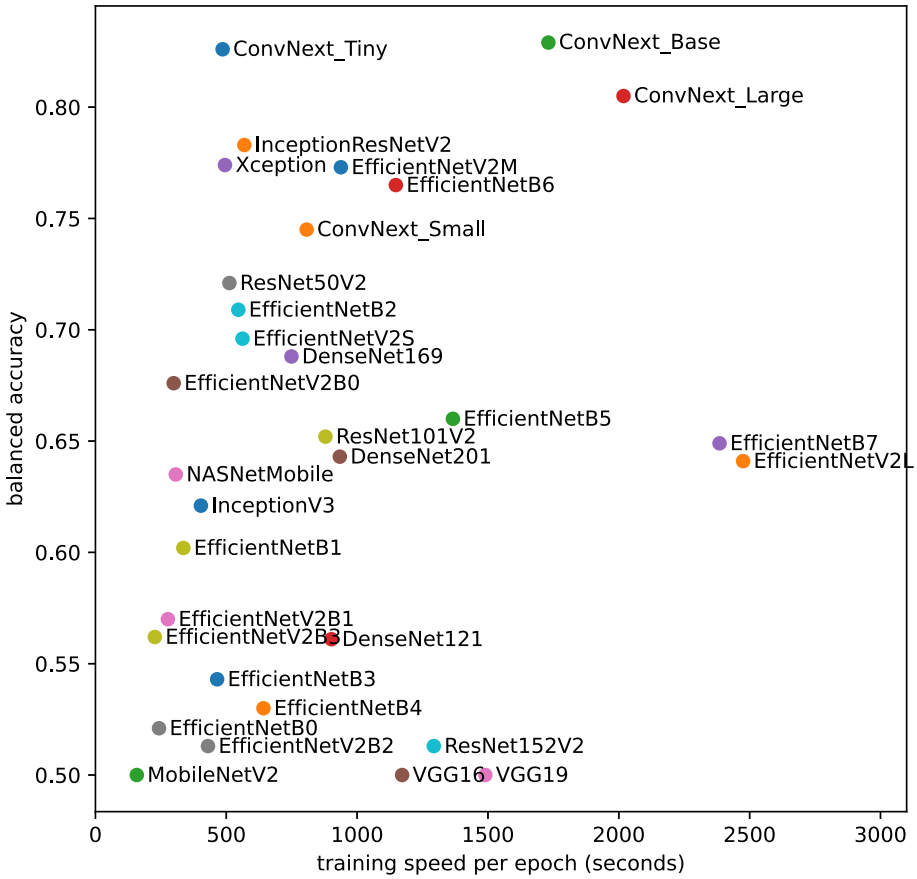

Fig. 5 Balanced accuracies and training speeds of various deep learning methods, including ConvNeXt family

allowed to be updated during training. Lastly, the *transfer and freeze* strategy involves transferring weights but keeping them frozen, preventing updates during backpropagation.

The balanced accuracies achieved on the training and validation datasets, as well as the corresponding training times, are shown in Fig 6(b,c,d), respectively. Each cell in these matrices displayed is associated with the outcomes of the transfer learning strategy conducted in Fig 6(a). For instance, the balanced accuracy on the training dataset (64.1%) corresponds to the scenario of transferring one layer and freezing none. Likewise, the balanced accuracy of 78.8% on the validation dataset corresponds to the scenario where all the layers are transferred but frozen. Moreover, based on the validation accuracies presented in Fig. 6(c), it can be observed that transferring more layers leads to higher accuracy in the validation dataset, as evident from the increased accuracies. An immediate result of this observation is that one should transfer all layers, as indicated by their performances in the last column of the matrices. Once the decision is made to transfer all of the layers, another decision has to be made on which layers should be frozen. There are five different options: freezing none of the layers, only the first, the first two, the first three, or all of the layers. By examining the validation scores in the last column, it can be observed that freezing the first, second, and third layers yields higher validation performance compared to the other options. To make a final decision, the training times required for these strategies were also taken into consideration. Among all the scenarios, freezing the first three layers requires the least amount of training time. Therefore, this strategy is selected for the remaining part of the study. As a result of adopting this optimal transfer learning strategy, the average balanced accuracy on the validation set shows a significant improvement, rising from 80.0% to 85.4% balanced accuracy.

8.4 Data Augmentation and Regularization

8.4.1 Label Smoothing

As a next step of the development process, label smoothing, a regularization technique, was performed with $\alpha = 0.1$. The same transfer learning strategy was chosen in the previous section, and the ConvNeXt Tiny model was used in the experiments. As a result of label smoothing, average balanced accuracy on training, validation, and testing sets are increased to 99.6%, 87.3%, and 86.5%, respectively, as shown in Table 6. Therefore, label smoothing is consistently employed throughout the remainder of this study.

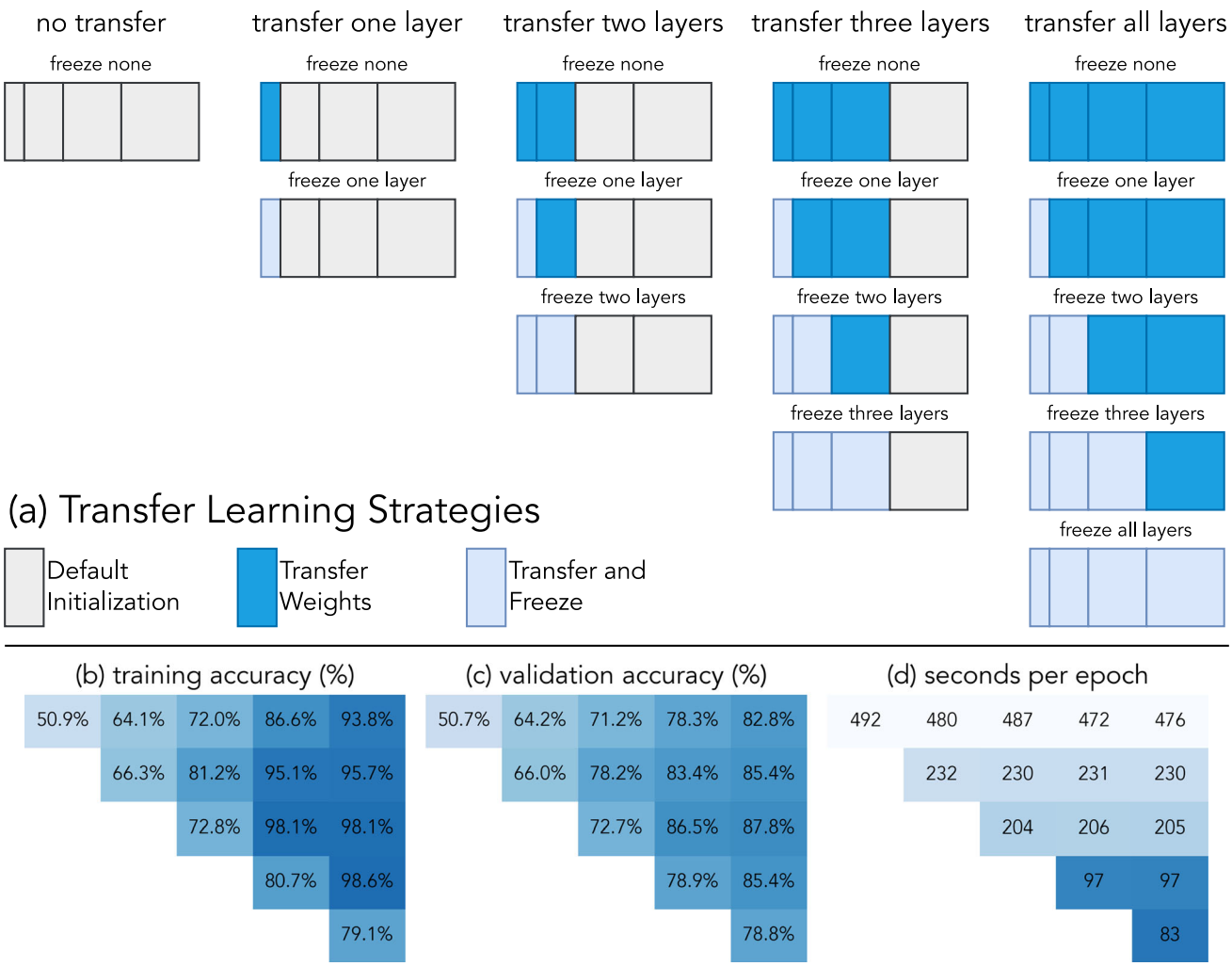

Fig. 6 (a) Transfer learning strategies, (b) training and (c) validation balanced accuracies, (d) training duration per epoch

8.4.2 Color Jittering

Apart from label smoothing, another form of implicit regularization, color jittering, was also carried out in the experiments. Brightness, saturation, and contrast were randomly changed using a common scale parameter of 0.20. To measure the effect of color jittering, the model, data, and all parameters except color jittering were kept the same, and ten random experiments were conducted. The average balanced accuracies obtained with color jittering are shown in Table 6. The results indicated that color jittering degrades the classification accuracy on all data splits. Hence, it is not applied in the remainder of this study.

8.4.3 Random Augmentation

The scarcity of training data has been recognized to have a negative effect on the performance of the structural damage classifier. To alleviate this effect and improve the classifier's performance further, the following set of random augmentations were applied: *identity, auto contrast, equalize, rotate, solarize, color, posterize, contrast, brightness, sharpness, shear X, shear Y, translate X, translate Y*. Table 6 shows the average balanced accuracies for training, validation, and testing sets: 90.4%, 87.8%, and 86.4%, respectively. Compared to the best performance obtained with label smoothing, validation accuracy is slightly improved, whereas testing accuracy is almost the same. The most significant change is the reduction in training performance without degrading validation and testing performance. This shows a regularization effect of random augmentations. Hence, despite the slight reduction in training performance, the decision has been made to retain the utilization of random augmentation for the remainder of the study.

Table 6 Summary of the experiments and their effects on the development process

Experiment	Balanced Accuracy trn/val/tst%	Observation
Starting Base	93.3/82.6/82.5	ConvNeXT Tiny w/o data aug.
Model Selection	93.3/82.6/82.5	ConvNeXT Tiny is the best model
Transfer learning	98.6/85.4/85.2	Transfer all, but freeze the first three
Label smoothing	99.6/87.3/86.5	0.1 smoothing is good
Color Jittering	99.3/86.6/85.8	no improvement
Random Augmentation	90.4/87.8/86.4	same accuracy with less memorization
Random Erasing	90.6/87.9/86.4	no improvement
MixUp and CutMix	69.3/83.6/83.1	worse performance

8.4.4 Random Erasing

In addition to random augmentation techniques, the effect of random erasing was also experimented. The default hyperparameters defined in TensorFlow implementation were used. While random erasing yields slight improvements in training and validation accuracies, as indicated in Table 6, it was deemed unfavorable to incorporate this technique for the remainder of the study due to the increased training time and the absence of significant performance advantages.

8.4.5 MixUp and CutMix

MixUp and CutMix were the last data augmentation techniques examined in this study. The default hyperparameters provided by the TensorFlow implementation were utilized for both methods. Since MixUp incorporates its own label smoothing mechanism, no additional external smoothing was applied. These two methods were sequentially applied during the experiments. However, the results presented in Table 6 indicate that both MixUp and CutMix resulted in a degradation of the classification performance. As a result, these techniques were not considered for further analysis in the study.

Figure 7 provides a comprehensive overview of the incremental steps carried out in this section as part of the development strategy. It also illustrates the training and validation performances attained at each step.

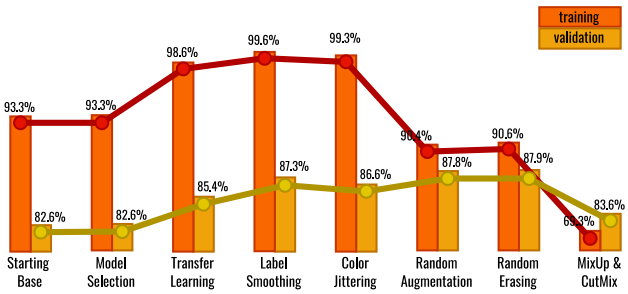

Fig. 7 Incremental development strategy used in this study

9 Model Inference

Upon completing the experiments in the development process, the final classification model is decided on the following configurations: ConvNext Tiny, data augmentation techniques with label smoothing, random augmentation, and transfer learning by transferring all the layers but freezing the first three. To demonstrate the model inference, we also created a publicly accessible user interface[1] that takes an input image, runs the model, and returns the structural damage assessment.

10 Explainability

It is important for a machine learning model to be accurate, but it is also important to understand how the model makes its predictions. To be specific, it was verified whether the model pays attention to the same regions as civil engineers or at least whether these regions contribute to the classification prediction. To accomplish this, various model-agnostic explainability tools are available, such as LIME (Local Interpretable Model-Agnostic Explanations) [59] and SHAP (Shapley Additive Explanations) [60], as well as model-specific methods that rely on class activation maps (CAM) [61], which are specifically designed for convolutional neural networks. The CAMs are much faster than model-agnostic approaches because they utilize the built-in activation values of CNNs to generate heatmaps. It should also be noted that these heatmaps provide valuable visualizations, enabling the identification of damage positions within the structure. As CAM and GradCAM are two widely used activation-based explainability techniques in the literature, both were utilized in this study.

[1] https://huggingface.co/spaces/hkayabilisim/deprem-hasar-tespit

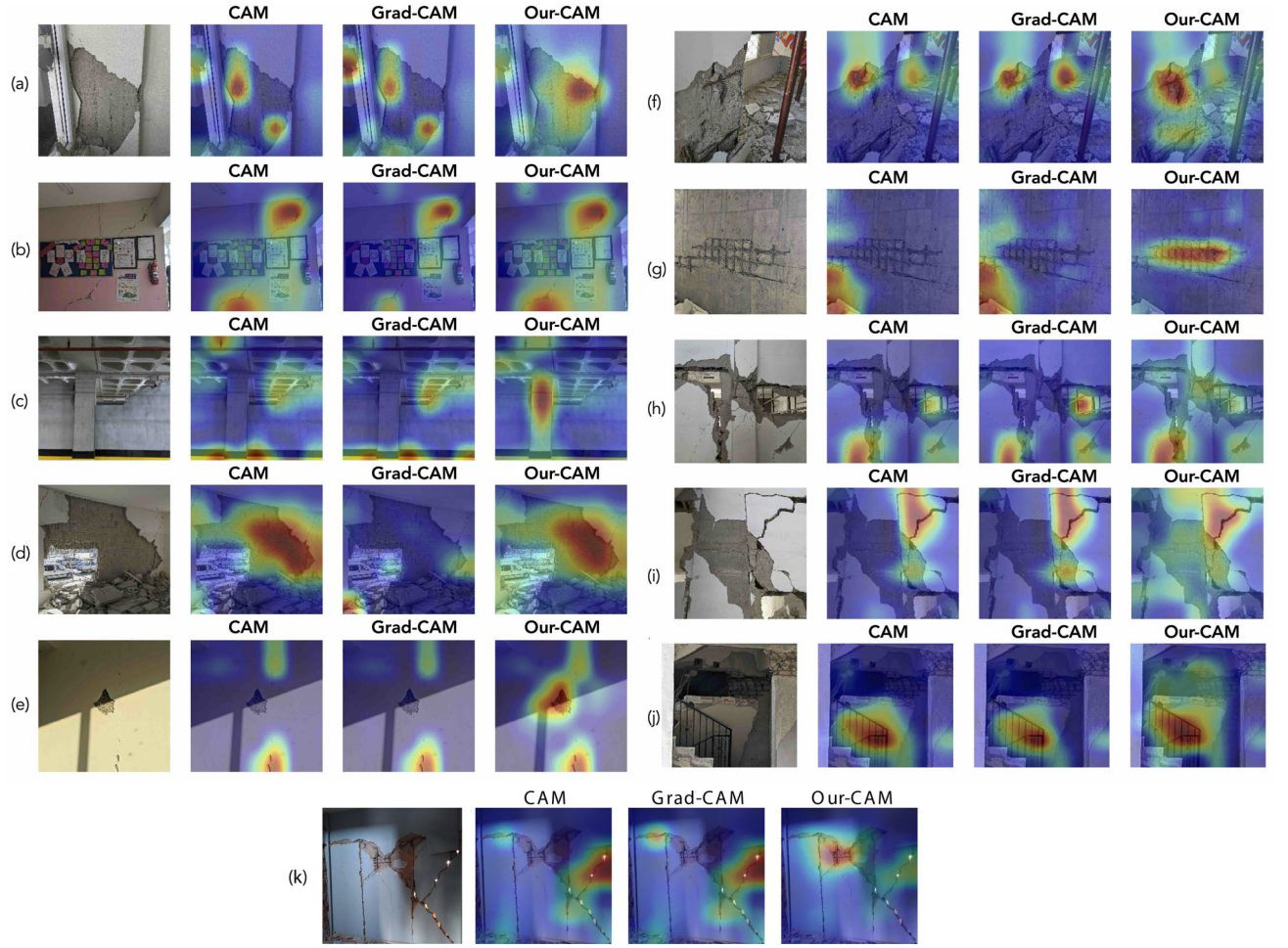

Fig. 8 Model inference results of the proposed approach. Classification of test images with their explanation using CAM, GradCAM, and our customized CAM method

The general approach of all CAM-based methods is to generate a heatmap based on the final activation values just prior to the head layer. In the ConvNext Tiny model, the final activations are stored in a $7 \times 7 \times 768$ tensor. The distinction between CAM-based methods lies in how they reduce the spectral channels. In traditional CAM, the global average pooling (GAP) technique is used to reduce the spectral channels, while GradCAM employs the gradient information of the output class to weight the feature maps. An alternative technique that is more faithful to the domain knowledge and, hence, more realistic and representative was introduced. Specifically, our custom CAM implementation leverages the final layer of the ConvNeXt model, wherein a summation of the absolute values of all spectral channels is performed to generate a 7×7 heatmap. This heatmap is mapped back to the initial image size, resulting in a clear and comprehensible visual representation that facilitates interpretation and comprehension. Fig. 8 shows samples of the prediction results of a new image data set collected from the recent Kahramanmaraş Earthquakes, which occurred in 2023. It also includes heatmaps generated through the use of CAM, GradCAM, and our customized CAM techniques. Based on the results in Fig. 8(a-k), the following can be drawn:

a) Although there are no cracks present in the column except for a significant plaster spalling, the image has been identified as showing structural damage. As a result, the nonstructural damage on the structural member has been incorrectly identified as nonstructural damage.
b) The image includes a very good example of a diagonal infill wall crack obstructed by obstacles. Despite the obstacles on the wall that are hindering the single continuous diagonal crack, it has been appropriately identified as nonstructural damage.
c) Partial out-of-plane infill wall failure, without any structural damage, is incorrectly classified as structural damage with a score of 0.501 (in terms of the score, 1.0 represents structural damage while 0.0 represents nonstructural damage).

d) Partial out-of-plane infill wall failure is correctly recognized as nonstructural damage.
e) Plaster failure with cracks on an infill wall is identified as structural damage with a score of 0.566.
f) The wavy surface of the radiator in the background makes it difficult to recognize the longitudinal rebar buckling and decreases the score to 0.357. It is detected as nonstructural damage.
g) Cover spalling in the weak axes of the shear wall due to pounding from an adjacent building is a highly uncommon damage mechanism of a shear wall and recognized as structural damage with a score of 0.935. It is worth noting that it is a very rare structural damage and not included in the training database.
h) Partial out-of-plane infill wall failure with structural damage is classified as nonstructural damage.
i) The nonstructural damage that occurred on the wall has concealed the structural damage at the column.
j) The systematically arranged iron bars of the staircase railing have prevented the detection of the severe damage that occurred in the columns, and it is recognized as nonstructural damage with a score of 0.202.
k) Structural and nonstructural damages are closely intertwined, and structural damage is correctly recognized with a score of 0.540.

As a result, the developed model yields quite good results. However, in cases where a partial infill wall collapses, systematically arranged repeated objects, and a structural member (with or without damage) is present in the same image with nonstructural damage, the model may produce inaccurate outcomes.

11 Generalization Capability

In machine learning, a crucial aspect is to ensure that the classification model developed by using images obtained from one region is capable of classifying images acquired from another region. The standard approach for assessing the generalization ability of such models involves evaluating their performance on unseen test samples. In accordance with this principle, the proposed trained model was used to classify test images obtained from the recent Kahramanmaraş Earthquakes, which occurred in 2023 Turkey. It should be noted that these images have never been used during the development and training phases of the proposed model to prevent evaluation bias. Table 7 shows the confusion matrix evaluated on the Kahramanmaraş dataset.

Based on Table 7, the overall classification accuracy is 90.7%, indicating satisfactory performance, even when the test images are acquired from a different region of Turkey.

Table 7 Confusion matrix on the Kahramanmaraş Earthquake dataset

Actual / Predicted	Negative (Nonstructural)	Positive (Structural)
Negative	True Negative	False Positive
71	68	3
Positive	False Negative	True Positive
70	10	60

These results suggest that the model performs better at correctly identifying negative samples (nonstructural-type damage) than positive samples (structural-type damage) as the true negative rate, 95.8%, is higher than the true positive rate, 85.7%. This observed phenomenon may be related to the fact that the proposed method was trained on a dataset characterized by an imbalanced class distribution, where the frequency of nonstructural samples significantly surpasses that of structural samples. Despite this, the proposed method achieves a balanced accuracy of 90.8%, indicating that the model is able to correctly classify a high percentage of both positive and negative samples. While false negatives (FN) can result in critical delays and safety risks by missing structural damage, false positives (FP) lead to unnecessary resource allocation, which, although less critical, can still have significant social and economic impacts, particularly in post-disaster settings. This trade-off underscores the importance of adjusting classification thresholds to minimize FN while considering the operational and social costs associated with FP.

12 Conclusions

Structural damage classification presents a unique challenge in computer vision, as it differs fundamentally from traditional image classification problems. Unlike typical image classification tasks, identifying damages, cracks, or other structural deficiencies in the images requires specialized knowledge and is often only recognizable by field experts.

Our study proposes a structural damage classifier that employs ConvNeXt, a state-of-the-art deep convolutional and residual neural network, to automatically evaluate post-earthquake images taken by field teams for structural damage assessment. The classifier was modified to differentiate between two damage types: structural and nonstructural. This modification could potentially enhance the speed of post-earthquake damage assessment. Transfer learning was utilized to fine-tune the existing ConvNeXt model with our dataset to achieve high accuracy and reliability. Furthermore, a customized CAM approach, a tool for explainable AI, was introduced to evaluate the reliability of predicted results as well as localize the crack's positions on the images. This

approach enabled us to determine the areas of the image where the machine learning model focuses on estimating the damage class. To train our model, the Elazığ dataset, which comprises 9645 images captured from 1789 reinforced concrete buildings, including 4235 distinct units, was utilized. All the images were used in their original form and were not subjected to any modifications such as cropping, stretching, sharpening, or smoothing. To enhance the performance of the structural damage classification while minimizing computational costs, various transfer learning strategies were implemented instead of training the deep learning network from scratch. Because our labeled dataset is smaller than those typically used in conventional computer vision problems, several data augmentation techniques were utilized to enhance the performance even further. Among all the strategies applied in this study, the classification model is fixed based on the following configurations: ConvNext Tiny, data augmentation techniques with label smoothing, random augmentation, and transfer learning by transferring all the layers but freezing the first three. The final model and configuration were tested on images acquired from the recent Kahramanmaraş Earthquakes. The outcomes demonstrated a consistent classification performance with domain knowledge, as well as a strong generalization performance on the Kahramanmaraş dataset. A publicly accessible user interface is also provided to run the model in inference mode. Currently, the model cannot determine the level of structural damage or repair costs without stereo camera disparity or at least four known real-world coordinates in the images. However, it serves as a preliminary base model and guiding tool for future studies. Furthermore, less experienced participants in post-earthquake damage assessments tend to classify all damages as structural; this model can serve as an auxiliary tool for these users. While the proposed model shows promising results, there are several limitations that need to be addressed. One key limitation is the exclusion of rare damage types from the training dataset, which may lead to misclassification or failure to detect less common structural failures. To address this, future work could focus on incorporating segmentation models that isolate structural elements from the background prior to classification. This approach may enhance the model's robustness and reduce the influence of irrelevant objects on classification accuracy. Additionally, in low-magnitude, widespread earthquakes, it can reduce experts' workload by enabling agreements on fixed compensation amounts for buildings with only non-structural damage.

Acknowledgements This research was supported by the Istanbul Technical University (ITU) Scientific Research Projects Coordination Unit (BAP) under the project number MDA-2023-44128. The computational resources were provided by the National Center for High Performance Computing of ITU (UHEM).

Funding Open access funding provided by the Scientific and Technological Research Council of Türkiye (TÜBİTAK).

References

1. German, S.; Brilakis, I.; DesRoches, R.: Rapid entropy-based detection and properties measurement of concrete spalling with machine vision for post-earthquake safety assessments. Advanced Engineering Informatics **26**(4), 846–858 (2012)
2. Tung, P.-C.; Hwang, Y.-R.; Wu, M.-C.: The development of a mobile manipulator imaging system for bridge crack inspection. Automation in construction **11**(6), 717–729 (2002)
3. Abdel-Qader, I.; Abudayyeh, O.; Kelly, M.E.: Analysis of edge-detection techniques for crack identification in bridges. Journal of Computing in Civil Engineering **17**(4), 255–263 (2003)
4. Chen, L.-C.; Shao, Y.-C.; Jan, H.-H.; Huang, C.-W.; Tien, Y.-M.: Measuring system for cracks in concrete using multitemporal images. Journal of Surveying Engineering **132**(2), 77–82 (2006)
5. Sinha, S.K.; Fieguth, P.W.: Automated detection of cracks in buried concrete pipe images. Automation in construction **15**(1), 58–72 (2006)
6. Abdel-Qader, I.; Pashaie-Rad, S.; Abudayyeh, O.; Yehia, S.: Pca-based algorithm for unsupervised bridge crack detection. Advances in Engineering Software **37**(12), 771–778 (2006)
7. Yu, S.-N.; Jang, J.-H.; Han, C.-S.: Auto inspection system using a mobile robot for detecting concrete cracks in a tunnel. Automation in construction **16**(3), 255–261 (2007)
8. Oh, J.-K.; Jang, G.; Oh, S.; Lee, J.H.; Yi, B.-J.; Moon, Y.S.; Lee, J.S.; Choi, Y.: Bridge inspection robot system with machine vision. Automation in Construction **18**(7), 929–941 (2009)
9. Yamaguchi, T.; Nakamura, S.; Saegusa, R.; Hashimoto, S.: Image-based crack detection for real concrete surfaces. IEEJ Transactions on Electrical and Electronic Engineering **3**(1), 128–135 (2008)
10. Zhu, Z.; Brilakis, I.: Concrete column recognition in images and videos. Journal of computing in civil engineering **24**(6), 478–487 (2010)
11. Zhu, Z.; Brilakis, I.: Machine vision-based concrete surface quality assessment. Journal of Construction Engineering and Management **136**(2), 210–218 (2010)
12. Zhu, Z.; German, S.; Brilakis, I.: Visual retrieval of concrete crack properties for automated post-earthquake structural safety evaluation. Automation in Construction **20**(7), 874–883 (2011)
13. Gul, M.; Catbas, F.: Critical issues, condition assessment and monitoring of movable bridges: Image processing for open gear monitoring. In: Structures Congress 2013: Bridging Your Passion with Your Profession, pp. 308–318 (2013)
14. Higgins, C.; Turan, O.T.: Imaging tools for evaluation of gusset plate connections in steel truss bridges. Journal of Bridge Engineering **18**(5), 380–387 (2013)

15. Kim, H.; Sim, S.-H.; Spencer, B.F.: Automated concrete crack evaluation using stereo vision with two different focal lengths. Automation in Construction **135**, 104136 (2022)
16. Wang, X.; Demartino, C.; Narazaki, Y.; Monti, G.; Spencer, B.F., Jr.: Rapid seismic risk assessment of bridges using uav aerial photogrammetry. Engineering Structures **279**, 115589 (2023)
17. Yeum, C.M.; Dyke, S.J.: Vision-based automated crack detection for bridge inspection. Computer-Aided Civil and Infrastructure Engineering **30**(10), 759–770 (2015)
18. Zhang, S.; Ni, P.; Wen, J.; Han, Q.; Du, X.; Fu, J.: Intelligent identification of moving forces based on visual perception. Mechanical Systems and Signal Processing **214**, 111372 (2024)
19. Zhang, S.; Ni, P.; Wen, J.; Han, Q.; Du, X.; Xu, K.: Automated vision-based multi-plane bridge displacement monitoring. Automation in Construction **166**, 105619 (2024)
20. Zhang, S.; Ni, P.; Wen, J.; Han, Q.; Du, X.; Xu, K.: Non-contact impact load identification based on intelligent visual sensing technology. Structural Health Monitoring **23**(6), 3525–3544 (2024)
21. Furuta, H.; Hattori, H.; Frangopol, D.M.: Damage assessment of reinforced concrete bridge decks using tam network. In: Advances in Engineering Structures, Mechanics & Construction, pp. 81–86 (2006). Springer
22. Nishikawa, T.; Yoshida, J.; Sugiyama, T.; Fujino, Y.: Concrete crack detection by multiple sequential image filtering. Computer-Aided Civil and Infrastructure Engineering **27**(1), 29–47 (2012)
23. O'Byrne, M.; Schoefs, F.; Ghosh, B.; Pakrashi, V.: Texture analysis based damage detection of ageing infrastructural elements. Computer-Aided Civil and Infrastructure Engineering **28**(3), 162–177 (2013)
24. Lee, B.Y.; Kim, Y.Y.; Yi, S.-T.; Kim, J.-K.: Automated image processing technique for detecting and analysing concrete surface cracks. Structure and Infrastructure Engineering **9**(6), 567–577 (2013)
25. Jahanshahi, M.R.; Masri, S.F.; Padgett, C.W.; Sukhatme, G.S.: An innovative methodology for detection and quantification of cracks through incorporation of depth perception. Machine vision and applications **24**, 227–241 (2013)
26. O'Byrne, M.; Ghosh, B.; Schoefs, F.; Pakrashi, V.: Regionally enhanced multiphase segmentation technique for damaged surfaces. Computer-Aided Civil and Infrastructure Engineering **29**(9), 644–658 (2014)
27. Gul, M.; Catbas, F.N.; Hattori, H.: Image-based monitoring of open gears of movable bridges for condition assessment and maintenance decision making. Journal of Computing in Civil Engineering **29**(2), 04014034 (2015)
28. Cha, Y.-J.; You, K.; Choi, W.: Vision-based detection of loosened bolts using the hough transform and support vector machines. Automation in Construction **71**, 181–188 (2016)
29. Cha, Y.-J.; Choi, W.; Büyüköztürk, O.: Deep learning-based crack damage detection using convolutional neural networks. Computer-Aided Civil and Infrastructure Engineering **32**(5), 361–378 (2017)
30. Cha, Y.-J.; Choi, W.; Suh, G.; Mahmoudkhani, S.; Büyüköztürk, O.: Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types. Computer-Aided Civil and Infrastructure Engineering **33**(9), 731–747 (2018)
31. Xue, Y.; Li, Y.: A fast detection method via region-based fully convolutional neural networks for shield tunnel lining defects. Computer-Aided Civil and Infrastructure Engineering **33**(8), 638–654 (2018)
32. Dung, C.V.; Anh, L.D.: Autonomous concrete crack detection using deep fully convolutional neural network. Automation in Construction **99**, 52–58 (2019)
33. Alipour, M.; Harris, D.K.; Miller, G.R.: Robust pixel-level crack detection using deep fully convolutional neural networks. Journal of Computing in Civil Engineering **33**(6), 04019040 (2019)
34. Yuan, J.; Ren, Q.; Jia, C.; Zhang, J.; Fu, J.; Li, M.: Automated pixel-level crack detection and quantification using deep convolutional neural networks for structural condition assessment. In: Structures, vol. 59, p. 105780 (2024). Elsevier
35. Krizhevsky, A.; Sutskever, I.; Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Communications of the ACM **60**(6), 84–90 (2017)
36. Simonyan, K.; Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
37. Badrinarayanan, V.; Kendall, A.; Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Transactions on pattern analysis and machine intelligence **39**(12), 2481–2495 (2017)
38. Yeum, C.M.; Dyke, S.J.; Ramirez, J.: Visual data classification in post-event building reconnaissance. Engineering Structures **155**, 16–24 (2018)
39. Gao, Y.; Mosalam, K.M.: Deep transfer learning for image-based structural damage recognition. Computer-Aided Civil and Infrastructure Engineering **33**(9), 748–768 (2018)
40. Liang, X.: Image-based post-disaster inspection of reinforced concrete bridge systems using deep learning with bayesian optimization. Computer-Aided Civil and Infrastructure Engineering **34**(5), 415–430 (2019)
41. Zhang, X.; Rajan, D.; Story, B.: Concrete crack detection using context-aware deep semantic segmentation network. Computer-Aided Civil and Infrastructure Engineering **34**(11), 951–971 (2019)
42. Gao, Y.; Mosalam, K.M.: Peer hub imagenet: A large-scale multiattribute benchmark data set of structural images. Journal of Structural Engineering **146**(10), 04020198 (2020)
43. Karaaslan, E.; Bagci, U.; Catbas, F.N.: Attention-guided analysis of infrastructure damage with semi-supervised deep learning. Automation in Construction **125**, 103634 (2021)
44. Dais, D.; Bal, I.E.; Smyrou, E.; Sarhosis, V.: Automatic crack classification and segmentation on masonry surfaces using convolutional neural networks and transfer learning. Automation in Construction **125**, 103606 (2021)
45. Chen, L.; Chen, W.; Wang, L.; Zhai, C.; Hu, X.; Sun, L.; Tian, Y.; Huang, X.; Jiang, L.: Convolutional neural networks (cnns)-based multi-category damage detection and recognition of high-speed rail (hsr) reinforced concrete (rc) bridges using test images. Engineering Structures **276**, 115306 (2023)
46. Dogan, G.; Arslan, M.H.; Ilki, A.: Detection of damages caused by earthquake and reinforcement corrosion in rc buildings with deep transfer learning. Engineering Structures **279**, 115629 (2023)
47. Ilki, A.; Halici, O.; Comert, M.; Demir, C.: The modified post-earthquake damage assessment methodology for tcip (tcip-dam-2020). In: Advances in Assessment and Modeling of Earthquake Loss, pp. 85–107. Springer, Cham. (2021)
48. AFAD (Disaster and Emergency Management Presidency): 24 Ocak 2020 Sivrice (Elazig) Depremi Raporu" . Accessed April 15, 2022 (2022). https://shorturl.at/cDES5
49. CSB (Ministry of Environment Urbanization and Climate Change: Bakan Kurum Elazığ'daki Hasar Tespit Çalışmalarını Anlattı" . Accessed April 15, 2022 (2022). https://shorturl.at/kABT3
50. Liu, Z.; Mao, H.; Wu, C.-Y.; Feichtenhofer, C.; Darrell, T.; Xie, S.: A convnet for the 2020s. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11976–11986 (2022)
51. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 618–626 (2017)
52. He, K.; Zhang, X.; Ren, S.; Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

53. Newell, A.: A tutorial on speech understanding systems. Speech recognition, 4–54 (1975)
54. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
55. Zhong, Z.; Zheng, L.; Kang, G.; Li, S.; Yang, Y.: Random erasing data augmentation. In: Proceedings of the AAAI Conference on Artificial Intelligence **34**, 13001–13008 (2020)
56. Zhang, H.; Cisse, M.; Dauphin, Y.N.; Lopez-Paz, D.: mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412 (2017)
57. Yun, S.; Han, D.; Oh, S.J.; Chun, S.; Choe, J.; Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6023–6032 (2019)
58. Glorot, X.; Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 249–256 (2010). JMLR Workshop and Conference Proceedings
59. Ribeiro, M.T.; Singh, S.; Guestrin, C.: " why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144 (2016)
60. Lundberg, S.M.; Lee, S.-I.: A unified approach to interpreting model predictions. Advances in neural information processing systems **30** (2017)
61. Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; Torralba, A.: Learning deep features for discriminative localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2921–2929 (2016)