

Article

A Minority Class Balanced Approach Using the DCNN-LSTM Method to Detect Human Wrist Fracture

Tooba Rashid ¹, Muhammad Sultan Zia ², Najam-ur-Rehman ³ , Talha Meraj ⁴ , Hafiz Tayyab Rauf ⁵ 
and Seifedine Kadry ^{6,7,8,*} 

¹ Department of Computer Science, The University of Lahore, Chenab Campus, Gujrat 50700, Pakistan

² Department of Computer Science, The University of Chenab, Gujrat 50700, Pakistan

³ Department of Human Resource Section, University of Gujrat, Gujrat 50700, Pakistan

⁴ Department of Computer Science, COMSATS University Islamabad-Wah Campus, Wah Cantt 47040, Pakistan

⁵ Centre for Smart Systems, AI and Cybersecurity, Staffordshire University, Stoke-on-Trent ST4 2DE, UK

⁶ Department of Applied Data Science, Noroff University College, 4612 Kristiansand, Norway

⁷ Artificial Intelligence Research Center (AIRC), Ajman University, Ajman 346, United Arab Emirates

⁸ Department of Electrical and Computer Engineering, Lebanese American University, Byblos P.O. Box 13-5053, Lebanon

* Correspondence: skadry@gmail.com

Abstract: The emergency department of hospitals receives a massive number of patients with wrist fracture. For the clinical diagnosis of a suspected fracture, X-ray imaging is the major screening tool. A wrist fracture is a significant global health concern for children, adolescents, and the elderly. A missed diagnosis of wrist fracture on medical imaging can have significant consequences for patients, resulting in delayed treatment and poor functional recovery. Therefore, an intelligent method is needed in the medical department to precisely diagnose wrist fracture via an automated diagnosing tool by considering it a second option for doctors. In this research, a fused model of the deep learning method, a convolutional neural network (CNN), and long short-term memory (LSTM) is proposed to detect wrist fractures from X-ray images. It gives a second option to doctors to diagnose wrist fracture using the computer vision method to lessen the number of missed fractures. The dataset acquired from Mendeley comprises 192 wrist X-ray images. In this framework, image pre-processing is applied, then the data augmentation approach is used to solve the class imbalance problem by generating rotated oversamples of images for minority classes during the training process, and pre-processed images and augmented normalized images are fed into a 28-layer dilated CNN (DCNN) to extract deep valuable features. Deep features are then fed to the proposed LSTM network to distinguish wrist fractures from normal ones. The experimental results of the DCNN-LSTM with and without augmentation is compared with other deep learning models. The proposed work is also compared to existing algorithms in terms of accuracy, sensitivity, specificity, precision, the F1-score, and kappa. The results show that the DCNN-LSTM fusion achieves higher accuracy and has high potential for medical applications to use as a second option.

Keywords: augmentation; convolution neural network; human wrist fracture; long short-term memory; lifesaving



Citation: Rashid, T.; Zia, M.S.; Najam-ur-Rehman; Meraj, T.; Rauf, H.T.; Kadry, S. A Minority Class Balanced Approach Using the DCNN-LSTM Method to Detect Human Wrist Fracture. *Life* **2023**, *13*, 133. <https://doi.org/10.3390/life13010133>

Academic Editors: Piotr Waż, Dorota Bielińska-Waż and Katarzyna Zorena

Received: 24 November 2022

Revised: 19 December 2022

Accepted: 27 December 2022

Published: 3 January 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

A bone fracture happens when a high force is applied against a bone. Fractures of a bone can be caused by trauma, osteoporosis, and overuse. According to a recent study by the World Health Organization (WHO), a fracture affects a significant number of people, and the consequences of a neglected fracture can lead to severe injury or even death. A wrist fracture is a quite common injury, and positive cases are increasing on a daily basis [1]. Wrist fracture can be the result of an accident, such as slipping over with an outstretched palm. Often, extensive injuries occur from intense physical trauma, such as automobile

accidents or falls from a roof. In osteoporosis, weak bone appears to crack more swiftly. Countless incidences need healthcare professionals to examine fractures. Both human and environmental factors, such as inexperienced physicians, physical fatigue, distractions, poor observation circumstances, and time constraints, promote radiographic analysis errors.

The advent of radiographic technology has significantly improved the early diagnosis of fractures. Some medical imaging technologies, such as X-ray, magnetic resonance imaging (MRI), computed tomography (CT), and ultrasound, are convenient to capture fractures. However, with relatively low cost and accessibility, X-rays are most generally implemented in bone fracture diagnosis. However, in comparison to the enormous incidence of fractural cases, the lack of skilled radiologists is huge. As a result, many radiologists are fatigued by the massive amount of medical imaging [2]. To overcome this concern, computer-aided diagnostic (CAD) technology has been used to assist physicians to analyze medical images. Recent studies have shown that incorporating machine learning and deep learning methods into CAD systems enhances the performance of healthcare professionals.

Machine learning is a sub-domain of artificial intelligence (AI) that uses various algorithms, such as a support vector machine (SVM) [3], decision tree (DT), K-nearest neighbour (KNN) [4], neural network (NN), and random forest (RF) [5], for automatic fracture diagnosis, classification, and prediction. However, when the number of features is significantly beyond the number of observations, these algorithms produce inadequate outcomes, which cannot fulfil the potential medical need. Nowadays, deep learning has emerged as the most powerful AI technology in the medical field [6].

Deep learning enables raw data to be seamlessly transferred into simulators and analyzed with multiple feature extraction and weighting levels [7]. Deep learning is widely used for its immense potential in intricate feature extraction and predictions in a variety of medical fields, including pathology [8], pharmacology, radiology, ophthalmology, and even feature detection, including both ulnar and radial fractures [9,10], pulmonary tuberculosis classification [11], cancer diagnosis [12], rib fracture [13], hip fracture [14] hand and wrist fracture [15], ankle fracture [16], and thoracolumbar fracture [17]. Convolutional neural networks (CNNs) tend to be a more efficient strategic method for feature detection and have rapidly acquired significance in the computer vision area in the past years. To modify clinical challenges, CNNs “train” distinguishing patterns using models, such as Inception V3 [18], ResNet [19], U-Net [20], Xception [21], and DenseNet [22]. Due to limited annotated data availability, data augmentation and other data generative methods can be used to increase the size of the dataset. The interpretation of a deep learning algorithm is still a challenge since its accuracy is highly dependent on the features of the learned data, including diagnostic precision and fracture extent. In general, specific input data and modification are frequently needed for pre-existing deep learning techniques before proposing any clinical development.

To automatically diagnose wrist fracture from X-ray images, this research attempts to provide a deep learning method that adopts the CNN and LSTM networks together to enhance the patient’s experience by enhancing performance, minimizing misdiagnosis, and reducing delayed treatment. Clinicians depend on X-ray images to detect where the fractures have occurred. The earlier approach could only identify fractures in a specific bone area, such as the distal radius [10]. However, various forms of bone fractures can be seen in an X-ray image.

The contributions of this study are mentioned next.

- (a) To alleviate clinicians’ cognitive load, the two-method model adopted in this research detects significant fracture areas in an X-ray image.
- (b) A fusion method, DCNN-LSTM, is proposed that can automatically diagnose wrist fractures to facilitate the radiologist.
- (c) The first model, the DCNN, is designed to enhance the CNN’s receptive field by using a dilation factor in the convolution operation, which lessens the complexity of the CNN training phase.

- (d) The fused model DCNN-LSTM enhances accuracy with augmented data to recognize anomalies using wrist X-ray images.
- (e) The entire research work is structured as follows: Section 2 presents related works on fractures. Section 3 provides an overview of the proposed methodology, comprising dataset acquisition and analysis. Section 4 includes a discussion of the results and an evaluation of the proposed methodology. Section 5 concludes the research work.

2. Related Works

Machine learning (ML) and deep learning (DL) enable AI systems to take a step ahead as they allow cognitive learning to take place within a framework dependent on prior studies or data analysis. As it progresses, the model executes challenging decision methods and evaluates past activities. Researchers have designed deep learning approaches to detect fractures, depending on clinical radiographs, CT scans, and X-rays of bone. Various bone fracture detection and classification methods have been proposed traditionally. However, they are statistical methods, they cannot detect the fracture location in the bone, and they need a number of pre-processing stages. This study discusses recently developed methodologies that use different machine learning and deep learning approaches to detect fractures.

Various imaging approaches are being developed to identify lower leg bone (tibia) fracture variants. Myint et al. [23] used the machine learning algorithms K-nearest neighbour (KNN) and decision tree (DT) to detect tibia fracture from X-ray images. The chest CT images of 1707 patients were used by Yao et al. [24] to classify rib fractures using a three-step algorithm and achieved an accuracy of 0.869 on 4496 CT images. The most popular object recognition deep learning model YOLOv3 was used by Choi et al. [25] to diagnose skull fractures in X-ray images and achieved an accuracy of 91.7% on a limited dataset. Tomita et al. [26] identified incidental OVs in chest, abdomen, and pelvic CT scans. The OVF detection system uses a deep convolutional neural network (CNN) to accumulate radiological features out of each CT slice, and the resulting features are then evaluated using the long short-term memory (LSTM) network to determine the final decision for a complete CT scan report. This approach achieved an accuracy of 89.2% and has the potential to minimize the time and cognitive load on radiologists for OVF monitoring, along with the risk of adverse outcomes. Kim et al. [27] used transfer learning from deep CNNs to diagnosis wrist fractures from X-ray images in four stages. The findings showed that the model performs efficiently on a small dataset. The findings of the system using deep learning techniques and level-set methods by Kim et al. [28] achieved efficient results on 160 lumbar X-ray images to identify and segment each lumbar vertebra. Guan et al. [29] used a novel deep learning model fast R-CNN to diagnose arm bone fractures on X-ray images. The algorithm fast R-CNN was trained on the MURA dataset comprising 4000 X-ray images with low quality and achieved a state-of-the-art average precision of 62.04% for diagnosing arm fractures, which is much faster than existing state-of-the-art deep learning methods. CNN is used in combination with other modules to recognize, localize, and categorize objects in images. Thian et al. [30] applied the Inception-ResNet Faster R-CNN model on 7356 wrist images. Even before conducting classification, they trained their algorithm on the wrist imaging dataset. For adequate and comprehensive classification and contextual localization of radius and ulna fractures in wrist radiographs, an object detection deep learning approach was optimal and achieved an accuracy of 91.8%. Guan et al. [31] introduced a novel deep learning technique termed “dilated convolutional feature pyramid network” (DCFPN) to identify thigh fracture. The algorithms were trained using 3484 X-ray images from Linyi People’s Hospital, and the learning algorithm was tested using 358 images. The DCFPN findings outperformed traditional algorithms of deep learning. The overall related work algorithms regarding fracture detection in different body organs are summarized in a visual illustration in Figure 1.

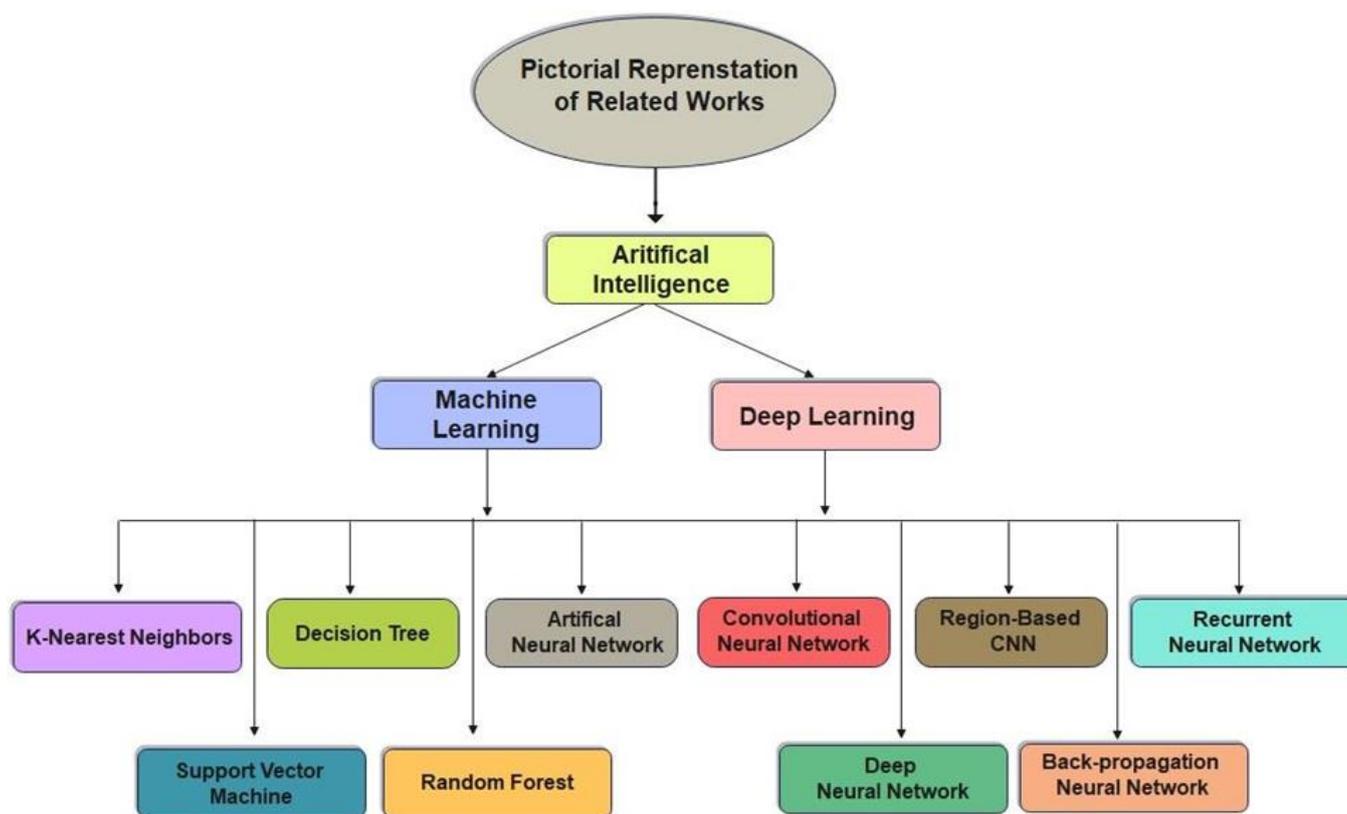


Figure 1. Pictorial representation of related works.

Ebism et al. [32] detected the radius in both posteroanterior PA and lateral wrist images using a random forest (RF) classifier. Guo et al. [33] collected CT images of orbital blowout fractures from the Shanghai Ninth People's Hospital and used the Inception-V3 convolutional neural network (CNN) framework with the XGBoost model to classify the orbital blowout fractures. Zeelan et al. [34] detected bone fracture from X-ray images using the machine learning models probabilistic neural network (PNN), backpropagation neural network (BPNN), and support vector machine (SVM) and classified the input images into the classes skull, head, chest, hand, and spine. A deep convolutional neural network (DCNN) model developed by Cheng et al. [35] not only detected hip fractures on plain frontal pelvic radiographs (PXR) with a good accuracy rate, but it was also good at localizing fracture areas. In the human body, bone fractures are common. Rathor et al. [36] used a deep neural network (DNN) to detect fractured bone from X-ray images and achieved a 92.44% accuracy, but the model still involves affirmation on a huge dataset. Ebsim et al. [37] trained a convolutional neural network (CNN) to detect wrist fractures in posteroanterior (PA) and lateral radiographs. Adigun et al. [38] combined two machine learning models, K-nearest neighbour (KNN) and support vector machine (SVM), to enhance each other's findings. The KNN-SVM attained better classification accuracy than earlier work. Myint et al. [39] trained support vector machine (SVM) on 40 X-ray images to classify two types of fractures (non-fracture and fracture or transverse fracture). According to the paper's conclusions, good outcomes were not achieved. Mondol et al. [40] built a model using VGG-19 and ResNet to detect four types of fractures: elbow, wrist, finger, and humerus on the MURA dataset. Lindsey et al. [41] developed a deep convolutional neural network (DCNN) model to detect and locate fractures in 135,845 PA or LAT wrist, foot, elbow, shoulder, knee, ankle, pelvis, hip, humerus, and shoulder radiographs. Chittajallu et al. [42] used a CNN model that was trained on 200 images of human hands, ribs, legs, and the neck to detect fracture, and the findings achieved adequate predictions. Vasilakakis et al. [43] proposed a novel approach using wavelet fuzzy phrases (WFP) for feature extraction and classification. The

classification performance of the model achieved better results, potentially minimized diagnostic errors, and improved the radiologists' performance. Dimililer et al. [44] designed an intelligent classification system using a backpropagation neural network (BPNN) that was capable of detecting and classifying bone fractures from 100 X-ray images. Hrzić et al.'s transfer learning model YOLOv4 was used to detect wrist fractures from X-ray images. The functional testing on three aspects demonstrated that the YOLOv4-based model significantly outperforms the state-of-the-art technique based on the U-Net framework and achieves better accuracy [45].

The summary of recently applied studies is shown in Table 1. All studies used different datasets on different organs of the human body to propose an efficient method for fracture detection. However, the studies did not mainly consider the class imbalance problem and annotated data availability to fulfil the requirement of deep learning input data. In addition, many of the studies did not consider multi-model usage to include multi-feature usage for precise detection of body fractures. Therefore, the proposed study firstly solves the class imbalance problem and secondly applies the multi-feature extraction approach to precisely classify wrist fractures.

Table 1. Recent work on automatic fracture detection using machine learning and deep learning techniques.

Fracture Type	Image Type	Feature Extraction Model	Classifier Used	Accuracy	Year
Leg bone fracture [23]	X-ray	Harris algorithm	Decision tree (DT) and KNN	82%	2018
Rib fracture [24]	CT scan	-	3D DenseNet	86.9%	2021
Skull fracture [25]	X-ray	-	YOLOv3	91.7%	2022
Osteoporotic vertebral fractures [26]	CT scan	ResNet34	CNN/LSTM)	89.2%	2018
Fracture detection [27]	X-ray	-	CNN	95.4%	2018
Wrist fracture [28]	-	Inception-ResNet version	Faster R-CNN	62.04%	2019
Lumbar vertebra compression fracture [29]	X-ray	Post-driven Learning	DL techniques and level-set methods (Pose-net and M-net)	91.60%	2020
Distal radius fractures [30]	Distal Radius Fractures	-	Random forest (RF)	91.4%	2017
Arm fracture [31]	X-ray	-	YOLOv4	81.91%	2021
Thigh fracture [32]	X-ray	-	Dilated convolutional feature pyramid network (DCFPN)	82.1%	2019
Orbital blowout fractures [33]	CT scan	Inception-V3 convolutional neural network (CNN)	Inception V3 CNN and XGBoost	92%	2019
Fracture detection [34]	X-ray	-	BPNN, SVM, and PNN	92.3%	2018
Hip fractures [35]	PXR	- Grad-CAM	Deep convolutional neural network (DCNN)	91%	2019
Bone dracture [36]	X-ray	-	Deep CNN (Adam optimizer and softmax)	92.44%	2020

Table 1. Cont.

Fracture Type	Image Type	Feature Extraction Model	Classifier Used	Accuracy	Year
Wrist dractures [37]	X-ray	-	Convolutional neural network (CNN)	96%	2018
Bone fracture [38]	X-ray	Hough transformation	KNN and SVM	90%	2020
Arm fracture [39]	X-ray	ResNet	Fast R-CNN		2020
Elbow fractures [40]	X-ray	-	CNN Xception (Vision model)	88%	2019
Wrist fraction [41]	X-ray	VGG-19 and ResNet	Deep CNN using VGG-19, ResNet	87.8%	2019
Fracture detection [42]	X-ray	-	DCNN	91.5%	2018
Wrist fracture [43]	X-ray	-	YOLOV4	96.5%	2022
Binary fracture [44]	X-ray	Binary entropy function	Convolutional neural network (CNN)	Hands 80.45% Legs 84.75% Ribs 80.65% Other 86.75%	2019
Bone fracture [45]	X-ray	2D DWT	Wavelet fuzzy phrases (WFP)	84%	2019

3. Proposed Methodology

To detect the presence of fractures from the wrist images of humans, we designed a combination of a dilated convolutional neural network (DCNN) and an LSTM network. The proposed network, DCNN-LSTM, has two methods, as illustrated in Figure 2. X-ray images of patients were acquired from the Mendeley data repository. Firstly, the images were passed through the pre-processing stage. Image resizing and enhancement were performed in the pre-processing stage. Secondly, the data augmentation approach was used to enlarge the dataset. Next, pre-processed images and augmented normalized data were fed into a 28-layer dilated CNN to extract valuable features and eliminate features that were not beneficial. Finally, the LSTM network was used to accurately classify the normal and fractured images.

3.1. Data Normalization and Pre-Processing

Normalization of data has always been an essential aspect of the pre-processing method, focused on eliminating data redundancy and inconsistency from the database in order to regulate the complexity of the network and provide accurate findings. The images captured in real time are of various sizes and destroyed for numerous factors. An image must be resized to accommodate the size of the model for which it is built. The X-ray images used in this study were in JPG format with different sizes. All images were resized to ones with a resolution of $512 \times 256 \times 3$ pixels using the bi-cubic interpolation method. In the bi-cubic method, 4×4 neighbouring pixels were used to evaluate the weighted average. The pixels that were nearest in both vertical and horizontal directions were weighted in computations. Assuming that (x, y) are the coordinates of the point from which we aimed to explore the new pixel, the value was determined using the following equation:

$$v(x, y) = \sum_{i=0}^3 \sum_{j=0}^3 a_{ij} \quad x^i y^j \quad (1)$$

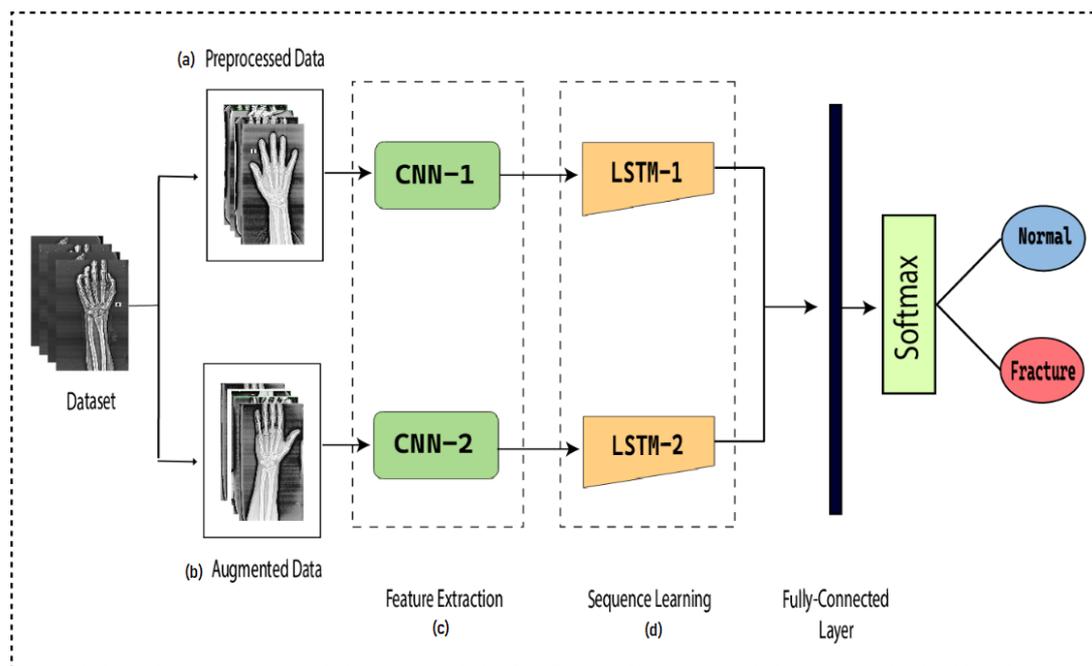


Figure 2. DCNN-LSTM main parts. (a) X-ray images are resized and enhanced in the pre-processing stage. (b) The diversity of the dataset increases in the data augmentation stage. (c) The dilated-CNN model for feature extraction from wrist X-ray images. (d) Extracted features are classified into normal and fractured images through the LSTM network.

To increase the brightness of the X-ray images, the image contrast enhancement method is effective to apply. Image enhancement was accomplished by modifying the image's histogram attributes, altering the image's histogram, enhancing the various grey levels, and increasing the image's contrast. To reduce data loss and distortion, we used the adaptive histogram equalization method in this study. It enhances the contrast of the image and edges to detect the area of interest.

3.2. Data Augmentation

Data augmentation, which estimates the data probability space by modifying input images, such as rotation, random crop, scaling, and noise disturbance, is an efficient and effective strategy to reduce the "overfitting" of the deep convolutional neural network due to inadequate training images. The model's performance can be enhanced in general by increasing the volume, quality, and variety of data in the dataset. The method used to augment images in this study was the affine transformation method. Affine transformation enhances the alteration of an image's geometric structure by maintaining line parallelism but not dimensions and angles. It uses a linear collection of translation, rotation, scaling, and/or shearing operations to process data into new data. In this study, we applied a rotation operation of affine transformation on the images to increase the size of the dataset. Rotation is most often optimized to enhance an image's visual effect, but it can also be applied as a modifier in systems that use directional operators. To rotate the image around a centre or an axis, we simply provide the angle of rotation. The angle of rotation is denoted by θ , which specifies how many degrees the image is rotating.

$$R = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2)$$

The original image is mapped to a new image by rotation at an angle of 0° , 90° , 180° , 270° , and 360° . The rotation operation increases the diversity of the dataset and thus enables

the deep learning model to identify the data intrinsic features quite comprehensively. Training based on augmented data can potentially enhance the performance of a deep CNN as compared to training without data augmentation.

3.3. Proposed Dilated CNN

The traditional CNN's primary function is to execute the convolution operation on the specific region of an image. The input layer, convolutional layer, batch normalization, ReLU, pooling layer, fully connected layer, and output layer constitute the CNN, as shown in Figure 3. When the number of feature channels increases, the number of parameters of the convolution kernel escalates as well, resulting in an increase in simulation. The standard convolutional neural network (CNN) was substituted in this study by the dilated convolution neural network (D-CNN). The D-CNN uses larger two-dimensional filters. Traditional CNNs often use smaller convolution filters (typically 2×2 or 3×3). The D-CNN uses dilated convolution filters, which are enlarged filters. A dilated convolution with a dilatation rate r comprises $r-1$ zeros between consecutive filter values, enlarging the size of a $k \times k$ filter to $[k + (k - 1)(r - 1)] \times [k + (k - 1)(r - 1)]$ [46]. The dilated convolution filters expand the CNN's receptive field without integrating any additional parameters. Thus, the D-CNN uses slightly fewer layers than the CNN to attain the same receptive size of the field, preventing the overfitting problem caused by a deep CNN.

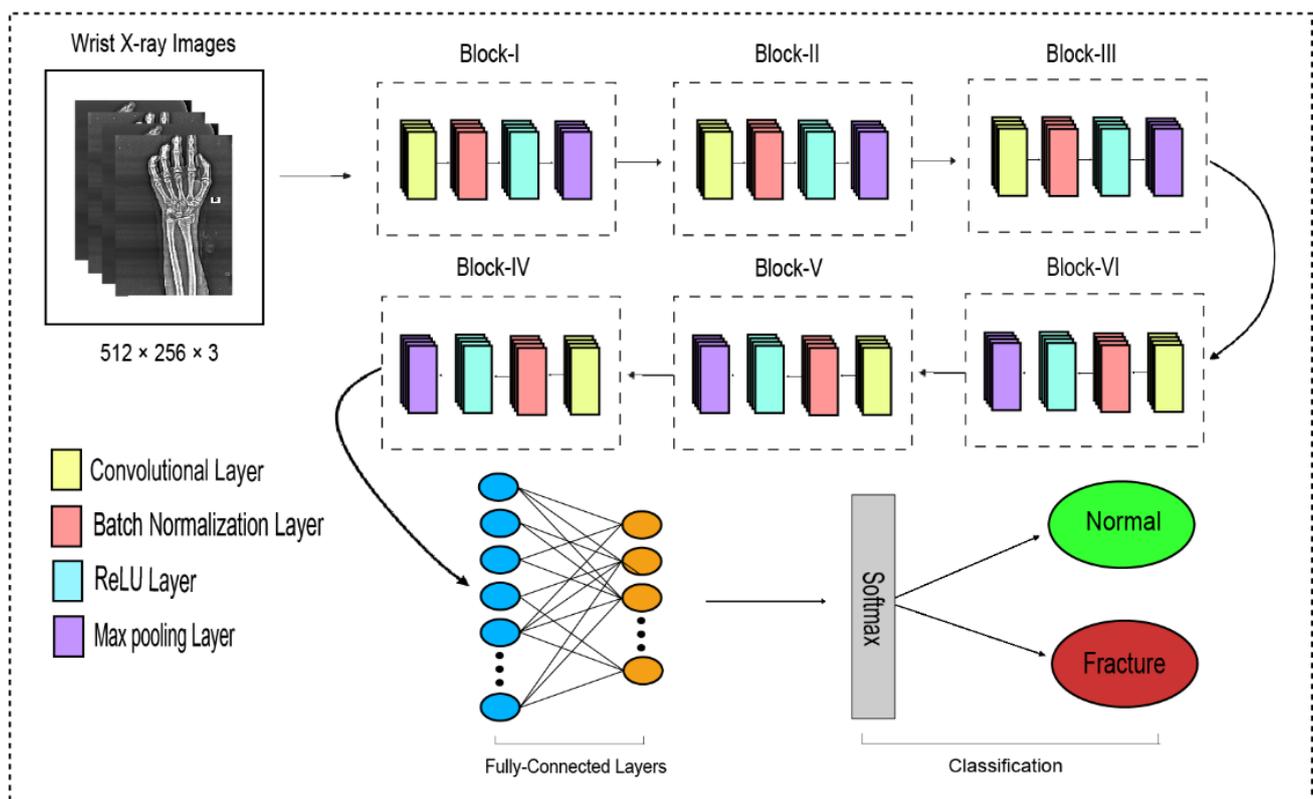


Figure 3. The dilated convolutional neural network for wrist fracture detection.

The proposed dilated convolution neural network (D-CNN) was based on a 28-layer architecture and had six blocks. The size of the input images was set to 512×256 . The pre-processed images were fed into D-CNN1, and augmented images were fed into D-CNN2, where (convolution, batch normalization, rectified linear unit, and max pooling) were performed with different parameters, as shown in Table 2. The input layer is the initial layer and feeds images to the system. The next layer is the convolution layer, which executes the convolution process on the obtained images by simply shifting a window across it with a stride. For our initial convolution layer, we constructed a 5×5 window and

used kernels to convolve the images; the number of filters was set to 16, and the amount of filtering increased as further convolutional layers were included. The receptive field size was enlarged by a dilation factor in the C2, C3, C4, C5, and C6 layers. Batch normalization was applied to normalize the previous layers' outcome and to avoid overfitting. Batch normalization facilitates each layer of the system to perform learning autonomously. After the convolution process is completed, the subsequent feature maps are used as input to the batch normalization layer, which is preceded by the rectified linear unit, which uses the function $f(x) = \max(0, x)$. The basic concept to use this function $f(x)$ is to set the values of neurons that are above a certain point to zero. This minimizes data redundancy and preserves essential features. A feature layer's size can be lessened by using max pooling on each feature map with stride 2×2 . All neurons in the fully connected layer are considered and associated with the neurons in the preceding max-pooling layer. The number of output classes is determined by the output from the fully connected layer, which is preceded by the softmax layer.

Input Layer:

Layers follow the tensor to modify and then rebuild the tensor in this layer. The input layer is the entire DCNN's input. It widely represents an image's pixel matrix in the deep learning of image processing, and the image's pixel values are kept in this layer. An image input layer feeds images to the system. A DCNN uses tensors of the shape image height, image width, and colour channels as input. In our proposed system, this was $512 \times 256 \times 3$, where the height of an image is 512, the width of an image is 256, and the number of colour channels is 3.

Convolution Layer:

The convolutional layer's primary function is to identify local conjunctions of features from the preceding layer and transfer their structure to a feature map. The input image is processed by a filter in this layer. By changing the dilated rate, the dilated convolution enlarges the receptive field of the convolution kernel. As a consequence, dilated convolution achieves multi-scale information by enlarging the receptive field of the convolution kernel. Whenever the dilation rate is 1, the receptive size of dilated convolution remains equivalent to that of conventional convolution. In our study, we applied six convolution layers with a dilation rate r that increased the size of the filter $k \times k$ to extract the features from the input images. The formula stated next is applied to compute the receptive field size of a dilated convolution [47,48]:

$$S = F_c + (F_c - 1)(R - 1) \quad (3)$$

S denotes the receptive field size, and R denotes the size of the dilation factor. When the dilation value is far more than 1, dilated convolution can acquire a relatively large receptive field size and accumulate more diverse visual information compared to conventional convolution. When numerous dilated convolutional layers are deployed in the framework, the receptive field size of the convolution kernel is determined layer by layer according to the conceptual framework, with the assumption that the dilation factor is specified appropriately.

Our first layer was a convolution layer comprising 16 feature maps and a 5×5 kernel size. Secondly, the convolution layer comprised 32 feature maps with a 5×5 kernel size with a dilation rate $r = 2$. Third was a convolution layer comprising 64 feature maps with a 5×5 kernel size with a dilation rate $r = 4$. Fourth was a convolution layer of 128 feature map with a 5×5 kernel size with a dilation rate $r = 8$. Fifth was a Convolution layer of 128 feature maps with a 5×5 kernel size with a dilation rate $r = 16$. Sixth was a convolution layer of 256 feature maps with a 5×5 kernel size with a dilation rate $r = 32$.

Table 2. The detailed information about the proposed dilated CNN.

Number	Layer Name	Activations	Kernel Size	Stride	Parameters	Feature Maps
1	Input layer	$512 \times 256 \times 3$	/	/	/	/
2	Convolutional layer (C1)	$512 \times 256 \times 16$	5×5	1	Weights = $5 \times 5 \times 3 \times 16$ Bias = $1 \times 1 \times 16$	16
3	Batch normalization (B1)	$512 \times 256 \times 16$	/	/	Offset = $1 \times 1 \times 16$ Scale = $1 \times 1 \times 16$	16
4	ReLU (R1)	$512 \times 256 \times 16$	/	/	/	/
5	Max-pooling layer (MP1)	$256 \times 128 \times 16$	2×2	2	/	/
6	Convolutional layer (C2)	$256 \times 128 \times 32$	5×5	1 Dilation factor = 2	Weights = $5 \times 5 \times 16 \times 32$ Bias = $1 \times 1 \times 32$	32
7	Batch normalization (B2)	$256 \times 128 \times 32$	/	/	Offset = $1 \times 1 \times 32$ Scale = $1 \times 1 \times 32$	32
8	ReLU (R2)	$256 \times 128 \times 32$	/	/	/	/
9	Max-pooling layer (MP2)	$128 \times 64 \times 32$	2×2	2	/	/
10	Convolutional layer (C3)	$128 \times 64 \times 64$	5×5	1 Dilation factor = 4	Weights = $5 \times 5 \times 32 \times 64$ Bias = $1 \times 1 \times 64$	64
11	Batch normalization (B3)	$128 \times 64 \times 64$	/	/	Offset = $1 \times 1 \times 64$ Scale = $1 \times 1 \times 64$	64
12	ReLU (R3)	$128 \times 64 \times 64$	/	/	/	/
13	Max-pooling layer (MP3)	$64 \times 32 \times 64$	2×2	2	/	/
14	Convolutional layer (C4)	$64 \times 32 \times 128$	5×5	1 Dilation factor = 8	Weights = $5 \times 5 \times 64 \times 128$ Bias = $1 \times 1 \times 128$	128
15	Batch normalization (B4)	$64 \times 32 \times 128$	/	/	Offset = $1 \times 1 \times 128$ Scale = $1 \times 1 \times 128$	128
16	ReLU (R4)	$64 \times 32 \times 128$	/	/	/	/
17	Max-pooling layer (MP4)	$32 \times 16 \times 128$	2×2	2	/	/
18	Convolutional layer (C5)	$32 \times 16 \times 128$	5×5	1 Dilation factor = 16	Weights = $5 \times 5 \times 128 \times 128$ Bias = $1 \times 1 \times 128$	128

Table 2. Cont.

Number	Layer Name	Activations	Kernel Size	Stride	Parameters	Feature Maps
19	Batch normalization (B5)	$32 \times 16 \times 128$	/	/	Offset = $1 \times 1 \times 128$ Scale = $1 \times 1 \times 128$	128
20	ReLU (R5)	$32 \times 16 \times 128$	/	/	/	/
21	Max-pooling layer (MP5)	$16 \times 8 \times 128$	2×2	2	/	/
22	Convolutional layer (C6)	$16 \times 8 \times 256$	5×5	1 Dilation factor = 32	Weights = $5 \times 5 \times 128 \times 256$ Bias = $1 \times 1 \times 256$	256
23	Batch normalization (B6)	$16 \times 8 \times 256$	/	/	Offset = $1 \times 1 \times 256$ Scale = $1 \times 1 \times 256$	256
24	ReLU (R6)	$16 \times 8 \times 256$	/	/	/	/
25	Max-pooling layer (MP6)	$8 \times 4 \times 256$	2×2	2	/	/
26	Fully connected layer (FC)	$1 \times 1 \times 2$	/	/	Weights = 2×8192 Bias = 2×1	2
27	Softmax	$1 \times 1 \times 2$	/	/	/	/
28	Classification layer	/	/	/	/	/

Batch Normalization Layer:

When the convolution process is completed, the cumulative feature maps are obtained and used as input to the batch normalization layer. Batch normalization facilitates each layer of the system to perform learning autonomously. The batch normalization method normalizes the input value x_i by first estimating μ_B and σ_B^2 , along with a small batch size to boost CNN training, while significantly reducing the network initialization accuracy. The normalized activations are then calculated using the following equation:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad (4)$$

The batch normalization layer's normalized outcome is maintained in \hat{x}_i . The outcome of the batch normalization layer is then used by the ReLU activation function.

Rectified Linear Unit (ReLU):

The ReLU function was used as the activation function in each convolutional layer to increase non-linearity in the yield, and it is represented as:

$$f(x) = \max(0, x) \quad (5)$$

It is a non-linear operation that transmits zero outcomes when handed a negative input and one outcome when handed a positive input. The ReLU has no finite possibilities for positive input values, and the gradients are either zeros or ones. This permits the ReLU to evaluate efficiently and yield relatively high accuracy.

Pooling Layer:

By conducting dimension reduction, the pooling layer strives to decrease the number of parameters in the convolutional yield. In our study, we applied max pooling with a 2×2 kernel size and with a stride of 2. The layer in max pooling typically works with the most significant feature in the feature map obtained from the convolutional layer.

The pooling layer accepts the input size of $W_1 \times H_1 \times D_1$ from the convolutional layer, where W_1 is the width, H_1 is the height, and D_1 is the depth. The pooling layer's window size in each block remains the same at a 2×2 kernel size and a stride of 2. The pooling layer needs two hyperparameters: the spatial extent (F) and the stride (S). The MAX operation resizes each slice of the input spatially.

The output size of the pooling layer is $W_2 \times H_2 \times D_2$, where:

$$W_2 = (W_1 - F) / S + 1 \quad (6)$$

$$H_2 = \frac{(H_1 - F)}{S + 1 - F} \quad (7)$$

$$D_2 = D_1 \quad (8)$$

Fully Connected Layer:

The fully connected layer connects each neuron from the preceding layer to each neuron from the succeeding layer. The output acquired from the preceding convolutional layer and pooling layer is generally flattened, which is transformed into a $1 - D$ array and associated with two fully connected layers. We attached the outcome extracted by the max-pooling layer to a fully connected network comprising two layers to complete the DCNN-BiLSTM model.

Softmax Layer:

The softmax layer is used to categorize the feature map outcome of fully connected layers, and the outcome is a vector indicating probability classification, with values varying from 0 to 1. The softmax activation function is mathematically represented as:

$$\sigma(x_i) = \frac{e^{x_i}}{\sum_{i=0}^n e^{x_i}} \quad (9)$$

where x_i represents the i -th element in array x and i represents the total number of elements in array i .

3.4. Feature Extraction

The pre-trained DCNN model's output structure consists of a fully connected layer and a softmax classifier. Convolving the fully connected layers of the proposed DCNN model yields the feature vector. The inclusion of the fully connected layer at the end of the DCNN has a number of significant advantages. In the concluding stage of the architecture, the feature map generated from the DCNN is transferred through transfer learning to the LSTM layer to retrieve time data. Deep features obtained from the pre-trained DCNN are transferred into a new process without any need to train the new process with a tremendous number of labelled data values, which is particularly accurate and convenient for image classification.

3.5. LSTM

An LSTM network is a recurrent neural network variant that uses memory blocks to operate more effectively and learn more efficiently than a traditional RNN. LSTM networks provide feasible alternatives to RNNs' vanishing and exploding gradient concerns. A network is composed of a sequence of memory blocks termed "cells," each of which contains three gates: input, output, and forget. Therefore, the LSTM network can remember the prior data and relate it to the current data. It also handles complex tasks for which prior RNNs were inadequate to discover a solution [49]. These gates enable information to flow through selectively. In other words, using the three gates (input, output, and forget), the LSTM can regulate which information is maintained, which information is rejected, and which information is delivered. The LSTM represents unidirectional input data; therefore, it cannot preserve structural localization and is more vulnerable to overfitting. It has a hidden layer h in the forward direction that processes the input from left to right by acquiring the left context of the current input. At each input sequence, an input vector is given into the LSTM, and the output is generated based on:

$$h_t = f(h_{(t-1)}, x_t) \quad (10)$$

where x_t represents the input state, h_t represents the present state, and $h_{(t-1)}$ represents the prior state.

3.6. Bidirectional LSTM

Although the LSTM is unidirectional, it only acquires a small number of the context, which significantly reduces the classification performance. To improve the classification performance of the long-term dependencies of time series data without affecting latency is to handle the information bidirectionally [50]. A BiLSTM uses various layers to handle contextual information in both directions, forward and backward. It has a hidden layer with the forward sequence \vec{h} and the backward sequence \overleftarrow{h} for the left and right contexts, respectively [51]. It must be ensured that each subsequent layer is taken from both the forward and backward layers. In the proposed network, the output of the CNN was fed into the BiLSTM layer to accurately the normal and fractured images. Let's look more deeply at BiLSTM layers and thus at how they operate independently.

Sequence Input Layer

The BiLSTM network begins with a sequence input layer, and it is the primary feature of a BiLSTM network. A sequence input layer is responsible for transmitting sequential data to a network. The input size of a sequence input layer is 2.

BiLSTM Layer

DCNN findings are transferred to the BiLSTM layer for further feature extraction. Figure 3 illustrates two LSTM layers. The BiLSTM is structured with a forward and a backward LSTM layer. The forward LSTM may acquire preceding information from the input sequence, while the backward LSTM can acquire forthcoming information from the input sequence, and the results from both hidden layers are then integrated.

$$h_t = \vec{h} \oplus \overleftarrow{h} \quad (11)$$

where \oplus symbolizes the feature computation used to add the forward and backward output features. The first BiLSTM layer contains 2000 hidden units, whereas the subsequent BiLSTM layer contains 1000 hidden units. Although it can use both prior and subsequent knowledge, a BiLSTM is more suitable than the LSTM and RNN.

Dropout Layer

A dropout layer is integrated after both BiLSTM layers to avoid the overfitting problem; the dropout rate is 0.5, signifying that 50% of the information is discarded. The decrease of 1000 hidden units after the first layer of the BiLSTM and the decrease of 1000 hidden units after the second layer of the BiLSTM are done to avoid overfitting.

Fully connected layer

We attached the outcome extracted by the BiLSTM layer to a fully connected network comprising two layers to complete the DCNN-BiLSTM model.

Softmax Layer

The entire DCNN-BiLSTM architecture is shown in Figure 4. At the end of the architecture, the softmax activation function was used to classify the output from the preceding layer. The softmax layer predicted the class of the images as fractured or non-fractured. Table 3 represents the parameter setting for proposed bidirectional LSTM.

Table 3. The detailed information about the proposed bidirectional LSTM.

Number	Layer Name	Activations	Parameters
1	Sequence input layer	2	/
2	BiLSTM	2000	Input weights = 8000×1 Recurrent weights = 8000×2000 Bias = 8000×2
3	Dropout layer	2000	/
4	BiLSTM	1000	Input weights = 4000×2000 Recurrent weights = 4000×1000 Bias = 4000×1
5	Dropout layer	1000	/
6	Fully connected layer	2	Input weights = 2×1000 Bias = 2×1
7	Softmax	2	/

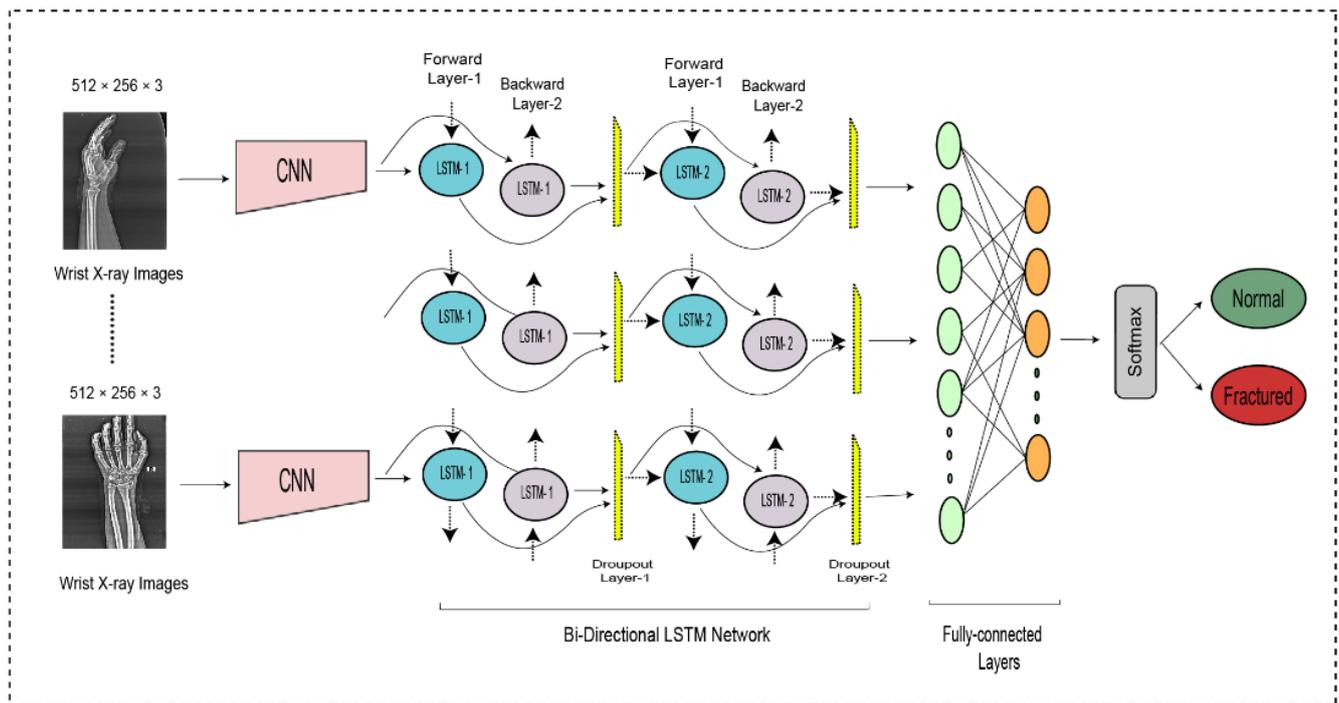


Figure 4. The bi-directional LSTM network for wrist fracture detection.

4. Results and Discussion

The proposed framework used a dataset with its original images at first in experiment 1 and then applied data augmentation to solve the class imbalance and data overfitting problem for deep learning in experiment 2. Both experiments used a different number of images but the same methods of classification. In each experiment, the DCNN method was applied to extract deep features and then fed to the BiLSTM network for classification. In total, two experiments and four methods were applied, with two methods in each experiment.

4.1. Dataset Description

The wrist fracture X-ray images of patients used in our research were acquired from the Mendeley dataset, which is openly available. Mendeley collects X-ray images from the Al-huda Digital X-ray Laboratory at Nishtar Road in Multan, Pakistan. The dataset consists of 111 fractured and 82 normal wrist X-ray images. The images captured in real time are of various sizes and in JPG format. All images were resized to ones with a resolution of 512×256 pixels to use in the system, and the dataset before and after pre-processing and augmentation is shown in Table 4.

Table 4. Dataset before and after pre-processing and augmentation.

Variables	Actual Data	Pre-Processed Data	Augmented Data
Wrist X-ray images	192	192	192
Fractured images	111	111	555
Normal images	82	82	410
Dimension	Different sizes, such as 344×596 , 252×650 , 328×624 , and 264×622	512×256	512×256
Format	JPG file	JPG file	JPG file
Total images	192	192	965

4.2. Performance Evaluation

In this work, after applying pre-processing, data augmentation, feature extraction and classification were performed on the dataset. The last stage was to determine the model's performance. There are four potential outputs when using a classifier on any case. These outputs are:

- True-positive (TP) outputs include positive images (fractured) that are accurately classified as fractured.
- True-negative (TN) outputs include normal images that are accurately classified as non-fractured.
- False-positive (FP) outputs include normal images that are inaccurately classified as fractured.
- False-negative (FN) outputs include positive images (fractured) that are inaccurately classified as non-fractured (normal).

The main purposes of measuring a classification model's prediction outcomes are (1) to evaluate the overall performance and (2) to improve the classifier's predictive potential by modifying the input variables of the model. In the mentioned description, the performance of the presented approach was analyzed in terms of accuracy, precision, sensitivity, specificity, the F-score, and kappa. Sensitivity indicates the percentage of all positive cases detected and evaluates the classifier's potential to detect positive cases, whereas recall is similar to sensitivity. Specificity reflects the percentage of all normal cases detected and indicates the classifier's capacity to comprehend normal cases. Precision shows the percentage of positive classes that are divided into positive cases. The F1-score is a comprehensive evaluation metric, and a significant value shows that the classifier is much more accurate. The Cohen kappa metric improves assurance in the outcomes. The formulas are mentioned next.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (12)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (13)$$

$$\text{Sensitivity} = \frac{TP}{FP + TN} \quad (14)$$

$$\text{Specificity} = \frac{TN}{FP + TN} \quad (15)$$

$$\text{F1 - Score} = \frac{2 \times (TP)}{(2 \times TP + FN + FP)} \quad (16)$$

$$\text{Kappa} = \frac{2 \times (TP \times TN - FN \times FP)}{(TP + FP) \times (FP + TN) + (TP + FN) \times (FN + TN)} \quad (17)$$

4.3. Experiment 1 (Non-Augmented Data)

To enhance the efficiency and results of the designed model, pre-processing was used in the first method in this study. As it is a major part of the classification model, the quality and significant knowledge gathered in the pre-processing process have a significant influence on the model's training. The experimental results obtained after applying the dilated CNN to the pre-processed data achieved an 84.48% accuracy, 87.50% precision, 84.85% sensitivity, 84% specificity, 86.15% F1-score, and 68.52% Cohen kappa. If we look at the different scores of the results achieved, we can analyze the true-positive over the false-positive score, which is precision, which was more than the accuracy score. However, the sensitivity and specificity were nearer to the accuracy score, which are rates of true positives and true negatives, respectively. The F1-score was higher at 86.15% compared to accuracy, which is good as it covers the class-imbalance-addressing measure too.

To enhance the outcomes of the dilated CNN, we transferred the findings into the LSTM model and achieved an accuracy of 86.21%, which increased as compared to the

DCNN method, although the purpose of the implemented LSTM is shown here as increased results by just feeding already trained features using the DCNN method. The sensitivity, F1-score, and Cohen kappa rates significantly improved in this method, whereas other measures remained the same. The results are shown in Table 5.

Table 5. Non-Augmented Data before and after LSTM.

Methods	Accuracy	Precision	Sensitivity	Specificity	F1-Score	Kappa
DCNN 1	84.48%	87.50%	84.85%	84%	86.15%	68.52%
DCNN-LSTM 1	86.21%	87.88%	87.88%	84%	87.88%	71.88%

4.4. Experiment 2 (Augmented Data)

In experiment 2, to increase the diversity of the dataset and to resolve the concerns of overfitting, the data augmentation method was applied in this study. The rotation operation of affine transformation was applied on the X-ray images' dataset. Both methods, dilated CNN and LSTM, were again applied to enhance the model's accuracy by increasing the volume, quality, and variety of data in the dataset and thus enable the proposed model to identify the data's intrinsic features quite comprehensively. The experimental results obtained after applying the dilated CNN on the augmented data achieved an accuracy of 86%, which is better than that of DCNN1 applied in experiment 1 on the non-augmented dataset. Similarly, other scores, such as sensitivity, F1-score, and Cohen kappa, also increased significantly. To improve the DCNN2 results, we transferred the observations into the BiLSTM. By using the BiLSTM, the model obtained an accuracy of 88.24%, which is better than that of the DCNN-LSTM1 methods applied in experiment 2. The other scores of the DCNN-LSTM 2 also improved and achieved up-to-the-mark results for wrist fracture detection. Training based on augmented data enhanced the computational performance of the model as compared to training without data augmentation.

The tabular description of experiment 2 results is shown in Table 6. The improvement of results showed that the model performance in terms of precise results improved by implementing a 2-step approach of classification by integrating CNN and LSTM methods.

Table 6. Experiment 2 results of augmented data before and after using the LSTM.

Methods	Accuracy	Precision	Sensitivity	Specificity	F1-Score	Kappa
DCNN-2	86.54%	86.93%	92.17%	81.30%	89.47%	74.25%
DCNN-LSTM 2	88.24%	87.93%	92.17%	82.93%	90%	75.7%

5. Conclusions

Many countries are experiencing resource limitations as fracture cases are increasing on a regular basis. It is essential to detect every individual positive case. The increasing rates of accidental cases have increased the need for medical staff in the emergency department, which is badly needed in many lower-income countries. The existing staff is less in number, and due to a shortage of doctors, the overload could lead them to less accurately detect fractures. Therefore, there is a need to develop a computer-assisted method for detection of body organ fractures in order to facilitate the second-option diagnosis for doctors. This study proposed a CAD system based on a DCNN-LSTM network approach to detect wrist fractures from X-ray images. For the proposed research, a dataset of 111 fractured and 82 normal wrist X-ray images from Mendeleev was used. Image data pre-processing operations were applied; for example, image enhancement was used to enhance the brightness of the X-ray images and to reduce the distortion. To prevent overfitting and to increase the diversity of the dataset, the data augmentation technique was used. We attempted to maintain the class balance by generating an equal number of images for both classes during the training process. Two experiments were conducted in this study to prove the improvement in results using data augmentation, in which a deep learning

method named “dilated CNN” was used to enhance the receptive field of the CNN with a dilated factor to extract valuable features from X-ray images. Secondly, an LSTM network was used as a classifier for the detection of wrist fracture. Furthermore, the capabilities of the proposed DCNN-LSTM models with and without augmentation were compared with those of other deep learning models. The overall accuracy of the proposed model with augmentation was 88.24%, and it was almost more than that without augmentation (86.21%). As the incidence of fracture cases continues to increase, we believe that our model will be useful for medical evaluation. An automated detection system will improve detection and classification and will contribute to the solution of the incident load issue faced by medical experts.

However, the limitations of data could be a limitation for the proposed method, as with increasing data, the applied approach may need to be fine-tuned in terms of architecture and parameter differences in order to achieve higher results. For the future, big data and big network architectures could be applied to different body organ fracture images to provide CAD systems for them.

Author Contributions: Conceptualization, T.R.; Methodology, M.S.Z.; Validation, N.-u.-R.; Investigation, S.K.; Resources, T.M. and H.T.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Brink, M.; Steenbakkens, A.; Holla, M.; de Rooy, J.; Cornelisse, S.; Edwards, M.J.; Prokop, M. Single-shot CT after wrist trauma: Impact on detection accuracy and treatment of fractures. *Skelet. Radiol.* **2019**, *48*, 949–957. [[CrossRef](#)] [[PubMed](#)]
2. Üreten, K.; Sevinç, H.F.; İğdeli, U.; Onay, A.; Maraş, Y. Use of deep learning methods for hand fracture detection from plain hand radiographs. *Turk. J. Trauma Emerg. Surg.* **2022**, *28*, 196–201. [[CrossRef](#)] [[PubMed](#)]
3. Bagaria, R.; Wadhvani, S.; Wadhvani, A.K. Bone fractures detection using support vector machine and error backpropagation neural network. *Optik* **2021**, *247*, 168021. [[CrossRef](#)]
4. Umadevi, N.; Geethalakshmi, S. Multiple classification system for fracture detection in human bone x-ray images. In Proceedings of the 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12), Coimbatore, India, 26–28 July 2012.
5. Cao, Y.; Wang, H.; Moradi, M.; Prasanna, P.; Syeda-Mahmood, T.F. Fracture detection in x-ray images through stacked random forests feature fusion. In Proceedings of the 2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI), Brooklyn, NY, USA, 16–19 April 2015.
6. Greenspan, H.; Van Ginneken, B.; Summers, R.M. Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Trans. Med. Imaging* **2016**, *35*, 1153–1159. [[CrossRef](#)]
7. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [[CrossRef](#)] [[PubMed](#)]
8. Esteva, A.; Robicquet, A.; Ramsundar, B.; Kuleshov, V.; DePristo, M.; Chou, K.; Cui, C.; Corrado, G.; Thrun, S.; Dean, J. A guide to deep learning in healthcare. *Nat. Med.* **2019**, *25*, 24–29. [[CrossRef](#)] [[PubMed](#)]
9. Olczak, J.; Fahlberg, N.; Maki, A.; Razavian, A.S.; Jilert, A.; Stark, A.; Sköldenberg, O.; Gordon, M. Artificial intelligence for analyzing orthopedic trauma radiographs: Deep learning algorithms—Are they on par with humans for diagnosing fractures? *Acta Orthop.* **2017**, *88*, 581–586. [[CrossRef](#)]
10. Yahalomi, E.; Chernofsky, M.; Werman, M. Detection of distal radius fractures trained by a small set of X-ray images and Faster R-CNN. In Proceedings of the Intelligent Computing—Proceedings of the Computing Conference, London, UK, 16–17 July 2019.
11. Xie, H.; Yang, D.; Sun, N.; Chen, Z.; Zhang, Y. Automated pulmonary nodule detection in CT images using deep convolutional neural networks. *Pattern Recognit.* **2019**, *85*, 109–119. [[CrossRef](#)]
12. Ragab, D.A.; Sharkas, M.; Marshall, S.; Ren, J. Breast cancer detection using deep convolutional neural networks and support vector machines. *PeerJ* **2019**, *7*, e6201. [[CrossRef](#)]
13. Zhang, J.; Li, Z.; Yan, S.; Cao, H.; Liu, J.; Wei, D. An Algorithm for Automatic Rib Fracture Recognition Combined with nnU-Net and DenseNet. *Evid.-Based Complement. Altern. Med.* **2022**, *2022*, 5841451. [[CrossRef](#)]

14. Urakawa, T.; Tanaka, Y.; Goto, S.; Matsuzawa, H.; Watanabe, K.; Endo, N. Detecting intertrochanteric hip fractures with orthopedist-level accuracy using a deep convolutional neural network. *Skelet. Radiol.* **2019**, *48*, 239–244. [[CrossRef](#)]
15. Joshi, D.; Singh, T.P.; Joshi, A.K. Deep learning-based localization and segmentation of wrist fractures on X-ray radiographs. *Neural Comput. Appl.* **2022**, *34*, 19061–19077. [[CrossRef](#)]
16. Kitamura, G.; Chung, C.Y.; Moore, B.E. Ankle fracture detection utilizing a convolutional neural network ensemble implemented with a small sample, de novo training, and multiview incorporation. *J. Digit. Imaging* **2019**, *32*, 672–677. [[CrossRef](#)] [[PubMed](#)]
17. Doerr, S.A.; Weber-Levine, C.; Hersh, A.M.; Awosika, T.; Judy, B.; Jin, Y.; Raj, D.; Liu, A.; Lubelski, D.; Jones, C.K. Automated prediction of the Thoracolumbar Injury Classification and Severity Score from CT using a novel deep learning algorithm. *Neurosurg. Focus* **2022**, *52*, E5. [[CrossRef](#)] [[PubMed](#)]
18. El-Saadawy, H.; Tantawi, M.; Shedeed, H.A.; Tolba, M.F. Deep Learning Method for Bone Abnormality Detection Using Multi-View X-rays. In Proceedings of the International Conference on Artificial Intelligence and Computer Vision, Settat, Morocco, 28–30 June 2021.
19. Chung, S.W.; Han, S.S.; Lee, J.W.; Oh, K.-S.; Kim, N.R.; Yoon, J.P.; Kim, J.Y.; Moon, S.H.; Kwon, J.; Lee, H.-J. Automated detection and classification of the proximal humerus fracture by using deep learning algorithm. *Acta Orthop.* **2018**, *89*, 468–473. [[CrossRef](#)]
20. Schudlo, L.C.; Xie, Y.; Small, K.; Graf, B. A novel CNN+ LSTM approach to classify frontal chest x-rays for spine fractures. In Proceedings of the Medical Imaging 2022: Computer-Aided Diagnosis, San Diego, CA, USA, 20 February–28 March 2022.
21. El-Saadawy, H.; Tantawi, M.; Tolba, M. Bone X-Rays Classification and Abnormality Detection using Xception Network. *Int. J. Intell. Comput. Inf. Sci.* **2021**, *21*, 82–95. [[CrossRef](#)]
22. Solovyova, A.; Solovyov, I. X-ray bone abnormalities detection using MURA dataset. *arXiv* **2020**, arXiv:2008.03356.
23. Myint, W.W.; Tun, K.S.; Tun, H.M. Analysis on leg bone fracture detection and classification using X-ray images. *Mach. Learn. Res.* **2018**, *3*, 49–59. [[CrossRef](#)]
24. Yao, L.; Guan, X.; Song, X.; Tan, Y.; Wang, C.; Jin, C.; Chen, M.; Wang, H.; Zhang, M. Rib fracture detection system based on deep learning. *Sci. Rep.* **2021**, *11*, 23513. [[CrossRef](#)]
25. Choi, J.W.; Cho, Y.J.; Ha, J.Y.; Lee, Y.Y.; Koh, S.Y.; Seo, J.Y.; Choi, Y.H.; Cheon, J.-E.; Phi, J.H.; Kim, I. Deep learning-assisted diagnosis of pediatric skull fractures on plain radiographs. *Korean J. Radiol.* **2022**, *23*, 343. [[CrossRef](#)]
26. Tomita, N.; Cheung, Y.Y.; Hassanpour, S. Deep neural networks for automatic detection of osteoporotic vertebral fractures on CT scans. *Comput. Biol. Med.* **2018**, *98*, 8–15. [[CrossRef](#)] [[PubMed](#)]
27. Kim, D.; MacKinnon, T. Artificial intelligence in fracture detection: Transfer learning from deep convolutional neural networks. *Clin. Radiol.* **2018**, *73*, 439–445. [[CrossRef](#)] [[PubMed](#)]
28. Kim, K.C.; Cho, H.C.; Jang, T.J.; Choi, J.M.; Seo, J.K. Automatic detection and segmentation of lumbar vertebrae from X-ray images for compression fracture evaluation. *Comput. Methods Programs Biomed.* **2021**, *200*, 105833. [[CrossRef](#)] [[PubMed](#)]
29. Guan, B.; Zhang, G.; Yao, J.; Wang, X.; Wang, M. Arm fracture detection in X-rays based on improved deep convolutional neural network. *Comput. Electr. Eng.* **2020**, *81*, 106530. [[CrossRef](#)]
30. Thian, Y.L.; Li, Y.; Jagmohan, P.; Sia, D.; Chan, V.E.Y.; Tan, R.T. Convolutional neural networks for automated fracture detection and localization on wrist radiographs. *Radiol. Artif. Intell.* **2019**, *1*, e180001. [[CrossRef](#)] [[PubMed](#)]
31. Guan, B.; Yao, J.; Zhang, G.; Wang, X. High fracture detection using deep learning method based on new dilated convolutional feature pyramid network. *Pattern Recognit. Lett.* **2019**, *125*, 521–526. [[CrossRef](#)]
32. Ebsim, R.; Naqvi, J.; Cootes, T. Fully automatic detection of distal radius fractures from posteroanterior and lateral radiographs. In *Computer Assisted and Robotic Endoscopy and Clinical Image-Based Procedures*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 91–98.
33. Li, L.; Song, X.; Guo, Y.; Liu, Y.; Sun, R.; Zou, H.; Zhou, H.; Fan, X. Deep convolutional neural networks for automatic detection of orbital blowout fractures. *J. Craniofacial Surg.* **2020**, *31*, 400–403. [[CrossRef](#)]
34. Zeelan Basha, C.; Maruthi Padmaja, T.; Balaji, G. Automatic X-ray image classification system. In *Smart Computing and Informatics*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 43–52.
35. Cheng, C.-T.; Ho, T.-Y.; Lee, T.-Y.; Chang, C.-C.; Chou, C.-C.; Chen, C.-C.; Chung, I.; Liao, C.-H. Application of a deep learning algorithm for detection and visualization of hip fractures on plain pelvic radiographs. *Eur. Radiol.* **2019**, *29*, 5469–5477. [[CrossRef](#)]
36. Yadav, D.; Rathor, S. Bone fracture detection and classification using deep learning approach. In Proceedings of the 2020 International Conference on Power Electronics & IoT Applications in Renewable Energy and its Control (PARC), Mathura, India, 28–29 February 2020.
37. Ebsim, R.; Naqvi, J.; Cootes, T.F. Automatic detection of wrist fractures from posteroanterior and lateral radiographs: A deep learning-based approach. In Proceedings of the International Workshop on Computational Methods and Clinical Applications in Musculoskeletal Imaging, Granada, Spain, 16 September 2018.
38. Oyeranmi, A.; Ronke, B.; Mohammed, R.; Edwin, A. Detection of Fracture Bones in X-ray Images Categorization. *Kwara State Univ. Malete Nigeria. Br. J. Math. Comput. Sci.* **2020**, *35*, 1–11. [[CrossRef](#)]
39. San Myint, K.T. Implementation of Lower Leg Bone Fracture Detection from X-ray Images. *Int. J. Trend Sci. Res. Dev.* **2019**, *3*, 2411–2415.
40. Mondol, T.C.; Iqbal, H.; Hashem, M. Deep CNN-based ensemble CADx model for musculoskeletal abnormality detection from radiographs. In Proceedings of the 2019 5th International Conference on Advances in Electrical Engineering (ICAEE), Dhaka, Bangladesh, 26–28 September 2019.

41. Lindsey, R.; Daluisi, A.; Chopra, S.; Lachapelle, A.; Mozer, M.; Sicular, S.; Hanel, D.; Gardner, M.; Gupta, A.; Hotchkiss, R. Deep neural network improves fracture detection by clinicians. *Proc. Natl. Acad. Sci. USA* **2018**, *115*, 11591–11596. [[CrossRef](#)]
42. Chittajallu, S.M.; Mandalaneni, N.L.D.; Parasa, D.; Bano, S. Classification of Binary Fracture Using CNN. In Proceedings of the 2019 Global Conference for Advancement in Technology (GCAT), Bangalore, India, 18–20 October 2019.
43. Vasilakakis, M.; Iosifidou, V.; Fragkaki, P.; Iakovidis, D. Bone fracture identification in x-ray images using fuzzy wavelet features. In Proceedings of the 2019 IEEE 19th International Conference on Bioinformatics and Bioengineering (BIBE), Athens, Greece, 28–30 October 2019.
44. Dimililer, K. IBFDS: Intelligent bone fracture detection system. *Procedia Comput. Sci.* **2017**, *120*, 260–267. [[CrossRef](#)]
45. Hržić, F.; Tschauer, S.; Sorantin, E.; Štajduhar, I. Fracture Recognition in Paediatric Wrist Radiographs: An Object Detection Approach. *Mathematics* **2022**, *10*, 2939. [[CrossRef](#)]
46. Zhang, X.; Zou, Y.; Shi, W. Dilated convolution neural network with LeakyReLU for environmental sound classification. In Proceedings of the 2017 22nd international conference on digital signal processing (DSP), London, UK, 23–25 August 2017.
47. Raghavendra, U.; Bhat, N.S.; Gudigar, A.; Acharya, U.R. Automated system for the detection of thoracolumbar fractures using a CNN architecture. *Future Gener. Comput. Syst.* **2018**, *85*, 184–189. [[CrossRef](#)]
48. Li, X.; Zhai, M.; Sun, J. DDCNNC: Dilated and depthwise separable convolutional neural Network for diagnosis COVID-19 via chest X-ray images. *Int. J. Cogn. Comput. Eng.* **2021**, *2*, 71–82. [[CrossRef](#)]
49. Passricha, V.; Aggarwal, R.K. A hybrid of deep CNN and bidirectional LSTM for automatic speech recognition. *J. Intell. Syst.* **2020**, *29*, 1261–1274. [[CrossRef](#)]
50. Xu, X.; Jeong, S.; Li, J. Interpretation of electrocardiogram (ECG) rhythm by combined CNN and BiLSTM. *IEEE Access* **2020**, *8*, 125380–125388. [[CrossRef](#)]
51. Kavianpour, P.; Kavianpour, M.; Jahani, E.; Ramezani, A. A cnn-bilstm model with attention mechanism for earthquake prediction. *arXiv* **2021**, arXiv:2112.13444.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.