Predicting skull fractures via CNN with classification algorithms

Md Moniruzzaman Emon
Department of Computer Science and Engineering, Shahjalal University of Science and Technology
Sylhet, Bangladesh
moniruzzaman5673@gmail.com

Tareque Rahman Ornob
Department of Computer Science and Engineering, Shahjalal University of Science and Technology
Sylhet, Bangladesh
ornob011@gmail.com

Moqsadur Rahman
Department of Computer Science and Engineering, Shahjalal University of Science and Technology
Sylhet, Bangladesh
moqsad-cse@sust.edu

ABSTRACT

Computer Tomography (CT) images have become quite important to diagnose diseases. CT scan slice contains a vast amount of data that may not be properly examined with the requisite precision and speed using normal visual inspection. A computer-assisted skull fracture classification expert system is needed to assist physicians. Convolutional Neural Networks (CNNs) are the most extensively used deep learning models for image categorization since most often time they outperform other models in terms of accuracy and results. The CNN models were then developed and tested, and several convolutional neural network (CNN) architectures were compared. ResNet50, which was used for feature extraction combined with a gradient boosted decision tree machine learning algorithm to act as a classifier for the categorization of skull fractures from brain CT scans into three fracture categories, had the best overall F1-score of 96%, Hamming Score of 95%, Balanced accuracy Score of 94% & ROC AUC curve of 96% for the classification of skull fractures.

CCS CONCEPTS

• **Computing methodologies** → **Object recognition**; **Computer vision problems**; **Computer vision**; **Artificial intelligence**;

KEYWORDS

Skull fracture, Deep learning, Convolutional Neural Network, Medical Image Analysis, Computer vision

ACM Reference Format:
Md Moniruzzaman Emon, Tareque Rahman Ornob, and Moqsadur Rahman. 2022. Predicting skull fractures via CNN with classification algorithms. In Proceedings of 2nd International Conference on Computing Advancements, March 10–12, 2022, Dhaka, Bangladesh, 8 pages. https://doi.org/10.1145/3542954.3543017

1 INTRODUCTION

Because a significant hit or blow to the head might result in a skull fracture as well as a brain injury, it's critical to figure out what kind of brain injury the skull fracture might cause as soon as possible so that the patient can get the care he or she needs. If the fracture occurs over a major blood vessel, significant bleeding may occur inside the brain, so head injury patients with skull fractures have far more intracranial hematomas than those without fractures.[10, 13] Classification is a method of identifying the lesion caused by skull fractures[7]. Computed tomography (CT) has become the primary diagnostic tool for suspected skull or brain injuries.CT images and radiology reports, in general, provide more information to a physician, allowing them to make an informed decision. In the typical diagnosis approach, the radiologist examines the image and notes the results, and the physician then chooses a treatment based on the diagnosis. All of this takes a lot of time. A radiologist uses a CT scan to assess whether there is a skull fracture and which category it belongs to. On CT scans, however, the skull fracture has the following characteristics: fractures typically appear as narrow slits, fractures can be seen in a variety of locations and lengths, and a large proportion of fractures are microscopic. All of these factors might make manual diagnosis and grading of skull fractures time-consuming and challenging. As a result, it's vital to propose a reliable automated skull fracture categorization and detection system. Skull fractures might be automatically detected and classified, which could aid in the diagnosis of other anomalies in CT scan brain imaging. Furthermore, many hospitals around the world are understaffed, resulting in delays in evaluating CT scan images. In the CQ-500 dataset [1], their three radiologists couldn't agree on whether a patient's skull was fractured or normal in many situations, much alone classify the fracture as a calvarial fracture or another fracture deterministically. Automatic classification could assist clinicians in a short-staffed hospital in identifying the most critical patients and prioritizing their treatment. Information contained in the Computed Tomography (CT) images is very important for assessing the severity and prognosis of the Classification of skull fracture. Each skull CT scan includes a large number of slices. To give clinicians with suggestions and predictions on diagnostic judgments and treatment planning, highly efficient and automated computational approaches are urgently needed to process and analyze all accessible medical data. These facts provide the motivation for this work. In addition to detecting fractured or normal cases from CT scan pictures, we provide a model for automated detection of common three skull fractures. These advantages could provide a good platform for retrieving content-based medical images for medical instruction or diagnosis

2 RELATED WORKS

Some approaches for identifying skull fractures have already been proposed. Shao et al.[11] have focused their efforts on CT brain segmentation for automated skull fracture diagnosis. They proposed

ICCA 2022, March 10–12, 2022, Dhaka, Bangladesh

ACM ISBN 978-1-4503-9734-6/22/03…$15.00
https://doi.org/10.1145/3542954.3543017

utilizing a region-growing approach to segment the brain image and then using the entropy feature to construct guidelines for identifying skull fractures. This strategy has received a lot of positive feedback. Its complexity and performance, on the other hand, can be simplified and improved computationally. Zaki et al.[14] used Sobel edge detection technique for the diagnosis of skull fractures. Despite the fact that the Sobel edge detection approach is superior in a variety of ways, it does yield some misshaping lines in some cases. It's worth mentioning that this method is incapable of handling large features. Prewitt is a straightforward approach to detect the boundaries based on gradient magnitude. However, when the amplitude of the gradient decreases, the accuracy will almost certainly decrease. Abubacker et al.[4] used histogram-based thresholding and nearby pixel connection search, presented a simple and fast automatic solution in Digital Imaging and Communications in Medicine (DlCOM) to extract the skull bone and diagnose the fracture. The experimental results for this approach are consistent, with a high detection rate. Chilamkurthy et al.[5] have developed a deep learning approach for detecting cerebral bleeding and its kinds (intraparenchymal, intraventricular, subdural, extradural, and subarachnoid), as well as calvarial fractures, midline shift, and mass effect. They demonstrated that deep learning systems were capable of doing this task with excellent precision. Emon et al.[6] developed a model called SkullNetV1 to classify seven skull fractures. However, SkullNetV1 lacks score in some classes. Kuang et al.[8] have proposed a way for more precisely detecting skull fractures in a short period of time. Skull R-CNN is the name of the proposed approach. Skull R-CNN has fewer false positives than earlier research on skull fracture diagnosis while keeping good sensitivity. Yamada et al.[12] developed a unique approach for automatically detecting linear skull fractures on head CT images by recognizing crack lines They used two types of phantoms to do a rudimentary examination. A crack line with a width of 0.35 mm was found in their experiment with a digital phantom.

All of the methods outlined above totally focused on the skull's local properties. As far as we know, there are just two methods for detecting linear skull fractures automatically. Only one method exists to automate the detection of skull fractures but no one has automated the classification of Linear and Depressed skull fractures with very high accuracy.

3 DATASET

With respective permission, we collected 142 patients' head CT scan from Medinova Medical Services Ltd.[3] and Ibn Sina Hospital Sylhet Limited.[2] Every CT scan image was in DICOM format, with 512x512 pixels in slice thicknesses of 0.75 mm, 1.0 mm, and 5.0 mm. The amount of 1.0 mm slices was 25678, those were used to construct the dataset. The CT images of all 142 patients were evaluated and annotated by expert radiologists. First, the radiologists divided the dataset into the fracture and normal cases (8628 CT scan slices), then the fractured instances were further categorized of into two categories i.e. Linear Fracture (4578 CT scan slices) and Depressed Fracture (8492 CT scan slices). The paired data (15189 CT scan slices and radiology reports of 107 patients) were used as training sets. The remaining 35 CT scans (6509 CT scan slices) were used as test sets while the associated radiology reports were used to evaluate the predictions. The data distribution between classes was moderately imbalanced.

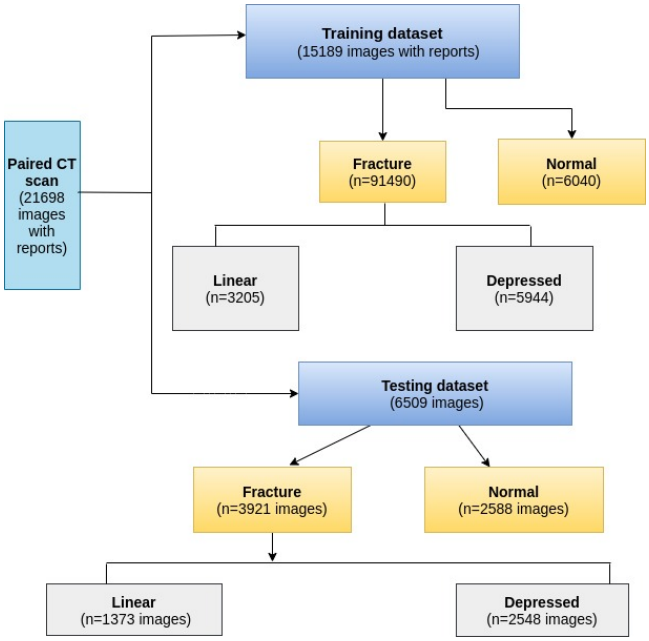

Figure 1: Data characteristics. Number of CT images and radiology reports for training and testing the system.

4 METHODOLOGY

4.1 Data Pre-processing

1 mm DICOM sequence from each DICOM series was extracted automatically. Best DICOM slices were annotated by the respective radiologists. During the preparation of images, several processes were taken. Transforming to HU, Removing Noises, Tilt Correction, Cropping Images, and Padding are the steps. Model accuracy improved significantly when these preprocessing techniques were applied to data. We converted our DICOM picture data to Hounsfield Unit form after loading it. Using Rescale Intercept and Rescale Slope headings, we were able to retrieve the HU. Removing noises is critical since the data is better after implementation, allowing us to view it more clearly. Tilt correction is the proposed alignment of the brain picture. When brain CT pictures are tilted, it might cause misalignment in medical applications. It's significant since the model can see all of the data via the same alignment when it's being trained. Cropping an image is required to center the brain image and remove superfluous sections of the image. Additionally, within the overall image, various brain images may be positioned in different locations. We ensured that almost all of the images are in the same area inside the general image by cropping it and adding pads. Each patient's CT scan picture slices were given a unique ID in a CSV file along with the slice class i.e. Linear Fracture, Depressed Fracture, or normal cases in another column. LabelEncoder was used to transform the string format of the patients' condition class into a one-row matrix i.e. 0, 1, 2. The encoded matrix was

then converted into a hot one encoded matrix by LabelBinarizer. Every CT scan image was stored in a NumPy array. Data was highly imbalanced across the classes as shown in the Figure 2:
Class=0 (Depressed Fracture), n=5944 (39.134%)
Class=1 (Linear Fracture), n=3205 (21.101%)
Class=2 (Not Fractured), n=6040 (39.766%)

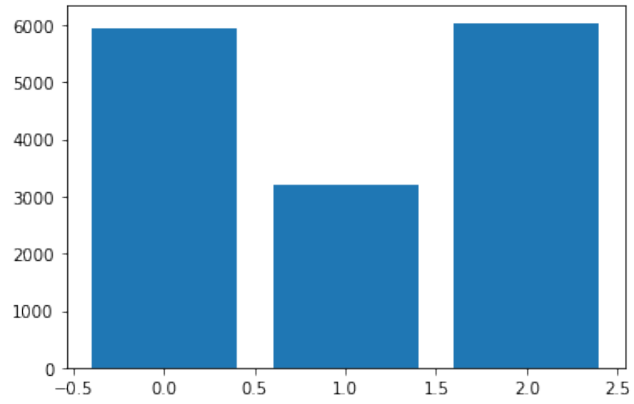

Figure 2: Data Distribution across the classes before balancing

To tackle this problem, two resampling techniques were implemented, Random oversampling and Random undersampling. Random oversampling involves randomly duplicating examples in the minority class, whereas random undersampling involves randomly deleting examples from the majority class. A pipeline was defined that first oversamples the minority class and under samples the majority class.

After applying this technique, the dataset became balanced as shown in the Figure 3:
Class=0 (Depressed Fracture), n=6040 (33.333%)
Class=1 (Linear Fracture), n=6040 (33.333%)
Class=2 (Not Fractured), n=6040 (33.333%)
A pie chart is given in Figure 4 to show the full transformation.

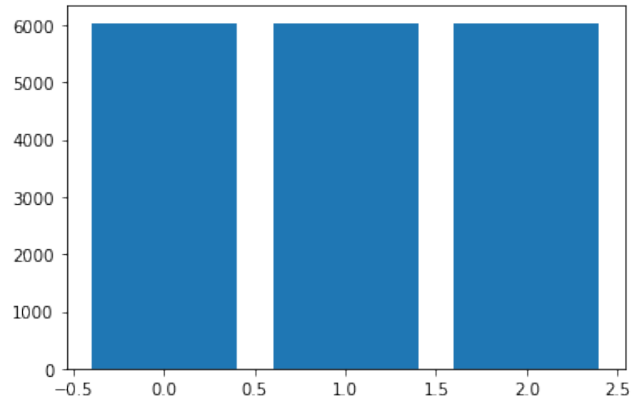

Figure 3: Data Distribution across the classes after balancing

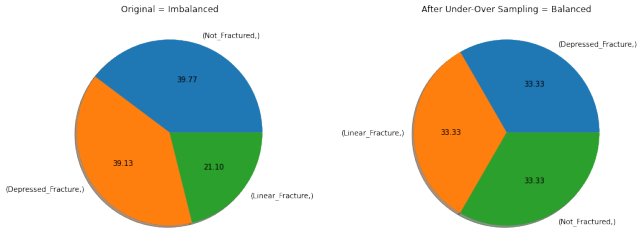

Figure 4: Data Distribution across the classes before & after balancing

4.2 Data Visualization

The estimated (Gaussian) noise standard deviation is 0.02 in our dataset. So, our data is almost noise-free, no other denoising technique is necessary to further denoise the data. To visualize our statement clearly, various noise filtering algorithm was performed on the dataset.

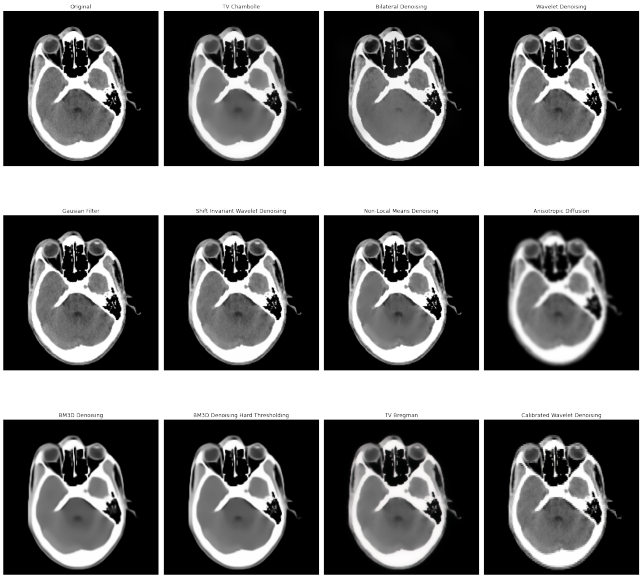

Figure 5: Visualization of an original random 2d slice after performing denoising

We can see from Figure 5 our original data is much clearer than after performing denoising filters.

4.3 Implementation and Graph

From different model-based approaches, we implemented:
Convolutional Neural Network (Transfer Learning Based)
Bagging algorithm
Decision Tree
Support Vector Machine

Convolutional Neural Network:
Xception, InceptionV3, ResNet50, and InceptionResNetV2 was our preferred CNN model due to their state-of-the-art architecture. Every model CNN was imported from TensorFlow and was fine-tuned by:

- Loading weights from available pre-trained models, included with Keras library.
- Stacking another network for training on top of any layers
- Inserting a layer in the middle of other layers
- Freezing multiple layers
- Removing multiple layers and inserting a new one in the middle

Learning Rate was kept low for Nadam, Adam, and SGD, momentum was used to avoid local minimum. Categorical Crossentropy was used as the loss function, the activation function was relu and the activation function in the output layer was softmax. F1-score was chosen as the metric. The dense layer was composed of 128 and 3 units, with l2 kernel regulizer for InceptionV3.

Training vs Validation Loss Graphs:

Xception:
Xception performed reasonably well until the 30 epoch. We can

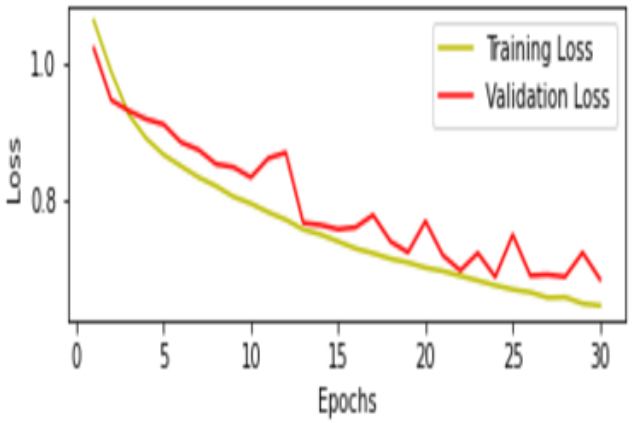

Figure 6: Cross Entropy loss of Xception

see that the validation curve is fluctuating because the validation dataset is quite small compared to the training dataset.

Table 1: Xception

Model	F1 Score (micro avg)	Hamming Score	Hamming Loss	Balanced Accuracy Score	Roc Auc	Kappa Score	Log Loss
Xception	0.37	0.36	0.42	0.39	0.55	0.105	21.87

InceptionV3:
The same result as Xception, but validation loss is less fluctuating.

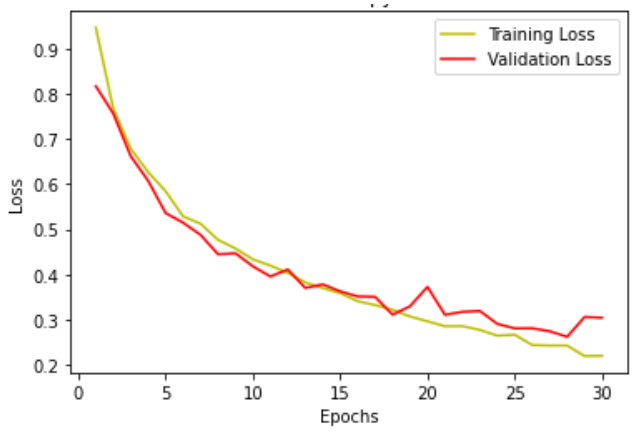

Figure 7: Cross Entropy loss of InceptionV3

Table 2: InceptionV3

Model	F1 Score (micro avg)	Hamming Score	Hamming Loss	Balanced Accuracy Score	Roc Auc	Kappa Score	Log Loss
InceptionV3	0.37	0.37	0.42	0.38	0.49	0.107	21.61

ResNet50:
Among four CNN model, ResNet50 performed best nevertheless not learning at the beginning. In the end, validation and training loss comes closer.

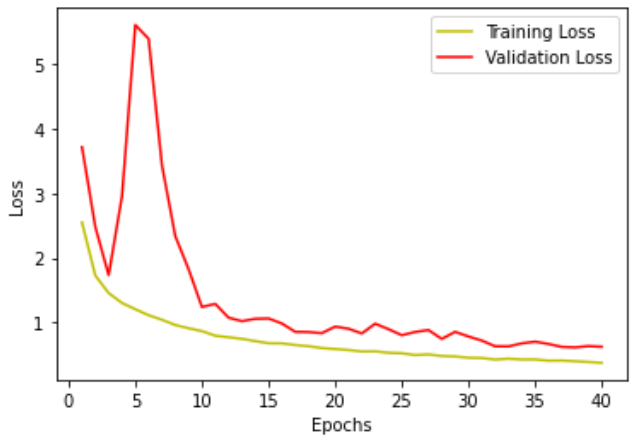

Figure 8: Cross Entropy loss of ResNet50

InceptionResNetV2:
InceptionResNetV2 took a lot of time to complete training as the parameter is around 54 million. It was trained until epoch 30, but

Table 3: ResNet50

Model	F1 Score (micro avg)	Hamming Score	Hamming Loss	Balanced Accuracy Score	Roc Auc	Kappa Score	Log Loss
ResNet50	0.39	0.387	0.408	0.32	0.49	-0.014	21.178

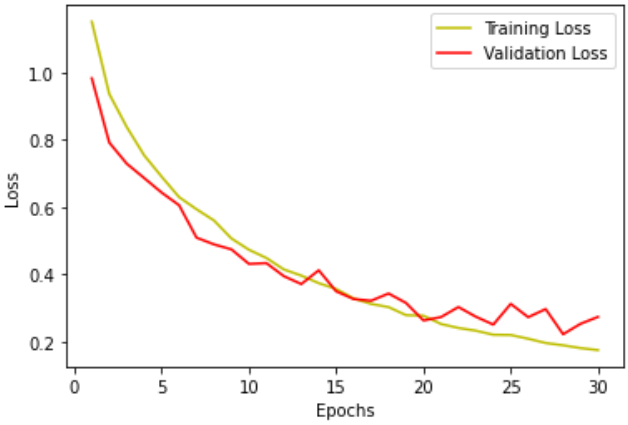

Figure 9: Cross Entropy loss of InceptionResNetV2

Table 4: InceptionResNetV2

Model	F1 Score (micro avg)	Hamming Score	Hamming Loss	Balanced Accuracy Score	Roc Auc	Kappa Score	Log Loss
InceptionResnetV2	0.37	0.37	0.42	0.31	0.5	-0.03	21.7

from 25 number epoch it started to fluctuate greatly.

Bagging algorithm:
Random forest is a machine learning technique that combines many different decision trees to get a more accurate and reliable forecast. The ideal split for each node is determined using a set of randomly generated candidate variables during the tree-building process. Following trees in the bagging process are not dependent on prior trees, and each one is built separately using a bootstrap sample of the data set. Finally, a simple majority vote is used to make the prediction.

Random Forest Classifier was used with CNN models instead of the dense layer as the Neural Networks will require much more data. 200 trees were used in the forest before taking the maximum voting. The extracted features from CNN were flattened to feed into Random Forest Classifier along with the target variable.

Decision Tree: XGBoost is a well-known gradient boosting approach (ensemble) that improves the performance and speed of tree-based machine learning algorithms (sequential decision trees). In Ensemble Learning, XGBoost is classified as a boosting strategy. Ensemble learning combines different models into a collection of predictors to improve prediction accuracy. The faults created by prior models are attempted to be repaired by subsequent models by adding weights to the models in the boosting strategy.

Table 5: Bagging algorithm

Model	F1 Score (micro avg)	Hamming Score	Hamming Loss	Balanced Accuracy Score	Roc Auc	Kappa Score	Log Loss
Inceptionv3 + RandomForest	0.85	0.84	0.10	0.78	0.85	0.75	5.38
ResNet50 + RandomForest	0.94	0.94	0.03	0.92	0.945	0.9	1.97
Xception + RandomForest	0.90	0.90	0.06	0.86	0.90	0.84	3.47
InceptionResnetv2 + RandomForest	0.88	0.87	0.08	0.82	0.87	0.80	4.30

XGBoost was used as the classifier along with CNN. 500 trees were used in the XGBoost.

Table 6: Decision Tree

Model	F1 Score (micro avg)	Hamming Score	Hamming Loss	Balanced Accuracy Score	Roc Auc	Kappa Score	Log Loss
Inceptionv3 + XgBoost	0.91	0.91	0.05	0.88	0.91	0.86	3.09
ResNet50 + XgBoost	0.96	0.95	0.02	0.94	0.96	0.93	1.47
Xception + XgBoost	0.93	0.92	0.04	0.90	0.93	0.88	2.48
InceptionResnetv2 + XgBoost	0.92	0.92	0.05	0.90	0.93	0.88	2.65

Support Vector Machine The purpose of the Linear SVC (Support Vector Classifier) is to fit the data provided and generate a "best fit" hyperplane that divides or categorizes the data. Following that, we can input some features to the classifier to check what the "predicted" class is after we've obtained the hyperplane. Despite the fact that SVC and Linear SVC are supposed to optimize the same issue, all liblinear estimators penalize the intercept, but libsvm estimators do not. LinearSVC's underlying estimators are liblinear, which penalizes the intercept. SVC makes use of libsvm estimators, which do not. Liblinear estimators are tailored for a linear (special) scenario, hence they converge faster than libsvm on vast amounts of data. This is because the linear kernel is a particular situation that is optimized for in Liblinear but not in Libsvm. The One-vs-All (also known as One-vs-Rest) multiclass reduction is used by LinearSVC, whereas the One-vs-One multiclass reduction is used by SVC. SVC fits N * (N - 1) / 2 models to multi-class classification problems, where N is the number of classes. Linear SVC, on the other hand, only fits N models.

Linear SVC were employed as the classifier for the CNN's extracted features.

Table 7: Support Vector Machine

Model	F1 Score (micro avg)	Hamming Score	Hamming Loss	Balanced Accuracy Score	Roc Auc	Kappa Score	Log Loss
Inceptionv3 + Linear SVC	0.90	0.897	0.07	0.88	0.91	0.84	3.55
ResNet50 + Linear SVC	0.93	0.925	0.04	0.925	0.945	0.88	2.56
Xception + Linear SVC	0.91	0.92	0.05	0.91	0.93	0.88	2.62
InceptionResnetv2 + Linear SVC	0.92	0.92	0.05	0.90	0.93	0.87	2.78

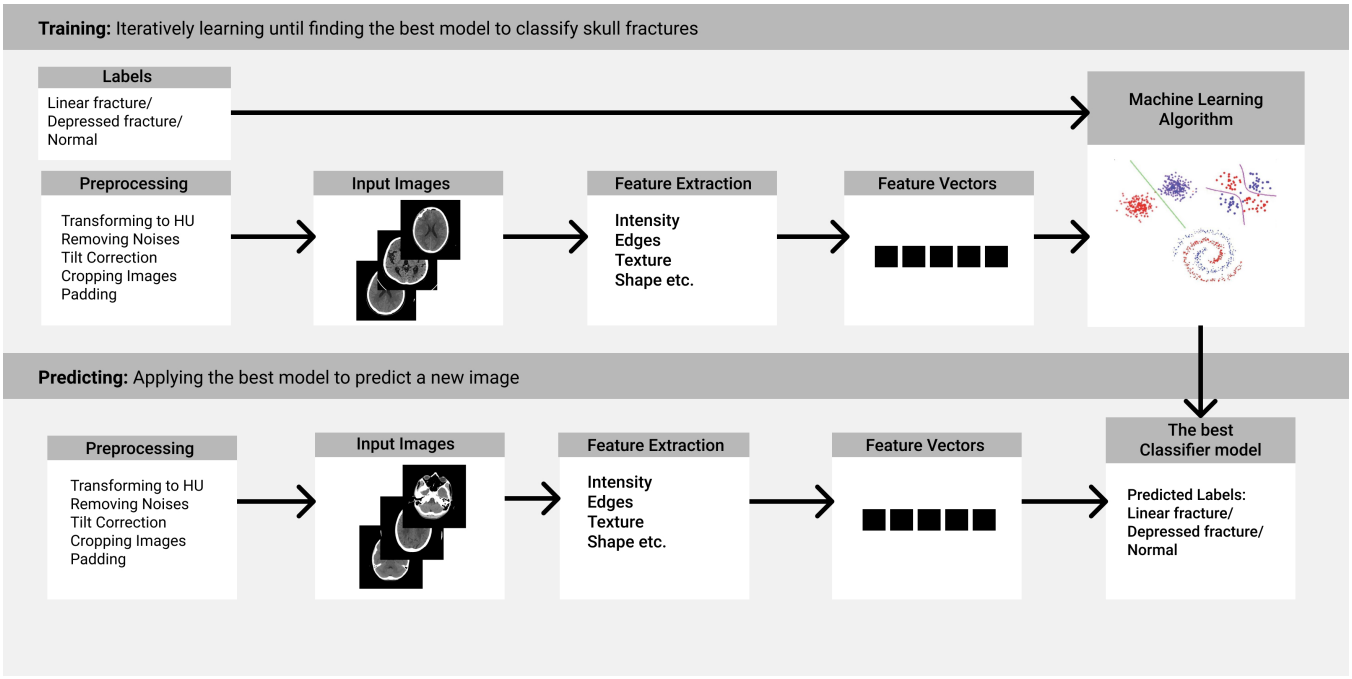

Figure 10: Model Training Pipeline

4.4 Metric

Because there are so many classes to predict, the notion of positive and negative words and associated terms are calculated for each kind in a one vs. rest way, and then the overall levels are averaged. As a result, we chose F1 scores to evaluate all models.

TP = True Positive, TN = True Negative, FP = False positive, FN = False Negative

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \tag{1}$$

$$Precision = \frac{TP}{(TP + FP)} \tag{2}$$

$$Recall = \frac{TP}{(TP + FN)} \tag{3}$$

$$F1 = 2 * \frac{\text{Precision * Recall}}{\text{Precision + Recall}} \tag{4}$$

Balanced Accuracy, Hamming Loss, Hamming Score, ROC AUC, Kappa score, Log Loss were also used to check the model performance.

4.5 Algorithmic details of the proposed system

We developed a model that comprises ResNet50 and XGBoost. By combining ResNet50 as a pre-trained feature extractor to automatically obtain features from input and XGBoost as a classifier on the top level of the network to provide results, the ResNet50-XGBoost model gives more precise output. All of the ResNet50's feature extraction layers, as well as the feature flattening layer, are kept. The XGBoost model takes the role of the fully connected neural network and performs the classification task using the extracted features. ResNet50 was used to construct feature vectors from raw images, which gives a low-dimensional and noise-resistant manner to represent these images. Images were loaded into a pre-trained ResNet50 to build feature vectors, and the representation for that image in the intermediate layers of the neural network was utilized. At the penultimate layer of the ResNet50 model pre-trained on the ImageNet dataset, our approach extracts representations of given images. XgBoost was utilized in the classification phase to create direct predictions based on the high-level features extracted by ResNet50.

5 EXPERIMENTAL ENVIRONMENTS

Ubuntu 20.04.2 LTS was utilized as the operating system. The AMD Ryzen Threadripper 1950X 16-Core Processor was used. The primary memory was 64 GB, and the graphics cards were two GeForce RTX 2080 Ti with 11 GB RAM each. TensorFlow 2.5.0rc2 was used as the deep learning framework.

6 EXPERIMENTAL RESULT

The dataset divided into three parts: 50% training, 20% validation and 30% testing. It was made sure no slices from one patient was present in all three dataset. Table 1,2 ,3,4 ,5 ,6 ,7 shows a comparison of the most well-known CNN models and CNN with supervised machine learning algorithm's classification performance in terms of micro average F1 Score, Hamming Score, Hamming Loss, Balanced Accuracy, ROC AUC, Kappa score and Log Loss.

From these 7 tables, we observe that fine tuned ResNet50 combined with XGBoost achieves the highest score in all of the metrics. It is likely because every other CNN model stopped to learn better

after 25 epoch, but ResNet50 was still learning quite well at 30th epoch.

ResNet50+XGBoost model's classwise score is demonstrated in Table 8:

Table 8: Classwise Score of Resnet50+XGBoost

	precision	recall	f1-score	support
Depressed_Fracture	0.93	0.98	0.95	2548
Linear_Fracture	0.96	0.87	0.91	1373
Not_Fractured	0.99	0.98	0.99	2588
micro avg	0.96	0.96	0.96	6509
macro avg	0.96	0.94	0.95	6509
weighted avg	0.96	0.96	0.96	6509
samples avg	0.96	0.96	0.96	6509

It is clear that our experimented model achieves a higher score in all three classes significantly.

7 DISCUSSION

To our knowledge, this is the first research to disclose the development of a system that can distinguish between three types of skull fractures and test it with a small number of samples. To clarify this, Table 9 shows a comparison with existing published reference models vs our proposed work. Radiology reports were employed as ground labels to increase classification performance over CT scan images alone and supervised deep learning architectures for adding auxiliary data during training were presented. The suggested approach was successful in categorizing a CT scan of the head depending on the kind of skull fracture. Despite having fewer image data than prior classification tests, our model performed wonderfully. Because the extracted features of images by CNN were fed into a powerful gradient boosting (GB) classifier that is an ensemble learning algorithm, which combines the predictions of multiple base learners (usually, each one being a fairly weak performer on its own) to generate one overall prediction for each input/example. This allows it to learn more complex relationships between the features and labels in the training set. This is a solid method when the dataset is very structured. As such, the final ensemble sequence can achieve (nearly) arbitrarily good performance on the training set. According to the outcomes of this investigation, our architecture has the potential to improve classification performance with high accuracy and F1 score. Because we have many more features than examples in our dataset, well-known CNN models with neural networks failed to perform effectively in this challenging multi-class classification task. We had a large number of weights to estimate, but the neural network couldn't since the NN's massive structure couldn't generalize effectively on our small, unbalanced medical dataset, and we required more data to learn such a large number of hidden weights. This is frequent in medical image processing since the design of well-known CNN models was too dense to learn from a small number of images, and they were not pre-trained on medical images. Despite having a higher overall F1-score, our dataset contains fewer Linear Fracture slices. As a result, our model did not perform as well on Linear Fracture as it did on other fractures, demonstrating that even with highly structured data, training with a short dataset by gradient boosters combined with CNN is difficult. Medical image processing software has a lot of capabilities. Medical image analysis software can correctly detect inconsistencies and identify potentially harmful anomalies using Machine Learning methods. These diagnostic technologies can take over some of the most time-consuming processes, allowing clinicians to focus on issues that require immediate treatment. As a result, proper application of AI technology might assist healthcare organizations in providing better and more timely treatment.

Table 9: Comparative performance of published model vs proposed model

Paper	Objective	Score, Metric
Shao and Zhao[11]	Automatically detect if the skull is fractured or not	100%, accuracy
Zaki et al.[14]	Segment fractured skull from 2D-CT brain image	95%, Normalized Euclidean Recall rate
Yamada et al.[12]	Detection of Linear skull fracture	80% accuracy for a crack line of width 1.05mm
Chilamkurthy et al. [5]	Detection of multiple Hemorrhage and skull fracture of only calvaria.	91.11%, AUC
Lee et al. [9]	Detection of femur fracture	86.78%, accuracy
Kuang et al. [8]	Faster detection of skull fracture more accurately	80%, precision recall score
Ours	Classification of three skull fractures	96%, F1-score

8 CONCLUSION

Using head CT images, we developed a deep learning method to automatically identify and classify skull fractures. The proposed method for automatically classifying head CT images is very fast and human-level accurate. In contrast to previous research that focused just on detecting skull fractures, our goal was to categorize skull fractures from images into three different class. Our model classifies skull fracture images trained with limited data, so it has

the advantages of effective utilization of little training data. We hope that by applying our technique to head CT images, we will be able to automate the head CT scan triage process. Our approach has the potential to make radiologists' jobs easier. This technique has shown to have a promising future in delivering second opinions to doctors and radiologists. The ongoing enhancement of the algorithm is one of our major areas of focus for future research. To balance the multi-class dataset, other improvements could be done, such as the addition of GAN and SMOTE method.

ACKNOWLEDGMENTS

Syed Md. Shakawath Hossain, Medical Technologist, CT Scan department, Medinova Medical Services Ltd.; Md Sad Udiin Sadik, assistant manager; MD Ismail Hossain, Medical Technologist, CT Scan Department, Ibn Sina Hospital Sylhet Limited; Md Sad Udiin Sadik, assistant manager; MD Ismail Hossain, Medical Technologist, CT Scan department, Ibn They provided the CT brain images utilized in this study with the approval of the appropriate authorities. Dr. Tahmina Sumi of Z. H. Sikder Women's Medical College, Professor Dr. Ashikur Rahman Majumder, HEAD OF THE DEPARTMENT, Department of Radiology & Imaging of Sylhet MAG Osmani Medical College, and Dr. Sajal Chandra Das of Ibn Sina Hospital Sylhet Limited, along with the respective radiologists, provided additional annotations of CT scan images. We'd also like to thank Sakib Alam Snigdha for letting us use his PC for our preliminary calculations.

REFERENCES

[1] Head CT CQ500 dataset. http://headctstudy.qure.ai/dataset. Accessed: 2021-06-06.

[2] Ibn Sina Hospital Sylhet Limited.. http://ibnsinahospitalsylhet.com.bd/. Accessed: 2021-06-06.

[3] Medinova Medical Services Ltd.. http://www.medinovasylhet.com/index.php?page=9. Accessed: 2021-06-06.

[4] Nirase Fathima Abubacker, Azreen Azman, Masrah Azrifah, and Shyamala Doraisamy. 2013. An approach for an automatic fracture detection of skull dicom images based on neighboring pixels. In *2013 13th International Conference on Intellient Systems Design and Applications*. IEEE, 177–181. https://doi.org/10.1109/ISDA.2013.6920731

[5] Sasank Chilamkurthy, Rohit Ghosh, Swetha Tanamala, Mustafa Biviji, Norbert G Campeau, Vasantha Kumar Venugopal, Vidur Mahajan, Pooja Rao, and Prashant Warier. 2018. Deep learning algorithms for detection of critical findings in head CT scans: a retrospective study. *The Lancet* 392, 10162 (2018), 2388–2396. https://doi.org/10.1016/S0140-6736(18)31645-3

[6] Md Moniruzzaman Emon, Tareque Rahman Ornob, and Moqsadur Rahman. 2022. Classifications of Skull Fractures using CT Scan Images via CNN with Lazy Learning Approach. *Journal of Computer Science* 18, 3 (mar 2022), 116–129. https://doi.org/10.3844/jcssp.2022.116.129

[7] National Collaborating Centre for Acute Care (UK and others). 2007. Head injury: triage, assessment, investigation and early management of head injury in infants, children and adults. (2007). https://pubmed.ncbi.nlm.nih.gov/25340248/

[8] Zhuo Kuang, Xianbo Deng, Li Yu, Hang Zhang, Xian Lin, and Hui Ma. 2020. Skull R-CNN: A CNN-based network for the skull fracture detection. In *Medical Imaging with Deep Learning*. PMLR, 382–392. http://proceedings.mlr.press/v121/kuang20a/kuang20a.pdf

[9] Changhwan Lee, Jongseong Jang, Seunghun Lee, Young Soo Kim, Hang Joon Jo, and Yeesuk Kim. 2020. Classification of femur fracture in pelvic X-ray images using meta-learned deep neural network. *Scientific reports* 10, 1 (2020), 1–12.

[10] Ruizhe Liu, Chew Lim Tan, Tze Yun Leong, Cheng Kiang Lee, Boon Chuan Pang, CC Tchoyoson Lim, Qi Tian, Suisheng Tang, and Zhuo Zhang. 2008. Hemorrhage slices detection in brain ct images. In *2008 19th International Conference on Pattern Recognition*. IEEE, 1–4. https://doi.org/10.1109/ICPR.2008.4761745

[11] Hong Shao and Hong Zhao. 2003. Automatic analysis of a skull fracture based on image content. In *Third International Symposium on Multispectral Image Processing and Pattern Recognition*, Vol. 5286. International Society for Optics and Photonics, 741–746. https://doi.org/10.1117/12.538780

[12] Ayumi Yamada, Atsushi Teramoto, Tomoko Otsuka, Kohei Kudo, Hirofumi Anno, and Hiroshi Fujita. 2016. Preliminary study on the automated skull fracture detection in CT images using black-hat transform. In *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 6437–6440. https://doi.org/10.1109/EMBC.2016.7592202

[13] W Mimi Diyana W Zaki, M Faizal Ahmad Fauzi, and Rosli Besar. 2008. Automated method of fracture detection in CT brain images. In *2008 3rd International Conference on Intelligent System and Knowledge Engineering*, Vol. 1. IEEE, 1156–1160. https://doi.org/10.1109/ISKE.2008.4731105

[14] Wan Mimi Diyana Wan Zaki, Mohammad Faizal Ahmad Fauzi, and Rosli Besar. 2009. A new approach of skull fracture detection in CT brain images. In *International Visual Informatics Conference*. Springer, 156–167. https://doi.org/10.1007/978-3-642-05036-7_16