Hybrid CNN-LSTM Approach in Skin Cancer Disease Detection

Souvik Pramanik
Department of Electrical and Computer Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24060

***Abstract*—Although deep learning framework have been used extensively in the medical field for classifying images to determine diseases, more work is needed to improve accuracy of the classification and detection of diseases. The goal of this work is to propose and implement a novel deep learning method to evaluate images about the presence of malignant and benign oncological diseases. The proposed model utilizes a hybrid approach by leveraging useful features of Convolutional Neural Networks (CNN/ConvNet) and Long Short-Term Memory (LSTM) which is not used for classification of images for Skin Cancer Diagnosis prior to this work. Data Augmentation is applied to images during the training. In addition, traditional CNN along with ResNet-50 architectures were implemented and evaluated for comparison purposes. Moreover, different hardware components such as Central Processing Unit (CPU), Graphics Processing Unit (GPU), and Tensor Processing Unit (TPU) are examined to determine additional improvements in performance.**

***Index Terms*—Machine Learning (ML), Convolutional Neural Networks (CNN/ConvNet), Long-Short-Term Memory (LSTM), Central Processing Unit (CPU), Graphics Processing Unit (GPU), Tensor Processing Unit (TPU), Recurrent Neural Network (RNN), Confusion Matrix, Skin Cancer Images, Accuracy, Precision, Recall, F1 Score, International Skin Imaging Collaboration (ISIC), Data Augmentation**

I. Background and Motivations

Machine Learning (ML) has been adopted across a wide range of industries from finance to marketing. Recently, it has been having an impact in the healthcare sector. Several studies have demonstrated the benefits of detecting diseases at early stages. For example, one study showed the power of image recognition technology based on ML by identifying white blood cells with a 90% accuracy rate [4]. One use of ML related to healthcare is the detection of skin cancer based on classification of images. This technique can be useful for identifying skin abnormalities and cancer like Melanoma.

Different ML methods are adopted for detection of diseases. Deep Learning architectures are used for obtaining results on image classification tasks. These architectures perform better with classification compared to other machine learning methods like Support Vector Machines (SVM) [9], [38]. CNN and its variants are the most common type of deep learning networks that have been found to perform well in image classification and facial recognition [24]. The model in [24] achieved a high testing accuracy for certain conditions but resulted in lower metrics for others. However, performance is dependent on dataset composition, with variations in accuracy and error rates observed in experiments with different datasets. The ongoing objective is to improve on the accuracy of detection of diseases based on image classification under any conditions as well as for similarly structured datasets.

In [5], the authors present how to classify images using LSTM but the significant challenge with this work is network pruning or the elimination of parameters in a network. Furthermore, the author in [5] acknowledges that combination of multiple methods is more likely to obtain higher accuracy scores such as an integration of CNN and RNN method. The authors in [33] utilize a CNN model for Pneumonia Detection using X-Ray images. However, this model resembles the VGG-16 architecture except in the use of dropout in the convolutional part of the network. But VGG-16 is slow to train and takes a lot of bandwidth and disk space. Moreover, this architecture has exploding gradient problems. The authors in [10] utilizes a layered LSTM-CNN structure that does better than a traditional CNN when tested for MNIST and Breast Cancer Datasets. The authors in [10] have implemented the hybrid combination LSTM-CNN for image classification. It is proven that LSTM-CNN performs better than CNN architectures. Because of the characteristics of LSTM, it will remember the long-term dependencies and shape of the input image in a particular pattern. But in [10], there has not been much exploration with respect to reversing the order of CNN and LSTM layer placements. The lack of experimentation of different orders of the combined CNN and LSTM layers with respect to classification of Skin Cancer Images was the motivation of the hybrid approaches, CNN-LSTM and LSTM-CNN, in this work.

ResNet architecture is a state-of-the-art neural network architecture utilizing the shortcut mechanism to solve the performance degradation problem that occurs with the increase of network depth by directly connecting inputs and outputs [24]. ResNet-50 provides superior performance in handling small image data due to its unique structure allowing the architecture to achieve excellent performance in image recognition tasks. While ResNet has its advantages in image classification in terms of improved accuracy, faster convergence in terms of gradient flow and vanishing gradients, better generalization, and transfer learning, it has drawbacks too. These downsides include increased complexity where skipped connections can lead to higher computing and memory requirements which may lead to whether the neural networks need these extra layers [40]. Moreover, there is the issue of overfitting where the structure is too complex and cannot be

sufficiently learned with fewer training instances or epochs. There is even the issue of interpretability which arises from the complexity of the neural network leading to difficulty in how the network arrives at the output decisions. The work in [41] trains the skin cancer disease images from International Skin Imaging Collaboration (ISIC) using InceptionV3, ResNet, and VGG19. However, the ResNet images do not perform as well in classifying the Skin Cancer ISIC Images compared to InceptionV3. But the problem with the InceptionV3 model is limited by its complexity and can lose its computational efficiency if the model is enlarged [42]. The proposed model implemented in this work aims to handle memory issues along with reduction of this type of model complexity.

The specific goal of this work is to propose a newer architecture for Skin Cancer disease detection. It is important to develop a real-time Skin Cancer detection system that can give better recommendations about treatments. An architecture is constructed that combines CNN and LSTM layers using different order structure of layers for better classification purposes. It is observed that LSTMs can complement the feature extraction ability of CNN when used in a layered order [10]. LSTMs have the capacity to selectively remember patterns for a long duration of time and CNNs can extract the important features out of it. LSTM is meant to handle sequential data like time series analysis. But its capabilities extend beyond time series data and are applied to data of sequential nature including image data by memorizing and capturing important features in images [27]. This feature makes LSTM good for medical image diagnosis and analysis in which accurate identification of key attributes is important. Furthermore, the authors in [27] were able to prove the capability of memorizing and capturing important features in image data using MedvLSTM for medical image classification. However, using LSTM alone does not always perform well given issues with dimensionality which is why combinations with other architectures as proposed by Bappy et al. [28] are preferred. The above-mentioned attributes of CNN and LSTM along with the dimensionality issues with singular LSTM is the motivation for integration of their important features resulting in the hybrid model. This combined approach has been implemented by experimenting with the CNN layer acting as an input to the LSTM layer and vice-versa.

This work proposes the combination of CNN and LSTM architectures resulting in a hybrid paradigm in the field of image classification for determining accuracy of detection of Skin Cancer Disease. Based on the literature studies above, this hybrid approach is not used to accomplish the task of image classification for skin cancer disease detection prior to this work. Moreover, CNN classifier and its variant Residual neural network (ResNet) are also implemented for comparisons with the integrated hybrid approach. The implemented networks are used to train the model and are evaluated to compare their performances.

II. METHODS

More Information about each deep learning network and the hardware components used is discussed below. These methods are used for implementing the architectures which will be used for training and testing the Skin Cancer ISIC Dataset.

A. Convolutional Neural Networks (CNN)

CNN or ConvNet is a class of neural networks, defined as multilayered neural networks designed for computer vision applications to detect complex features in data [7]. CNN is known for extracting important features from an image by applying filters that scan across an image which helps in identifying patterns and spatial relationships [8]. These patterns are important in classifying objects provided in images. These patterns are derived from concepts like linear algebra specifically matrix multiplication [9]. An illustration of how Convolutional Neural Networks progress through each layer is shown in Fig. 1 below. An input pixel travels through different feature maps which are sub-sampled as part of the feature extraction process [9]. The sub-sampled data passes through the fully connected layers for classification. In addition to CNN architecture, there are variants that have been used for image classification like ResNet architecture which is renowned for the residual connections to extract high level features from images of skin diseases using transfer learning and has been used for Skin Cancer detection as provided in [24]. The expressions for CNN are presented in [30] where x_t represents the input image pixels instead of the input time series.

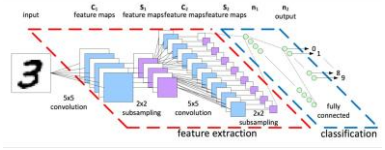

Fig. 1. Diagram of ConvNet presenting how input image navigates through the network.

B. Long Short-Term Memory (LSTM)

LSTM is a type of Recurrent Neural Network (RNN). It is typically used in sequential data analysis like analyzing and processing time series datasets because it can remember short-term and long-term dependencies. Fig. 2 shows a diagram of an LSTM architecture. Input data passes through a combination of Input, Forget, and Output gates along with a hidden state and memory cell state. These gates selectively determine the data to be retained or discarded for each cell. Data passes through a chain of sequences [18]. Activation functions are used to provide a new memory and hidden state. RNN can be used for image classification; however, few of these implementations have been done [5]. Traditional RNN can be jittery especially with backpropagating gradients which may cause gradient explosion and/or vanishing [5]. LSTM handles the issues caused by RNN by having each ordinary recurrent node replaced by memory cells, which can store relevant information so that the model does not have to utilize long-range

dependencies in the data. The full process explaining the input, forget, and output gates and their respective equations can be found in [29] and [30].

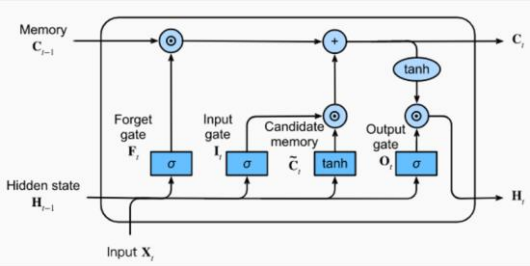

Fig. 2. Diagram of an LSTM Architecture (Source: Taken from [18].)

C. Hardware Components

Recent developments in Deep Learning show that neural networks are getting larger and more complex. This is due to growing amounts of data and more powerful computer processing, data storage, and large-value predictions that can guide better decisions in real time without needing human intervention. ML plays a role in developing models that can predict in real-time. However, with big data comes memory issues when running at a large scale [6]. The issue present is that the workload for training the data is higher due to factors such as dense matrix multiplication, convolution operation, and recurrent operation [6]. This hardware experimentation will be used to compare performance by examining the speed of the building blocks like matrix multiplication, convolution, and data/memory paths [6]. Fig. 3 provides a plot of the different hardware used throughout the years for training compared with throughput and energy consumption.

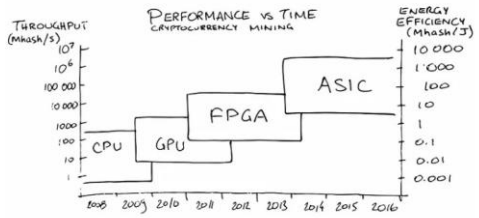

Fig. 3. Performance vs. Time graph for different hardware comparing throughput and energy efficiency (Source: Taken from [6]).

For this implementation, hardware to be used to train and test the dataset with the architectures are CPU (Central Processing Unit), GPU (Graphics Processing Unit), and TPU (Tensor Processing Unit). CPU is flexible in terms of programmability and workloads [6]. GPU is a good option for training machine learning models because of its large storage to perform large computations faster [6]. TPU is used by Google for Machine Learning and is known for its fast inference through high throughput 8-bit arithmetic [6], [11]. It performs dense-matrix multiplication and uses a 32-bit accumulator to sum the result [11].

III. Research Design and Experiments

The cancer detection system utilized the steps of preprocessing, implementing the CNN, LSTM-CNN, CNN-LSTM and ResNet models, and training, testing, evaluating, and visualizing results. In addition, experiments with finding distribution of classes in the training dataset or class imbalances are also performed. These research designs and experiments are shown below. An algorithm of the steps for the system is shown in Appendix A.

A. Dataset

The Dataset used is taken from the International Skin Imaging Collaboration (ISIC). The organization is known for collaborations between computer vision and dermatology by using digital skin imaging to detect and mitigate skin cancers [19]. The set consists of 2357 images of malignant and benign skin diseases. They are sorted according to the classifications provided by the ISIC. The subsets are divided into the same number of images using preprocessing methods. Visualizations of the dataset is shown in Fig. 4. The dataset is made up of the images of the following nine diseases: Actinic keratosis, Basal cell carcinoma, Dermatofibroma, Melanoma, Nevus, Pigmented benign keratosis, Seborrheic keratosis, Squamous cell carcinoma, and Vascular lesion [2].

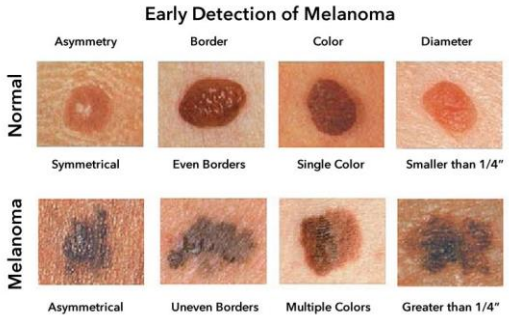

Fig. 4. Images distinguishing between normal skin and melanoma (Source: Taken from [2] and [43]).

B. Preprocessing

Details of the images in the dataset will be visualized as shown in Fig. 4. In addition, images will be resized to a suitable dimension. Furthermore, there will be normalization applied to the pixels for each image. Min-Max Scalar Normalization is used for the pixels as provided in Equation (1) where x represents the current image pixel value, x_{min} represents the lowest pixel value, x_{max} represents the highest pixel value, x_{new} represents the normalized pixel values [22].

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

C. Training, Testing, and Validation

Data is split into training, testing, and validation sets. The training set is used for training the model. The testing set is used for testing the trained data. The system is measured using validation set to check for model overfitting using accuracy and loss plots (Fig. 5). The Testing Set is used to check the models using evaluation techniques like Accuracy, Precision, Recall, and F1 scores and if they predict the unseen data correctly.

The data is split in a 4:1 Train Test Ratio (80 % Training and 20 % Testing). In addition, the skin cancer images are resized by their original sizes to 120x120 (image height and image width respectively). There is a batch size provided or the number of images to be trained at once. For all architectures, the batch size is set to 32 which is standard in for training deep learning models.

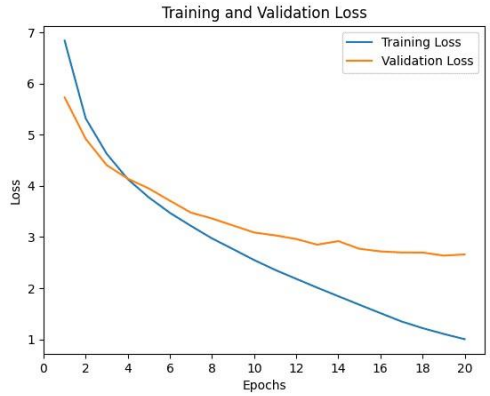

Fig. 5. Training and Validation Loss Plot vs. Epochs (Source: Taken from [43]).

D. CNN-LSTM Architecture

Fig. 6 presents the CNN-LSTM architecture. This model involves using CNN layers for feature extraction on input data combined with LSTMs to perform sequence prediction on feature vectors. The images are preprocessed using normalization and filtering techniques. The output shape from the normalization and filtering is the input shape for the CNN or Convolutional Layer. A Convolutional kernel is created which produces a tensor by convolving with the layer input over a single spatial dimension and extract over important features [20]. The tensor produced by the convolutional layer will be fed into a max pooling layer to reduce spatial dimension of feature maps [21]. The output of the max-pooling layer is then transferred into the LSTM Architectures to achieve the output shape.

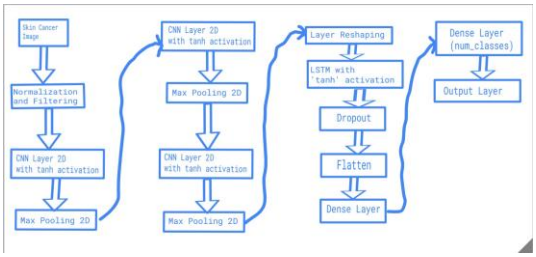

Fig. 6. CNN-LSTM Network Architecture.

For this CNN-LSTM architecture, input values are rescaled from values in the [0, 255] range to the [0, 1] range. The rescaled input values are passed into the CNN layer with 64 filters and a pooling size of with a hyperbolic tangent (tanh) activation function. The output of this CNN layer is passed into a MaxPooling 2D Layer. The output of the Max Pooling 2D layer is passed through two additional CNN layers with the same filters, pooling sizes, and activation functions. The output of the third CNN layer is reshaped based on the number of features in the image. This is passed into the LSTM layer with 32 units with a return sequence of True. A return sequence of True means that the output of the hidden state is relied upon for a time-step [39]. The output of the LSTM layer is inputted to the Dropout layer with a value of 0.2. The output of the dropout layer is flattened and passed into two dense layers, one with 128 units and a ReLu Activation function and another specifying the number of classes to be identified.

E. LSTM-CNN Architecture

Like the CNN architecture, the input values are rescaled from values in the [0, 255] range to the [0, 1] range with a batch size of 32. The input of the dataset is reshaped based on the timesteps and features before being passed into the LSTM layer. After this, the structure will follow from the LSTM-CNN implementation from [10] where output from the LSTM layer is directly provided to the convolutional layer. The diagram for the LSTM-CNN implementation is followed from [10]. After passing into the Convolutional layers, the output is flattened before passing into a dense layer.

F. ResNet-50 Architecture

ResNet architecture is employed after fine-tuning on the skin cancer image dataset to better suit the task of skin cancer disease identification to compare with the better performing proposed model in this work. The structure of the ResNet architecture is provided in Fig. 7 which contains different layer names, output sizes, and types. Like the CNN-LSTM architecture, the input data was rescaled from [0, 255] to [0, 1]. The input data is passed into the ResNet-50 layer with **ImageNet** weights. **ImageNet** is the pre-trained weights that come from a database used in image recognition research. The model will start with a baseline of understanding different features. **ImageNet** is advantageous in that it enables transfer learning leading to faster convergence, improved performance, and better results for skin cancer detection. The output of the

ResNet-50 layer is passed into the Global Average Pooling 2D layer and finally to the dense layer with the number of classes being identified. This Dense layer includes a Softmax activation function. Softmax is designed for multiclassification problems like detecting skin cancer types. The probability distribution of the multiple classes sums to 1. This is helpful in probability interpretation and understanding the model's confidence in predictions and making informed decisions based on the results [35]. The ResNet-50 architecture is provided in Fig.8 providing these features which are compatible with the Skin Cancer Dataset.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Fig. 7. ResNet-50 Structure (Source: Taken from [26])

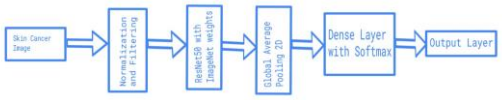

Fig. 8. ResNet-50 Network Architecture.

G. CNN Architecture

Fig. 9 presents the CNN architecture. Like the CNN-LSTM architecture, the input values are rescaled from values in the [0, 255] range to the [0, 1] range and passed into the CNN 2D Layer with 64 filters using a tanh activation function. The output after the CNN 2D Layer is passed into the Max Pooling 2D layer. The output of the Max Pooling 2D layer is passed into two additional CNN 2D layers with 64 filters and two additional Max Pooling 2D layers. The output of the last Max Pooling 2D layer is flattened and passed into a fully connected or dense layer with 32 units. This is followed by the Dense Layer based on the output of the number of skin cancer disease (`num_classes`). This architecture is used as a baseline comparison with the CNN-LSTM, LSTM-CNN, and ResNet-50 models.

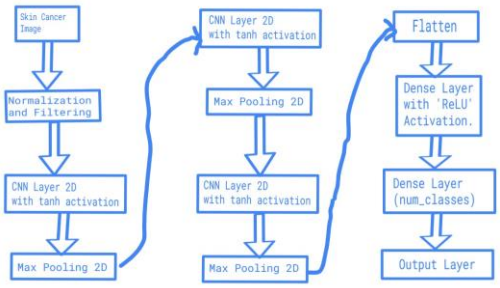

Fig. 9. CNN Network Architecture.

H. Evaluation

Several Metrics are used to evaluate the Skin Cancer Detection Models. They are confusion matrices, accuracy, precision, recall, and F1 scores. All of the implementations correct the issues with imbalanced data using Data Augmentation. These experiments with accuracy and data augmentation are helpful in determining the architecture improvements.

1) Confusion Matrix: Confusion Matrix is generally provided as a 2D matrix in classification problems to check the performance of a system by showing the number of correctly and wrongly classified data helping to identify classes of data, which might be misplaced (Fig. 10 and [12]). Confusion Matrices are used to determine which images for cancer detection are classified correctly.

In a confusion matrix, there are four characteristics which represent the measurement metrics of the classifier (Fig. 10 and [12]). These combinations are True Positive (*TP*), True Negative (*TN*), False Positive (*FP*), and False Negative (*FN*). *TP* and *TN* mean that the classification correctly identified the positive and negative classes respectively. *FP* and *FN* mean that the classification predicted a positive or negative class respectively, but represents the opposite class ([12]- [13]).

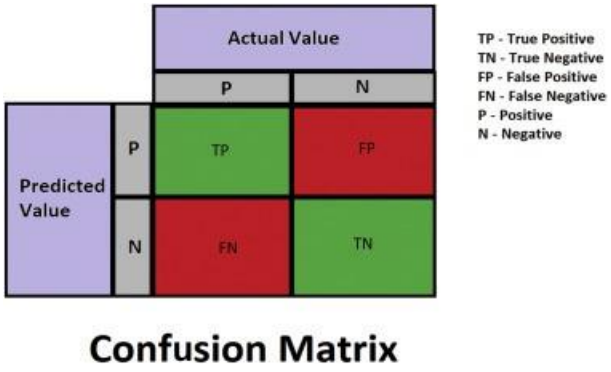

Fig. 10. Confusion Matrix showing Actual and Predicted Values (Source: Taken from [12]).

2) Accuracy, Precision, Recall, and F1 Scores: Accuracy, Precision, Recall, and F1 scores are used for classification of the CNN-LSTM model for the Skin Cancer images. They provide assessments of a model's predictive capabilities, each focusing on different aspects of classification performance [15].

Accuracy is used to calculate percentage of the images that are correctly classified (2). This will be used for the testing images.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

However, when there are imbalanced datasets, accuracy may not always be a good metric. Precision and Recall scores are usefulfor these situations ([14] - [15]). *Precision* determines how correct are the predictions by taking the number of true positive predictions *TP* and dividing by all of the relevant positive predictions (*TP* and *FP*) as provided in Equation (3).

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

Recall checks for instances that are actually correct among instances that might have been missed (4).

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

*F*1*Score* combines both precision and recall providing an overall view of the model's accuracy as shown in Eq. (5). This is especially useful with respect to false positives (*FP*) and false negatives (*FN*) [15].

$$F1Score = \frac{2 \times Precision \times \ Recall}{Precision + Recall} \tag{5}$$

I. Data Augmentation

Data Augmentation is a statistical technique which allows maximum likelihood estimation from incomplete data. Datasets are inflated based on their size and quality so that better deep learning models can be trained with them ([22] and Fig. 11). This is useful in Bayesian Analysis and to reduce overfitting on machine learning models. Data Augmentation improves on model generalization and performance [36]. Real-life datasets can have a class imbalance where one class might have a disproportionate number of samples which could affect the final model [36]. For all the architectures presented, a distribution check is performed using data augmentation technique **Augmentor. Augmentor** adds more across all classes so that none of the classes have few samples [37]. In Python, **Augmentor** is instantiated using the pipeline directory containing the initial image data. The number of operations are performed in the dataset using the Pipeline object by calling the `sample()` method. 500 samples were added per class for even distribution. The augmented images are stored in the output sub-directory of each of the sub-directories of skin cancer types. This balanced dataset was combined with architectures for training. Data Augmentation is helpful to address overfitting issues with the dataset distribution.

In addition to applying **Augmentor**, images were randomly flipped, rotated, and zoomed. This is a common technique used for data augmentation to allow the model to learn the images from different perspectives and angles and ensure the model is robust to variations in real-world data [34]. Performing these operations also prevents overfitting by exposing the image to different image variations and conditions like camera angles or slight changes in position [34].

Fig. 11. Data Augmentation visual (Source: Taken from [22]).

J. Hardware Components and State of the Art Implementations

The CNN-LSTM and ResNet-50 implementations are experimented with other hardware components like CPU, GPU, and TPU and evaluated using the Accuracy, Precision, Recall, F1 scores, Confusion matrices. The CNN-LSTM implementation is compared with other state-of-the-art implementations such as those provided in [5], [7], [33], [10] utilizing Accuracy scores. These implementations were selected as they were tested using images such as Fashion-MNIST [5], Pneumonia Detection [7], [33], MNIST, and Breast Cancer IDC [10]. All architectures utilize data augmentation techniques.

IV. Results

For the implementation of the CNN-LSTM, CNN, and LSTM-CNN architectures, the training and testing of the data was operated on Google Colab Notebook. The data was normalized and trained with the original data using the architectures. After training with the original data, operations such as zooming, flipping, and rotation were performed with the data and the modified dataset was retrained with the implemented models. After this, the dataset was corrected for imbalances using **Augmentor**. The Accuracy, Precision, Recall, and F1 scores, and Confusion Matrices shown below were taken after re-training the datasets with **Augmentor** with the implemented models. These results give a better outlook on which architecture performs better with Skin Disease

Classification. Training deep learning architectures can take time due to calculations across multiple layers, backpropagation, forward propagation, hardware component usage, and the amount of data needed for training. In Google Colab, while GPU and TPU are utilized, they are usage limits for these components while utilizing the CPU operation takes longer to obtain results with increases in the number of epochs, lower batch sizes, number of images, and image size. These have been factors when obtaining the following results below.

The Accuracy, Precision, Recall, and F1 Scores for the architectures are shown in Appendix B. The dataset was normalized and augmented prior to the training of the CNNLSTM, CNN, LSTM-CNN, and ResNet-50 architectures. In addition, these architectures were also trained for different hardware architectures such as CPU, GPU, and TPU. The hardware architectures varied for different epochs. Table I presents the accuracy scores for 10, 20, 50, and 100 epochs for CNN, CNN-LSTM, and LSTM-CNN architectures. While CNN performs better than CNN-LSTM and LSTM-CNN for 10 epochs, CNN-LSTM performs better than the CNN and LSTM-CNN architectures from [2] and [10] respectively as the number of epochs used for training increases with an achievement of 87.28 % accuracy for 100 epochs. This proves the superiority of CNN-LSTM architecture with respect to the other ones.

The CNN-LSTM architecture was also compared to the ResNet-50 architecture using **ImageNet** weights. Tables II and III show the Accuracy, Precision, Recall and F1 scores for these architectures for 40 and 10 epochs respectively. For time constraints, only 10 and 40 epochs were chosen as ResNet50 could not operate for epochs higher than this. Also, these epochs were chosen to compare the hardware components of CPU and GPU/TPU. 10 epochs were chosen to operate CPU (Table III) and 40 epochs were chosen to operate GPU (Table II). This is because a GPU completes each epoch significantly faster due to its parallel processing capabilities, allowing for more epochs to be trained in the same amount of wall-clock time [32]. Although the accuracy of ResNet-50 is better than CNN-LSTM, CNN-LSTM performs better than the ResNet-50 in-terms of Precision, Recall, and F1 scores. This shows that CNN-LSTM is better at predicting model accuracy, identifying true positives, and handling imbalanced datasets compared to ResNet-50.

Accuracy and loss plots for 10 Epochs are shown for CNNLSTM and ResNet-50 in Figs. 12 and 13. The accuracy plots for the training and validation datasets in Figs. 12 and 13 show an increase in the accuracy over time. However, it is the loss plots that are important for both the CNN-LSTM and ResNet50 models. The loss plots show that with the CNN-LSTM data, the training and validation data hold up well indicating that the CNN-LSTM data is better at tracking unseen data. However, with the ResNet-50 architecture, there is an increase in the validation loss compared to the training loss indicating there is some overfitting involved with this model. This is tied somewhat to the Precision, Recall, and F1 scores which show that CNN-LSTM doing better than ResNet-50 with these metrics.

Fig. 12. Training and Validation Accuracy and Loss Plots for 10 Epochs for CNN-LSTM Architecture.

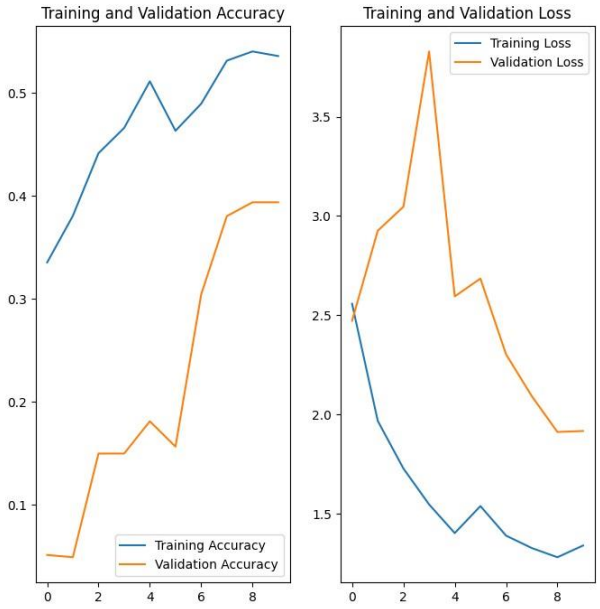

Fig. 13. Training and Validation Accuracy and Loss Plots for 10 Epochs for ResNet-50 Architecture.

Confusion Matrices are also presented for the CNN-LSTM and ResNet-50 architectures in Figs. 14-17. Figs. 14 and 16 show the CNN-LSTM and ResNet-50 architectures for 10 epochs operating in CPU while Figs. 15 and 17 present these same architectures with 40 epochs in GPU. For 10 epochs using CPU, the confusion matrix for the ResNet50 architecture in Fig. 16 shows every image classified as
vascular lesion while the CNN-LSTM architecture in Fig. 14 shows most of the true positives clustered around melanoma,

nevus, and pigmented benign keratosis. This may indicate the possibility that the ResNet-50 model does not classify images well which is also reflected in the Precision, Recall, and F1 scores. With 40 epochs using GPU, both architectures improve. However, CNN-LSTM architecture shows the most improvement with more true positive results across the right diagonal while the ResNet-50 model continues to show more incorrect classifications despite an increase in true positive classifications. CNN-LSTM having more True Positives compared to ResNet-50 is reflected in the Precision, Recall, and F1 scores.

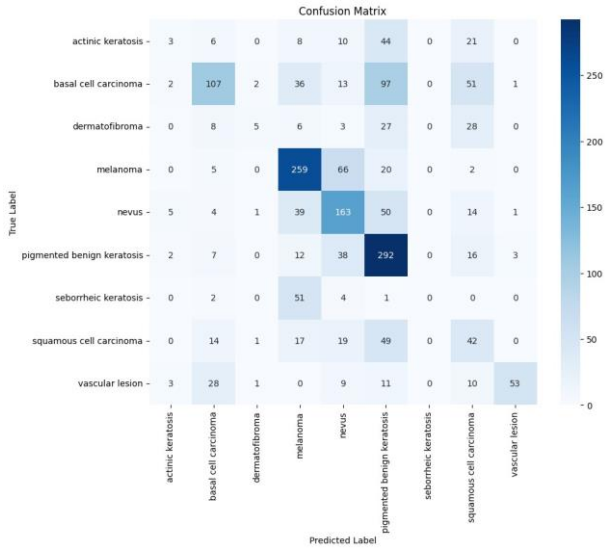

Fig. 14. Confusion Matrix for CNN-LSTM implementation trained for 10 Epochs using Training Data and CPU.

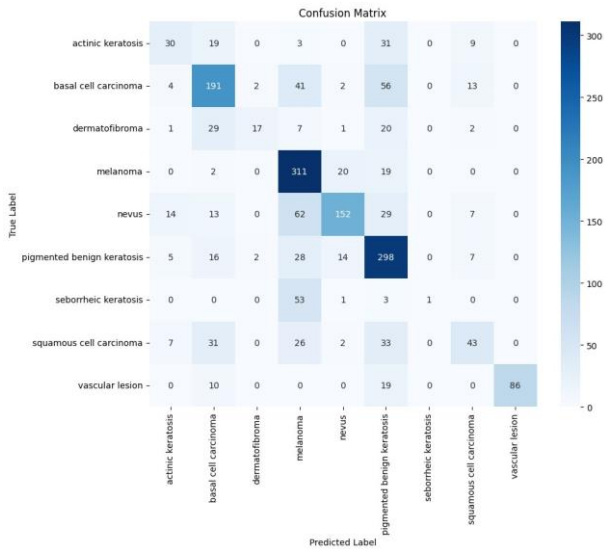

Fig. 15. Confusion Matrix for CNN-LSTM implementation trained for 40 Epochs using Training Data and GPU.

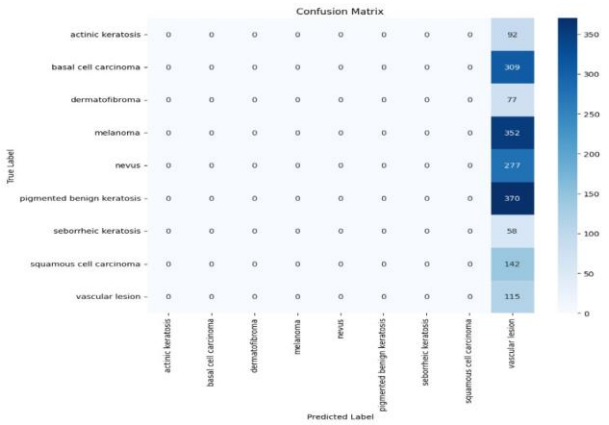

Fig. 16. Confusion Matrix for ResNet-50 implementation trained for 10 Epochs using Training Data and CPU.

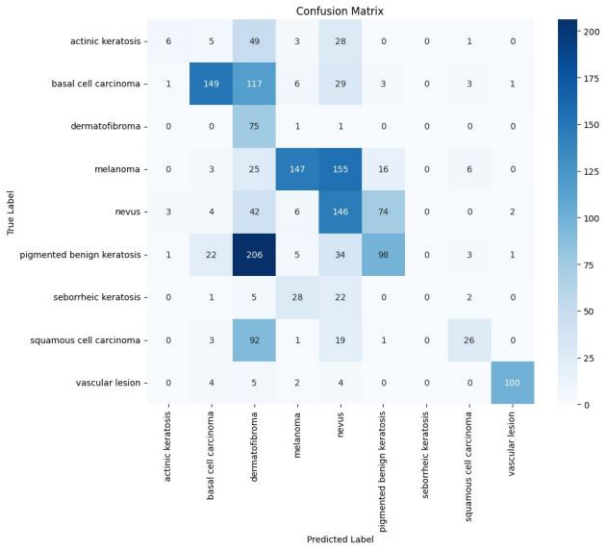

Fig. 17. Confusion Matrix for ResNet-50 implementation trained for 40 Epochs using Training Data and GPU.

V. Conclusion and Future Works

In this work a novel CNN-LSTM approach has been proposed for detection and classification of Skin Cancer Disease detection. The results show that CNN-LSTM outperforms in terms of accuracy compared to the CNN and LSTM-CNN architectures for this image classification task. The performance of the CNN-LSTM model might improve further with more training iterations. In addition, while ResNet-50 architecture with **ImageNet** Weights performed better in-terms of Accuracy compared to the CNN-LSTM architecture, CNN-LSTM model implemented in this work has a better F1 score with more training epochs than the state-of-the-art classifier, ResNet-50, demonstrating a performance that correctly identifies positive cases while maintaining a good balance between precision and recall. This means that the hybrid CNN-LSTM paradigm is the better performing model

for detection of Skin Cancer Diseases compared to the other models.

For future work improvements can be done by training the models with increased image sizes, upgrading hardware components like GPU and TPU to expedite the training, and experimenting with Computer Vision techniques like masking, image denoising, and edge detection for data augmentation and comparing these to the standard Data Augmentation methods like zooming, flipping, and rotating. Experimentation can also be done with initializing the training weights randomly in the network in ResNet-50 architecture for better image classification. The default classifier can also be replaced by efficient classifier built from scratch in the ResNet model. Finally, the CNN-LSTM architecture can be applied to other medical imaging datasets like Brease Cancer and Pneumonia Detection as provided in [7, 10].

REFERENCES

[1] R. O. Duda, P. E. Hart, and D. G. Stork, Pattern Classification. John Wiley & Sons, 2012

[2] J. Ahuja, "Skin Cancer Detection using CNN," Kaggle.com, 2024. https://www.kaggle.com/datasets/jaiahuja/skin-cancer-detection

[3] "GPU vs CPU - Difference Between Processing Units - AWS," Amazon Web Services, Inc. https://aws.amazon.com/compare/the-differencebetween-gpus-cpus/

[4] "How to Build a Disease Detection System Using ML," Akkio. https://www.akkio.com/post/disease-detection-using-machine-learning

[5] K. Zhang, "LSTM: An Image Classification Model Based on Fashion-MNIST Dataset." Available: https://users.cecs.anu.edu.au/Tom.Gedeon/conf-˜ /ABCs2018/paper/ABCs2018 paper 92.pdf

[6] S. Pangre, "Hardware for Machine Learning and Neural Network," Medium, Feb. 19, 2020. https://medium.com/sayalipangre123/hardwarefor-machine-learning-and-neural-network-864544e54e4f

[7] "Pneumonia Detection using CNN with Implementation in Python," Analytics Vidhya, 18-Sep-2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/09/pneumonia-detectionusing-cnn-with-implementation-in-python/.

[8] H. Shi, A. Wei, X. Xu, Y. Zhu, H. Hu, and S. Tang, "A CNN-LSTM based deep learning model with high accuracy and robustness for carbon price forecasting: A case of Shenzhen's carbon market in China," Journal of Environmental Management, vol. 352, p. 120131, Feb. 2024, doi: https://doi.org/10.1016/j.jenvman.2024.120131.

[9] Piotr Skalski, "Gentle Dive into Math Behind Convolutional Neural Networks," Medium, Apr. 12, 2019. https://towardsdatascience.com/gentledive-into-math-behind-convolutional-neural-networks-79a07dd44cf9

[10] Aditi, & Nagda, Mayank & Eswaran, Poovammal. (2019). Image Classification using a Hybrid LSTM-CNN Deep Neural Network. International Journal of Engineering and Advanced Technology. 8. 13421348. 10.35940/ijeat.F8602.088619.

[11] "Slides26," Cornell.edu, 2020. https://www.cs.cornell.edu/courses/cs4787/2021sp/notebooks/Slides26.html (accessed Oct. 11, 2024).

[12] A. Kulkarni, "Confusion Matrix - an overview — ScienceDirect Topics," Sciencedirect.com, 2019. https://www.sciencedirect.com/topics/engineering/confusion-matrix

[13] Tonatiuh Hernandez-Del-Toro, Fernando Mart´ ´ınez-Santiago, Arturo Montejo-Raez, Chapter 7 - Assessing classifier's performance, Ed-´ itor(s): Alejandro A. Torres-Garc´ıa, Carlos A. Reyes-Garc´ıa, Luis Villasenor-Pineda,˜ Omar Mendoza-Montoya, Biosignal Processing and Classification Using Computational Learning and Intelligence, Academic Press, 2022, Pages 131-149, ISBN 9780128201251, https://doi.org/10.1016/B978-0-12-820125-1.00018-X.

[14] Is accuracy a good measure of model performance? — Fiddler AI," www.fiddler.ai. https://www.fiddler.ai/model-accuracy-vs-modelperformance/is-accuracy-a-good-measure-of-model-performance

[15] S. Walker, "F-Score: What are Accuracy, Precision, Recall, and F1 Score? — Klu," klu.ai, Jul. 04, 2023. https://klu.ai/glossary/accuracyprecision-recall-f1

[16] M. Pharr, W. Jakob, and G. Humphreys, "Sampling and Reconstruction," Elsevier eBooks, pp. 401–504, Oct. 2016, doi: https://doi.org/10.1016/b978-0-12-800645-0.50007-5.

[17] Matt Pharr, Wenzel Jakob, Greg Humphreys, 07 - Sampling and Reconstruction, Editor(s): Matt Pharr, Wenzel Jakob, Greg Humphreys, Physically Based Rendering (Third Edition), Morgan Kaufmann, 2017, Pages 401-504, ISBN 9780128006450, https://doi.org/10.1016/B978-012-800645-0.50007-5.

[18] S. N. Qasem and S. M. Alzanin, "An Effective Forecasting Approach of Temperature Enabling Climate Change Analysis in Saudi Arabia," International Journal of Advanced Computer Science and Applications, vol. 15, no. 3, 2024, doi: https://doi.org/10.14569/ijacsa.2024.0150372.

[19] "ISIC Archive," www.isic-archive.com. https://www.isic-archive.com/

[20] Ying Chen, Yiqi Huang, Zizhao Zhang, Zhen Wang, Bo Liu, Conghui Liu, Cong Huang, Shuangyu Dong, Xuejiao Pu, Fanghao Wan, Xi Qiao, Wanqiang Qian, Plant image recognition with deep learning: A review, Computers and Electronics in Agriculture, Volume 212, 2023, 108072, ISSN 0168-1699, https://doi.org/10.1016/j.compag.2023.108072.

[21] L. Zhao and Z. Zhang, "A improved pooling method for convolutional neural networks," Scientific Reports, vol. 14, no. 1, p. 1589, Jan. 2024, doi: https://doi.org/10.1038/s41598-024-51258-6.

[22] T. A. Team, "How, When, and Why Should You Normalize/Standardize/Rescale Your Data? – Towards AI — The Best of Tech, Science, and Engineering," Towards AI, May 16, 2019. https://towardsai.net/p/data-science/how-when-and-whyshould-you-normalize-standardize-rescale-your-data-3f083def38ff

[23] G. Boesch, "Image Data Augmentation for Computer Vision in 2022 (Guide)," viso.ai, Jun. 07, 2022. https://viso.ai/computer-vision/imagedata-augmentation-for-computer-vision/

[24] N. Gaffoor and S. Soomro, "Skin Disease Detection and Classification Using ResNet-50 and Support Vector Machine: An Effective Approach for Dermatological Diagnosis," 2023 IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS), Bali, Indonesia, 2023, pp. 140-145, doi: 10.1109/IoTaIS60147.2023.10346059. keywords: Support vector machines;Sensitivity;Scalability;Eczema;Melanoma;Skin;Medical diagnosis;Skin Diseases;CNN;Classify;Feature Extraction;SVM algorithm;Accuracy,

[25] C. Qi, Z. Yang and Y. Wen, "Improved ResNet-50 Model for AI Image Recognition Based on Multi-Scale Attention Mechanism," 2024 6th International Conference on Communications, Information System and Computer Engineering (CISCE), Guangzhou, China, 2024, pp. 825-829, doi: 10.1109/CISCE62493.2024.10653161. keywords: Image recognition;Attention mechanisms;Accuracy;Face recognition;Media;Robustness;Task analysis;ResNet-50;AI image recognition;two-channel pooling layer;multi-scale attention mechanism,

[26] R. ZHU, B. XIN, N. DENG, and M. FAN, "Semantic Segmentation Using DeepLabv3+ Model for Fabric Defect Detection," Wuhan University Journal of Natural Sciences, vol. 27, no. 6, pp. 539–549, Dec. 2022, doi: https://doi.org/10.1051/wujns/2022276539.

[27] I. Salehin et al., "Real-Time Medical Image Classification with ML Framework and Dedicated CNN–LSTM Architecture," Journal of Sensors, vol. 2023, p. e3717035, Dec. 2023, doi: https://doi.org/10.1155/2023/3717035.

[28] Jawadul H. Bappy, Cody Simons, Lakshmanan Nataraj, B. S. Manjunath, and Amit K. Roy-Chowdhury. Hybrid lstm and encoder–decoder architecture for detection of image forgeries. IEEE Transactions on Image Processing, 28(7):3286–3300, 2019.

[29] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on LSTM recurrent neural network," IEEE Trans. Smart Grid, vol. 10, no. 1, pp. 841–851, Jan. 2017.

[30] S. P. Sone, J. J. Lehtomaki and Z. Khan, "Wireless Traffic Usage¨ Forecasting Using Real Enterprise Network Data: Analysis and Methods," in IEEE Open Journal of the Communications Society, vol. 1, pp. 777-797, 2020, doi: 10.1109/OJCOMS.2020.3000059. keywords: Forecasting;Time series analysis;Wireless networks;Data analysis;Machine learning;Resource management;5G;CNN;CNNGRU;CNN-LSTM;forecasting;GRU;holt-winters;LSTM;neural network;real network data;SARIMA;spatio-temporal;temporal;time series analysis;WLAN,

[31] S. Savarese, A. Sadeghian, K. Fang, D. Xu, F. Xia, and J. Gao, Computer Vision, From 3D Reconstruction to Recognition, Standford Univ., Standford, CA, USA, 2018. [Online]. Available: https://https://web.stanford.edu/class/cs231a/lectures/intro˙cnn.pdf

[32] "Epochs, Batch Size, Iterations - How they are Important," Sabrepc.com, 2024. https://www.sabrepc.com/blog/Deep-Learningand-AI/Epochs-Batch-Size-Iterations?srsltid=AfmBOopdQ6PxkxSm13rkck-m1cNQVKS0rPQWCg65jQ˙x0n8xJGXdf9w(accessed Dec. 12, 2024).

[33] P. Szepesi and L. Szilagyi, "Detection of pneumonia using con-´ volutional neural networks and deep learning," Biocybernetics and Biomedical Engineering, vol. 42, no. 3, Aug. 2022, doi: https://doi.org/10.1016/j.bbe.2022.08.001.

[34] A.A. Awan, "A Complete Guide to Data Augmentation," www.datacamp.com, Nov. 2022. https://www.datacamp.com/tutorial/complete-guide-data-augmentation

[35] "Neural networks: Multi-class classification," Google for Developers, 2024.https://developers.google.com/machine-learning/crashcourse/neural-networks/multi-class

[36] G. O. Assunc¸ao, R. Izbicki, and M. O. Prates, "Is Augmentation˜ Effective in Improving Prediction in Imbalanced Datasets?," Journal of Data Science, pp. 1–16, Jan. 2024, doi: https://doi.org/10.6339/24jds1154.

[37] B. Marcus, Augmentor Documentation Release 0.2.12. 2023. Accessed: Dec. 13, 2024. [Online]. Available: https://augmentor.readthedocs.io/˙/downloads/en/stable/pdf/

[38] E. Shevchenko, "Why deep learning may be preferred over support vector machines (SVMs)," Medium, Jan. 05, 2023. https://medium.com/@eugenesh4work/why-deep-learning-may-bepreferred-over-support-vector-machines-svms-9d2d3021aa29 (accessed Dec. 14, 2024).

[39] J. Brownlee, "Difference Between Return Sequences and Return States for LSTMs in Keras," Machine Learning Mastery, Oct. 23, 2017. https://machinelearningmastery.com/return-sequences-and-returnstates-for-lstms-in-keras/

[40] C. T. Kien, "Skip Connection and Explanation of ResNet," *Medium*, Oct. 08, 2022. https://chautuankien.medium.com/skip-connection-and-explanation-of-resnet-afabe792346c

[41] Maad M. Mijwil. 2021. Skin cancer disease images classification using deep learning solutions. Multimedia Tools Appl. 80, 17 (Jul 2021), 26255–26271. https://doi.org/10.1007/s11042-021-10952-7

[42] Y. Jing *et al.*, "A comprehensive survey of intestine histopathological image analysis using machine vision approaches," *Computers in Biology and Medicine*, vol. 165, p. 107388, Oct. 2023, doi: https://doi.org/10.1016/j.compbiomed.2023.107388.

[43] iavesh, "Melanoma (Cancer) Detection with 85% Acc (CNN)," *Kaggle.com*, Sep. 17, 2023. https://www.kaggle.com/code/iavesh/melanoma-cancer-detection-with-85-acc-cnn (accessed Dec. 14, 2024).

VI. Appendix

A. Algorithm of CNN, CNN-LSTM, LSTM-CNN, ResNet-50 implementation

Algorithm 1: Training Architecture using Skin Cancer ISIC Dataset

1: Load Data
2: Split training data to training and validation data (80% Training and 20% Testing)
3: Set up `batch_size` and resize images (`img_height` and `img_width`)
4: Normalization: Using `MinMax`
5: Visualize images
6: Model Implementation (CNN, CNN-LSTM, LSTM-CNN, ResNet-50)
7: Train Model (`batch_size` and `epochs`)
8: Evaluation: `acc, val_acc, loss, val_loss`
9: Data Augmentation: Image rotation and zoom
10: Model Implementation with Data Augmentation
11: Train Model with Data Augmentation (`batch_size` and `epochs`)
12: Evaluation using Data Augmentation: `acc, val_acc, loss, val_loss`
13: Check Class Imbalance and Rebalance distribution of images in Training Data (**Augmentor**)
14: Model Implemtation using **Augmentor**
15: Train Model with **Augmentor** (`batch_size` and `epochs`)
16: Evaluation using **Augmentor**: `acc, val_acc, loss, val_loss`
17: Precision, Recall, F1 Scores and Confusion Matrix

B. Accuracy, Precision, Recall, and F1 Scores.

TABLE I
Accuracy Scores for 10, 20, 50, and 100 Epochs.

Architecture	10 Epochs (%)	20 Epochs (%)	50 Epochs (%)	100 Epochs (%)
CNN-LSTM	48.31	**59.26**	**74.94**	**87.28**
LSTM-CNN [10]	46.32	54.35	60.44	65.74
CNN [2]	**50.95**	52.96	66.07	78.35

TABLE II
Precision Recall and F1 Scores for CNN-LSTM and ResNet-50 Architectures for 40 Epochs (Performed Using GPU/TPU).

Architecture	Accuracy (%)	Precision	Recall	F1
CNN-LSTM	64.17	**0.6631**	**0.6300**	**0.6066**
LSTM-CNN [10]	**89.12**	0.5817	0.4169	0.4391

TABLE III
Precision Recall and F1 Scores for CNN-LSTM and ResNet-50 Architectures for 10 Epochs (Performed Using CPU).

Architecture	Accuracy (%)	Precision	Recall	F1
CNN-LSTM	48.31	**0.5088**	**0.5156**	**0.4825**
LSTM-CNN [10]	**74.83**	0.0041	0.0642	0.0077