MACHINE LEARNING METHODS FOR PNEUMONIA DIAGNOSIS WITH CHILDREN'S CHEST X-RAY

Vernise Ang Veng Yan, Jessica Perchman, Alicia Gunawan Sanggalo

1. INTRODUCTION

Pneumonia is an infection caused by bacteria, fungi or viruses, and it causes the alveoli of the lungs to fill up with fluid [1]. According to the World Health Organization (WHO), pneumonia was the leading infectious cause of death among children globally in 2017, accounting for 15% of deaths in children under the age of five. Early diagnosis is crucial to reduce the morbidity and mortality rate. Chest X-ray images are the most utilised modality for diagnosing pneumonia. However, a study has addressed the poor interobserver agreement among radiologists, recommending a more standardised approach to pneumonia detection. Machine learning (ML) and deep learning (DL) methods are proven by various studies to improve diagnostic accuracy for pneumonia, offering non-invasive and reliable methods for early detection of the disease.

According to Bakarat et al.'s study, the quadratic SVM model is the best performing model to predict pneumonia from children's chest X-ray images. It achieved an accuracy of 97.58%, outperforming all 26 machine learning models evaluated in the study [2]. To validate this result, a quadratic SVM model will be tested on the dataset obtained from Kermany's study.

Convolutional Neural Networks (CNNs) have shown immense potential in the field of computer vision. Several studies have explored the use of CNNs for various medical imaging tasks, including classification of disease, segmentation and image analysis. For example, study by T. Rahman [3], it was found that their CNN could accurately classify X-ray images as being healthy or containing viral or bacterial pneumonia with up to 95% accuracy. This study used 'transfer learning', a common technique where a pre-trained model is used on a new problem. Some of these models included DenseNet2021, SqueezeNet and AlexNet. Furthermore, other research has indicated another common technique, data augmentation can also be used to accurately classify chest X-ray images, for example, a study by M. Maram et al. They used several data augmentation [3] techniques to classify COVID-19 X-Ray images with over 95% accuracy.

In the study 'Assessing the Role of Random Forests in Medical Image Segmentation' presented by Hartmann, random forests (RF) are evaluated against deep CNNs (DCNNs) for medical image segmentation. The research demonstrates that while CNNs achieved the highest performance, RFs also performed remarkably well, offering a viable alternative in clinical settings, when computational resources like GPUs are limited [5].

The use of a Random Forest (RF) model combined with a feature extraction architecture is highlighted as an effective method for medical image classification, as RFs do not require the extensive computational power that CNNs demand, making them accessible for institutions with limited resources. This approach offers a robust alternative to deep learning models, providing several advantages in terms of performance, interpretability, and resource efficiency.

The study also shows RFs achieving competitive accuracy and segmentation quality on both medical imaging datasets in the article, achieving 98% accuracy in cellular imaging and 95% accuracy in retinal imaging. They are also relatively simple to implement and tune compared to the complex architectures of CNNs. Additionally, RFs can be scaled efficiently across different types of medical imaging tasks without significant modifications to the underlying algorithm.

2. MACHINE LEARNING ALGORITHMS

2.1 Support Vector Machine (SVM)

Read and pre-process image dataset

Chest X-ray images from the training and testing datasets are read into empty arrays. This is done by appending the file directory to the images' filenames. After the images are read, the following image pre-processing steps are undertaken to ensure consistency across all images:

1. All images are resized to (1000,1000) with anti-aliasing. Anti-aliasing retains as much of the original quality of the images as possible, while reducing artefacts and smoothening edges.
2. RGB images (images with 3 dimensions) are converted to grayscale.
3. Images with pixel intensity ranges of 0-1 are converted to a range of 0-255.

Pre-processed images are then appended to the 'normal array', 'bacteria-infected pneumonia array' and 'virus-infected pneumonia array' according to their class.

The dimensions (shape) and the pixel intensity ranges of the first image from each class are checked to ensure the pre-processing algorithm has been applied. The first chest X-ray image from each class is also displayed.

Images from each class are then combined into large arrays called 'all_training_images' and 'all_testing_images'. Arrays of labels are created for the training and testing datasets, with normal X-ray images labeled as '0', bacteria-infected pneumonia X-ray images labeled as '1', and virus-infected pneumonia X-ray images labeled as '2'. These arrays are called 'all_training_labels' and 'all_testing_labels'. All arrays are converted to numpy arrays for more efficient processing.

Feature extraction
Relevant features are extracted from the X-ray images. A study using the permutation feature importance technique identified skewness, GLCM maximum probability, GLCM contrast, and GLCM difference variance as the most significant variables [6]. Utilising this information, first-order statistics and GLCM texture features are extracted for this dataset. The set of features extracted are:

- First-order statistics
    - Mean
    - Minimum
    - Maximum
    - Variance
    - Skewness
    - Kurtosis (shape of distribution)
    - Entropy
- GLCM features
    - Contrast
    - Dissimilarity
    - Homogeneity
    - Energy
    - Correlation
    - Angular second moment (ASM)

For GLCM features, the distance between pixel pairs is set to one pixel. The co-occurrence is computed at four different angles, which are 0 degrees, 45 degrees, 90 degrees and 135 degrees. The number of intensity levels are set to 256 as the pixel instensity range for all X-ray images in the dataset is 0-255. The GLCM is normalized to ensure probabilities are displayed in the GLCM matrix. The symmetric parameter is set to 'true' to average the co-occurrence of pixel pairs in both directions. This can enhance the robustness of texture features extracted.

All features are stored in arrays called 'all_training_features' and 'all_testing_features'. The arrays are then standardised to ensure uniformity across all X-ray images. The first set of features from both the training and testing dataset is displayed.

Model creation
To build a quadratic SVM model, a polynomial kernel function of degree 2 is used. This model searches for a quadratic hyperplane that best classifies the image features. A polynomial kernel is particularly effective when classes cannot be linearly separated, thus accommodating more complex feature patterns [6]. The training dataset and their corresponding labels are used to train the quadratic SVM model.

Model Evaluation
The trained model is used to make predictions on the testing dataset. The X-ray images are classified as '0', '1' or '2' based on its predictions. These predictions are then compared with the actual labels (ground truth) to assess the performance of the quadratic SVM model. The accuracy and classification report containing the precision, recall and f1-score of the model is presented.

Perform cross-validation
To maintain consistency with Barakat et al.'s study, cross-validation of 10 folds is performed on the training dataset. The cross-validation scores and the average accuracy is presented.

2.2 Convolutional Neural Networks (CNN)

Data Preparation
The training, testing and validation data was loaded using the 'tf.keras.preprocessing.image_dataset_from_directory' function which automatically loads images from a directory and labels them based on the folder structure, in this case, 'NORMAL', 'VIRUS' and 'BACTERIA'. 20% of each set was spit for validation and the images were resized into (224, 224) for computational purposes. The prefetch size was set to 10.

Next, the data goes through augmentation, this is a common method for transforming images and has been known to increase accuracy. In this instance, the sequence of layers goes through a random flip, random rotation and random zoom. This can also help prevent overfitting [8].

Model Architecture
The first layer of this model is a base model known as 'ResNet50'. This is a widely used model that has 50 layers and has commonly been used in medical contexts [9]. On top of the base model, there is then a:

1. Input layer – takes in the images, allowing 224x224 image with 3 channels to be processed
2. Global average pooling layer - this reduces each feature map to a single number, by taking average of values
3. Fully connected (dense) layers – There are two dense layers. One is a later with 1024 neurons, and the other uses ReLU activation to introduce non-linearity to learning process
4. Output layer – one containing 5 neuron and the other containing SoftMax activation

Model Compilation and training
The model was compiled with 'Adam Optimizer' and sparse categorical crossentropy. The accuracy and loss were also both calculated. Finally the learning rate was set to 0.001 and

number of epochs was set to 10 for the first iteration and 20 for the second iteration to improve learning.

2.3 Random Forest

Image Pre-Processing
The chest x-ray images from both the training and testing datasets are loaded into empty arrays by reading and appending the file directory into lists that correspond to each label/class.

The images are then read and undergo pre-processing to prepare for feature extraction:

1. Conversion of RGB images into grayscale to ensure that the arrays are 2-Dimensional.
2. Images of pixel intensity ranging 0-1 are scaled to a range of 0-255.

These images are then appended to the 'normal lungs', 'bacterial pneumonia lungs' and 'viral pneumonia lungs' arrays based on their respective labels.

Feature Extraction
Research has demonstrated the efficacy of integrating a comprehensive set of features, including both first-order statistics and GLCM texture features, in machine learning tasks involving medical image analysis [9]. RFs are robust to noisy data and capable of handling high-dimensional feature spaces, making them well-suited for processing diverse sets of features like those derived from X-ray images [10]. These features provide a holistic representation of the underlying characteristics within X-ray images, enabling models to capture intricate patterns and variations crucial for accurate classification or diagnosis. 10 features of each feature type are extracted from the x-ray images:

First-order statistics:

- **Mean:** The average pixel intensity.
- **Minimum:** The lowest pixel intensity value.
- **Maximum:** The highest pixel intensity value.
- **Variance:** A measure of how pixel intensities vary across the image.
- **Skewness:** Indicates the asymmetry of the intensity distribution.
- **Kurtosis:** Describes the shape of the distribution (peak-ness or flatness).
- **Entropy:** Measures the randomness or complexity of the texture.
- **Median:** The middle value when all pixel intensities are sorted.
- **Standard Deviation:** A measure of the spread of pixel intensities.
- **Range:** Shows span of pixel intensity values, potential outliers.

GLCM features:

- **Contrast:** Measures the local variations in pixel intensity.
- **Dissimilarity:** Reflects the difference between neighbouring pixel intensities.
- **Homogeneity:** Indicates how similar neighbouring pixels are.
- **Energy:** Represents the uniformity of pixel pairs.
- **Correlation:** Measures the linear dependence between pixel pairs.
- **Angular Second Moment (ASM):** Represents the uniformity of pixel pairs.
- **Entropy:** Quantifies texture randomness.
- **Cluster Prominence:** Describes the asymmetry of pixel value clusters.
- **Cluster Shade:** Reflects the skewness of pixel value clusters.
- **Maximum Probability:** The most frequently occurring pixel pair.

The inclusion of GLCM texture features enhances the model's ability to discern patterns and textures within the images, potentially improving its predictive performance [13]. The GLCM feature extraction process involves setting the pixel pair distance to one and computing co-occurrences at four distinct angles: 0, 45, 90, and 135 degrees. Intensity levels are configured to 256, corresponding to the full pixel intensity range of 0-255 in the X-ray images. Normalization of the GLCM ensures that probabilities are accurately represented within the matrix. By setting the symmetric parameter to 'true,' co-occurrences are averaged across both directions, contributing to the robustness of extracted texture features.

These features are extracted and stored in arrays 'normal_features', 'bacteria_features', 'virus_features', corresponding to each label.

Training and Testing Dataset
Arrays containing labels are generated for both the training and testing datasets. Normal X-ray images are assigned the label '0', while X-ray images showing bacterial pneumonia are labeled as '1', and those displaying viral pneumonia are labeled as '2'. These arrays are then converted into numpy arrays to enhance processing efficiency.

The dataset is divided using an 80:20 train-test split for each feature set. Subsequently, the training labels are aggregated into a single array, and the same procedure is applied to the test labels. Feature scaling is then implemented to ensure that all features were brought to a similar range to prevent features with larger scales from dominating the learning process.

Implementing the Model
The RF classifier is initialized and trained using the training datasets to make predictions on the test dataset. The predictions are subsequently compared with the ground truth labels to evaluate the performance of the RF model. This evaluation includes assessing its accuracy, through a classification report, and through a confusion matrix. To

combat data over-fitting, a 10-fold cross-validation on accuracy scores is performed on the training dataset.

3. RESULTS

3.1 Support Vector Machine (SVM)

Figure 1 shows the performance of the quadratic SVM model.

```
Accuracy:  0.5849358974358975
Classification report:
              precision    recall  f1-score   support

           0       0.88      0.39      0.54       234
           1       0.49      0.96      0.65       242
           2       0.91      0.28      0.42       148

    accuracy                           0.58       624
   macro avg       0.76      0.54      0.54       624
weighted avg       0.73      0.58      0.56       624
```

Fig. 1. Output of classification report of quadratic SVM model.

Evaluation Metrics

Accuracy	58.49%
Precision	73%
Recall	58%
F1-score	56%

Table. 2. Evaluation metrics for quadratic SVM model.

The overall accuracy of the model is 58.49%. This indicates that the quadratic SVM model correctly predicts the class labels of chest X-ray images slightly over half of the time. This accuracy is significantly lower compared to the accuracy reported in Bakarat et al.'s study.

The model achieves a high precision of 0.88 but a low recall of 0.39 for class 0 (normal images), resulting in an F1-score of 0.54. This indicates that the model is good in identifying normal images when it predicts them but it missed a large number of actual normal cases. The model performs similarly for class 2 (virus-infected pneumonia images) too, where it predicts virus pneumonia cases accurately when it predicts them but fails to identify most of the actual cases.

For class 1 (bacteria-infected images), the precision is significantly low at 0.49, but it has a high recall of 0.96, leading to a F1-score of 0.65. This shows that the model is effective at identifying most bacteria-infected pneumonia but also incorrectly labels many normal and virus-infected pneumonia cases as bacteria-infected pneumonia cases.

```
Cross-validation scores:  [0.70676692 0.71804511 0.69548872 0.70300752 0.64661654 0.7481203
 0.68421053 0.68045113 0.71052632 0.64285714]
Average accuracy:  0.693609022556391
Highest accuracy:  0.7481203007518797
```

Fig. 2. Output of cross-validation results of quadratic SVM model.

Accuracy of Model After Cross-validation

Average accuracy	69.36%
Highest accuracy	74.81%

Table. 2. Cross-validation results of quadratic SVM model.

After performing cross-validation on the training dataset, the accuracy of model has improved. The highest score is 74.81% and the average score across all folds is 69.36%. However, this result is still lower than the accuracy Barakat et al. obtained. There are several reasons that may have caused this accuracy difference:

1. Insufficient data

The training dataset may be too small for the model to correctly learn all feature patterns. This could lead to poor generalisability and overfitting of the model.

2. Noisy images

The X-ray images may contain a lot of noise, such as artefacts, which could mislead the training process and result in the model learning incorrect patterns. To improve the model's performance, image segmentation can be employed to segment the area of interest before using the images to train the model. This method is also mentioned in Bakarat et al.'s study, suggesting using Minitab to trace back from the ROI locations to find the most key features [2].

3. Inappropriate model selection

The dataset used in Barakat et al.'s study differs from the dataset used in this research. Barakat et al.'s dataset may include features that are more amenable to a quadratic SVM model, while the features in this study's dataset may not be best suited for this model.

In conclusion, the model exhibits imbalance performance for precision and recall. Further model tuning may be required, such as data augmentation and image segmentation, to enhance the performance of the quadratic SVM model.

3.2 Convolutional Neural Networks (CNN)

Figures 3 and 4 show the results from the first iteration of the CNN model.

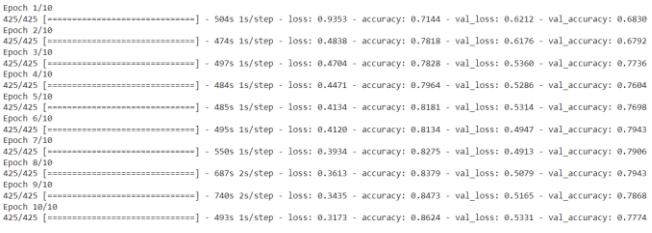

Fig. 3. Output of results for CNN model with 10 Epochs for training and validation dataset

```
87/87 [==========] - 82s 870ms/step - loss: 0.7337 - accuracy: 0.7593
Test accuracy: 0.7592592835426331
```

Fig. 4. Output of results for CNN model with 10 Epochs for testing dataset

The first CNN model was trained over 10 epochs, showing consistent improvement in training accuracy, with training

accuracy going from 71.44% to 86.33% (Figure 4). It also showed improvement in loss over the epochs, going from 0.9353 to 0.3173 (Figure 4), which indicates that model was learning effectively over time. Validation accuracy improved from 68.30% to a peak of 77.74% at epoch 10. The validation loss fluctuated, but ended at 0.5331 in the final epoch.

The gradual improvement in accuracy and the decrease in loss in the training set indicates good convergence. However, in the validation set, the loss fluctuates, specifically in epochs 5 and 10. This could indicate overfitting and confirms there is variability in performance on different sets of data.

The model achieved an overall accuracy of 75.9% and a loss of 0.733 which is quite a bit lower than the highest score for the training dataset. This could indicate that the model is too general and has difficulty accurately classifying images that may not necessarily have similar features to other ones in its class. It suggests that there is some degree of overfitting.

The model was then retrained using 20 epochs. Figures 5 and 6 show the results from the second iteration of CNN Model

```
Epoch 1/20
425/425 [==============================] - 483s 1s/step - loss: 0.2922 - accuracy: 0.8808 - val_loss: 0.5899 - val_accuracy: 0.7642
Epoch 2/20
425/425 [==============================] - 480s 1s/step - loss: 0.2861 - accuracy: 0.8718 - val_loss: 0.5873 - val_accuracy: 0.7358
Epoch 3/20
425/425 [==============================] - 480s 1s/step - loss: 0.2505 - accuracy: 0.9020 - val_loss: 0.6436 - val_accuracy: 0.7453
Epoch 4/20
425/425 [==============================] - 477s 1s/step - loss: 0.2312 - accuracy: 0.9095 - val_loss: 0.5920 - val_accuracy: 0.7736
Epoch 5/20
425/425 [==============================] - 474s 1s/step - loss: 0.2136 - accuracy: 0.9119 - val_loss: 0.7548 - val_accuracy: 0.7113
Epoch 6/20
425/425 [==============================] - 536s 1s/step - loss: 0.1813 - accuracy: 0.9345 - val_loss: 0.6342 - val_accuracy: 0.7849
Epoch 7/20
425/425 [==============================] - 521s 1s/step - loss: 0.1700 - accuracy: 0.9373 - val_loss: 0.7811 - val_accuracy: 0.7283
Epoch 8/20
425/425 [==============================] - 538s 1s/step - loss: 0.1999 - accuracy: 0.9222 - val_loss: 0.6563 - val_accuracy: 0.7302
Epoch 9/20
425/425 [==============================] - 475s 1s/step - loss: 0.1435 - accuracy: 0.9444 - val_loss: 0.7994 - val_accuracy: 0.7491
Epoch 10/20
425/425 [==============================] - 527s 1s/step - loss: 0.1231 - accuracy: 0.9510 - val_loss: 0.7580 - val_accuracy: 0.7736
Epoch 11/20
425/425 [==============================] - 474s 1s/step - loss: 0.1223 - accuracy: 0.9496 - val_loss: 0.7947 - val_accuracy: 0.7679
Epoch 12/20
425/425 [==============================] - 473s 1s/step - loss: 0.1229 - accuracy: 0.9505 - val_loss: 1.0240 - val_accuracy: 0.7434
Epoch 13/20
425/425 [==============================] - 532s 1s/step - loss: 0.0807 - accuracy: 0.9722 - val_loss: 0.8949 - val_accuracy: 0.7736
Epoch 14/20
425/425 [==============================] - 469s 1s/step - loss: 0.1465 - accuracy: 0.9477 - val_loss: 0.8272 - val_accuracy: 0.7717
Epoch 15/20
425/425 [==============================] - 472s 1s/step - loss: 0.1078 - accuracy: 0.9614 - val_loss: 0.9958 - val_accuracy: 0.7321
Epoch 16/20
425/425 [==============================] - 473s 1s/step - loss: 0.0828 - accuracy: 0.9694 - val_loss: 1.6054 - val_accuracy: 0.6566
Epoch 17/20
425/425 [==============================] - 481s 1s/step - loss: 0.0751 - accuracy: 0.9741 - val_loss: 1.1424 - val_accuracy: 0.7340
Epoch 18/20
425/425 [==============================] - 529s 1s/step - loss: 0.0778 - accuracy: 0.9713 - val_loss: 1.0389 - val_accuracy: 0.7604
Epoch 19/20
425/425 [==============================] - 482s 1s/step - loss: 0.0902 - accuracy: 0.9665 - val_loss: 1.1123 - val_accuracy: 0.7717
Epoch 20/20
425/425 [==============================] - 484s 1s/step - loss: 0.0771 - accuracy: 0.9727 - val_loss: 1.0442 - val_accuracy: 0.7774
```

Fig. 5. Output of results for CNN model with 20 Epochs for training and validation dataset

```
87/87 [==============================] - 77s 870ms/step - loss: 2.0722 - accuracy: 0.6273
Test accuracy: 0.6273148059844971
```

Fig. 6 Output of results for CNN model with 20 Epochs for training and validation dataset

During the training, the accuracy was very high, starting at 88.01% and peaking at 97.27% in the final epoch. The loss also improved, starting at 0.2922 in first epoch and finishing at 0.0771. However, the validation accuracy and loss were significantly lower, starting at 76.42% and peaking at 78.49%. Furthermore, the validation loss increased from 0.5899, ending at 1.0442. This loss increase and lower accuracy could be attributed to overfitting.

This is further corroborated by the test set performance, which is significantly worse. The overall accuracy ended up being 62.73% and the loss 2.0722, indicating that the model performs poorly on unseen data.

The primary concern from this model is overfitting, evident from the high accuracy and minimal loss during training, and low accuracy during testing. There are several potential improvements that could be implemented to prevent this. This could include model implementing regularisation techniques such as L1 regularisation (Lasso regression), Dropout and batch normalisation [11]

3.3 Random Forest (RF)

As shown in the figures below, the overall accuracy of the RF model is 80.34%, which is a strong performance, suggesting the model is generally reliable.

```
Accuracy: 0.803371

Classification Report:
              precision    recall  f1-score   support

         0.0       0.96      0.97      0.96       137
         1.0       0.77      0.87      0.82       260
         2.0       0.68      0.50      0.58       137

    accuracy                           0.80       534
   macro avg       0.80      0.78      0.79       534
weighted avg       0.80      0.80      0.80       534

Confusion Matrix:
 [[133   2   2]
 [  3 227  30]
 [  3  65  69]]
```

Fig. 7. Output of Evaluations of Random Forest model

Evaluation Metrics

Accuracy	80.34%
Precision	80%
Recall	80%
F1-score	80%

Table. 3. Evaluation metrics of Random Forest model

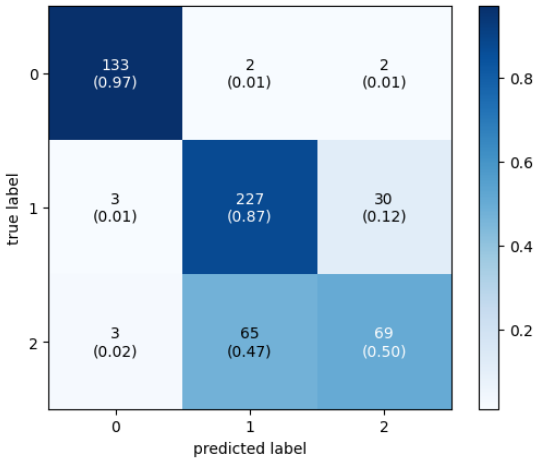

Fig. 8. Graphical Representation of Confusion Matrix of Random Forest model

Examining the performance across different classes, the model achieves a high precision of 0.96 and a recall of 0.97 for normal images (class 0), resulting in a very high F1-score of 0.96. This shows the model is highly accurate in identifying normal X-rays and rarely misclassifies other types of images as normal.

For bacterial pneumonia (class 1), the precision is 0.77, and the recall is 0.87, leading to an F1-score of 0.82. This indicates that the model effectively identifies bacterial pneumonia cases, although it does misclassify some images. The high recall suggests that most bacterial pneumonia cases are correctly identified, even though there are some false positives.

The performance for viral pneumonia (class 2) is less robust, with a precision of 0.68 and a recall of 0.50, resulting in an F1-score of 0.58. This indicates that while the model can identify viral pneumonia cases to some extent, it misses a significant number of actual cases and misclassifies other conditions as viral pneumonia more frequently.

The confusion matrix provides further insights into these metrics. For normal images, there are 133 true positives (TP), 2 false positives (FP), and 2 false negatives (FN), indicating very high accuracy. For bacterial pneumonia, the model has 227 TPs, 30 FPs, and 3 FNs, reflecting its good performance but also stress some misclassifications. For viral pneumonia, the model identifies 69 TPs but has 65 FPs and 3 FNs, showing its struggle with accurately identifying this class.

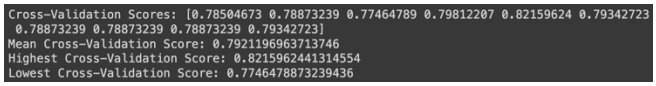

Fig. 9. Graphical Representation of Confusion Matrix of Random Forest model

The cross-validation scores indicate that the model's performance is fairly stable across different subsets of the data, with scores ranging from approximately 0.7746 to 0.8216. The mean cross-validation score of 0.7937 suggests that, on average, the model maintains a high level of accuracy across various splits of the dataset. The relatively narrow range between the highest and lowest scores further demonstrates that the model does not significantly overfit to any subset, affirming its generalizability.

3.4 Comparison of the Three Models

Evaluation Metrics

	Quadratic SVM	CNN	RF
Accuracy	58.49%	75.9%	80.34%
Precision	73%	-	80%
Recall	58%	-	80%
F1-score	56%	-	80%

Table. 4. Evaluation metrics of all three Models

The Quadratic SVM model has the lowest performance across all provided metrics. With an accuracy of 58.49%, it is not highly reliable. While the model is decent at making positive predictions, it misses many actual positives, resulting in a lower overall performance.

The Random Forest model shows the best performance across all available metrics with an accuracy of 80.34%. This consistency indicates that the Random Forest model is well-balanced and reliable for this classification task, excelling in both making correct positive predictions and identifying all actual positives.

The CNN, with its good accuracy, could be promising, but additional metrics are needed for a thorough evaluation. Overall, the Random Forest model emerges as the most robust and reliable among the three, demonstrating high accuracy and a balanced performance in precision, recall, and F1-score.

4. CONCLUSION

In conclusion, this report outlines three methods of classifying chest X-Ray images into three classes using various machine learning techniques. These include SVM, a CNN and Random Forest. Within the non-CNN models, various features were extracted, in accordance with literature recommendations. These included first order statistic features and GLCM features for the random forest and SVM models. The X-Ray images also were pre-processed, by being resized to (1000,1000) and converted to grayscale image arrays. The CNN model was built using a ResNet50 base model, with several layers and data augmentation, as used in several other studies. Whilst a review on literature discusses the high accuracy of CNNs, it often required high computing power, and a large amount of data, which was a limitation in this experiment due to the available technology. Whilst the CNN model performed well during training, in the testing dataset, it was clear that the model was overfitting and future improvements could be made such as adjusting the layers to compensate for overfitting or using a larger dataset. The SVM model initially performed at 58% accuracy but after cross-validation improved to 69.36%. Potential improvements in the model could include data segmentation and using a larger dataset. Finally, the Random Forest model, whilst performing the most accurately, there were still some improvements that could be made, to more accurately distinguish between classes. From Table 4, it is evident that the Random Forest model performed the best in terms of accuracy. In a clinical setting, this is advantageous, due to the little computing power needed to run the model, compared to more complex models such as CNNs.

14. REFERENCES

[1] B. Normandin and J. Seladi-Schulman, “Pneumonia: Symptoms, causes, treatment, and more,” Healthline, 2015. https://www.healthline.com/health/pneumonia

[2] N. Barakat, M. Awad, and B. A. Abu-Nabah, “A machine learning approach on chest X-rays for pediatric pneumonia detection,” Digital Health, vol. 9, p. 205520762311800, Jan. 2023, doi: https://doi.org/10.1177/20552076231180008

[3] M. M. A. Monshi, J. Poon, V. Chung, and F. M. Monshi, "CovidXrayNet: Optimizing data augmentation and CNN hyperparameters for improved COVID-19 detection from CXR," Computers in Biology and Medicine, vol. 133, 2021, Art. no. 104375.Available: https://doi.org/10.1016/j.compbiomed.2021.104375

[4] T. Rahman, M. E. H. Chowdhury, A. Khandakar, K. R. Islam, K. F. Islam, Z. B. Mahbub, M. A. Kadir, and S. Kashem, "Transfer Learning with Deep Convolutional Neural Network (CNN) for Pneumonia Detection Using Chest X-ray," Applied Sciences, vol. 10, no. 9, p. 3233, 2020. https://doi.org/10.3390/app10093233.

[5] D. Hartmann, D. Müller, I. Soto-Rey, and F. Kramer, “Assessing the Role of Random Forests in Medical Image Segmentation.” https://arxiv.org/pdf/2103.16492

[6] Y. J. Kim, “Machine Learning Model Based on Radiomic Features for Differentiation between COVID-19 and Pneumonia on Chest X-ray,” Sensors, vol. 22, no. 17, p. 6709, Sep. 2022, doi: https://doi.org/10.3390/s22176709.

[7] I. Dagher, “Quadratic kernel-free non-linear support vector machine,” Journal of Global Optimization, vol. 41, no. 1, pp. 15–30, Jun. 2007, doi: https://doi.org/10.1007/s10898-007-9162-0.

[8] S. Chatterjee, P. Tummala, O. Speck, and A. Nurnberger, "Complex Network for Complex Problems: A comparative study of CNN and Complex-valued CNN," Faculty of Computer Science, Otto von Guericke University Magdeburg; Data and Knowledge Engineering Group, Otto von Guericke University Magdeburg; Biomedical Magnetic Resonance, Otto von Guericke University Magdeburg; German Center for Neurodegenerative Disease, Magdeburg; Center for Behavioral Brain Sciences, Magdeburg, Germany.

[9] Showkat S, Qureshi S. Efficacy of Transfer Learning-based ResNet models in Chest X-ray image classification for detecting COVID-19 Pneumonia. Chemometr Intell Lab Syst. 2022 May 15;224:104534. doi: 10.1016/j.chemolab.2022.104534. Epub 2022 Mar 11. PMID: 35291673; PMCID: PMC8913041.

[10] Zainab Loukil, A. Mirza, W. Sayers, and I. Awan, “A Deep Learning based Scalable and Adaptive Feature Extraction Framework for Medical Images,” *Information Systems Frontiers*, Jul. 2023, doi: https://doi.org/10.1007/s10796-023-10391-9.

[11] Q. Wang, T.-T. Nguyen, J. Z. Huang, and T. T. Nguyen, “An efficient random forests algorithm for high dimensional data classification,” *Advances in Data Analysis and Classification*, vol. 12, no. 4, pp. 953–972, Mar. 2018, doi: https://doi.org/10.1007/s11634-018-0318-1.

[12] A. Demircioğlu, “Predictive performance of radiomic models based on features extracted from pretrained deep networks,” *Insights into Imaging*, vol. 13, no. 1, Dec. 2022, doi: https://doi.org/10.1186/s13244-022-01328-y.

[13] Kim SM, Kim Y, Jeong K, Jeong H, Kim J. Logistic LASSO regression for the diagnosis of breast cancer using clinical demographic data and the BI-RADS lexicon for ultrasonography. Ultrasonography. 2018 Jan;37(1):36-42. doi: 10.14366/usg.16045. Epub 2017 Apr 14. PMID: 28618771; PMCID: PMC5769953.