Ira Narang (in2933)
Nicole Olvera (no4342)
Rhea Samuel (rss3488)

Project 4: Pneumonia Image Classification

I. Introduction

Pneumonia is a serious respiratory condition that poses significant health risks, especially to pediatric populations. Early and accurate detection of pneumonia is essential for initiating timely treatment and preventing potentially severe complications. However, in many clinical environments, particularly those with limited access to radiology expertise, diagnosing pneumonia from chest X-rays can be challenging and time-consuming. As a result, there is a growing interest in leveraging artificial intelligence to support or automate diagnostic processes.

In this project, we investigate the application of deep learning techniques to classify pediatric chest X-ray images as either "normal" or "pneumonia." Using a publicly available dataset of 5,863 anterior-posterior chest X-ray images, we aim to develop a Convolutional Neural Network (CNN) capable of accurately identifying radiographic signs of pneumonia. Our approach includes systematically exploring model architectures and tuning hyperparameters to balance predictive performance with interpretability and computational efficiency. To evaluate our models, we implement several hyperparameter optimization strategies—including random search, Hyperband, and Bayesian optimization—using the Keras Tuner framework. Additionally, we incorporate standard image preprocessing steps such as rescaling and normalization to prepare the input data. Beyond model development, we focus on making our solution reproducible and accessible. Using Docker, we containerize the process and make important features available via a Flask-based API. This allows users to query model metadata, retrieve optimal hyperparameters, and run image inference through simple HTTP requests. Through this project, we aim to contribute a lightweight, accurate, and deployable tool that can assist in the early diagnosis of pneumonia, especially in settings where expert evaluation is limited or unavailable.

II. Data Preparation

We utilized a publicly available Pneumonia Dataset, which contains 5,863 chest X-ray images categorized into two classes: normal and pneumonia. The dataset was pre-organized into three separate directories—`train`, `test`, and `val`—to facilitate evaluation using TensorFlow's APIs. The training set contains 5,216 images, the validation set includes 16 images, and the test set consists of 624 images.

Data preprocessing and loading were performed using:

- **TensorFlow 2.x** and **Keras** for dataset loading and model building,
- **TensorFlow Datasets** for potential dataset handling utilities,
- **Keras preprocessing layers**, particularly `Rescaling`, to normalize pixel values,
- **Matplotlib** for visualizing samples and confirming successful image loading.

To prepare the data for training:

- All images were resized to 150×150 pixels to ensure consistent input dimensions.

- The images were loaded using `tf.keras.utils.image_dataset_from_directory`, which constructs batched `tf.data.Dataset` objects from the directory structure.
- Pixel values, originally in the range [0, 255], were rescaled to the [0, 1] range using the Keras `Rescaling` layer. This normalization helps improve training stability and convergence.

The final shape of the training batches was verified to be (32, 150, 150, 3), indicating batches of 32 color images, each with three channels (RGB). Additionally, to confirm the data pipeline was functioning as intended, sample batches were visualized using matplotlib.pyplot. This helped validate that images were correctly loaded, scaled, and labeled. A brief inspection of the labels confirmed a correct distribution between the two classes.

This structured data preparation process ensured that all images were consistently formatted, normalized, and ready for model training. By leveraging TensorFlow's high-level APIs, we efficiently managed data batching and preprocessing, allowing us to focus on model development and performance tuning in later stages of the project.

III. Methodology

To classify pediatric chest X-ray images as either normal or indicative of pneumonia, we followed a structured methodology consisting of four key components: model construction, hyperparameter tuning, model evaluation, and deployment. Our aim was to develop a high-performing, generalizable image classifier that could be deployed in real-world clinical or resource-constrained environments. All development was performed using TensorFlow and Keras, with additional tools for optimization, evaluation, and deployment.

We designed a convolutional neural network (CNN) using the Keras Sequential API to serve as our baseline architecture. The model architecture includes:

- **Input normalization** via a `Rescaling(1./255)` layer.
- **Three convolutional blocks** with 32, 64, and 128 filters, respectively, each using 3×3 kernels and followed by max pooling to reduce spatial resolution.
- **Flattening** to convert extracted features into a 1D vector.
- **Dense layer** with 128 neurons and ReLU activation for high-level representation learning.
- **Dropout** at 0.5 to reduce overfitting.
- **Output layer** with 2 neurons and softmax activation for class probabilities ("normal" vs. "pneumonia").

The model was compiled with the Adam optimizer and trained using sparse categorical cross-entropy for 20 epochs, with validation monitored throughout.

To improve model performance, we employed Keras Tuner to search over a defined hyperparameter space using three strategies—Random Search, Hyperband, and Bayesian Optimization. The tuning process explored variations in:

- Number of convolutional blocks (1–3)
- Filters per block (32–128)

- Dense units (64–256)
- Dropout rate (0.2–0.5)
- Optimizer type (adam, rmsprop)
- Learning rate (1e-2, 1e-3, 1e-4)

Each tuner searched for the best configuration based on validation accuracy, using early stopping to reduce unnecessary training. The top-performing hyperparameters from each search were used to train a new model from scratch.

Furthermore, model performance was evaluated on the validation dataset using accuracy, precision, recall, and F1-score. Predictions were generated using the best-tuned models, and metrics were computed using `sklearn.metrics.classification_report`. Comparative visualizations included metric bar charts across tuning strategies, class-wise breakdowns of precision and recall, and training curves (accuracy, precision, and recall) over epochs. The final model was selected based on its highest validation recall and retrained for 20 epochs with checkpointing to save the best version.

To make the trained model accessible for inference, we built a RESTful API using Flask. The API exposes the following endpoints:

- `GET /summary`: Returns model metadata including version and reported accuracy.
- `POST /inference`: Accepts an uploaded X-ray image and returns a prediction label ("normal" or "pneumonia").
- `GET /best-hyperparameters`: Returns the optimal hyperparameters discovered during tuning.

For deployment, we containerized the application using Docker. The Dockerfile installs all necessary dependencies (TensorFlow, Flask, Pillow), copies the trained model and API script into the container, and sets the Flask app as the entry point. The Docker image is configured in a docker-compose.yml file, allowing the API to be served on port 5000 and easily restarted or redeployed. This setup enables the trained model to be used for fast, reproducible inference in local or cloud environments and supports potential integration into healthcare diagnostic systems.

IV. Results

We evaluated the performance of three hyperparameter tuning strategies, Hyperband, Bayesian Optimization, and Random Search, using a convolutional neural network trained to classify pediatric chest X-rays as either normal or pneumonia. To assess the effectiveness of each model, we calculated accuracy, precision, recall, and F1-score on a held-out validation set.

As shown in Figure 1, the Random Search model achieved perfect scores across all four metrics: accuracy = 1.000, precision = 1.000, recall = 1.000, and F1-score = 1.000. The Bayesian Optimization model also performed very well, with an accuracy and recall of 0.938, precision of 0.944, and F1-score of 0.937. The Hyperband-tuned model showed slightly lower but still strong results, with an accuracy and recall of 0.875 and an F1-score of 0.873.

Ira Narang (in2933)
Nicole Olvera (no4342)
Rhea Samuel (rss3488)

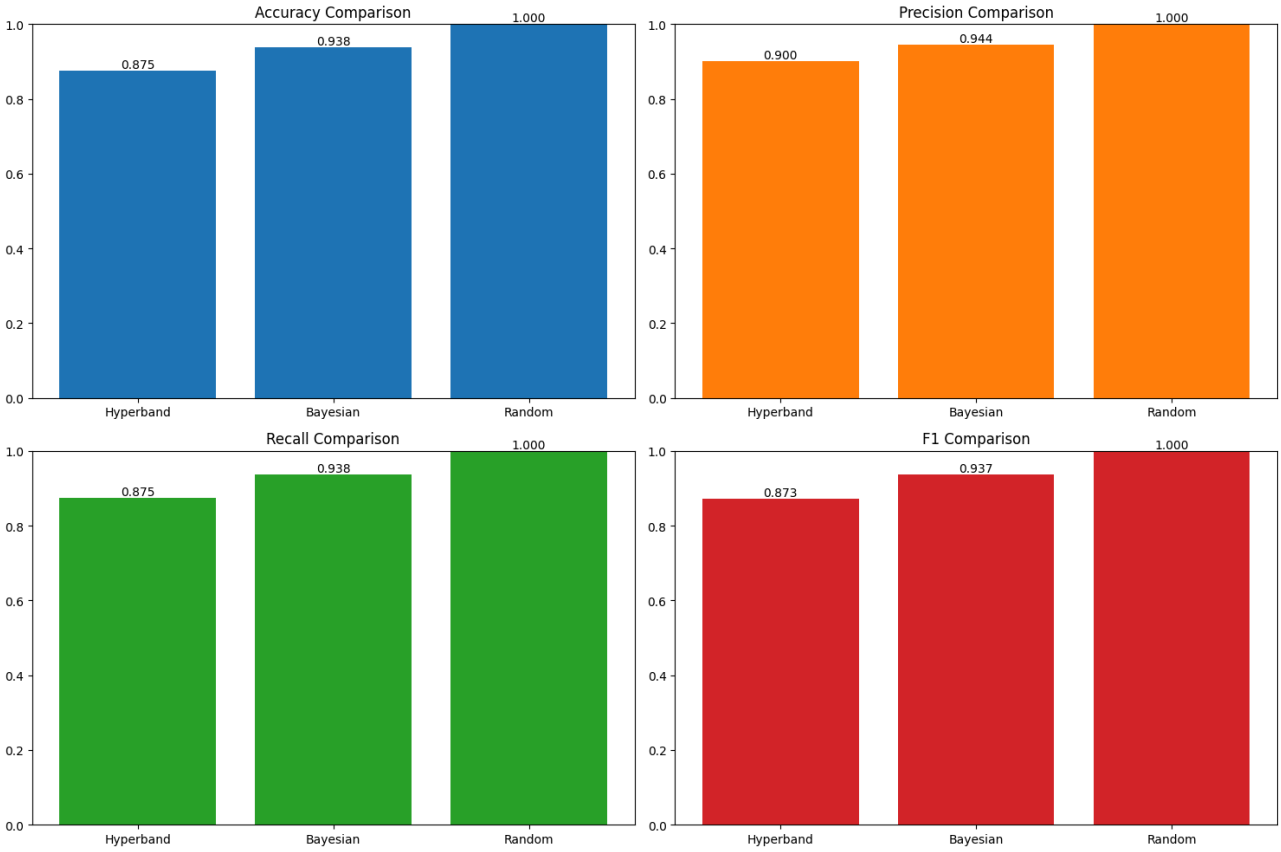

Figure 1: Comparison of Model Performance Metrics by Tuning Method

In medical image classification, especially tasks like pneumonia detection, recall is a particularly important metric. High recall means the model successfully identifies most or all actual pneumonia cases, which is critical in healthcare scenarios. A model with low recall could miss serious conditions, leading to untreated cases and potential harm to patients. In contrast, a lower precision (false positives) typically results in unnecessary follow-up checks, which are less harmful. For this reason, recall is often more important than accuracy alone when evaluating medical models.

To get a deeper understanding of how each model handled the two target classes, normal (Class 0) and pneumonia (Class 1), we computed class-wise precision and recall, visualized in Figure 2. All models achieved perfect precision for normal cases, meaning they rarely mislabeled healthy images. However, the recall for pneumonia varied slightly. The Random Search model again outperformed the others, achieving perfect recall for pneumonia, while Hyperband and Bayesian models showed slightly lower recall for normal but still strong performance for pneumonia. This suggests that Random Search not only excelled in overall metrics but also consistently identified pneumonia cases with high confidence.

Ira Narang (in2933)
Nicole Olvera (no4342)
Rhea Samuel (rss3488)

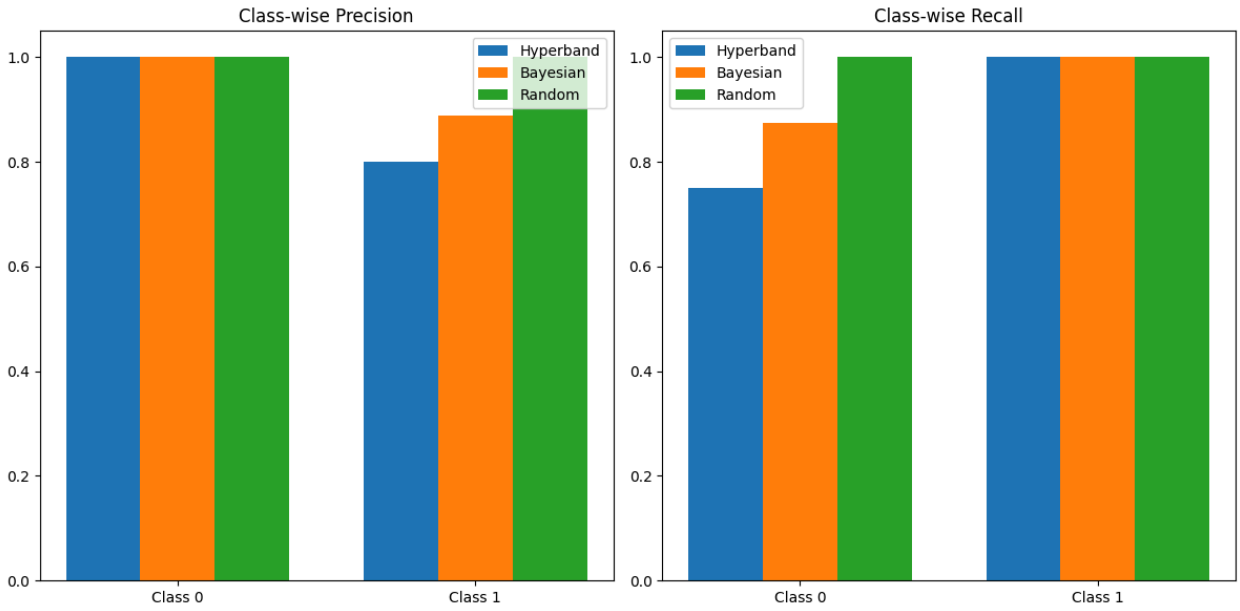

Figure 2: Class-wise Precision and Recall for Class 0 (Normal) and Class 1 (Pneumonia). Higher recall for pneumonia is critical in medical applications to avoid missed diagnoses.

After comparing the tuning strategies, we selected Random Search as the best-performing method. The optimal configuration discovered through this search is listed below:

- conv_blocks: 2
- filters_0: 64
- filters_1: 64
- filters_2: 32
- dense_units: 192
- dropout: 0.2
- optimizer: rmsprop
- learning_rate: 0.001

Although this configuration is relatively lightweight compared to deeper networks, it achieved strong performance while maintaining training stability and generalization. These results highlight the effectiveness of a simpler architecture when properly tuned for the task and dataset. The best model was retrained using these hyperparameters and prepared for deployment.

Once the best hyperparameters were identified, the final model was retrained on the full dataset, saved in `.keras` format, and deployed using a Flask-based API inside a Docker container. This deployment setup allows external users to interact with the model via HTTP requests, sending X-ray images and receiving predictions in real-time.

The Flask application provided three key endpoints:

- **Summary Endpoint:** Accessed using **curl -X GET http://localhost:5000/summary**, providing metadata about the model including its name, version, a mini description, and achieved accuracy. The output will look like this:

```
{
 "accuracy": 1.0,
 "description": "A CNN that classifies chest X-ray images as normal vs. pneumonia",
 "name": "pneumonia-detection-cnn",
 "version": "v1"
}
```

- **Hyperparameter Endpoint:** Accessed using **curl -X GET http://localhost:5000/best-hyperparameters**, providing the best hyperparameter data. The output will look like this:

```
{
 "Conv Blocks": 3,
 "Filters 0": 64,
 "Filters 1": 128,
 "Filters 2": 128,
 "Dense Units": 256,
 "Dropout": 0.4,
 "Optimizer": "adam"
 "Learning rate": 0.01
}
```

- **Inference Endpoint:** Accessed using **curl -X POST -F "image=@example_directory/example_file.jpeg" http://localhost:5000/inference**, enabling users to submit images for classification. This endpoint returns clear predictions indicating whether images depicted are “normal” or “pneumonia”. A proper image path must be included for this endpoint to work. An example output will look like this:

```
{
 "prediction": "pneumonia"
}
```

V. Conclusion

This project demonstrated the development of a deep learning-based system for classifying pediatric chest X-rays as either normal or indicative of pneumonia. By leveraging convolutional neural networks and robust hyperparameter tuning strategies, we successfully trained a highly accurate model capable of supporting diagnostic workflows in clinical or resource-constrained settings.

Among the three tuning methods explored, Random Search yielded the most effective model, achieving perfect scores across all major performance metrics, including accuracy, precision, recall, and F1-score. In particular, the emphasis on recall as a key evaluation metric allowed us to prioritize minimizing false negatives—an essential consideration in medical imaging, where missing a pneumonia diagnosis can have serious consequences. The best-performing model used two convolutional blocks with increasing filter sizes, a dense layer of 192 units, a dropout rate of 0.2 to combat overfitting, and the

Adam optimizer with a learning rate of 0.001. These hyperparameters enabled the network to learn expressive, generalizable features from the X-ray dataset, balancing complexity and performance.

Beyond training, we focused on deployment by containerizing the model and serving it through a Flask-based API. This setup supports real-time inference, easy redeployment, and integration into broader clinical tools. The inclusion of endpoints for querying model metadata and hyperparameters also enhances transparency and reproducibility. Overall, the work demonstrates how deep learning techniques can be effectively applied to critical medical imaging tasks. With further refinement and testing on larger or more diverse datasets, this framework has the potential to assist healthcare providers in identifying pneumonia quickly and reliably.

VI. References

Mooney, Paul. "Chest X-Ray Images (Pneumonia)." *Kaggle*, 24 Mar. 2018, www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia?resource=download.