



# Spark-Based Deep CNN for Pneumonia Detection in Chest X-Rays

Olivia Gette and Andrea Wroblewski

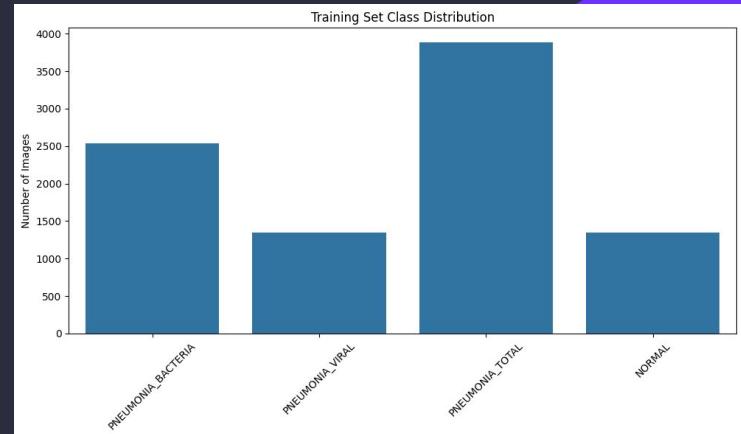
# Project Overview:

**Goal:** Classify chest X-rays in Normal or Pneumonia classes

**Clinical Importance:** Early pneumonia detection improves treatment outcomes.

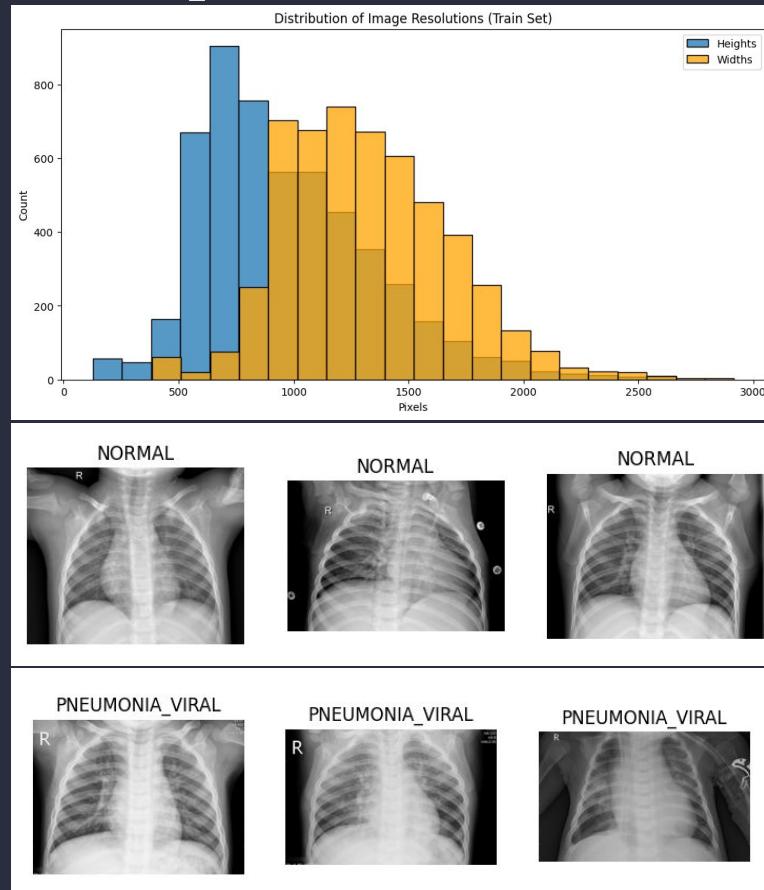
## Dataset:

- Chest X-Ray Pneumonia Dataset (Kaggle)
- 5,856 pediatric X-rays
- Imbalanced dataset ( $\approx 3:1$  pneumonia-heavy)
- Images vary in resolution



# Exploratory Data Analysis

- Imbalanced: 3,883 pneumonia vs 1,349 normal (train)
- Images vary in resolution -> all resized during preprocessing to 64 x 64
- Mean pixel intensity similar across classes (around 122-123)
- Pneumonia has higher intensity variation, means it causes bright/opaque regions
- Histogram of image sizes confirms need for resizing



# Challenges

## Data Challenges:

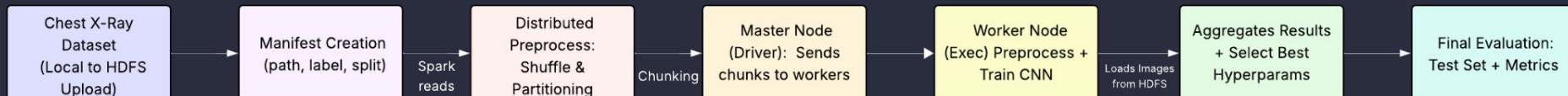
- Image variability (brightness, resolution, orientation)
- Class imbalance
- Risk of overfitting on small dataset

## Technical Challenges:

- VM storage limitations
- CPU-only cluster (slow training)
- Memory constraints
- Occasional Spark job timeouts

# Data Pipeline

- Dataset uploaded to HDFS for scalable access across Spark cluster
- Spark creates a manifest with image paths, labels, and split assignments
- Data is shuffled and randomly split into train/validation/test sets using Spark
- Training data is chunked to prevent memory overload and enable parallel processing
- Worker nodes preprocess and train CNNs on their assigned chunks
- Master node aggregates metrics and selects best hyperparameters for final evaluation



# Distributed Preprocessing on Spark Cluster

## Cluster Setup:

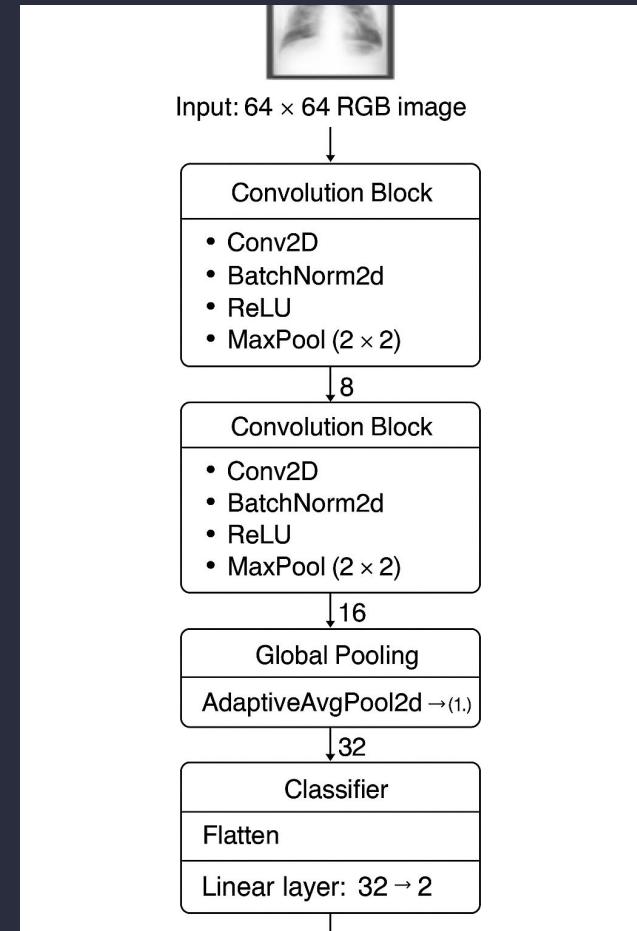
- 1 Master
- 3 Workers

## Preprocessing Steps:

- Convert image directories to Spark DataFrame (path, label, index)
- Shuffle to avoid directory-order bias
- Resize images to 64×64
- Save train/val/test splits as partitioned Parquet files in HDFS

# CNN Architecture

- **3 Convolutional Blocks:**
  - Conv2D
  - BatchNorm2d
  - ReLU
  - MaxPool (2x2)
- **Global Pooling:**
  - AdaptiveAvgPool2d → (1, 1)
- **Classifier:**
  - Flatten
  - Linear layer: 32 → 2 classes (Normal vs Pneumonia)



# Hyperparameters

Category	Key Choices
Input	64x64 resized, batch size = 8, normalize (mean=0.5, std=0.5)
Training	Adam, CrossEntropyLoss, LR $\in \{0.001, 0.01\}$ , 1 epoch/chunk
Distributed	20 chunks (~292 images each), 6 partitions/chunk
Regularization	BatchNorm, MaxPool, AdaptiveAvgPool

# Distributed Training & Tuning

## Distributed Strategy:

- Dataset split into ~20 chunks (~292 images each)
- Each chunk further divided into 6 partitions
- Each worker trains on its assigned subset
- 1 epoch per chunk to prevent long runtimes

## Hyperparameter Tuning:

- Workers randomly select LR from {0.001, 0.01}
- Each worker logs loss + processed images
- Master aggregates results to identify best LR
- Final LR chosen based on lowest avg distributed loss

# Results

-  Accuracy: 94%
-  Normal Precision: 0.96
-  Normal Recall: 0.81
-  Normal F1: 0.88
-  Pneumonia Precision: 0.94
-  Pneumonia Recall: 0.99
-  Pneumonia F1: 0.96
-  Macro Avg F1: 0.92
-  Weighted Avg F1: 0.94

Actual	Normal	Pneumonia
Normal	252	59
Pneumonia	11	850
Predicted		

	precision	recall	support
Normal	0.96	0.81	311
Pneumonia	0.94	0.99	861
accuracy		0.94	1172
macro avg	0.95	0.92	1172
weighted avg	0.94	0.94	1172

# Conclusion & Future Work

## Conclusion

- Built distributed CNN pipeline with Spark and PyTorch
- Achieved a 94% accuracy and 99% recall for pneumonia
- Chunk-based training enabled scaling without VM crashes
- Demonstrated feasibility of medical image classification on limited hardware

## Future Work

- Add data augmentation (flips, rotations)
- Explore higher resolutions (128 x 128, 224 x 224)
- Test transfer learning models (DenseNet121, ResNet18, EfficientNet)
- Integrate Grad-CAM heatmaps for interpretability
- Apply dropout/L2 regularization for deeper models