COMP 448/548 – MEDICAL IMAGE ANALYSIS
FINAL PROJECT
A Comparison of Traditional Machine Learning and Deep Learning Approaches for Chest X-Ray Analysis for Pneumonia Detection

1. Introduction

Medical image analysis has increasingly leaned on towards deep learning due to its impressive performance in various applications. However, traditional machine learning algorithms still offer valuable insights, especially in scenarios with limited data or computational resources.

The goal is to evaluate the performance and reliability of these two approaches. While deep learning models like Unet are popular for their ability to learn complex patterns, traditional methods such as SVM combined with effective feature extraction techniques remain relevant. This project will explore various feature extraction techniques for the SVM approach and compare them to the features learned by the Unet architecture. The dataset includes 10,192 normal lung X-ray images and 1,345 pneumonia X-ray images. To address class imbalance, three different techniques were implemented: undersampling, oversampling, and class weighing. For each method, four feature extraction techniques were experimented: raw pixel values, histogram of oriented gradients (HOG), local binary patterns (LBP), and intensity histograms.

To choose the optimal solution considering class balancing and feature extraction techniques, I developed a metric called the Combined Performance-Time Score(CPTS). This metrics focuses on accuracy, F1 score, and the time spent on training and feature extraction. The CPTS prioritizes evaluation metrics over time spent to ensure that statistical performance remains the primary focus.

Dataset: https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database/data

2. Support Vector Machine (SVM)

In this section, I will explain the methods implemented in the SVM and share the results obtained.

2.1. Optimal Parameter Selection for Feature Extraction

This part of the project was one of the most time-consuming due to the complexity of handling four different feature extraction techniques, since each requires a unique set of parameters. For each feature extraction technique, different sets of parameters were experimented with to optimize the feature representation. Given the total of 11,537 images, testing each set of parameters for every technique would be computationally intensive. Therefore, after addressing the class imbalance through undersampling, and oversampling techniques, a random subset containing 300 X-ray images of normal and pneumonia images, along with their corresponding labels, was selected for experimentation. Additionally, a random subset for class weighting was selected without applying any undersampling or oversampling. Here is the list of parameters for each technique:

- **Raw Pixel Values:** Image Size
- **Histogram of Oriented Gradients (HOG):** Image Size, Orientations, Pixels Per Cell, Cells Per Block, Block Norm
- **Local Binary Patterns (LBP):** Radius, Number of Points, Histogram Bins
- **Intensity Histogram:** Bins

For each of these techniques, different sets of parameters were experimented with, and the parameters that produced the optimal solutions considering both evaluation scores and time consumption were selected. The final version of these parameters is shared in the FP_SVM_85020.ipynb file. Using these parameters, the SVM model was trained and tested under three different conditions to address the class imbalance problem: undersampling, oversampling, and class weighting. For each condition, the Feature Extraction Time for the subsets are listed below:

Method	Number of Image Samples	Feature Extraction Time (sec)
Raw Pixel Features	300	62.52
Histogram of Oriented Gradients	300	4.61
Local Binary Patterns	300	22.61
Intensity Histogram	300	2.08

Table 2.1.1 Feature Extraction Time on the Undersampled Subset

Method	Number of Image Samples	Feature Extraction Time (sec)
Raw Pixel Features	300	77.71
Histogram of Oriented Gradients	300	5.36
Local Binary Patterns	300	19.36
Intensity Histogram	300	2.39

Table 2.1.2 Feature Extraction Time on the Oversampled Subset

Method	Number of Image Samples	Feature Extraction Time (sec)
Raw Pixel Features	300	111.30
Histogram of Oriented Gradients	300	5.29
Local Binary Patterns	300	19.00
Intensity Histogram	300	2.97

Table 2.1.3 Feature Extraction Time on the Subset (Imbalanced Subset)

Raw Pixel Features took the longest time for feature extraction. This is expected because this method involves flattening the entire image into a feature vector, and it is computationally intensive due to the large number of pixels.

The HOG method is significantly faster than raw pixel extraction. This is because HOG focuses on capturing essential edge and gradient information rather than processing every pixel. The efficiency of HOG makes it a practical choice for feature extraction in terms of balancing both speed and the quality of features extracted.

LBP takes a moderate amount of time for feature extraction compared to the other methods. This technique captures texture information by computing local binary patterns and histograms.

The Intensity Histogram method is the fastest among all the techniques. It involves calculating the distribution of pixel intensities.

2.2. Train, Validation and Test for SVM

The train_svm function is designed to train and evaluate an SVM classifier. Extracted features and their corresponding labels are provided to the train_svm function. The function creates an SVM pipeline with standard scaling and, optionally, class weighing to handle imbalanced classes. It contains a grid of hyperparameters for optimization, including kernel types, regularization parameters, and kernel coefficients. Using k-fold cross validation and grid-search, the function determines the best hyperparameters and the corresponding mode. It then trains the best model on the entire training set and evaluates the model on the test set.

The cross-validation metrics with standard deviations are presented for each feature extraction technique for the subsets are listed below:z

Method	CV Accuracy	CV Precision	CV Recall	CV F1 Score
Raw Pixel Features	0.942 ± 0.031	0.947 ± 0.026	0.942 ± 0.031	0.942 ± 0.030
Histogram of Oriented Gradients	0.933 ± 0.016	0.938 ± 0.012	0.933 ± 0.016	0.934 ± 0.015
Local Binary Patterns	0.896 ± 0.013	0.897 ± 0.013	0.896 ± 0.013	0.896 ± 0.013
Intensity Histogram	0.850 ± 0.055	0.855 ± 0.054	0.850 ± 0.055	0.850 ± 0.053
Table 2.2.1. Cross-Validation Metrics on the Undersampled Subset				

Method	CV Accuracy	CV Precision	CV Recall	CV F1 Score
Raw Pixel Features	0.946 ± 0.017	0.951 ± 0.014	0.946 ± 0.017	0.945 ± 0.017
Histogram of Oriented Gradients	0.938 ± 0.019	0.941 ± 0.020	0.938 ± 0.019	0.937 ± 0.019
Local Binary Patterns	0.879 ± 0.024	0.884 ± 0.028	0.879 ± 0.024	0.878 ± 0.024
Intensity Histogram	0.825 ± 0.069	0.837 ± 0.072	0.825 ± 0.069	0.825 ± 0.069
Table 2.2.2 Cross-Validation Metrics on the Oversampled Subset				

Method	CV Accuracy	CV Precision	CV Recall	CV F1 Score
Raw Pixel Features	0.967 ± 0.021	0.967 ± 0.022	0.967 ± 0.021	0.963 ± 0.023
Histogram of Oriented Gradients	0.950 ± 0.021	0.955 ± 0.018	0.950 ± 0.021	0.952 ± 0.020
Local Binary Patterns	0.921 ± 0.055	0.912 ± 0.063	0.921 ± 0.055	0.914 ± 0.060
Intensity Histogram	0.921 ± 0.048	0.918 ± 0.051	0.921 ± 0.048	0.916 ± 0.052
Table 2.2.3 Cross-Validation Metrics on the Class-Weighted Subset				

The test metrics with training time are presented for each feature extraction technique for the subsets are listed below:

Method	Test Accuracy	Test Precision	Test Recall	Test F1 Score	Training Time (sec)
Raw Pixel Features	0.967	0.969	0.967	0.967	76.10
Histogram of Oriented Gradients	0.933	0.936	0.933	0.934	231.39
Local Binary Patterns	0.867	0.867	0.867	0.867	5.63
Intensity Histogram	0.850	0.850	0.850	0.850	7.27
Table 2.2.4 Test Metrics on the Undersampled Subset					

Method	Test Accuracy	Test Precision	Test Recall	Test F1 Score	Training Time (sec)
Raw Pixel Features	0.950	0.954	0.950	0.950	37.19
Histogram of Oriented Gradients	0.850	0.869	0.850	0.847	228.54
Local Binary Patterns	0.867	0.881	0.867	0.865	3.84
Intensity Histogram	0.783	0.798	0.783	0.779	9.73
Table 2.2.5 Test Metrics on the Oversampled Subset					

Method	Test Accuracy	Test Precision	Test Recall	Test F1 Score	Training Time (sec)
Raw Pixel Features	0.983	0.985	0.983	0.984	168.77
Histogram of Oriented Gradients	0.950	0.869	0.965	0.950	846.45
Local Binary Patterns	0.967	0.967	0.967	0.967	13.26
Intensity Histogram	0.950	0.948	0.950	0.948	17.95
Table 2.2.6 Test Metrics on the Class-Weighted Subset					

The CPTS values are presented for each feature extraction technique for the subsets are listed below:

CPTS = $\alpha \cdot$(Accuracy + F1 Score) + $\gamma \cdot (1/$(Training Time+Feature Extraction Time), α : 0.35, γ: 0.3

Method	Raw Pixel	HOG	LIB	IH
Undersampled Subset	0.6791	0.6547	0.6175	0.6271
Oversampled Subset	0.6676	0.5952	0.6191	0.5715
Class-Weighted Subset	0.6896	0.6654	0.6862	0.6786
Table 2.2.7. CPTS For Different Features and Subsets				

The trade-off between computational time and accuracy must be considered when choosing the optimal method. The results suggest that while Raw Pixel Features can still achieve high performance, the choice of feature extraction and handling class imbalance significantly affects the overall efficiency and effectiveness of the model.

Method	CV Accuracy	CV Precision	CV Recall	CV F1 Score
Raw Pixel Features	0.987 ± 0.005	0.989 ± 0.005	0.988 ± 0.005	0.988 ± 0.005
Table 2.2.8 Cross-Validation Metrics on the Class-Weighted Complete Dataset				

Method	Test Accuracy	Test Precision	Test Recall	Test F1 Score	Training Time (sec)
Raw Pixel Features	0.983	0.985	0.983	0.984	168.77
Table 2.2.9 Test Metrics on the Class-Weighted Full Complete Dataset					

The improvement in cross-validation and test metrics indicates a significant enhancement in the model's performance. The reduced standard deviation suggests that more consistent model performance across different folds. The test metrics also improved. We can infer that the model generalizes well to unseen data.

Feature Extraction Time: 5389,25 sec.

Training Time: 6498,33 sec.

Total Time: 11878,58 sec.

CPTS value for SVM model: 0.6885

3. CNN Architecture (ResNet)

Training neural networks from scratch is often a data-hungry process. I experimented with different setups to train a CNN model. However, it did not go well. Initially starting a run with 10 epochs with a modest 300 samples(same as in SVM part), the performance of the model likely reflected the limited ability of neural networks to generalize from small datasets. As I incrementally increased the dataset size to 1000 and then 5000 samples, I began to observe some improvement in model performance. I used the entire dataset for training, I obtained slight improvements because the model needed to run more epochs than 10, which I needed more computational power.

To determine the optimal hyperparameters for the model, I experimented with a series of trials varying key parameters such as learning rate, batch size, and optimizer type. Each trial was evaluated based on both its loss and accuracy during training and its ability to generalize as indicated by the validation loss and other metrics such as precision, recall and F1 score. Each trial was run for multiple epochs to ensure

robustness in the findings. After approximately 40 minutes of testing, I concluded that the trial with a learning rate of 0.01, batch size of 16, and the SGD optimizer yielded the best results. The combination achieved the highest validation accuracy and the best balance among precision, recall and F1 score. Considering that balance, it was inferred that the model was trained well to generalize without overfitting. The experiment was conducted on 1000 samples.

In this project, I addressed the class imbalance in the dataset by implementing class weighting for both SVM and neural network model. Class imbalance can lead to model bias towards frequent classes. To correct this behavior, I first analyzed the class distribution and calculated weights inversely proportional to class frequencies. These weights were then integrated into Cross-Entropy Loss. This approach adjusts the training process to emphasize the minority classes by penalizing misclassifications of these classes more than those of the majority class.

Method	Accuracy	Precision	Recall	F1 Score
Train Evaluation Metrics (Mean)	0.915 ± 0.043	0.922 ± 0.044	0.911 ± 0.044	0.917 ± 0.044
Validation Evaluation Metrics (Mean)	0.920 ± 0.043	0.927 ± 0.044	0.916 ± 0.044	0.922 ± 0.044
Test Evaluation Metrics	0.952	0.960	0.948	0.954

Table 3.1.1 Evaluation Metrics on the pretrained ResNet

Parameter optimization: 2482,8 sec

Training + Validation + Test Time: 11238,41 sec

CPTS value for pretrained ResNet model: 0.6671

4. Conclusion

Considering the CPTS metric, SVM performed better than ResNet by achieving a score of 0.6885 compared to ResNet's 0.6671. This metric indicates that SVM not only provided a better balance between efficiency and accuracy, but also adapted more effectively to the constraints of limited data and computational resources. In terms of the computational demands of SVM and ResNet, SVM clearly stands out for its lower dependency high-end hardware like GPU. Unlike ResNet, SVM can operate effectively on a CPU.

The choice between SVM and ResNet for a particular application will significantly depend on the available computational resources, the size and complexity of the dataset, and the specific needs for manual feature extraction. Where computational resources are limited and datasets are smaller or domain-specific, SVMs manual feature extraction method may be more practical. Pretrained models such as ResNet bring a big advantage because they learn from huge datasets beforehand, which means they need less data and effort to adapt to new tasks. This makes them super handy for quickly tackling new problems. On the other hand, SVM starts from zero every time, requiring careful setup and tweaking for each specific task. While pretrained models speed things up significantly, their success depends on how similar the new task is to what they've already learned. SVM, while more hands-on, is great for unique challenges where you need to craft the solution more precisely.