Data Mining and Convolutional Neural Networks for Chest Disease Prediction

Mert Gülşen
Faculty of Computer and Informatics
Istanbul Technical University
Istanbul, Turkey
gulsen20@itu.edu.tr

***Abstract*——In this project, data mining methods for classification and state of the art convolutional neural networks (CNNs) are compared. Classification task for this project is detecting the type of disease a person has given their chest x-ray image. CNNs achieve state of the art results on image classification tasks. Aim of this project is to compare these state of the art results with data mining classification algorithms and see which algorithms score better accuracy and why it is the case.**

***Index Terms*—data mining, computer vision, image, classification, prediction**

I. Introduction

The problem for this project is performing a multi-class classification for detecting the type of disease a person has.

Background: Importance of this problem has increased with the recent COVID-19 pandemic. Chest diseases have become very common and patient numbers have increased. Due to these factors, fast automated diagnosis of diseases have become a very useful task.

Overall Purpose: This project considers the problem of disease prediction and studies different algorithms in order to get a better understanding. Goal is to use CNNs and data mining techniques separately or together to understand their impacts on the problem.

A. The problem

The problem is predicting class labels of 4 different types of patients. These are normal, COVID-19, pneumonia and tuberculosis patients. In the dataset for this problem, there are a total of 7135 images. The dataset is split into 3 parts which are train, validation and test sets. In each set, 4 different classes of images are found for each type of patient.

Automating the task of diagnosing a patient is very important because population of the world continues to grow but there are only limited number of medical institutions. Therefore automating this task by machine learning is essential to provide faster medical attention for patients.

B. Proposed solution

In this project, decision tree, random forest and CNN models are proposed. First, these models will be trained and tested separately. And then another model is proposed which is a combination of CNN and decision trees. In this model, CNN will extract spatial features via convolution operation and decision tree will be used to perform classification on extracted features. This model could get a better score than separate models because CNNs can extract spatial information and decision trees can perform better classification on them.

II. Related Work

Decision trees and random forests are supervised learning algorithms used for classification problems. Random forest is an ensemble method which was developed in 2001. [1] It combines multiple decision trees to get majority vote on classification.

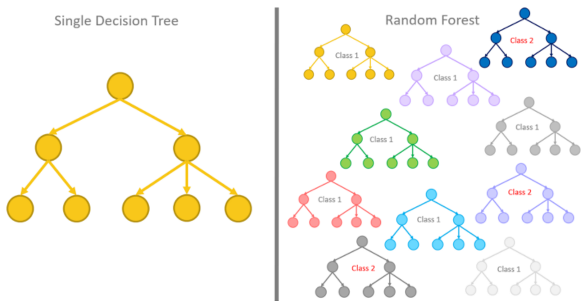

Fig. 1. Decision tree and random forest diagrams.

Decision trees have nodes at each level that splits samples using information on an attribute. Random forest models have multiple decision trees trained with bootstrap sampling as illustrated in [2, Fig. 1].

Previous work on disease prediction with x-ray images consider CNNs and deep learning models directly the best solution for the problem. In the study [3], a CNN model is proposed. CNNs can perform on images better on their own because they consider spatial information of images. But in the study [3], they only compare the recent CNN models to find optimal solution to the disease prediction problem. However data mining algorithms are not compared to these models. Another study on disease prediction [4], uses CNN and decision trees together to classify patients. In this project, data mining algorithms are both tested separately and along with CNNs to compare and see their performance.

III. Proposed Work

A. Dataset and tools

In the dataset there are 6326 images for train split, 38 images for validation split and 771 images for test split. Test size to train size ratio is approximately 0.12. Goal is training models on train split and doing hyper-parameter search on validation set. Finally getting a score on test split.

Dataset is downloaded from Kaggle and read with python. At first step, train, validation and test split are read and stored in python dictionaries.

Images in the dataset are in different shapes. All images have different sizes and there are images with 1, 3 and 4 channels. Pixel values are between 0 and 255.

Experiments are carried out on python language with jupyter notebook. At the first part of the project, scikit-learn library along with other libraries are used to read dataset and train decision tree models.

B. Preprocessing steps

Each disease type is encoded where normal patient label is 0, COVID-19 patient label is 1, pneumonia patient label is 2 and tuberculosis patient label is 3. In the process images of reading and storing images, they are converted to gray-scale first and resized into a pre-defined square shape. Size of the images is a hyperparameter to search. After converting image n to gray-scale, each pixel

$$I_{ij}^{(n)} \in [0, 255] \tag{1}$$

has a value between 0 and 255 where i represents the row number and j represents the column number and n is the sample index of the image in dataset.

After resizing, images are standardized. For decision tree based models, standardization is not effective but to train CNNs, standardization is required to get better results. Standardization operation

$$X' = \frac{(X - \mu)}{\sigma} \tag{2}$$

is done on dataset X where μ is the mean and σ is the standard deviation of the features. Finally, train, validation and test split are shuffled to get a random dataset.

C. Decision Tree and Random Forest Models

For data mining algorithms part of the project, two models are trained:

- Decision Tree Classifier
- Random Forest Classifier

Goal is using decision tree and random forest models to train a function

$$f : I \longrightarrow \{0, 1, 2, 3\} \tag{3}$$

that takes an image I as input and outputs one of the class labels.

Decision tree and random forest models are trained on CPU. Decision tree classifier took approximately 180 seconds to train. Images were resized to 128 by 128. Processed training dataset shape was 6326 by 16384. Maximum depth of decision tree was unlimited but minimum sample split number was set to 2 by default. Random forest model took approximately 49 seconds to train. Decision tree estimators were set to 10 as default.

At this stage of the project, validation split was not used because there were 38 images. Instead, models were tested directly on test split to get evaluation scores. Using the models, prediction was done on test data and predictions were compared with ground truth test labels.

D. Random Forest Model with CNN Feature Extraction

Random forest models can not use spatial features of images on their own. To improve random forest model scores, instead of training with flattened images, a CNN model is used to extract features. Then using these extracted features, a random forest model is trained. To extract the features from images, MobileNetV2 [6] is used.

Images are resized to 256 instead of 128 to extract more features from them. First convolution layer of MobileNetV2 is modified so that it accepts 1 channel images instead of 3 channel images. This is because all images were converted to grayscale. After the last layer of MobileNetV2, 2 MaxPool layers are added to reduce features to 1 dimensional vectors. Shape of features with all images after extracting is 6326 by 1280. Where 6326 is the length of dataset.

After extracting the features, a random forest model is trained with these features. In this random forest model, estimator number were set to 75. Extraction and training took around 30 minutes combined.

E. Convolutional Neural Network Model

To compare results with decision tree based models, CNN model is trained. For the CNN model, ResNet50 [5] is trained with PyTorch library.

Training set split is used for training with batches and validation set is used for calculating validation loss and accuracy while training. Images are resized to 128 by 128. The model is trained on GPU. Loss function used for the model is cross entropy loss.

Using PyTorch library, a custom dataset object and dataloaders were created. Also because all the images in dataset was converted to gray-scale, first convolution layer of ResNet50 model was modified to accept images with 1 channel which was done by reducing in-filter size to 1 from 3. Training took approximately 15 minutes. Training details of CNN model is listed as follows:

- Epochs = 50
- Batch size = 64
- Image size = 128
- Learning rate = 0.01
- Optimizer = Adam
- Regularization strength = 10^{-5}

After training the model for 50 epochs (loops over training set), test accuracy on test set is calculated and learning curves are plotted.

Fig. 2 and Fig. 3 shows loss and accuracy curves of CNN model. When learning curves are analyzed, it can be concluded that model converged and performs well on training

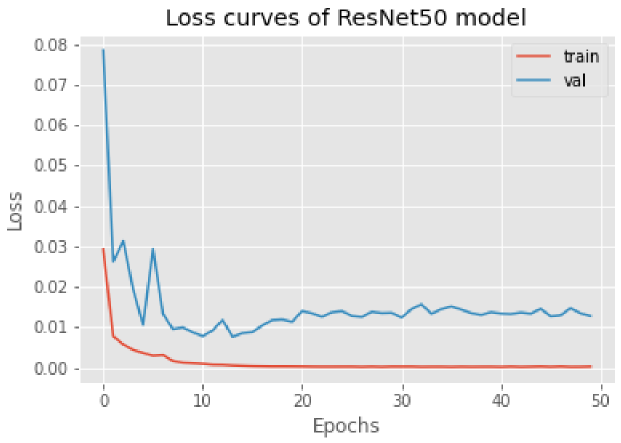

Fig. 2. Loss curves of CNN model.

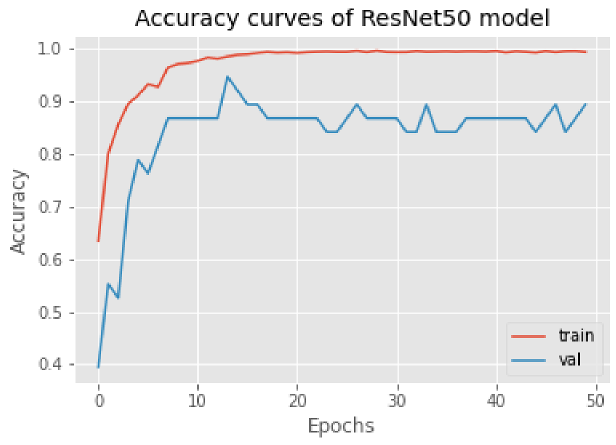

Fig. 3. Accuracy curves of CNN model.

set. Validation loss above the training loss curve which is as expected. Similarly, training accuracy is higher than validation accuracy and plateaued around 1.

IV. Experimental Results

Using scikit-learn library, a classification report and a confusion matrix was created on decision tree and random forest models. This was done by using labels predicted by each model and ground truth labels of test split. As evaluation metrics, precision, recall, f1-score, accuracy and confusion matrix will be used. In the classification reports of models, precision, recall and f1 score of each class will be analyzed.

TABLE I

	Test Accuracy
Decision Tree Model	0.68
Random Forest Model	0.74
Random Forest with CNN	0.76
CNN Model	0.79

Accuracy scores of different models on test set.

TABLE II

	Precision	Recall	F1-score	Support
0	0.68	0.32	0.44	234
1	0.74	0.63	0.68	106
2	0.70	0.91	0.79	390
3	0.42	0.66	0.51	41

Classification report of decision tree model.

TABLE III

	Precision	Recall	F1-score	Support
0	0.95	0.33	0.49	234
1	0.94	0.75	0.84	106
2	0.72	0.98	0.83	390
3	0.45	0.80	0.58	41

Classification report of random forest model.

TABLE IV

	Precision	Recall	F1-score	Support
0	0.94	0.47	0.63	234
1	1.00	0.54	0.70	106
2	0.69	0.98	0.81	390
3	0.85	0.80	0.83	41

Classification Random Forest with CNN extraction model.

Accuracy scores of the models are presented in Table I. Decision tree model has an accuracy of 0.68 and random forest model has an accuracy of 0.74. It shows that random forest model is better in terms of accuracy. Random forest model with CNN feature extraction achieved test accuracy of 0.76. There is an improvement compared to decision tree and random forest models. This indicates that use of CNN in models improves accuracy scores.

For this dataset, class labels are imbalanced. For the test split there are 234 normal patients, 106 COVID-19 patients, 390 pneumonia patients and 41 tuberculosis patients. To evaluate models better, we can check precision, recall and f1 scores.

For a class label i, precision tells how many of samples that are predicted as i has ground truth label as i. Recall score tells out of all ground truth label i samples, how many of them are predicted as i. F1-score is harmonic mean of precision and recall therefore, it takes both of them into consideration.

Table II shows precision, recall and f1-scores of decision tree model on test data. For normal patients, decision tree model has precision score of 0.68 and low recall score which is 0.32. This means that normal patient prediction of decision tree model is accurate, however, model can't detect all of normal patients accurately.

Instead of comparing each class scores, we can directly check f1-scores to get an intuition of model in terms of precision and recall. Class label 2, which is pneumonia, has the best f1-score among all classes which is 0.79. Model can classify pneumonia patients very accurately. Normal patient and tuberculosis patient classes have relatively low f1-scores. This shows the effect of imbalance of the dataset.

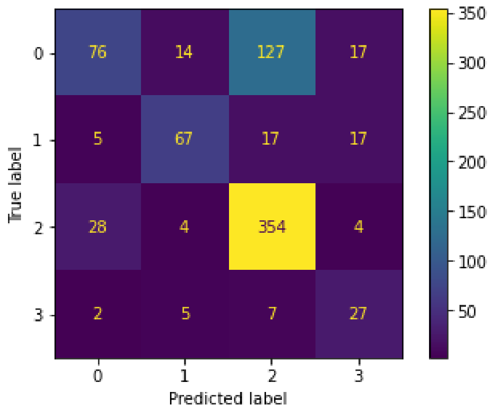

Fig. 4. Confusion matrix of decision tree model.

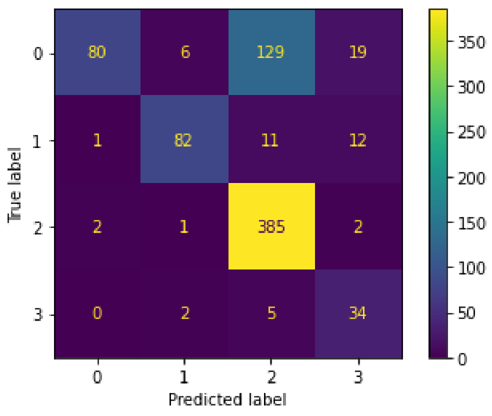

Fig. 5. Confusion matrix of random forest model.

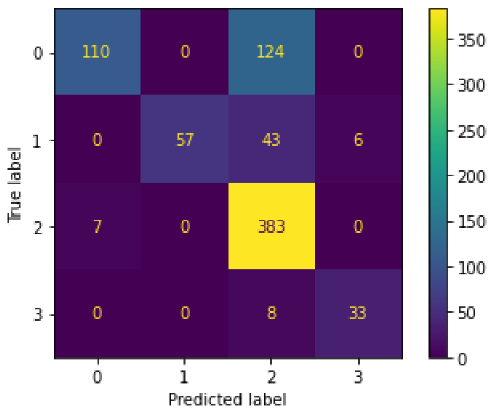

Fig. 6. Confusion matrix of random forest with CNN extraction model.

Table III shows precision, recall and f1-scores of random forest model on test data. Compared to decision tree scores, all scores have improvement. Precision of the model is significantly better along with an improvement on recall. Consequently, f1-score of model is better. This shows that random forest model performs better on imbalanced dataset than decision tree model.

Table IV shows precision, recall and f1-scores of random forest with CNN extraction model on test data. When compared to decision tree model, accuracy has increased from 0.68 to 0.76 and there is a clear improvement in precision, recall and f1-scores. When random forest model is compared, accuracy has increased from 0.74 to 0.76 and precision, recall and f1-scores are generally better.

Finally, confusion matrix of the models are evaluated. Fig. 4 shows confusion matrix plot of decision tree model. On the diagonal elements of matrix, there are true positive labels. To see where model is confused, off-diagonal entries should be analyzed.

It is seen that decision tree model is performing worst at class label 0 which is normal patients. Specifically, model predicted label 2 on 127 samples which is pneumonia, when it should have predicted label 0 which is normal patient.

In the Fig. 5, confusion matrix of random forest model can be seen. As it was also seen from classification report, random forest performs better on the dataset. However, there wasn't any significant improvement on classification of class label 0 which is normal patient.

In the Fig. 6, confusion matrix of random forest with CNN extraction model can be seen. Compared to decision tree and random forest models, there is an clear improvement on all labels except the label 1. Model predicts COVID-19 patients as Tuberculosis patients a lot more.

V. Conclusion

At this stage of the project, decision tree and random forest models are trained and evaluated. To train the models, dataset was analyzed and preprocessed with appropriate techniques. Therefore, a pipeline was created to solve the problem. Models were evaluated on test data and results are reported. Models are compared based on the results. It is found that random forest model performs better than decision tree model. Random forest model handles imbalanced dataset better for this problem.

When CNN models are used on the dataset, results have improved. To improve random forest model results, CNN extraction is used to train random forest model. And to compare these models to CNNs, a CNN model is trained and accuracy score was calculated. To improve the results of all models, augmentation methods can be applied to dataset. It can be concluded that CNNs perform better on image data due to extraction of spatial features.

References

[1] L. Breiman, Machine Learning, vol. 45, no. 1. Springer Science and Business Media LLC, pp. 5–32, 2001.

[2] R. Silipo, "From a single decision tree to a random forest," Medium, 08-Oct-2019. [Online]. Available: https://towardsdatascience.com/from-a-single-decision-tree-to-a-random-forest-b9523be65147. [Accessed: 01-Dec-2022].

[3] F. M. J. M. Shamrat et al., "LungNet22: A Fine-Tuned Model for Multiclass Classification and Prediction of Lung Disease Using X-ray Images," Journal of Personalized Medicine, vol. 12, no. 5, p. 680, April 2022

[4] S. H. Yoo et al., ‘Deep Learning-Based Decision-Tree Classifier for COVID-19 Diagnosis From Chest X-ray Imaging’, Frontiers in Medicine, vol. 7, 2020.

[5] He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*. pp. 770-778, 2016.

[6] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. & Chen, L. Mobilenetv2: Inverted residuals and linear bottlenecks. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*. pp. 4510-4520, 2018.