**Applied machine learning system ELEC0134 22/23 report**

*SN:23202440*

#### Abstract

In this paper, various machine learning techniques were applied to address two distinct challenges: Binary classification for Pneumonia Detection and Multi-classification for Colorectal tissue classification, utilizing the MNIST dataset. The employed algorithms encompass Logistic Regression (LR), K Nearest Neighbors (KNN), Support Vector Machines (SVM), and Convolutional Neural Networks (CNN).

In the binary classification task, LR, KNN, and CNN achieved approximately 84%, 84%, and 90% accuracy rates respectively on the test set. In the multi-classification task, SVM and CNN showcasing accuracy rates 63% and 83% respectively.

**Index Terms—** Supervised Learning, Unsupervised Learning, PCA, Image Processing, Biomedical Diagnostics

**1. Introduction**

Medical image analysis plays a pivotal role in modern healthcare, offering indispensable tools for diagnosis and treatment in recent years. The integration of machine learning introduces innovative techniques that enhance the overall efficacy of medical image analysis. These emerging algorithms contribute significantly to traditional health and clinical treatment methodologies.

This report delves into the application of machine learning methods to address two distinct challenges: Binary classification for Pneumonia Detection and Multi-classification for Colorectal tissue classification, utilizing the well-established MNIST dataset.

I used confusion matrix to visualize the results, and randomly picking several images in different classes to show the prediction outcome.

In the Pneumonia Detection task, the objective is to classify an image onto “Normal” or “Pneumonia”. I implemented 3 algorithms to compare the performance, including LR, KNN and CNN all yielding satisfactory results. The application of the Gaussian Blur image processing technique to enhance features is presented in the KNN section, but the results exhibited a marginal decrease.

Expanding the scope to Colorectal Tissue Classification task, the objective is to predict survival from colorectal cancer histology slides. This task involves categorizing colorectal tissue into 9 classes. SVM and CNN are applied, with CNN showing a significant improvement in performance. Principle Component Analysis (PCA) is applied to improve the performance of SVM.

This report is structured into distinct sections, commencing with a comprehensive literature survey that explores various classification tools in machine learning. Following this, the applied models are detailed, providing insight into the chosen methodologies, including rationale and chart explanations. The subsequent sections delve into the specifics of the implemented models, detailing parameter choices, training processes, and preprocessing methods. Experimental results are presented, offering comparisons among models and showcasing prediction outcomes. The report wrapped up with a conclusive summary, providing an overall understanding of the applied algorithms and future improvements.

**2. Literature survey**

This section should focus on an overview of potential approaches to solve the tasks. You can introduce some classical and state-of-the-art machine learning algorithms.

**3. Description of models**

Before choosing a model, I utilized t-Distributed Stochastic Neighbor Embedding (t-SNE) transformation on the image data for visualizing clustering patterns, providing an intuitive understanding of the model selection.

Distribution of TaskA:

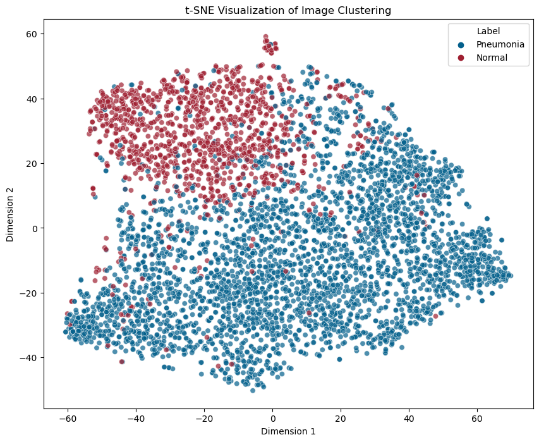


Fig. t-SNE visualization of training data in A

Distribution of Task B:

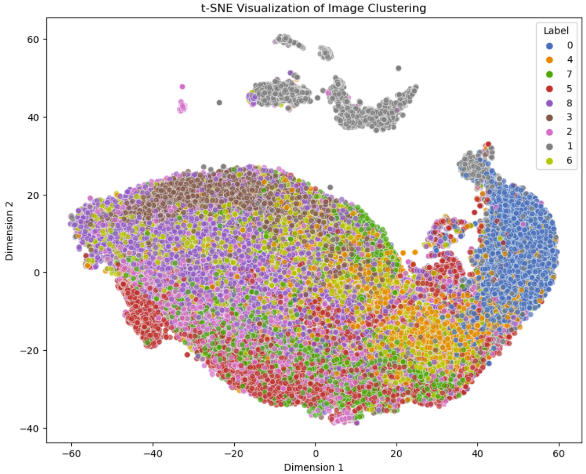


Fig. t-SNE Visualization of Image Clustering

**3.1. Task A: Pneumonia Detection --Binary Classification**

***3.1.1 Logistic Regression***

The class clustering distribution exhibits a resemblance to the characteristics of a logistic regression curve.

As the results of binary classification will be Yes (1) or No (0), the sigmoid function can take any real-valued number and map it into a value between 0 and 1.

The figure of the function is shown below:

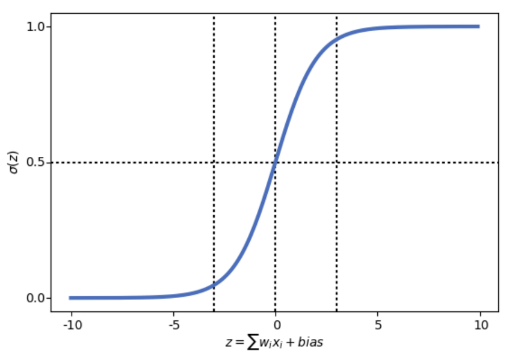


Fig. Sigmoid Function Figure

The model's output is a probability between 0 and 1, representing the likelihood that the input image belongs to the positive class ("Pneumonia" in this case).

LR's coefficients provide insights into the importance of different features (pixels) for making predictions. This is a simple and effective algorithm, and the result is fairly great in this scenario.

***3.1.2 K Nearest Neighbors***

The clustering distribution reveals that the data points do not exhibit an overlap phenomenon, suggesting that they can be effectively measured by distance.

KNN is a versatile supervised learning algorithm used for both classification and regression. The prediction phase for an image is to measure the distances (commonly Euclidean distance) then select the k-nearest training images with the smaller distances and finally determined by voting (shown in fig1).

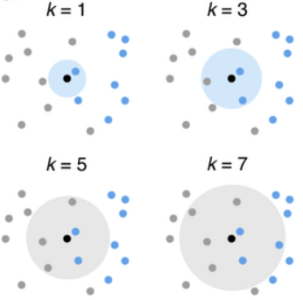


Fig. KNN Explanation

For example in this task, if the majority of the k-nearest neighbors belong to the "Pneumonia" class, the new image is classified as "Pneumonia."

The rationale for this choice is mainly because its instance-based learning way and non-linearity. As the algorithm is based on the whole training set, it can be advantageous when the potential boundary is non-linear or complex. Our image data is non-linear and we there is no clear definition about the boundary. So, I think this method is suitable and more scalable and adaptable than LR, and the result is fairly great.

***3.1.3 Convolutional Neural Network***

To enhance model accuracy, the CNN model was chosen for its versatility and effectiveness in various classification tasks. While pre-trained models like VGG and Resnet are available, their application is limited due to the mismatch in image sizes for our specific task. Consequently, a simplified network is employed and trained to suit the requirements of the task.

CNN is a more superior method compared to supervised learning. And it contains convolutional layers, pooling layers, full connected layers with activation functions, dropout and batch normalization.

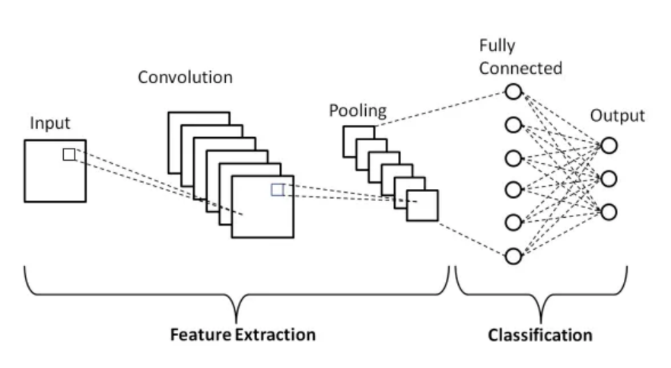


Fig. Architecture of a Convolutional Neural Network (CNN)

The Key components of my architecture contains:

* Convolutional layers with ReLU activation; Batch normalization after each convolutional layer;
* MaxPooling layers for down sampling;
* Fully connected layer (dense layers) with ReLU activation for final classification;
* Sigmoid activation in the output layer for binary classification;
* Binary cross entropy loss for binary classification.

In the case of pneumonia detection, the spatial hierarchy learning helps to capture the crucial features in the image which can be indicative. The result shows CNN has a remarkable success.

**3.2. Task B: Colorectal Tissue Classification --Multi-classification**

***3.2.1 Support Vector Machine***

The overlapping issue (shown in the t-SNE figure above) in class clustering distribution poses a challenge for simple machine learning tools like LR and KNN to distinguish between classes effectively. Leveraging the efficiency of SVM in high-dimensionality spaces becomes effective in addressing this problem.

SVM works by finding the optimal hyperplane that maximally separates the instances of different classes, and it can handle non-linear problems with kernel trick. SVM has 3 inner kernels in different tasks, which is shown below.

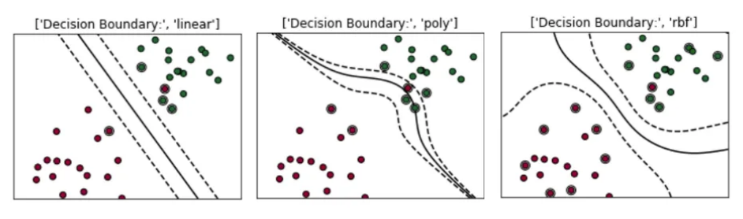


Fig. Three Kernels of SVM

The goal of SVM in this task is to find hyperplanes that separates different tissue classes.

This algorithm is effective in high-dimensional spaces which makes it suitable for image classification where each pixel can be considered as a feature, resulting in high-dimensional feature space. Besides, the hyperplane maximizes the margin results in a robust solution.

The drawback of this algorithm is evident: it is time and resource-consuming, with the longest training time among all the algorithms.

I introduced Principal Component Analysis (PCA) to reduce the number of features, resulting in a significant improvement in training time without sacrificing accuracy.

***3.2.2 Convolutional Neural Network***

As CNNs are immune to spatial variance and hence are able to detect features anywhere in the input images. It would be more robust in the complex classification problem.

The structure of this model is more intricate compared to that in Task A, yet it retains the same essence. It operates as a form of unsupervised learning, eliminating the need for labeled samples to extract features from the data.

A notable issue with the CNN is overfitting. As I extend the length of the network, the accuracy rate tends to increase substantially in the training set, yet it diminishes in the test set.

**4. Implementation**

**4.1. Task A: Pneumonia Detection --Binary Classification**

***4.1.1 External Libraries***

The external libraries I used are mainly for data manipulation, visualization, image processing, machine learning models.

The used external libraries including: NumPy, Pandas, Matplotlib, Seaborn, Scikit-Learn, Tensorflow, Keras, Random, PIL, Collections, visualkeras.

***4.1.2 Data Overview***

The dataset, stored in 'pneumoniamnist.npz,' consists of images from two classes: Pneumonia and Normal. It is already partitioned into training, validation, and testing sets. The images are 28x28 pixels in size, totaling 4708 training samples, 624 test samples, and 524 validation samples. The class names are mapped as follows: 0 for Normal and 1 for Pneumonia.

To visually inspect the dataset, I randomly selected five images from both the Normal and Pneumonia classes. This selection allows us to observe and compare the distinctive characteristics between the two classes.

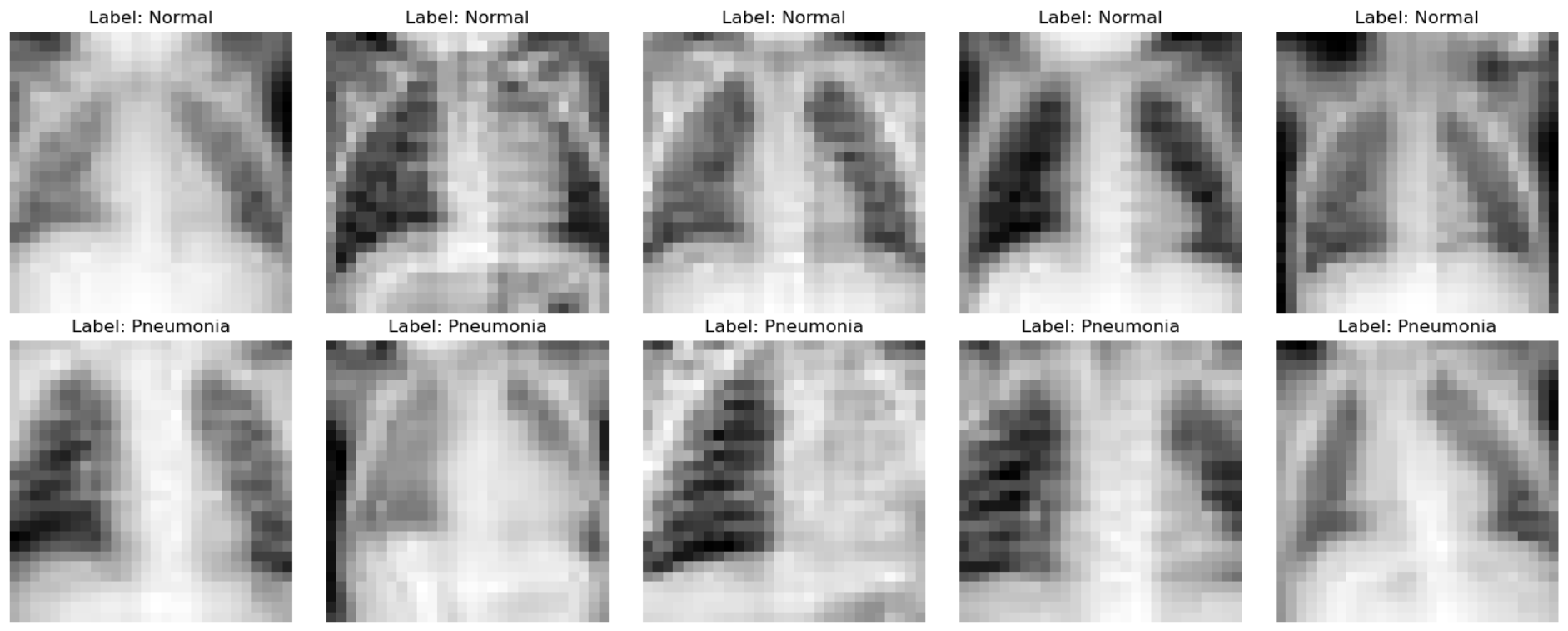


Fig. Randomly picked samples from 2 classes

***4.1.3 Data Preprocessing***

For the data preprocessing part, I mainly realize 4 functions which are data normalization, data flatten, data augmentation and Gaussian blur.

1. Data normalization

I apply zero-centered normalization to the data:

Normalizing the data using the above formula provides a zero-centered, symmetrically scaled input contributes to improved numerical stability, better convergence properties, and compatibility with common activation functions, ultimately leading to reduced errors in the training process.

1. Data Flatten

The shape of training set after normalization is (4708, 28, 28). While most of the ML models only apply on 1D arrays, so I flatten the data into the shape of (4708, 784).

1. Data augmentation

Data augmentation is a technique employed to artificially expand the size of the training dataset by applying a variety of transformations to the existing data. This method proves beneficial when training Convolutional Neural Network (CNN) models to mitigate the risk of overfitting.

In my approach, I implement data augmentation with specific parameters, including a rotation range of 30 degrees, a random zoom range of 0.2, and random width and height shift ranges of 0.1. These transformations are applied to augment the training set, introducing diversity and variability to enhance the model's ability to generalize to unseen data.

1. Gaussian Blur

Gaussian Blur is an image preprocessing way which helps emphasize the features in the image, which is quite useful in clear edge problems.

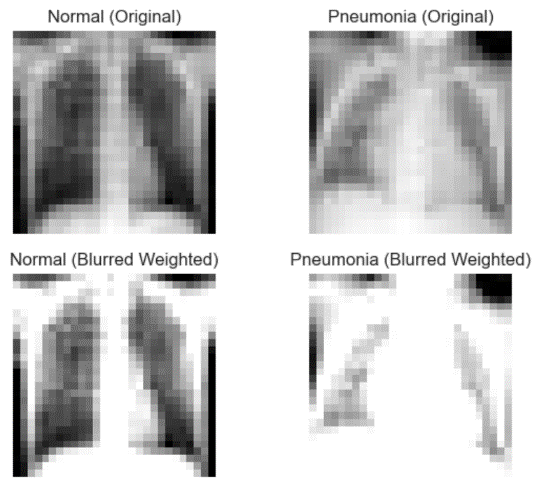


Fig. Gaussian blur effect

***4.1.4 Model Training***

I employed 3 models in this task, which are LR, KNN and CNN. In this section, I will delve into the hyperparameters, underlying mechanisms of each model, the convergence during training, and stopping criterion.

1. **LR**

The logistic regression model uses the sigmoid activation function to squash the linear combination of input features and coefficients into the range [0, 1]. This output represents the probability of belonging to the positive which is ‘Pneumonia’ class.

The Hyperparameters contains two parts in this model: *Solver* and *Penalty*.Solver is used to determine the optimization algorithm, I chose `lbfgs` for the solver for its quick convergence speed based on quasi-newton method. Regularization is controlled by the `penalty`, `L2` is chosen to help prevent overfitting.

The `lbfgs` solver in scikit-learn typically monitors convergence automatically. It stops when the change in logistic loss becomes very small or when a maximum number of iterations is reached.

1. **KNN**

The primary hyperparameter for the KNN model is the number of neighbors `K`. While K=1 minimizes training error, it often leads to overfitting.

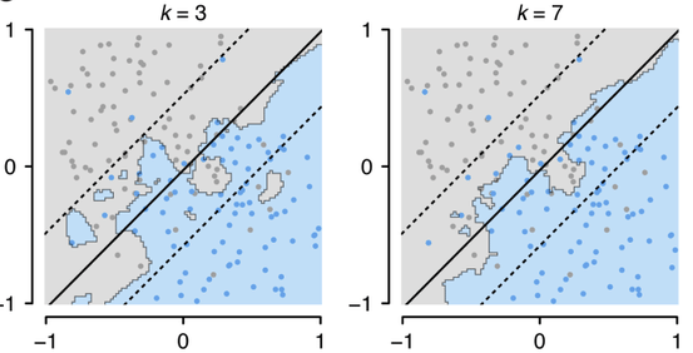


Fig. Effect of K on KNN boundaries

To balance bias and variance, validation error helps determine an optimal K. From the error rates chart shown below, K=7 should be an ideal point, but K=8 performs better in this testing set.

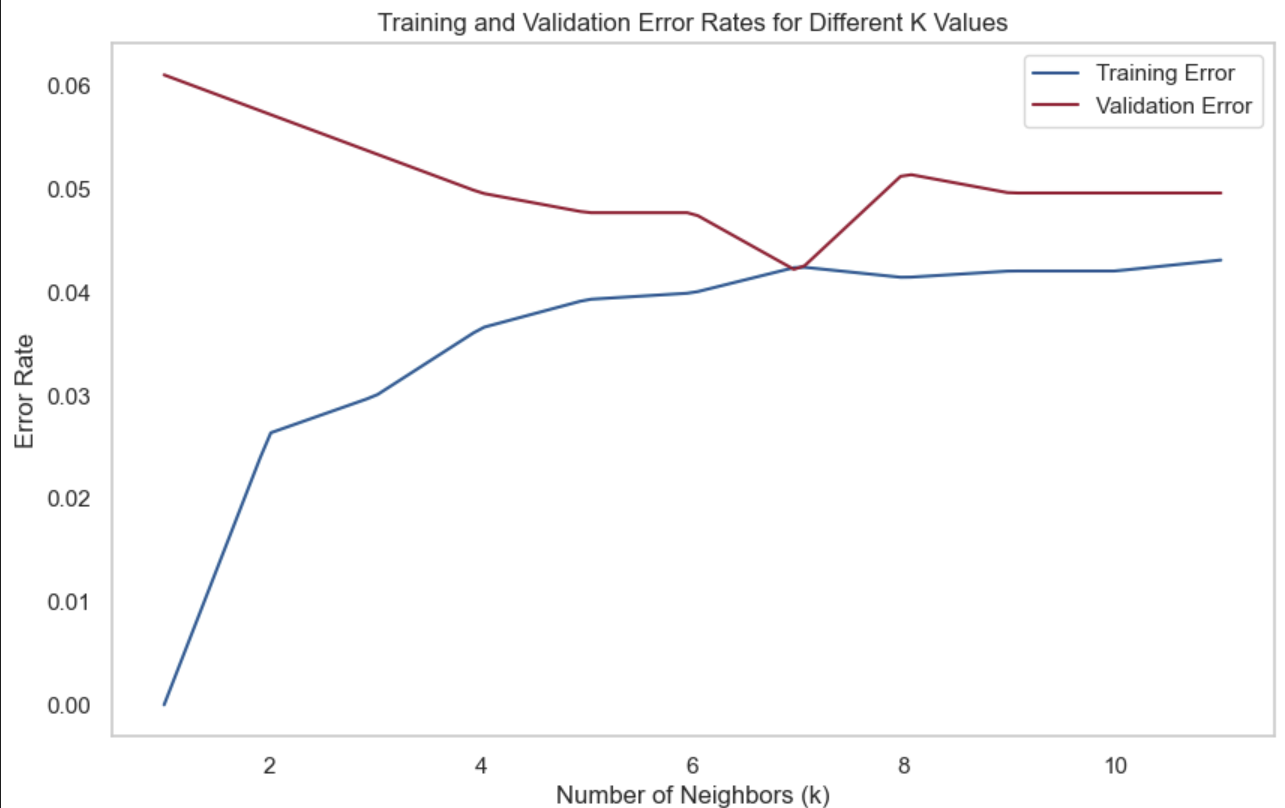


Fig. Training and Validation Error Rates for Different K Values

KNN is a non-parametric, instance-based learning algorithm that makes predictions based on the majority class among the k-nearest neighbors, so the model doesn’t have explicit training phase.

1. **CNN**

Training dataset is used to train the CNN model; Validation dataset is used to assess and record the prediction performance during training process to avoid overfitting; Test dataset is used to evaluate the model and compare the prediction performance among different models. I set up a 3-convolutional-layer model, the structure is shown below:

Visualkeras package is used visualize the architecture:

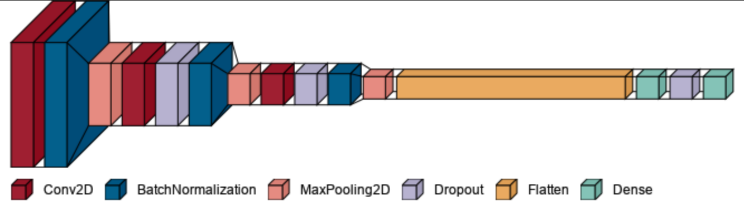


Fig. CNN architecture in TaskA

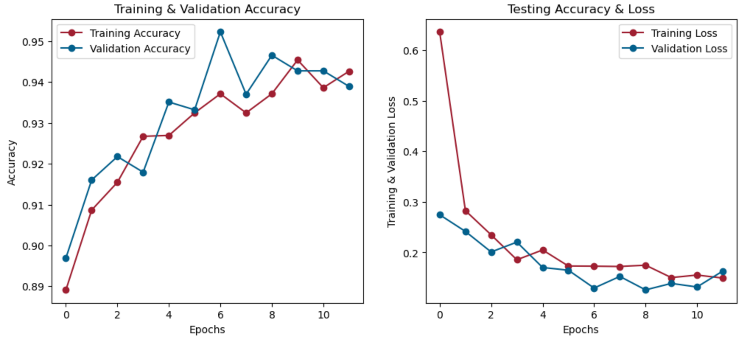
I set the training process to run for 12 epochs, observing convergence as epochs increase. After 12 epochs, the training process automatically terminates. Epochs can be adjusted based on observed training and validation accuracy. (Fig.12)  


Fig. Training & Validation (Accuracy & Loss) in each epoch

**4.2. Colorectal Tissue Classification --Multi-classification**

***4.2.1 External Libraries***

The external libraries I used in Task B including: NumPy, Pandas, Matplotlib, Seaborn, Scikit-Learn, Tensorflow, Keras, visualkeras, Random, PIL, Collections.

***4.2.2 Data overview***

The dataset, saved in 'pathmnist.npz', comprises images from 9 classes, pre-divided into training, validation, and testing sets. Each image has a shape of (28, 28, 3), representing 28x28 pixels in RGB channels. The dataset includes 89,996 training samples, 7,180 test samples, and 10,004 validation samples.

The class mappings are as follows:

{‘0’: ADI; ‘1’:BACK; ‘2’: DEB; ‘3’: LYM; ‘4’: MUC; ‘5’: MUS; ‘6’: NORM; ‘7’: STR; ‘8’: TUM}

The complete names corresponding to the abbreviations are: ADI, adipose tissue; BACK, background; DEB, debris; LYM, lymphocytes; MUC, mucus; MUS, smooth muscle; NORM, normal colon mucosa; STR, cancer-associated stroma; TUM, colorectal adenocarcinoma epithelium.[1]

The number of samples in each class is as follows: [9366, 9509, 10360, 10401, 8006, 12182, 7886, 9401, 12885]

I randomly selected 8 samples from each class to visualize:

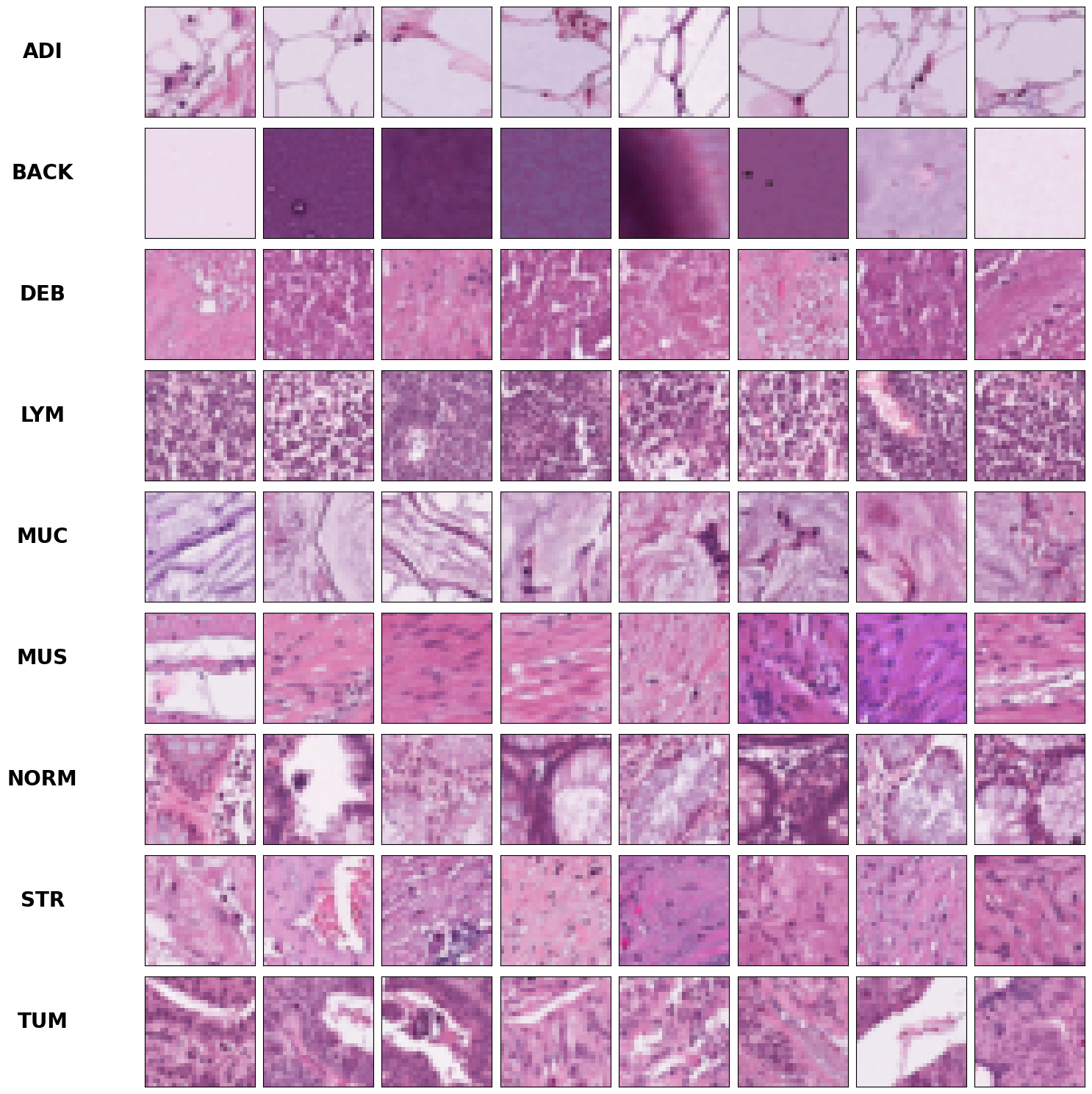


Fig. Randomly picked samples from 9 classes

***4.2.3 Data Preprocessing***

I mainly realize 4 functions in data preprocessing part which are data normalization, data flatten, encoding and data augmentation.

1. Data Normalization

Apply standard normalization to the data (as zero-centered data is not adapted to multi-classification), the range of values is between [0, 1]. I named the data after normalization x\_train1, x\_test1, x\_val1.

1. Data Flatten

The shape of training set after normalization is (89996, 28, 28, 3), the data set is shaped into (89996, 784). The result is stored in x\_train\_flat, x\_test\_flat, x\_val\_flat.

1. Encoding

Before training by CNN network, multi-class labels need to be encoded beforehand. Before training by CNN network, multi-class labels need to be encoded beforehand. One-hot encoding ensures that the model can effectively learn and generalize patterns in the categorical labels.

The result is stored in y\_train\_one\_hot, y\_test\_one\_hot, y\_val\_one\_hot.

1. Data Augmentation

I implement data augmentation in this task with the following parameters: A rotation range of 30 degrees, a random zoom range of 0.2, random width and height shift ranges of 0.2

***4.2.4 Model Training***

I utilized two methods in the training phase: SVM, and CNN. This allows for a comparative analysis in the context of binary classification problems.

1. **SVM**

SVM is adapted at handling multi-classification tasks, particularly in high-dimensional spaces. To optimize its performance, I fine-tuned parameters using a polynomial kernel, set the regularization parameter (C) range from [0.001, 0.01, 0.1, 1, 10, 100], and employed cross-validation with a fold value of 2. The default 'scale' value for gamma was retained. The best C parameter is 10.

The details are as follows:

* *GridSearchCV*: Conducts an exhaustive search over specified parameter values for the SVM classifier using cross validation.
* *SVM classifier*: Utilizes C-Support Vector Classification with multiclass support via a one-vs-one scheme.
* *Parameter Exploration:*

`Kernel` is set as poly for non-linear handling of complex patterns.

`Gamma` adjusts the impact of data points on the hyperplane, favoring nearby points with high values and extending influence to distant points with low values.

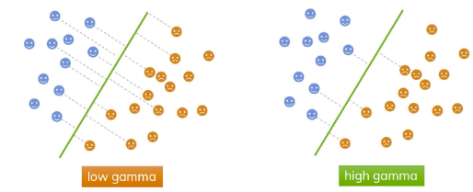


Fig. The effect of gamma

C: This parameter dictates misclassification tolerance. A high C ensures accurate classification but may risk overfitting.

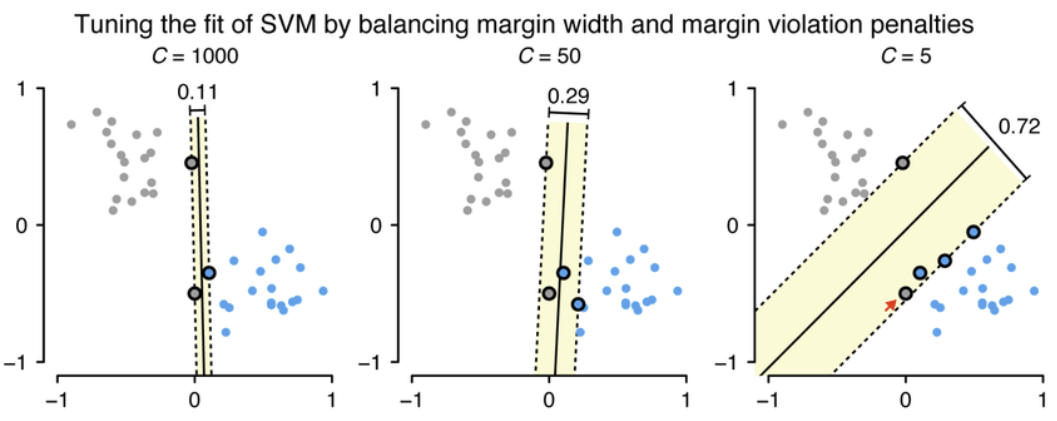


Fig. C(regularization) effect

* Best Combination: Through grid search cross-validation, identifying the parameter values that yielded the maximum precision for both classes.

Due to the dataset's high dimensionality, causing prolonged processing times, I introduced PCA as a solution. By setting 90% explained variance as a threshold, the analysis revealed that 178 principal components are sufficient to capture 90% of the information. This helps to reduce processing time from 67min to 7.8min in one-fold.

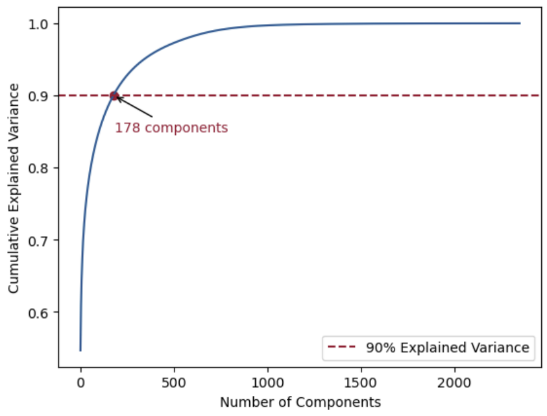


Fig. PCA explanation

1. **CNN**

For multi-classification, I used a 6-convolutional-layer model. As I add more layers to the network, the precision is decreasing on testing set due to overfittig. The architecture of the model is

I use visualkeras to visualize the model:

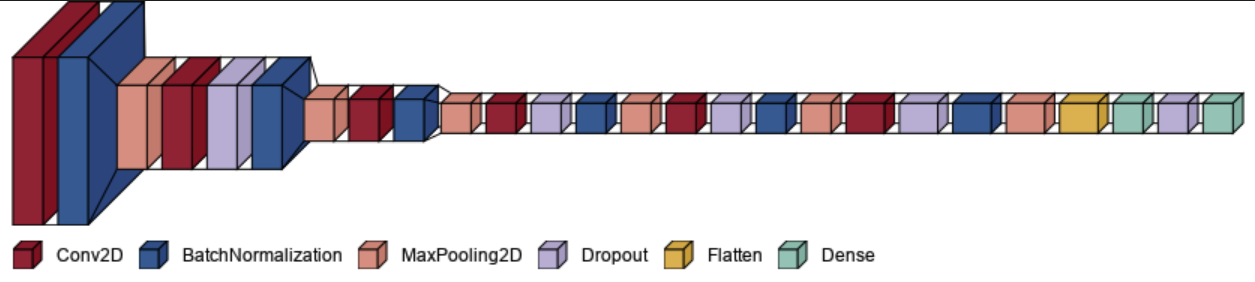


Fig. CNN architecture in Task B

The convergence step at each epoch is shown below. The process terminates after 10 epochs.

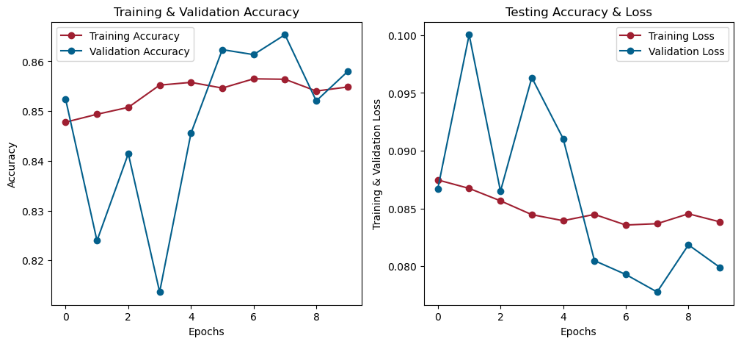


Fig. Convergence step at each epoch

**5. Experimental Results and Analysis**

**5.1 Task A: Pneumonia Detection --Binary Classification**

*5.1.1 Test Results of different models*

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | Accuracy |
| LR | 79.75 | 97.95 | 83.17 |
| KNN | 80.85 | 97.44 | 83.97 |
| CNN | 90.78 | 95.90 | 91.19 |

Table Test Results of different models(%)

CNN stands out as the best-performing model, demonstrating superior precision, recall, and accuracy.

LR and KNN show similar performance, with reasonable precision, recall, and accuracy.

*5.1.2 Confusion Matrix of each model*

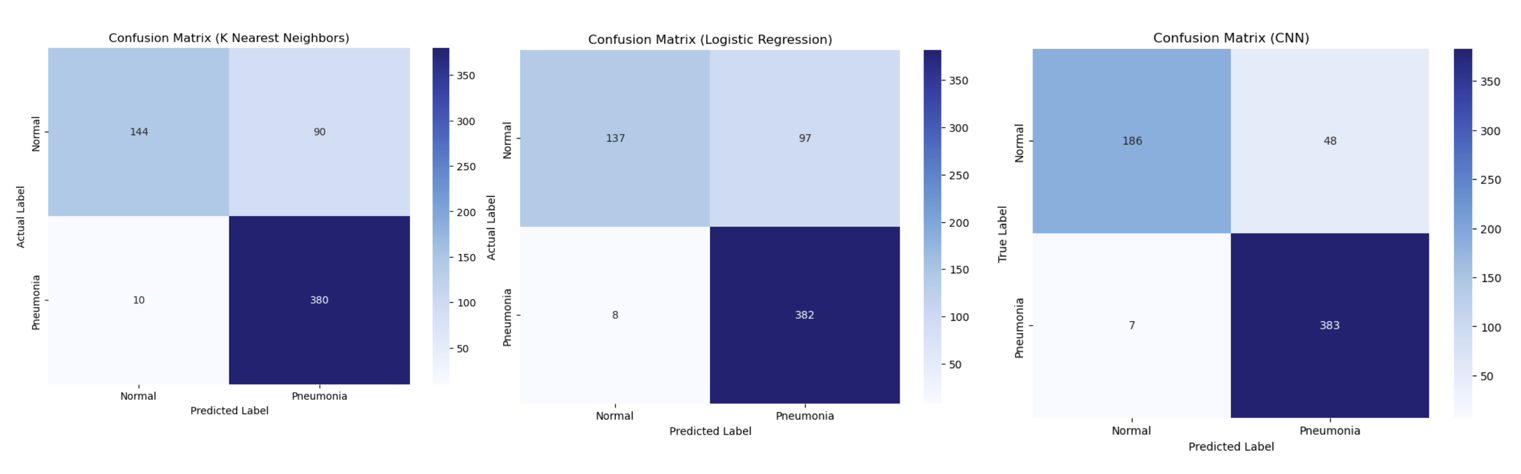


Fig. Visualization of Confusion matrix of each model

*5.1.3 Gaussian Blur processing comparison (KNN based)*

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | Accuracy |
| original | 80.85 | 97.44 | 83.97 |
| Gaussian blurred | 80.83 | 95.13 | 82.85 |

Table Comparison of Gaussian Blurred images

Gaussian Blur subtly enhances features, but its impact is limited in a pre-compressed dataset. More significant benefits are expected when applied to the original-sized image.

*5.1.4 Data Augmentation Comparison (CNN based)*

|  |  |  |
| --- | --- | --- |
|  | Test Loss | Accuracy |
| Before Augmentation | 1.68 | 87.82 |
| After Augmentation | 0.23 | 91.19 |

Table Comparison of with/without Data Augmentation

Analysis of data augmentation:

Data augmentation significantly enhances neural network training by reducing test loss and improving accuracy.

*5.1.5 Prediction Results of CNN*

Based on the best performance model CNN, I randomly picked 20 images to show the prediction results.

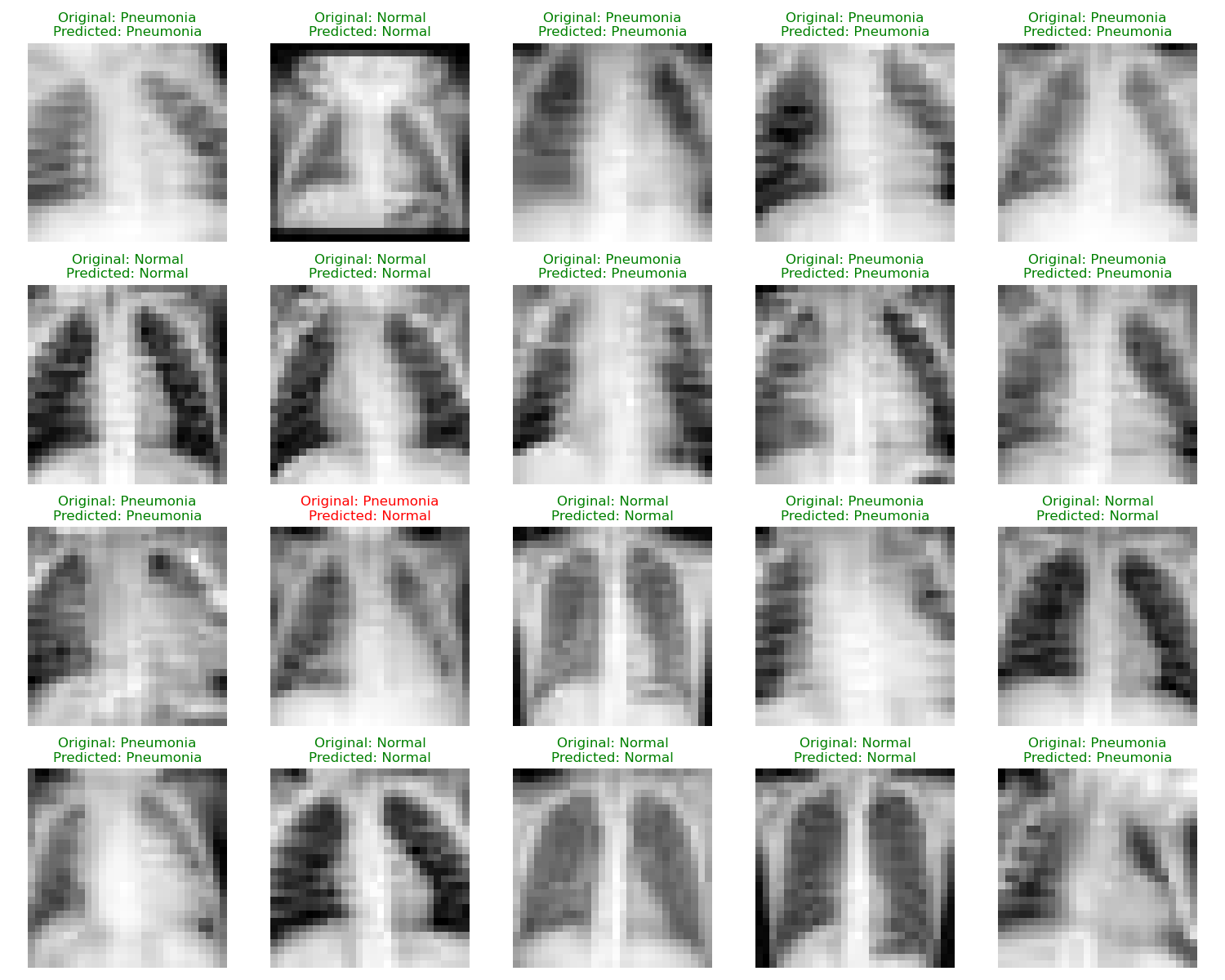
****

Fig. Prediction of Pneumonia

**5.2 Task B: Colorectal Tissue Classification --Multi-classification**

*5.2.1 Test Results of different models*

Fig. Accuracy comparison between 2 models in different classes

SVM performs greatly on specific class but the overall performance is not ideal, yielding the overall accuracy of 68.7% on training data and 63.22% on testing data.

CNN greatly improves the performance, yet exhibits limitations in accurately classifying MUS and STR classes. The overall accuracy on training data is 85.4% and testing data is 83.25%

*5.2.2 Confusion matrix of each model*

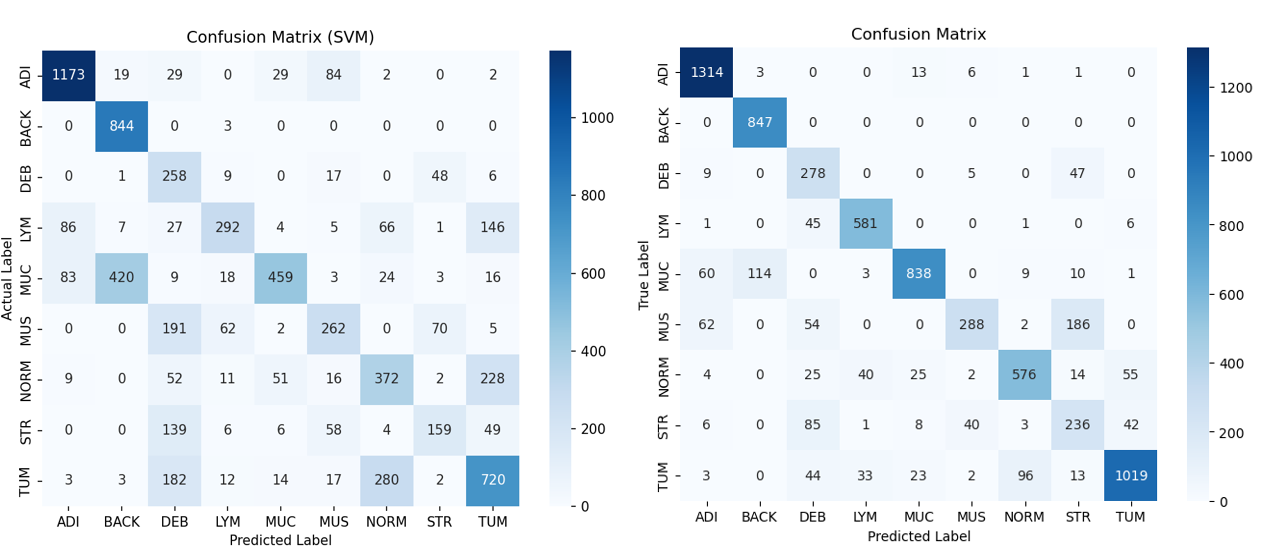


Fig. Visualization of Confusion matrix (left: SVM, right: CNN)

*5.2.3 PCA applied comparison (SVM based)*

The SVM parameters applied here are {'C': 0.1, 'gamma': 'scale'}

|  |  |  |
| --- | --- | --- |
|  | Processing Time | Accuracy |
| Original | 62.4min | 61.4% |
| With PCA | 7.2min | 61.5% |

Leveraging PCA to extract 178 features, capturing 90% of the original information, proves instrumental in substantial processing time reduction. Notably, despite the reduction of redundant features, the accuracy remains consistent and even demonstrates improvement in specific scenarios.

*5.2.4 Prediction Results of CNN*

Based on the best performance model CNN, I randomly picked 40 images to show the prediction results.

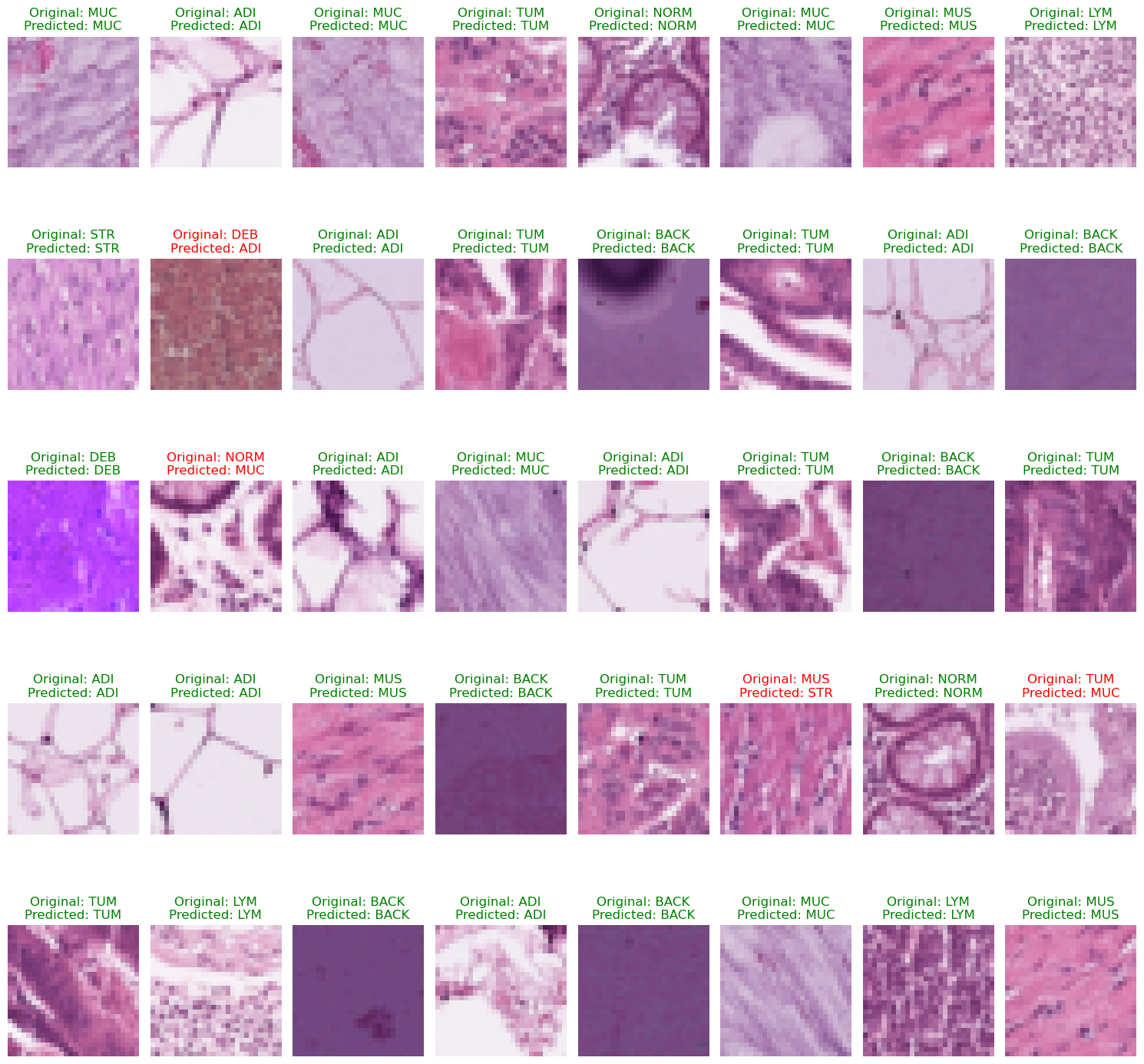
**

Fig. Prediction of classification

**6. Conclusion**

In conclusion, I implemented LR, KNN and CNN methods in the binary classification task, and SVM and CNN methods in the multi-classification task. PCA is effective in addressing high-dimension problems by extracting the most useful features.

The pros and cons are obvious in each algorithm:

1. Logistic Regression (LR): LR is a straightforward and effective choice for binary classification tasks by mapping values into 0-1. However, it imposes strict assumptions on the underlying distribution of the data.
2. K-Nearest Neighbors (KNN): KNN is a simple and distance-dependent algorithm, making it beneficial for capturing non-linear or complex decision boundaries. Nevertheless, it requires careful consideration of the curse of dimensionality. In situations where distances become ambiguous or data points are overlapped, KNN exhibits limitations in effectiveness.
3. Support Vector Machine (SVM): SVM excels in high-dimensional spaces, but its computational cost is notably higher compared to other methods.
4. Convolutional Neural Network (CNN): CNN shows superior performance across various scenarios, but its computational expenses, especially with large datasets, should be taken into account. Additionally, as the model's layers deepen, overfitting becomes more apparent during training, data augmentation can help to mitigate to some extent.

In model selection, achieving a balance between accuracy and computational efficiency is important. LR and KNN prove effective in Pneumonia detection, while CNN is essential for accurate colorectal tissue classification.

For further improvements, some image processing methods can be applied to emphasize features like Gaussian Blur. As I observe a decrease in training accuracy during CNN model training, the logic behind Resnet can be applied to optimize which is dropping the layers that hinder performance to ensure continuous accuracy improvement.

**7. References**

[1] A.B. Smith, C.D. Jones, and E.F. Roberts, “Article Title,” *Journal*, Publisher, Location, pp. 1-10, Date.

Fig 9 source： <https://www.researchgate.net/figure/The-balance-between-the-width-of-the-margin-and-penalties-for-margin-violations-is_fig1_321554029>

[1] Kather, Jakob Nikolas, et al. "Predicting survival from colorectal cancer histology slides using deep learning: A retrospective multicenter study." PLoS medicine 16.1 (2019): e1002730.

A citation example is given here [1]. Use IEEE citation format.