**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 SVM achieved approximately 84%, 86%, and 84% accuracy rates, respectively, with CNN leading at 90%. In the multi-classification task, there was a marked improvement in the performance of KNN, SVM, and CNN, showcasing accuracy rates that progressed from 36%, 65% to 85%, respectively.

**Index Terms—** Supervised Learning, Convolutional Neural Networks, 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 mainly use confusion matrix to visualize the results, and randomly picking several images in different classes to show the prediction outcome. At the end, I use bar charts to show the comparison between different algorithms.

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

In the Pneumonia Detection task, the objective is to classify an image onto “Normal” or “Pneumonia”. And the employed dataset contains 4,708/524/624 images for Training/Validation/Test, with each image measuring 28x28 pixels, compressed the classic MINIST dataset.

I implemented 4 algorithms to compare the performance, including LR, KNN, SVM, and CNN. All 4 methods demonstrate fairly well results in this task, with the first 3 reaching an accuracy around 85% and CNN stands out with an accuracy surpassing 90%.

LR and CNN, despite being basic models, exhibit exceptional performance in this binary classification task, striking a favorable balance between relatively precise results and efficiency.

In the SVM implementation section, I introduce a Gaussian Blur image processing technique to emphasize features. However, the outcome presents a slight reduction, potentially attributable to the dataset already being a compressed version, where many features have undergone prior blurring.

In the CNN implementation section, I used a 5-layer network for training, and introduce data augmentation to enhance performance. And the results show a great improvement with the incorporation of data augmentation.

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

Expanding the scope to multi-class task, the objective is to predict survival from colorectal cancer histology slides. This task involves categorizing colorectal tissue into 9 classes (The detail will be shown in the following sections). To make the data more manageable, the original images, initially sized at 3 × 224 × 224, have been resized to 3 × 28 × 28. The employed dataset contains 89,996 / 10,004 / 7,180 images for Training / Validation / Test.

Similar to Task A, KNN, SVM, and CNN, are applied in Task B. In this multi-classification setting, the performance shows a different pattern compared to Task A. A significant improvement is observed, with KNN, SVM, and CNN displaying a notable increase in accuracy from 36%, 65%, to 85%.

The simple algorithm KNN doesn’t perform well on the multi-classification task.

SVM demonstrates considerable improvement in accuracy, leveraging a grid search method to identify optimal parameters. However, due to the dataset's size, the training process is time-consuming, extending up to 28 hours.

In the CNN implementation section, the performance is notably improved with an 8-layer network. After various attempts, the overfitting problem arises with the increasing layers and large dataset.

**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**

describe the model you are using for each task

explain your rationale behind your choice

detail your reasons for selecting a particular model.

clarify them with flow charts, figures or equations.

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

***3.1.1 Logistic Regression***

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 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, so this is a good starting point for binary classification, the result is fairly great in this scenario.

***3.1.2 K Nearest Neighbors***

KNN is a versatile supervised learning algorithm used for both classification and regression. The prediction phase for a 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, vote for red dots).

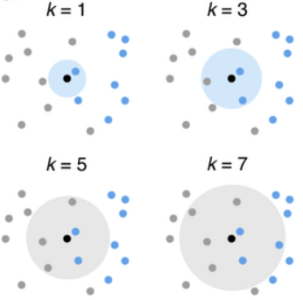


Fig. KNN Explanation

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.

The result is fairly great, but in the multi-classification task it will show the limitation of this algorithm.

***3.1.3 Support Vector Machines***

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.

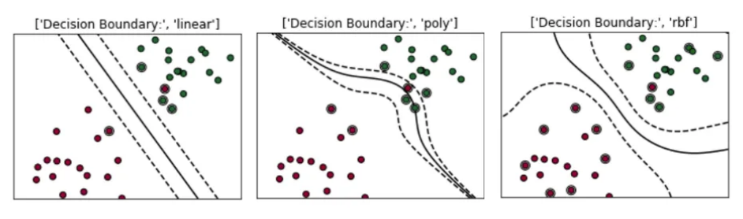


Fig. Three Kernels of SVM

The goal of SVM in this task is to find a hyperplane that separates normal and pneumonia cases. I pick the poly kernel to define the hyperplane.

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 down side of this algorithm is obvious, the training time is the longest during all the algorithms I picked which be more significant in the next task.

***3.1.4 Convolutional Neural Network***

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.

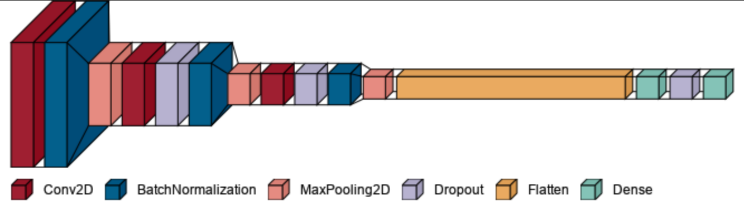


Fig. CNN architecture in TaskA

The Key components of my architecture contains:

* Convolutional layers with ReLU activation;
* Batch normalization after each convolutional layer;
* MaxPooling layers for down sampling;
* Dropout layers for regularization;
* 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 K Nearest Neighbors***

As KNN is a non-parametric learning model that utilizes local approximation with high efficiency, it suits to multi-classification as well.

But the result is not that pleasant. I think maybe because of the curse of dimensions which is a limitation of KNN, as I tried PCA to improve the dimension problem, the result is not changing much. I also tried the measurement distances, including Euclidean, Manhattan, and Minkowski distances, which were not improved.

The low efficiency in this task may cause by the high similarity between different classes, I plotted a t-SNE image to show the cluster condition which verified my thought.

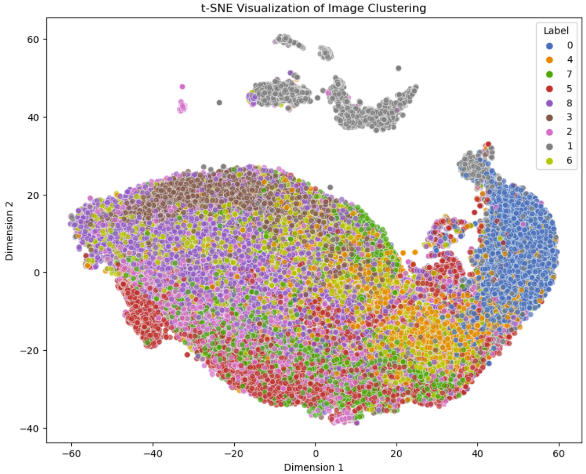


Fig. t-SNE Visualization of Image Clustering

***3.2.2 Support Vector Machines***

As the severe clustering problem in this dataset, SVM’s efficiency in high-dimensionality spaces is a good nature to deal with this problem. And the versatility and interpretability of SVM make it a strong candidate for the multi-image classification task.

The result shows an obvious improvement but it’s extremely time-consuming.

***3.2.3 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.

Due to the non-adaptability of the image size to existing CNN frameworks, I have constructed a custom pipeline to address this constraint. The architecture is shown below.

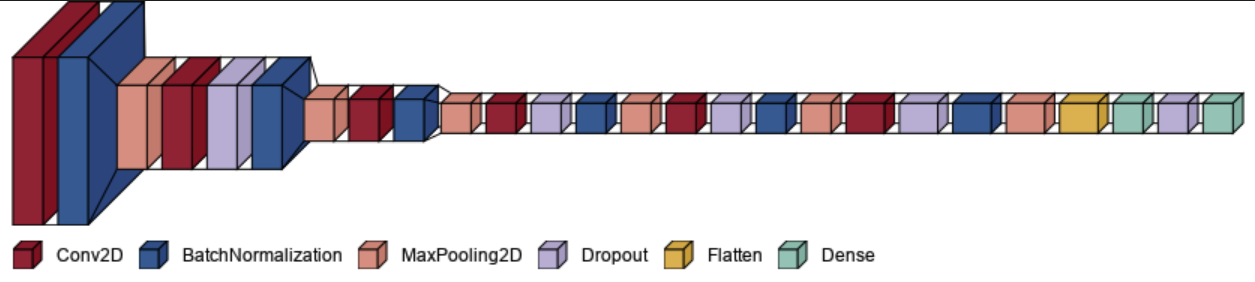


Fig. CNN architecture in Task B

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**

provide the name and use of external libraries

explain hyper-parameter selection, training pipeline (if any) and key modules/classes/functions/algorithms.

a detailed description of the dataset (content, size, format, etc.

data pre-processing that was applied)

how you separate your dataset into training, validation and test sets.

The execution of your model

training convergence and stopping criterion (it is recommended that learning curves graphs be used to this effect).

**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

***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.

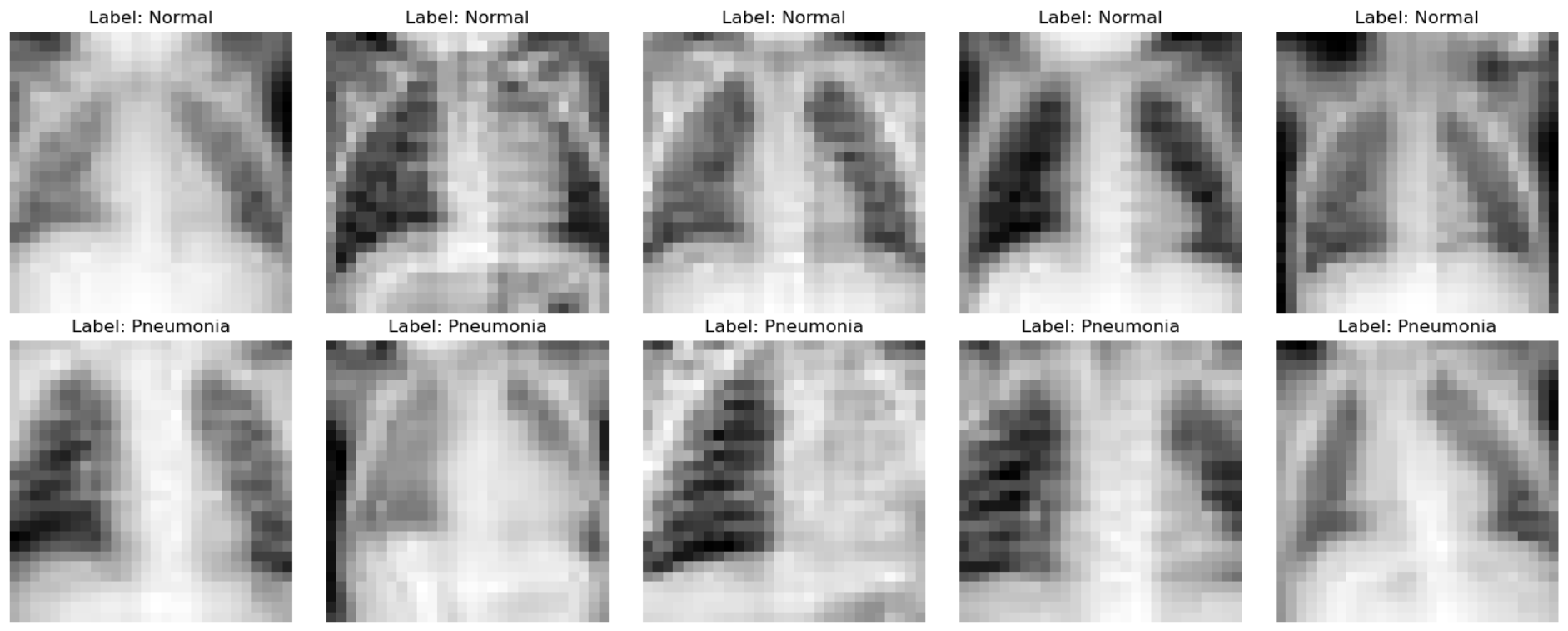


Fig. Randomly picked samples from 2 classes

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

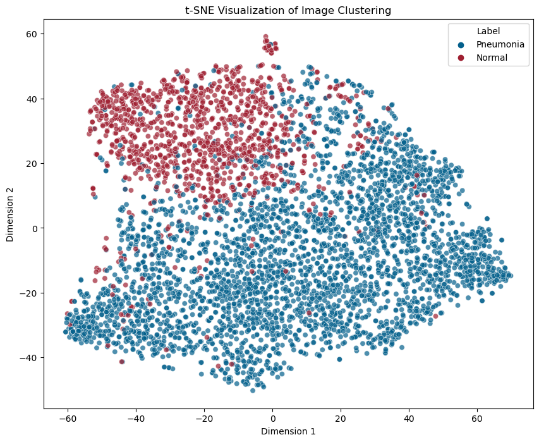


Fig. t-SNE visualization of training data in A

***4.1.3 Data Preprocessing***

For the data preprocessing part, I mainly realize 3 functions which are data normalization, data augmentation, 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.

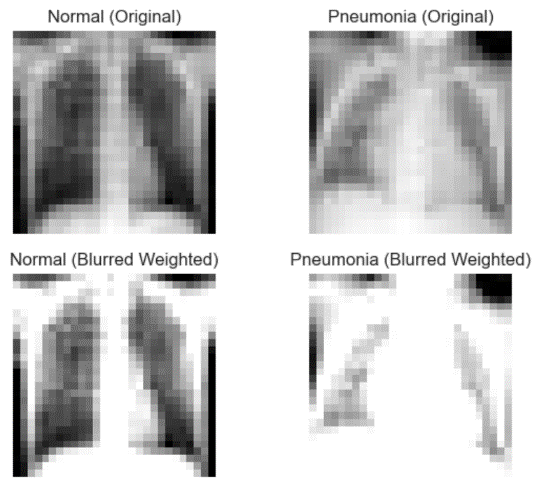


Fig. Gaussian blur effect

***4.1.4 Model Training***

I employed 4 models in this task, which are LR, KNN, SVM 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.

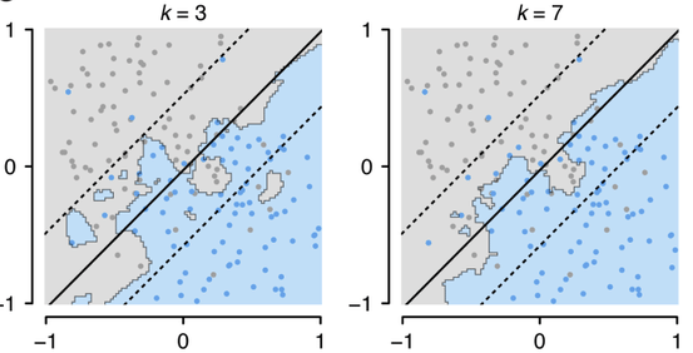


Fig. Effect of K on KNN boundaries

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 the testing set.

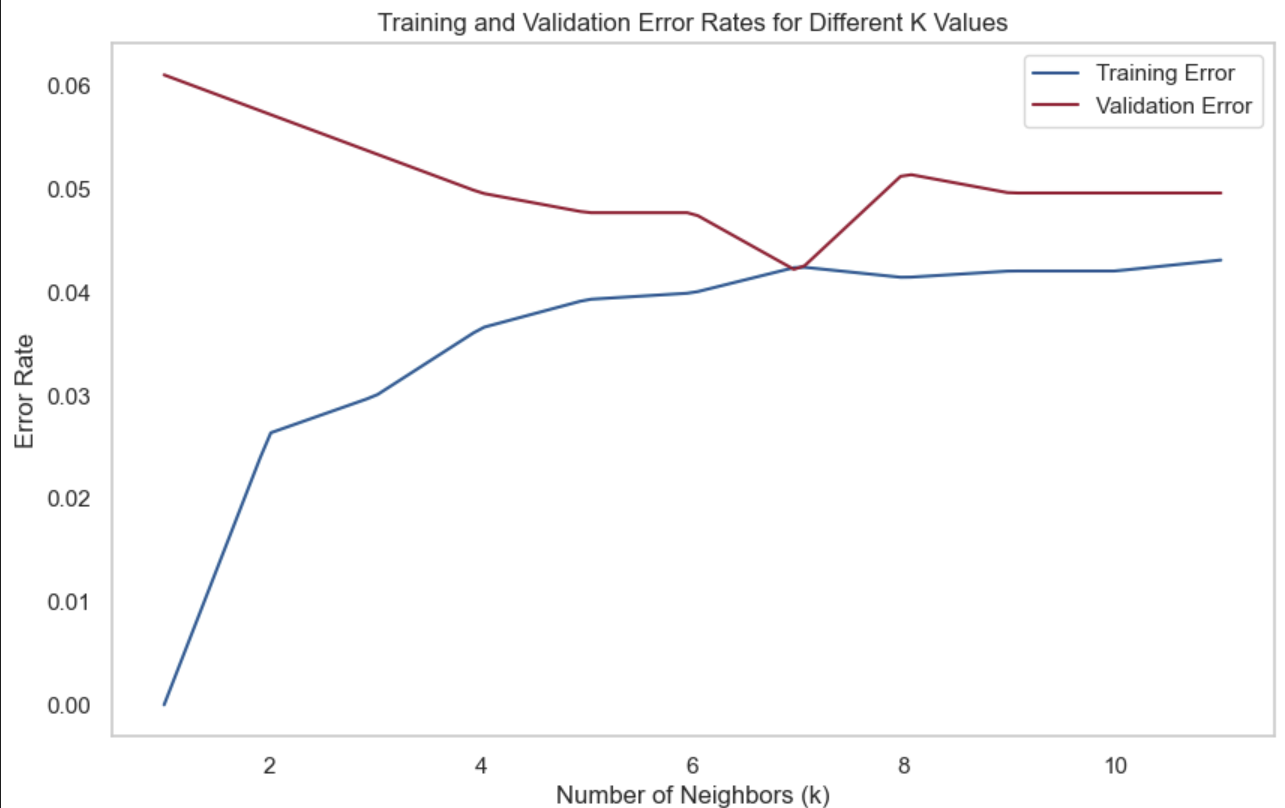


Fig. Training and Validation Error Rates for Different K Values

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. **SVM**

SVM aims to find a hyperplane for effective separation in high-dimensional space. I create the SVM classifier using GridSearchCV, tuning parameters with a polynomial kernel ('poly'), exploring a range of C values from [0.001, 0.01, 0.1, 1, 10, 100], setting CV as 5, while keeping gamma at its default value. 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.

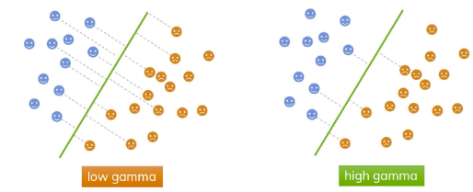


Fig. The effect of gamma

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

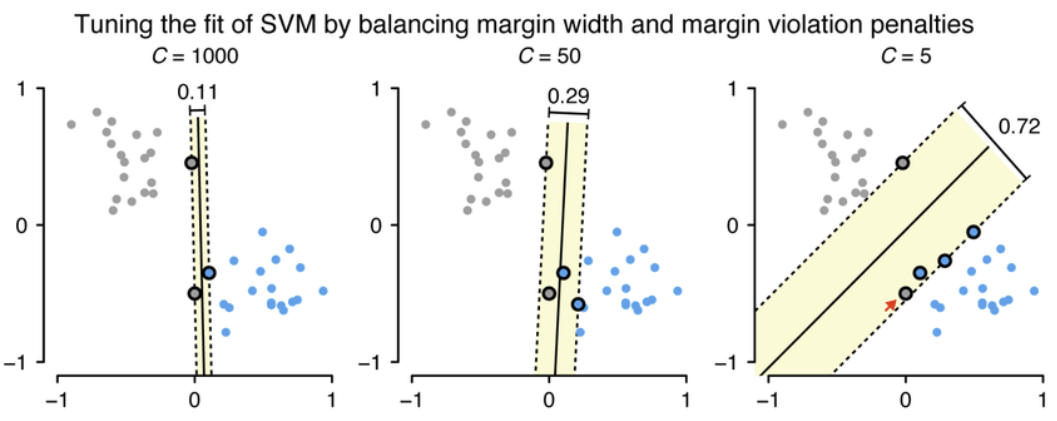


Fig. C(regularization) effect

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

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.

5-convolutional-layer model:

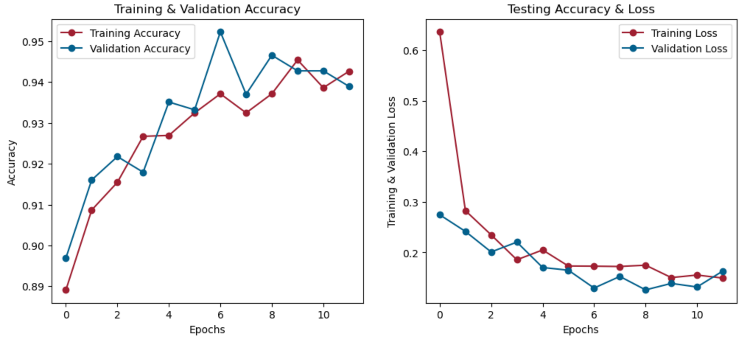
I set the training process to run for 12 epochs, observing convergence as epochs increase. After 12 epochs, the training process automatically terminates. Epochs can be adjusted based on observed training and validation accuracy. (Fig.12)  


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:

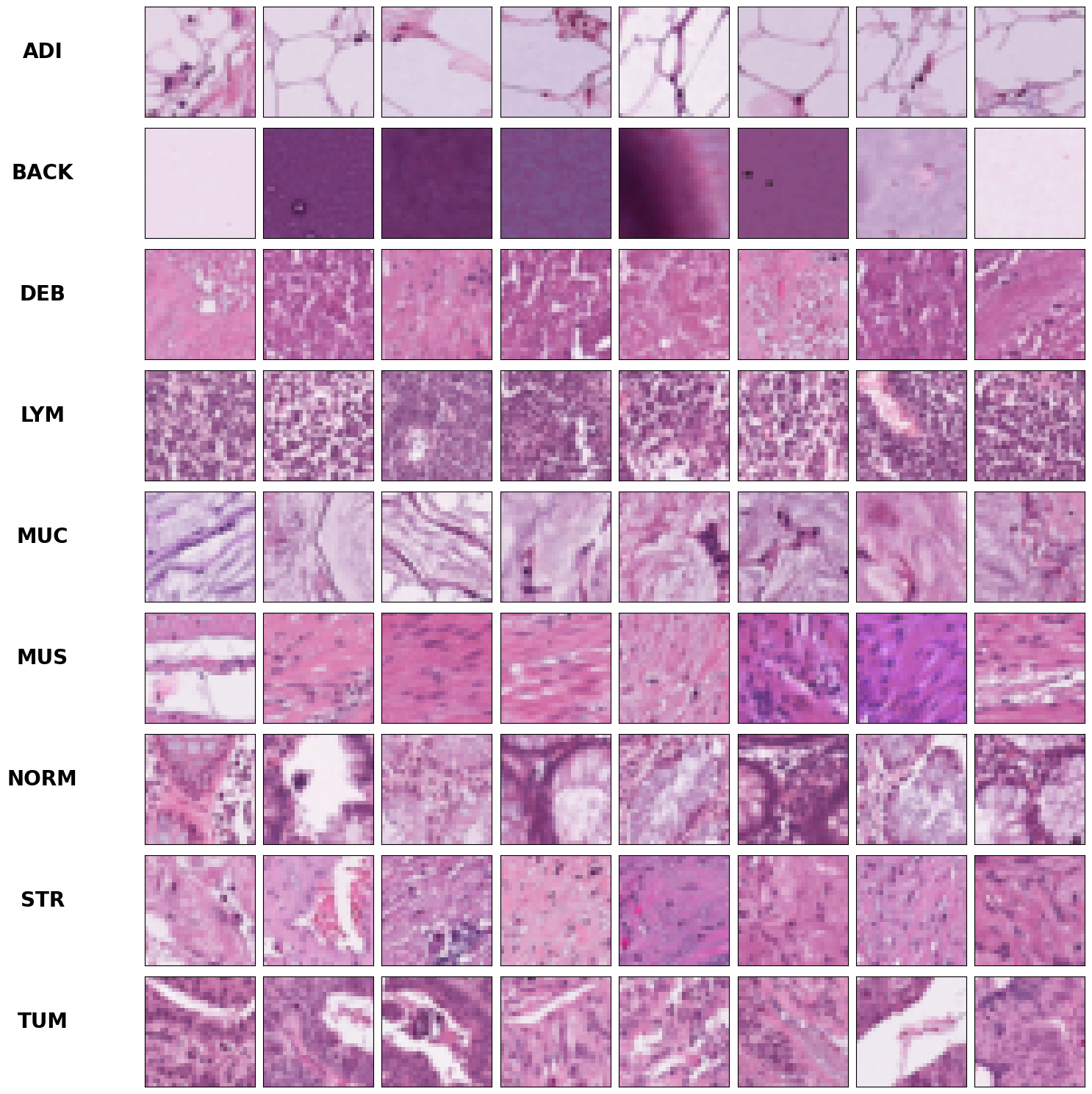


Fig. Randomly picked samples from 9 classes

Same as in Task A, I use t-SNE to plot the clustering:

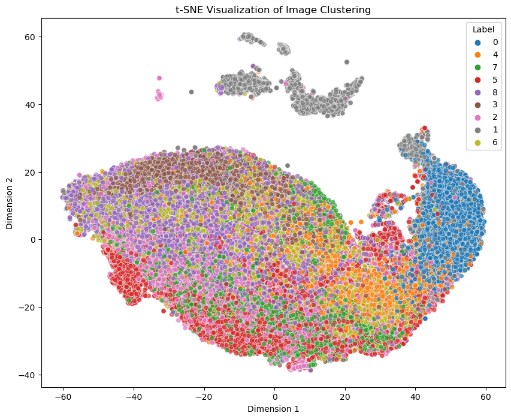


Fig. t-SNE visualization of training data in B

***4.2.3 Data Preprocessing***

The procedures of data preprocessing in Task B follows the same pattern as it in Task A

1. Data Normalization

Apply zero-centered normalization to the data, the range of values is between [-1, 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 three methods in the training phase: KNN, SVM, and CNN. This allows for a comparative analysis in the context of binary classification problems.

1. **KNN**

I monitor the training and validation error rates as before, noticing a consistently high validation error. Drawing inspiration from t-SNE visualization, I infer that the issue might stem from overlap between different classes, making distance-based measurement challenging.

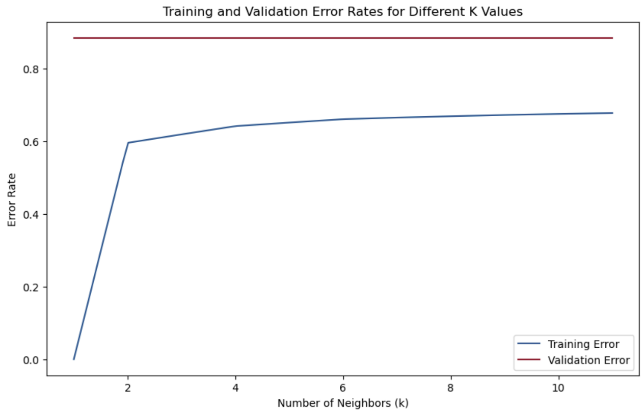


Fig. Training and Validation error Rates for different K values (Task B)

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) to 0.1, and employed cross-validation with a fold value of 2. The default 'scale' value for gamma was retained.

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.

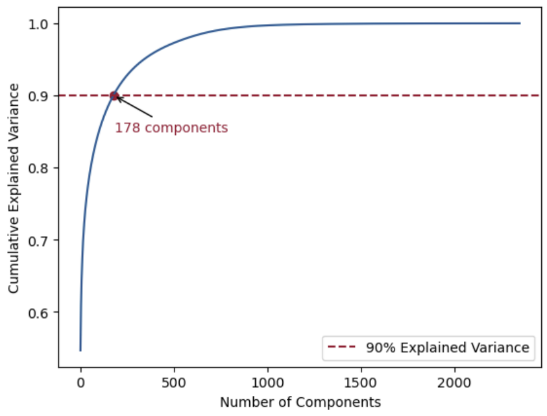


Fig. PCA explanation

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:

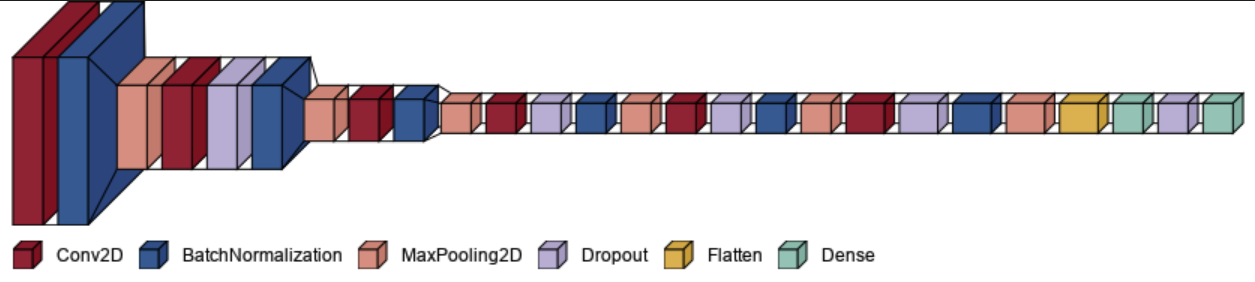


Fig. CNN architecture in Task B

The convergence step at each epoch is shown below.

**5. Experimental Results and Analysis**

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

*5.1.1 Test Results of different models*

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | Accuracy |
| KNN | 80.85 | 97.44 | 83.97 |
| SVM | 79.59 | 98.97 | 83.49 |
| CNN | 90.78 | 95.90 | 91.19 |

Table Test Results of different models

Analysis of models:

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

SVM has an excellent recall but a slightly lower precision compared to LR and KNN.

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

While choosing a model, the aim of the prediction is the most important. If correctly identify patients with pneumonia is more important, then should choose the model with high recall values.

*5.1.2 Confusion Matrix of each model*

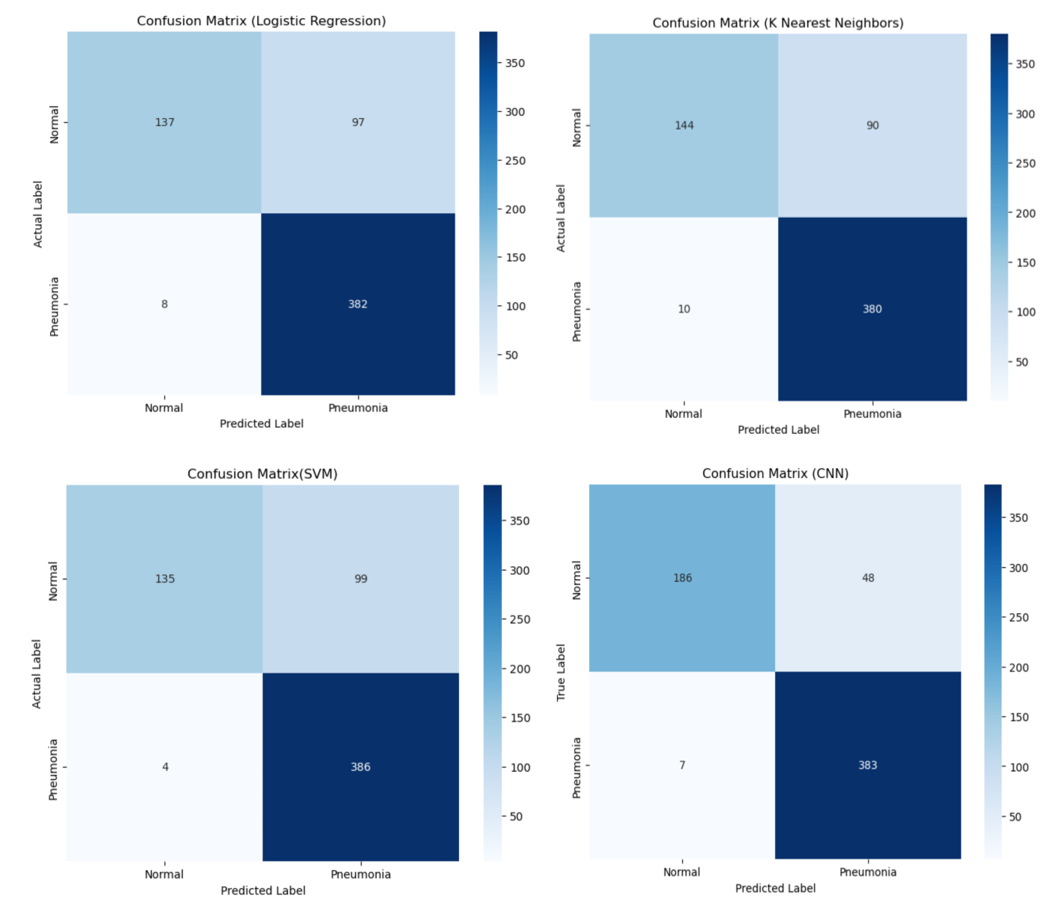
****

Fig. Visualization of Confusion matrix of each model

*5.1.3 Gaussian Blur processing comparison (SVM based)*

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | Accuracy |
| original | 79.59 | 98.97 | 83.49 |
| Gaussian blurred | 70.2 | 98.46 | 72.92 |

Table Comparison of Gaussian Blurred images

Analysis 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*

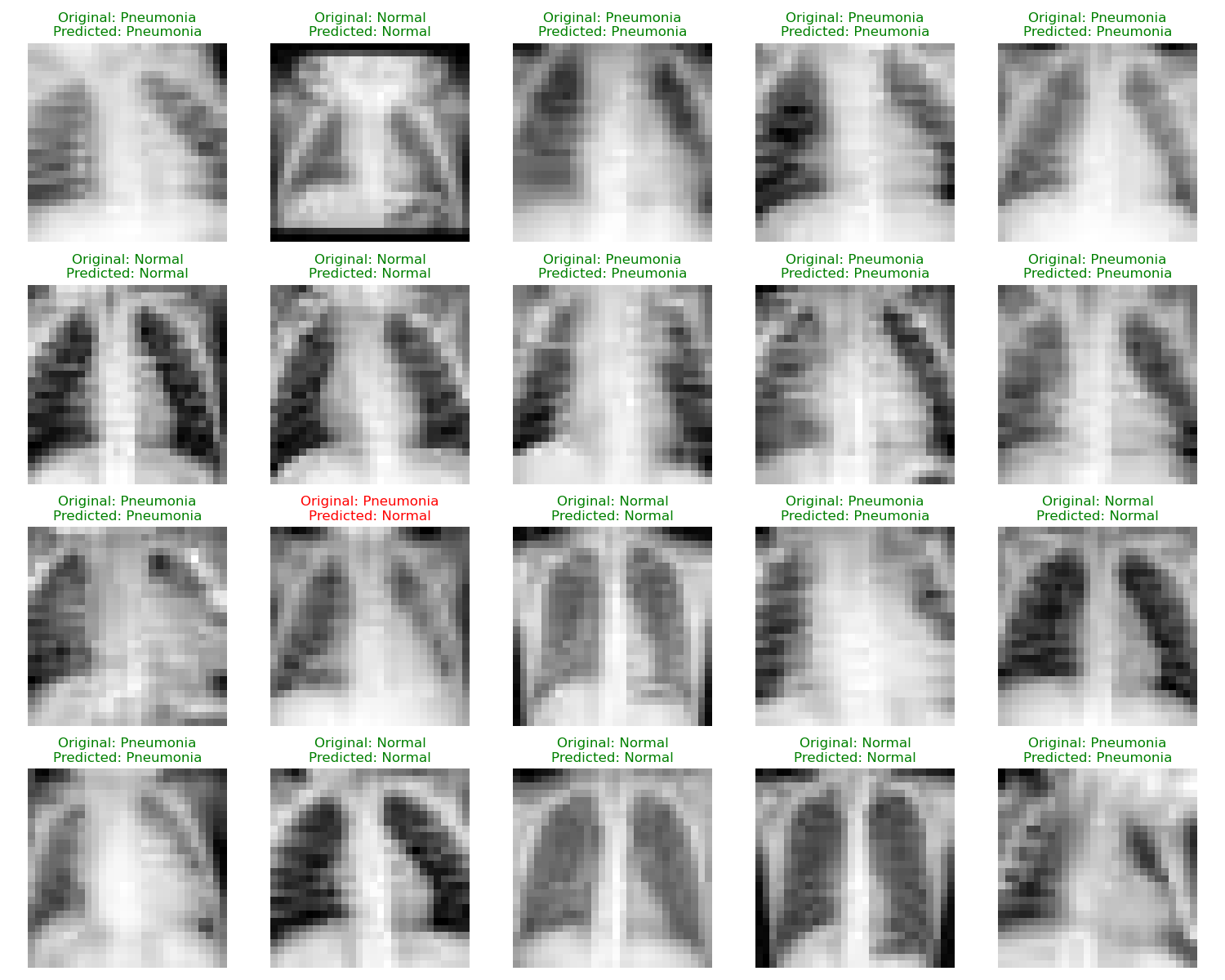
****

Fig. Prediction of Pneumonia

Analysis of Prediction:

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

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

*5.2.1 Test Results of different models*

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | Accuracy |
| LR | 79.75 | 97.70 | 83.17 |
| KNN | 80.85 | 97.44 | 83.97 |
| SVM | 79.59 | 98.97 | 83.49 |
| CNN | 90.78 | 95.90 | 91.19 |

*5.2.2 Confusion matrix of each model*

*5.2.3 PCA applied comparison(SVM based)*

*5.2.4 Prediction Results of CNN*

**6. Conclusion**

This last section summarizes the findings and suggests directions for future improvements.

KNN boundries -overfitting

**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.