Assignment 2 - Group: (98)

Tutors: (Yixuam Zhang, Nicholas Rhodes)

Group members:

Yingbin Mo (SID510202271)

Songling Cheng (SID500540969)

THE UNIVERSITY OF SYDNEY

School of Information Technologies
Faculty of Engineering & IT

ASSIGNMENT/PROJECT COVERSHEET - GROUP ASSESSMENT

Unit of Study: COMP5318 - Machine Learning and Data Mining

Assignment name: Assignment 2

Tutorial time: TUT09 & 15 **Tutor name:** Yixuam Zhang Nicholas Rhodes

DECLARATION
We the undersigned declare that we have read and understood the *University of Sydney Student Plagiarism: Coursework Policy and Procedure*, and except where specifically acknowledged, the work contained in this assignment/project is our own work, and has not been copied from other sources or been previously submitted for award or assessment.

We understand that understand that failure to comply with the *Student Plagiarism: Coursework Policy and Procedure* can lead to severe penalties as outlined under Chapter 8 of the *University of Sydney By-Law 1999* (as amended). These penalties may be imposed in cases where any significant portion of my submitted work has been copied without proper acknowledgement from other sources, including published works, the internet, existing programs, the work of other students, or work previously submitted for other awards or assessments.

We realise that we may be asked to identify those portions of the work contributed by each of us and required to demonstrate our individual knowledge of the relevant material by answering oral questions or by undertaking supplementary work, either written or in the laboratory, in order to arrive at the final assessment mark.

Project team members				
Student name	**Student ID**	**Participated**	**Agree to share**	**Signature**
1. Yingbin MO	510202271	Yes / No	Yes/No	Yingbin Mo
2. Songling Cheng	500540969	Yes / No	Yes / No	Songling Cheng
3.		Yes / No	Yes / No	
4.		Yes / No	Yes / No	
5.		Yes / No	Yes / No	
6.		Yes / No	Yes / No	
7.		Yes / No	Yes / No	
8.		Yes / No	Yes / No	
9.		Yes / No	Yes / No	
10.		Yes / No	Yes / No	

Level 2, SIT Building, J12
The University of Sydney
NSW 2006 Australia

T +61 2 9351 3423
F +61 2 9351 3838
E sit.info@sydney.edu.au
sydney.edu.au/it

ABN 15 211 513 464
CRICOS 00026A

Abstract:

This experiment uses the Chars74K data set. This data set mainly distinguishes numbers 0-9 and uppercase and lowercase letters A-Z. By using this data set, the accuracy of different classifiers on the data set can be obtained. By comparing the accuracy, we can analyze the advantages and disadvantages of different classifiers on this data set. This experiment uses three classification methods: SVM, Random Forest, deep learning CNN, these three methods. Each method uses standardization for preprocessing. It is concluded that the CNN method has the highest accuracy and the fastest running speed. And after this experiment, it can be supposed that the methods used when processing the same data provide experience for future machine learning courses.

Introduction：

1. The aim of the study

This project is based on The Chars74K dataset. When selecting data, we chose EnglishFnt.tgz in The Chars74K dataset. Characters from computer fonts. There are four variants (combination of italic, bold, and normal). For EnglishFnt.tgz in The Chars74K data set, this data set has 60,000 68x68 pictures, and there are 62 types of 0-9, A-Z, a-z. This project uses three classifiers, SVM, random forest, and CNN, to analyze the data. The purpose is to solve the classification problem of the EnglishFnt.tgz data set.

The classification problem has always been the core problem of ML. By analyzing the EnglishFnt.tgz data set to find the best method, we can get the methods and parameters that should be used when processing the same data set.

2. Experiment overview

The main process of this project research is data preprocessing and hyperparameter adjustment. The data is preprocessed by using grayscale, resize and other preprocessing methods, and cross-validation (GridSearchCV) is used to find the hyperparameters with the highest accuracy among all combinations. In order to improve the optimization speed of GridSearchCV, 100 pictures of each category were randomly intercepted in this experiment. In the end, through experiments to find the best method for comparison is CNN.

Previous work：

1. KNN：

We found the KNN method through the experiment of finding the relevant data set on GitHub. The previous work of the KNN classification method is as follows：

```
import numpy as np
import matplotlib.pyplot as plt
import os.path
import string
from skimage import io, color, filters, feature, transform
from sklearn.decomposition import PCA
from sklearn.metrics import accuracy_score
from sklearn.neighbors import KNeighborsClassifier

def imread_convert(f):
        return io.imread(f).astype(np.uint8)

def load_data(data_amnt):
        if (data_amnt > 1016):
            print("data amount invalid")
            return -1
        ic = io.ImageCollection("English/Fnt/Sample0*/img*-00*.png", conserve_memory=True, load_func=imread_convert)
        data = io.concatenate_images(ic)
        data = data[0:62*data_amnt]
        labelNames = string.digits + string.ascii_uppercase + string.ascii_lowercase
        labels = np.empty(data_amnt*62, str)
        for l in range(len(labelNames)):
            labels[l*data_amnt : l*data_amnt + data_amnt] = np.full(data_amnt, labelNames[l])
        shuffled_idx = np.random.permutation(data.shape[0])
        cutoff = int(data.shape[0]*0.8)
        train_X = data[shuffled_idx[:cutoff]]
        test_X = data[shuffled_idx[cutoff:]]
        train_Y = labels[shuffled_idx[:cutoff]]
        test_Y = labels[shuffled_idx[cutoff:]]
        return train_X, train_Y, test_X, test_Y

def preprocess(imgs):
        l = []
        for img in imgs:
            img = color.rgb2gray(img)
            img = transform.resize(img, (32, 32), anti_aliasing=True, mode='reflect') #anti_aliasing automatically blurs before resize
            # fd, img = feature.hog(img, orientations=10, pixels_per_cell=(5, 5), cells_per_block=(2, 2), visualize=True, block_norm='L2-Hy
            img = np.array(img).flatten()
            l.append(img)
        return np.array(l)
```

Figure 1. Chars74KDataAnalysis(Ewrobel, 2019)

```
def knn_error(train_X, train_Y, test_X, test_Y, n_pca=49, n_neighbors=4):
    train_X = preprocess(train_X)
    test_X = preprocess(test_X)

    # pca = PCA(n_components=n_pca)
    # pca_model = pca.fit(train_X)
    # pca_train = pca_model.transform(train_X)
    # pca_test = pca_model.transform(test_X)

    knn_model = KNeighborsClassifier(n_neighbors=n_neighbors)
    # knn_model.fit(pca_train, train_Y)
    knn_model.fit(train_X, train_Y)
    # train_prediction = np.array(knn_model.predict(pca_train))
    train_prediction = np.array(knn_model.predict(train_X))
    # test_prediction = np.array(knn_model.predict(pca_test))
    test_prediction = np.array(knn_model.predict(test_X))
    train_accuracy = accuracy_score(train_Y, train_prediction)
    test_accuracy = accuracy_score(test_Y, test_prediction)
    return train_accuracy, test_accuracy


def main():
    train_X, train_Y, test_X, test_Y = load_data(500)
    train_accuracy = []
    test_accuracy = []
    max_k = 10
    for k in range(max_k):
        tr_acc, tst_acc = knn_error(train_X, train_Y, test_X, test_Y, n_neighbors=k+1)
        print(tr_acc, tst_acc)
        train_accuracy.append(tr_acc)
        test_accuracy.append(tst_acc)
    plt.title("KNN Accuracy", fontsize=16, fontweight='bold')
    plt.xlabel("K value")
    plt.ylabel("Accuracy")
    xaxis = np.linspace(1, max_k, max_k)
    plt.xticks(np.arange(0, max_k+1))
    plt.plot(xaxis, train_accuracy, c='b')
    plt.plot(xaxis, test_accuracy, c='r')
    plt.show()
```

Figure 2. Chars74KDataAnalysis(Ewrobel, 2019)

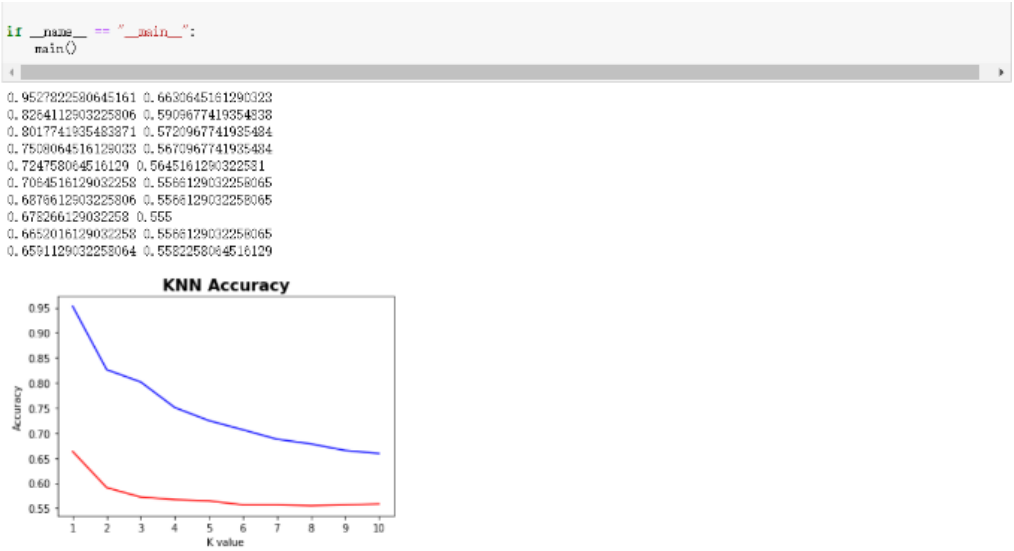


Figure 3. Chars74KDataAnalysis(Ewrobel, 2019)

The previous work used the KNN method. In terms of preprocessing, the previous model work performed resize to change the size of the image to 32x32. And in terms of confirming the hyperparameters, the value of k is set to 1-10 to classify the Chars74K data set. From the previous work model, we can see that the accuracy of KNN's train ranges from 95% to 65%, and the accuracy of the test ranges from 66% to 55.5%. The larger the value of k, the lower the accuracy.

Compared with the previous work model in terms of preprocessing, this experiment uses the resize method and uses a grayscale image, and loads the EnglishFnt data as a grayscale image to reduce the processing time. Moreover, standardized the data to improve accuracy.

Compared with the previous work model, the random forest, SVM, CNN and KNN models selected in this experiment are simple, easy to understand, but have lower accuracy. This is because in the hyperparameter attributes, some categories are highly correlated, and some are weakly correlated. If the similarity of samples is calculated based on the same weight of all attributes, this will reduce the accuracy. Similarly, when the sample is unbalanced, the prediction bias is relatively large.

2. CNN

Complex classification problems are limited by the system's capacity and cannot reach a high level. However, the emergence of deep learning and neural network (CNN) provides a theoretical basis for us to solve this problem. CNN is a part of deep learning, mainly clustering images through similarity of images and showing surprising

enhancement in field font recognition (Kadir, Hidayah & NorasiahMohammad, 2019). The algorithm has also been applied to many daily tasks, such as automatic recognition of car license plates (Arroyo-Pérez et al, 2020). The model they created bases on the Chars74k database (use 33849 images) and character images extracted from license plates (3549 images), and performs 98% accuracy ultimately 2.

Layer (type)	Filters	Size/stride	Output shape
Conv2d_1 (conv2d)	64	7x7/1	(74 ; 34 ; 64)
Max_pooling2d_1 (maxpooling2)		2x2/1	(37 ; 17 ; 64)
Conv2d_2 (conv2d)	128	3x3/1	(35 ; 15 ; 128)
Max_pooling2d_2 (maxpooling2)		2x2/1	(17 ; 7 ; 128)
Conv2d_3 (conv2d)	256	3x3/1	(15 ; 5 ; 256)
Max_pooling2d_3 (maxpooling2)		2x2/1	(7 ; 2 ; 256)
Flatten_2(flatten)			(3584)
Dense_1 (dense)			(1024)
Dropout_1 (dropout)			(1024)
Activation_4 (activation)			(1024)
Dropout_2 (dropout)			(1024)
Dense_2 (dense)			(36)
Activation_5 (softmax)			(36)

Figure 4. Previous research CNN structure(Arroyo-Pérez, Álvarez-Canchila, Patiño-Saucedo, González & Patiño-Vanegas, 2020)

First, analyze the structure of the convolutional neural network they created. It consists of 3 convolution layers followed by a Relu activation function and a max-pooling layer. Then, it is followed by flattening, dense, dropout layers ending with a softmax activation layer with 36 outputs (26 letters and 10 numbers). Specifically, an 80*40 pixels image is applied to this model will generate 64 feature maps of 74*34 pixels via a 7*7 filter in Conv2d_1 layer. Afterward, these feature maps are applied to 2*2 max-pooling filter to resize as 37*17 feature maps. Subsequently, the same steps were performed twice (the filter size and max-pooling hyperparameters are different). Lastly, this model extract 256 feature maps of 7*2 from an 80*40 pixels image. After extracting features stage, the classification stages are combined by flattening, dropout, and dense layer separately ends with the softmax function. This function can assign probabilities to each image to the respective classification.

Compared to that CNN model, my CNN model has only two convolution layers. Because the images in Fnt dataset are relatively similar, too many convolutional layers will lengthen the operation time. In addition, I add a normalized layer between each convolution layer and max-pooling layer. This layer is mainly designed to Improve training speed. Moreover, there are only flattened and dense layers without dropout layers, since batch normalization also plays a role in preventing overfitting.

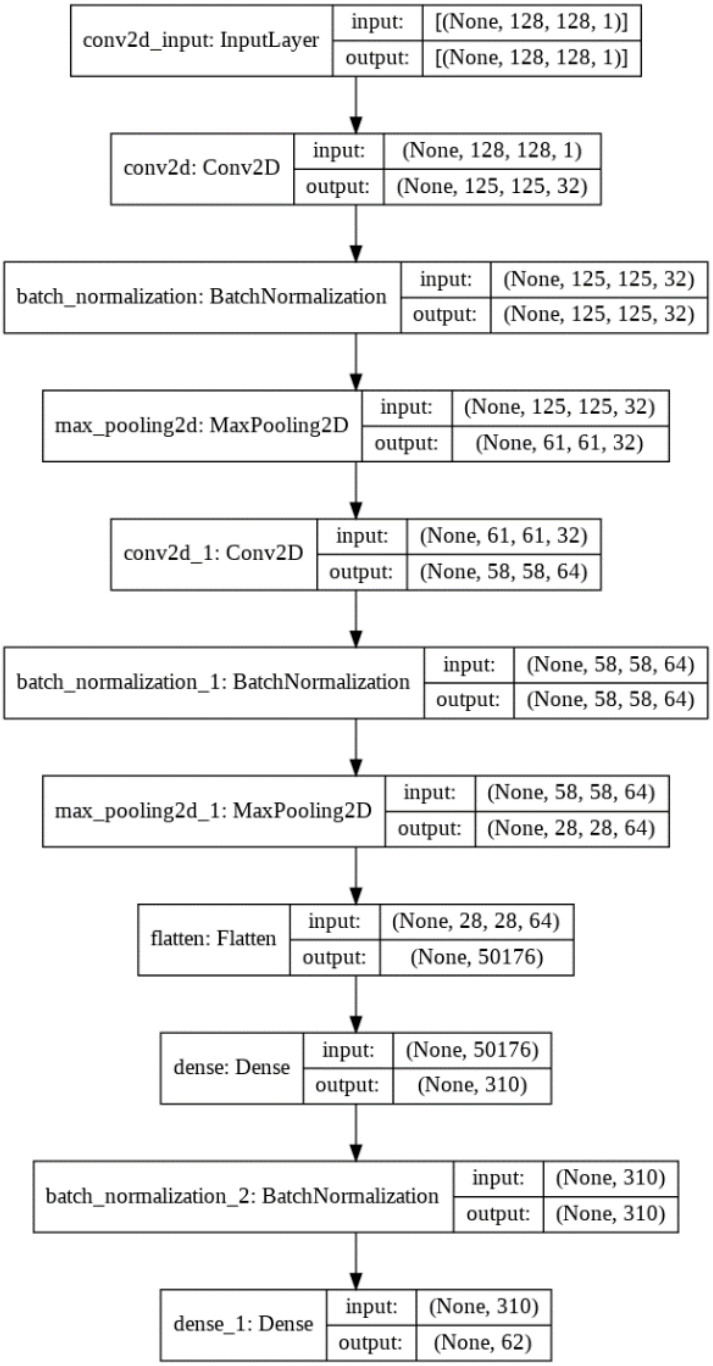


Figure 5. CNN structure

Methods

1. Data pre-processing

First, this experiment loads EnglishFnt data as a grayscale image. The color value of each pixel on the grayscale image is also called grayscale, which refers to the color depth of the point in the black and white image. The range is generally from 0 to 255, white is 255, and black is 0. The so-called gray value refers to the intensity of the color,

and the gray histogram refers to the number of pixels with the gray value corresponding to each gray value in a digital image. Image grayscale processing can be used as a preprocessing step of image processing to prepare for subsequent operations such as image segmentation, image recognition, and image analysis.

By using the grayscale image for preprocessing, the time for processing data in various methods can be greatly reduced. This preprocessing method provides a more concise image for optimizing the classification model. Therefore, this experiment uses grayscale images as the first preprocessing method.

After loading as a grayscale image, this experiment performed resize preprocessing. The purpose of this is also to reduce the processing time of the classification method. After researching, we found that when the pixel is 50x50 after resizing, the running time of the method will be reduced, and the accuracy will not change. Therefore, in this experiment, the pixel size after resize is set to 50x50, and the INTER_CUBIC method is used for interpolation (3 times interpolation based on 4x4 pixel neighborhood).

PCA (Principal Component Analysis) is a standard data analysis method, often used for dimensionality reduction of high-dimensional data. It can be used to extract the main feature components of the data. Although PCA will reduce the dimensionality of the data, it may cause pixel loss, but it can greatly reduce the running time of the classification method. Compared with the data that is only standardized, the accuracy rate is not greatly reduced after the PCA is performed, but the running time is greatly reduced. After combining the two situations, this experiment decided to use PCA to preprocess the data.
By drawing a graph explaining the relationship between variance and dimensionality, this experiment shows that n_components=0.9.

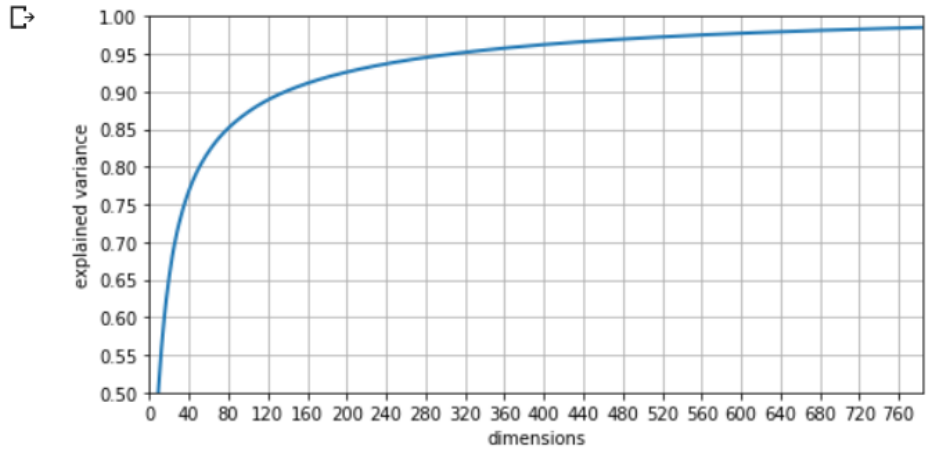


Figure 6. Diagram of relationship between variance and dimension

Image Data Generator
When training the model, generally, all training data will be loaded into the memory. However, this method is no longer available when memory is limited, and the amount

of data is too large. In order to solve this problem, one of the essential pre-processing techniques is to create an ImageDataGenerator, and then call the flow_from_directory function, which can select the batch sizes extracted images from the directory to trained model in each epoch rather than loading them all. For example, I set 128 images in each step. This greatly reduces the memory burden. Moreover, ImageDataGenerator has another practical function- data augmentation. This means the batch_size sample data can be enhanced in each batch to expand the size of the data set and enhance the model's generalization ability, such as rotation, transformation, normalization, and so on. Apart from the training model, ImageDataGenerator also can be used to fit and evaluate the model. All of the mentioned have been used in CNN code.

2. Classifier methods algorithms

SVM:

Support vector machine is a two-classification model. Its basic model is a linear classifier with the most significant interval defined in the feature space. SVM is also a non-linear classifier. The learning method of SVM is to maximize the gap, and the learning algorithm of SVM is the optimal algorithm for solving convex quadratic programming.

Support vector machine (SVM) finds the best separation hyperplane in the feature space to maximize the interval between positive and negative samples on the training set. Use this hyperplane as the classification interval.

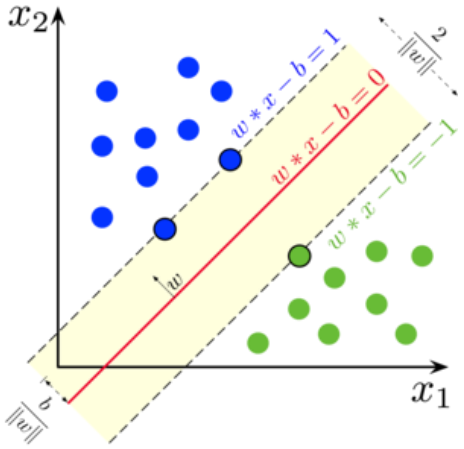


Figure 7. Analysis and Derivation of Support Vector Machine (SVM) (Fing, 2020)

Support vector machine (SVM) finds the best separation hyperplane in the feature space to maximize the interval between positive and negative samples on the training set. Use this hyperplane as the classification interval.

In this figure, the blue circle and the green circle ask for two different data types, respectively, and the dividing line in the middle of the two parallel dashed lines is the

optimal decision surface. The vertical distance between the two dashed lines is the classification interval corresponding to this optimal decision surface. There is an optimal decision surface in every direction that may divide the data set correctly, and the classification interval of the optimal decision surface in different directions is usually different. The decision surface with the "maximum interval" is the most critical decision surface SVM is looking for. Excellent solution. The sample points crossed by the dotted lines on both sides of the proper optimal solution are the support sample points in the SVM, which are called "support vectors."

Random Forest:

Forest is an algorithm that integrates multiple trees through the idea of ensemble learning. Its basic unit is a decision tree, and its essence belongs to the ensemble learning method in machine learning. In a random forest, each tree is a classifier. For an input sample, N trees will have N classification results. The random forest integrates all the classification voting results, and the category with the most votes is designated as the final output, which is Random Forest.

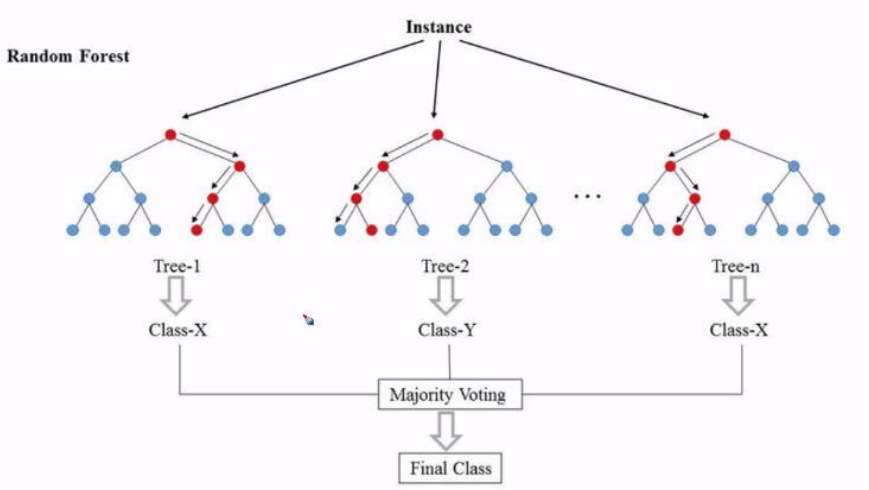


Figure 8 Overview Random Forest. (2018)

N trees and N classification results are on the way, and these N results are voted on. Finally, the voting result is used as the final classification result of the instance. Advantages of the random forest:

• Has a very high accuracy rate.

• The introduction of randomness makes the random forest not easy to over fitting

• The introduction of randomness makes the random forest have a good anti-noise ability.

Because of these advantages, this experiment chose to use Random Forest as one of the classification methods to analyze the EnglishFnt data set in The Chars74K.

CNN:

[1] Convolutional Layer

The convolution layer is composed of multiple feature maps, and each feature maps is composed of various neurons, each of which is connected with the local region of the feature plane of the next layer through the convolution kernel. The convolution kernel is a weight matrix (Yan et al, 2017). The pink cuboid in the figure below (left) is a convolution layer, and a rectangle inside represents the feature graph. The front side of the feature graph can be seen in the figure below (right). A feature graph contains multiple neurons (represented by white circles in figure 9).

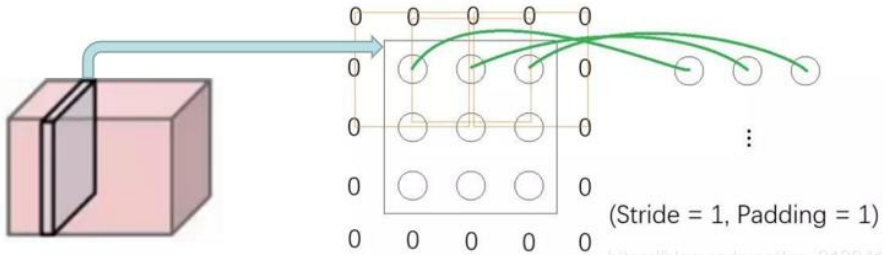


Figure 9. Convolutional Layer

Features are extracted by translation of the convolution kernel on the original image. It's like looking at an image, scanning it up and down to extract the information you want.

Therefore, I create two convolutional layers to extract features. Specifically, different convolutional layers have different sizes and numbers of filters. In general, the size of the filter is set to 1*1, 3*3, 5*5, or 7,7, while the latter convolution layer has twice as many filters as the previous one.

[2] Batch Normalization Layer

In a deep neural network, basically, all network parameters will be updated after each iteration, so in the training process, the input distribution of each hidden layer always changes, and with the deepening of the network layer, the input distribution of the deeper hidden layer network will be more different. This is the so-called "Internal Covariate Shift" phenomenon. One of the influences of this phenomenon is that we dare not set a large learning rate when training, which makes it difficult for us to practice, and the convergence is getting slower and slower, and the gradient is easy to disappear or explode.

In recent years, the batch normalization method has been introduced. It can affect normal distribution with a mean of 0 and a variance of 1. This method aims to alleviate the gradient explosion phenomenon in the deep neural network and speed up the training speed of model (Ioffe, Szegedy, 2015). The other advantage is to enhance the

classification effect. One explanation is that this is a regularization expression similar to Dropout to prevent overfitting so that the effect can be achieved without Dropout layer.

In order to prevent batch normalization from overcorrecting, scale hyperparameter and shift hyperparameter are introduced to fine adjust distribution. Therefore, batch normalization also simplifies the hyperparameter tuning process, resulting in the model is less sensitive to the weight of initialization parameters, and allowing to apply a large learning rate.

[3] Max Pooling Layer

The pooling layer is usually used to retain the main features while reducing the parameters and computation of the next layer. Generally speaking, there are maximum pooling and average pooling. The figure below shows 2*2 maximum pooling, which means that the 2*2 area in the feature graph is compressed into a value, so as to retain the main features and reduce the parameters of the next layer. If you use average pooling, you take the average of the region, as in the yellow area above. If you use average pooling, the output is 2, not 3.

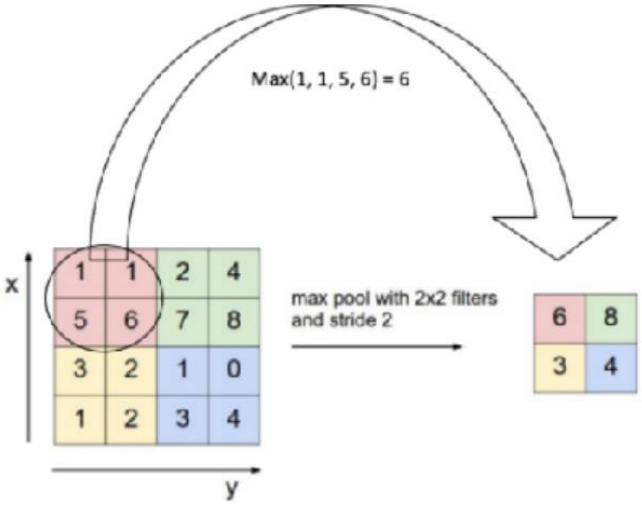


Figure 10. Zero basic understanding of NN and CNN.(2018)

In addition to the convolutional layer, convolutional networks also often use a pooling layer to reduce the size of the model, improve the computing speed, and improve the robustness of extracted features (Albawi, Mohammed, Al-Zawi, 2017).. the hyperparameters of pooling include filter size f and step s. The commonly used parameter values are f=2, s=2. One thing to note is that there are no parameters to learn during the pooling process since pooling can be seen as a static property of a layer in a computational neural network.

Experiments and Discussion

Table 1. Model running time and accuracy

Model	Time	Accyracy
SVM	2min28s	79%
RandomForest	4min 37s	73%
CNN	11min	91.1%

1. SVM

By comparing the advantages and disadvantages of SVM, the experiment found that using the SVM algorithm in the case of small data sets can better analyze the data. This is because when the problem is complex and there are many samples, support vectors may be possible. When there are not many samples, the model can distinguish well. The EnglishFnt data set of The Chars74K is 60,000, which also meets this characteristic, so we choose SVM as one of the methods to analyze this data set.

The next project needs to find the most accurate SVC hyperparameters, so this project carried out GridSearchCV (100 pictures per category) to find the optimal parameters, and used a 5fold method for random crossover, and finally determined that it is accurate when C=3 of. The highest degree is 79%, and the operating time is 2min28s. At the same time, through experiments, we also found that without using PCA, the calculation time of the model is several times that of using PCA, and the accuracy rate does not change much. Therefore, from the perspective of optimization, this experiment uses the PCA method.

From the confusion matrix diagram, we can clearly see the various errors of the SVM classifier.

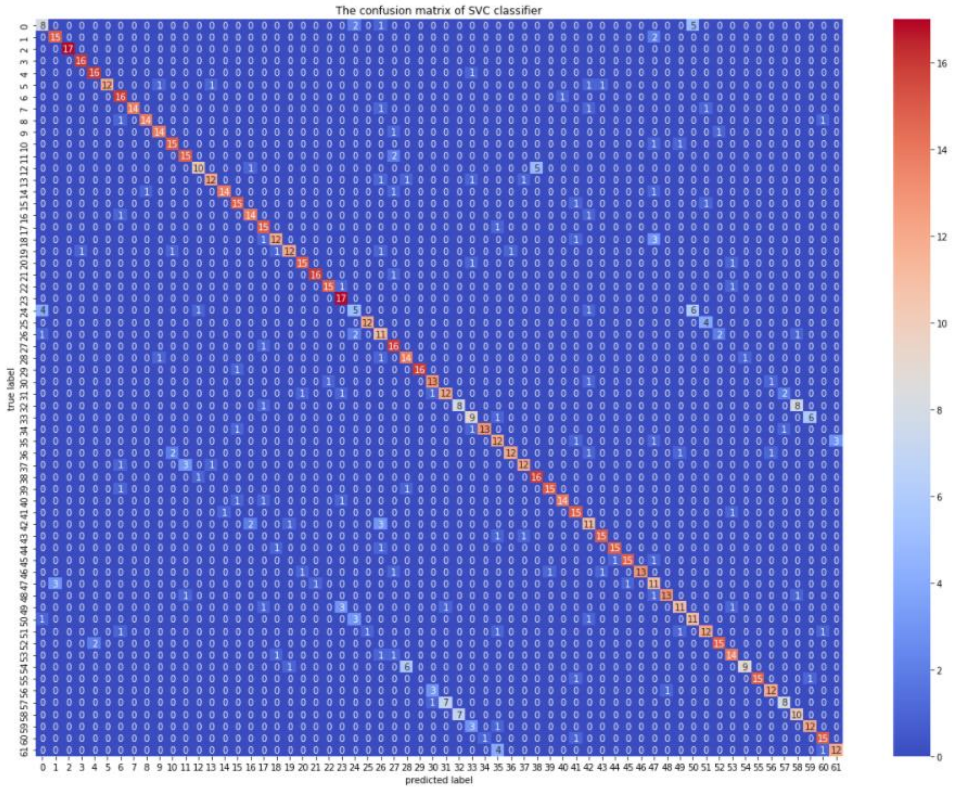


Figure 11 Confusion matrix of SVM

There are bright spots from the 28 rows, 33 rows, and 35 rows of the confusion matrix. From the classification label, we can see that the reasons for these dots are uppercase S and lowercases, uppercase X, lowercase x, uppercase Z, and lowercase z. It can be seen that the SVM classifier has many errors in distinguishing similar uppercase and lowercase letters. Not only that but there is also an error in the 0th line. This error is to classify 0 as an uppercase O. It can be seen that this classifier also has errors when distinguishing some dissimilar letters and numbers. It is these errors that cause the accuracy of the SVM to be 79%.

By comparing other experimental models, this experiment concludes. The SVM model is relatively general in accuracy and time, and it is different from CNN in terms of computing time (11min), but it is acceptable. Compared with RandomForest (4min 37s), there is little difference. In terms of accuracy, SVM still has a good performance (79%) compared with CNN (91.1%), which is far behind, and compared with RandomForest's 73%. It can be seen that these three methods differ greatly in accuracy. Although there is a gap in time, the accuracy rate is quite different from that of CNN (91.1%). Therefore, SVM is not a very good method compared with CNN.

The accuracy of SVM is 79% because SVM is sensitive to missing data and sensitive to the selection of parameters and kernel functions. Therefore, the performance of the support vector machine mainly depends on the choice of the kernel function, so for a practical problem, how to choose the appropriate kernel function according to the actual

data model to construct the SVM algorithm. At present, the choice of kernel function and its parameters are. It is artificial, selected based on experience, and has a certain degree of arbitrariness. The kernel function should have various forms and parameters in different problem domains, so domain knowledge should be introduced when selecting, but there is no suitable method yet. To solve the problem of choosing the kernel function. Therefore, the parameter with the highest accuracy may not be selected when selecting the parameters, which leads to errors.

To improve the SVM classifier, the focus should still be placed on the selection of parameters. Only the most accurate and accurate parameters can be selected to achieve the best accuracy. In the confusion matrix, we have also found out the types of errors (letters with similar upper and lower case). This experiment should perform better preprocessing on these categories to improve accuracy.

2.Random Forest:

For random forest, because it cannot control the internal operation of the model, it can only try between different parameters and random seeds. Moreover, for small data or low-dimensional data (data with fewer features), it may not produce a good classification. (Processing high-dimensional data, processing missing feature data, and processing unbalanced data are the strengths of random forests).

The EnglishFnt data set in The Chars74K consists of 60,000 numbers and letters written by computers, and the characteristics are not very obvious. In addition, 60,000 samples are not a particular data set, so Random Forest is more suitable for processing this data than other methods.

Random forest also has these many hyperparameters such as gini", "entropy, etc., so this experiment also performed GridSearchCV on Randomforest to find the optimal parameters and also used a 5fold method for random crossover, and finally determined when criterion =' entropy' The highest accuracy rate is 73%, and RandomForest is 4min 37s in terms of running time.

From the confusion matrix diagram, we can see the various errors produced by the Random Forest classifier.

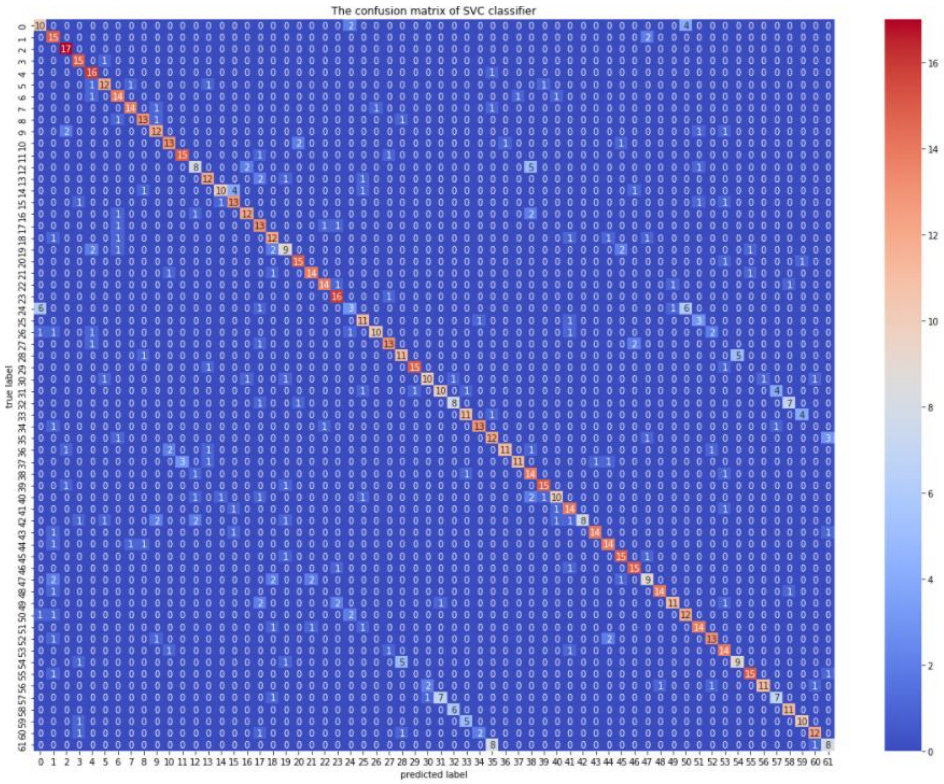


Figure 12. Confusion matrix of Random Forest

From the confusion matrix, this experiment found that the error points of RandomForest are similar to the SVM classifier. There is a significant error in the processing of categories with similar uppercase and lowercase letters. However, the RandomForest classifier has more errors in lines 37-44, and most of these errors are classification errors between letters. For example, line 38 would classify c as e and line 37 b as e. These classification errors resulted in the accuracy of RandomForest classifier being only 73%.

Compare other methods:
From the perspective of accuracy, RandomForest (73%) is similar to SVM (79%) in accuracy compared to the other two methods. However, RandomForest is not the optimal model after a comprehensive consideration of time and accuracy.
Although RandomForest (4min 37s) is different from CNN (11min) in terms of time, there is a big difference in accuracy between the two. Therefore, RandomForest is not a good method compared to CNN.

The reasons why the Randomforest classifier is not ideal are as follows:
Random forest is like a black box and cannot control the internal operation of the model. You can only try between different parameters, and random seeds, which will result in the calculation that the selected parameter is not the best.
There may be many similar decision trees that obscure the real results, and for small data or low-dimensional data (data with fewer features), it may not produce good

classification. These two points will directly lead to the low accuracy of the Randomforest classifier.

Because Randomforest has the above shortcomings, this experiment should try more to find the parameters with the highest accuracy. Like SVM, we also found the type of error (misclassification of letters) from the confusion matrix. This experiment should use a more appropriate method when dealing with these labels—for example, changing the preprocessing method to improve the accuracy of these data.

3. CNN

The pre-processing method in this algorithm is different from the other two because it uses ImageDataGenerator class to batch load data, which means it can process the data and train the model simultaneously. In this case, it saves much more memory and time than the other two models. The average running time of each epoch is less than 1 minute. Moreover, the data is normalized, and the color mode is set to grayscale as well as the labels are converted to categorical type (2D one-HOT encoded label). These can be achieved in a function, reflecting the simplicity of the code.

Figure 13. running time

Before randomly selecting batch data, splitting the data into a train set and test set is needed. Therefore, I execute some operations to achieve train, validation, and test set to store in different files with stratified sampling. Specifically, a validation set is used to fine-tune the model.

As for the hyperparameters in CNN model, I choose ReLU as activation function
$f(x) = \text{Max}(0, x)$
ReLU is mainly used as an activation function in the hidden layer of the neural network, and it will turn all negative activation to zero, which means the nonlinear characteristics of the model and the whole neural network are added, and the expression effect of the convolution layer will not be affected. Thus, it can solve the problem of slow

convergence of neural network learning caused by the disappearance of gradient.

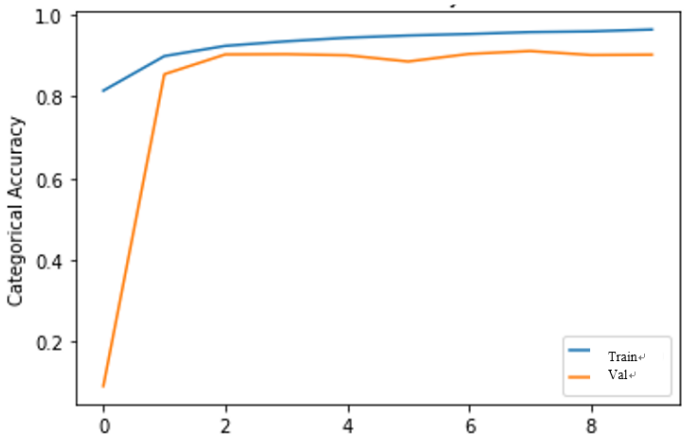


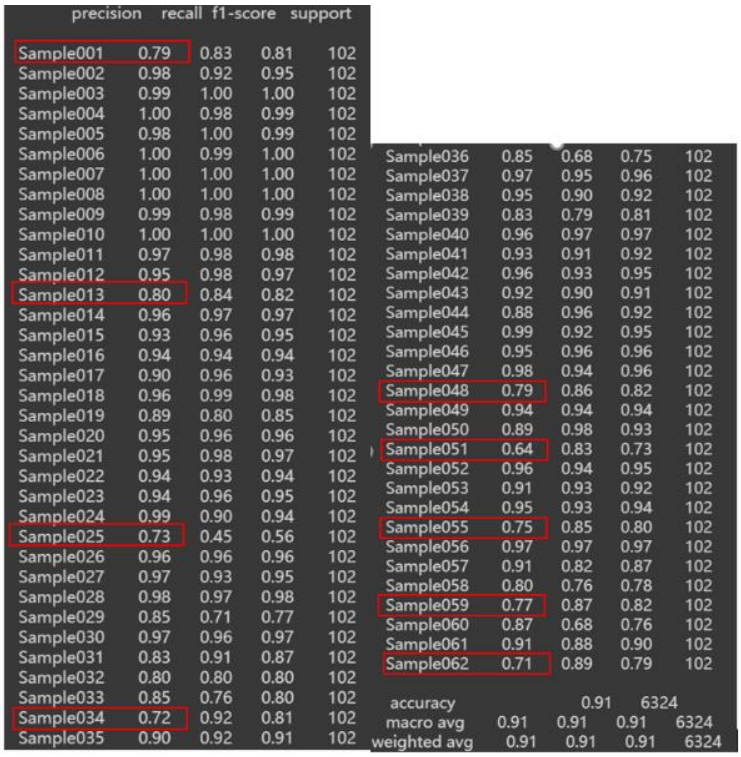
Figure 14. Model Accuracy

As for the result, it can be seen that after one epoch, the validation accuracy remains at a similar level (90%~95%) to train accuracy, which means the model is set up reasonably and trained effectively. Moreover, between 4 and 6 epochs, there is a slight decrease in validation accuracy, but train accuracy keeps growing. Perhaps it's because of model overfitting or improper learning rate.

```
pr_model performance:
loss for test dataset is : 0.32
categorical_accuracy for test dataset is : 0.91
```

Figure 15. Test data accuracy

From my perspective, 91% test accuracy is a relatively good model. But if I want to improve the model accuracy further, one of the possible methods is to add more convolution layers to obtain more extracted information. As shown in Figure 7. the accuracy of each of them lower than 0.8 will be marked in red. For example, samples 001(class=’0’) and 025(class=’O’) are very similar. Therefore, adding an appropriate convolutional layer model is likely to extract more essential information and fit the model to higher accuracy.



	precision	recall	f1-score	support
Sample001	0.79	0.83	0.81	102
Sample002	0.98	0.92	0.95	102
Sample003	0.99	1.00	1.00	102
Sample004	1.00	0.98	0.99	102
Sample005	0.98	1.00	0.99	102
Sample006	1.00	0.99	1.00	102
Sample007	1.00	1.00	1.00	102
Sample008	1.00	1.00	1.00	102
Sample009	0.99	0.98	0.99	102
Sample010	1.00	1.00	1.00	102
Sample011	0.97	0.98	0.98	102
Sample012	0.95	0.98	0.97	102
Sample013	0.80	0.84	0.82	102
Sample014	0.96	0.97	0.97	102
Sample015	0.93	0.96	0.95	102
Sample016	0.94	0.94	0.94	102
Sample017	0.90	0.96	0.93	102
Sample018	0.96	0.99	0.98	102
Sample019	0.89	0.80	0.85	102
Sample020	0.95	0.96	0.96	102
Sample021	0.95	0.98	0.97	102
Sample022	0.94	0.93	0.94	102
Sample023	0.94	0.96	0.95	102
Sample024	0.99	0.90	0.94	102
Sample025	0.73	0.45	0.56	102
Sample026	0.96	0.96	0.96	102
Sample027	0.97	0.93	0.95	102
Sample028	0.98	0.97	0.98	102
Sample029	0.85	0.71	0.77	102
Sample030	0.97	0.96	0.97	102
Sample031	0.83	0.91	0.87	102
Sample032	0.80	0.80	0.80	102
Sample033	0.85	0.76	0.80	102
Sample034	0.72	0.92	0.81	102
Sample035	0.90	0.92	0.91	102
Sample036	0.85	0.68	0.75	102
Sample037	0.97	0.95	0.96	102
Sample038	0.95	0.90	0.92	102
Sample039	0.83	0.79	0.81	102
Sample040	0.96	0.97	0.97	102
Sample041	0.93	0.91	0.92	102
Sample042	0.96	0.93	0.95	102
Sample043	0.92	0.90	0.91	102
Sample044	0.88	0.96	0.92	102
Sample045	0.99	0.92	0.95	102
Sample046	0.95	0.96	0.96	102
Sample047	0.98	0.94	0.96	102
Sample048	0.79	0.86	0.82	102
Sample049	0.94	0.94	0.94	102
Sample050	0.89	0.98	0.93	102
Sample051	0.64	0.83	0.73	102
Sample052	0.96	0.94	0.95	102
Sample053	0.91	0.93	0.92	102
Sample054	0.95	0.93	0.94	102
Sample055	0.75	0.85	0.80	102
Sample056	0.97	0.97	0.97	102
Sample057	0.91	0.82	0.87	102
Sample058	0.80	0.76	0.78	102
Sample059	0.77	0.87	0.82	102
Sample060	0.87	0.68	0.76	102
Sample061	0.91	0.88	0.90	102
Sample062	0.71	0.89	0.79	102
accuracy			0.91	6324
macro avg	0.91	0.91	0.91	6324
weighted avg	0.91	0.91	0.91	6324

Figure 16. Test set classification report

Another method is data augmentation, such as rotation/reflection and flips in the category marked in red. Data augmentation will expand the data set size and enhance the generalization ability of the model.

Conclusion

Through the above analysis, we determined that the CNN algorithm would be our finally best algorithm. Moreover, it has an accuracy of 91%, much higher than the SVM algorithm and Random Forest algorithm (79%, 73%, respectively). Apart from the accuracy, time cost also is a critical metric in evaluating the model. It only takes 1 minute to run an epoch. The CNN algorithm can use various methods to accelerate the training model, such as load batch data instead of all data, pre-process data and train model at the same time, max-pooling layer, activation function, and so on. Therefore, even though CNN uses all data (62992), its running speed is not much slower than SVM and random forest (only part of 6200).

In order to further improve the accuracy of CNN, adding more convolutional layer and data augmentation can be considered. The CNN pre-training model with complex and fixed structures, such as AlexNet, LeNet, GoogLeNet, can be applied to the same dataset and then compared their result.

References

Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. *In 2017 International Conference on Engineering and Technology (ICET)* (pp. 1-6). Ieee.Overview Random Forest. (2018). Retrieved from https://gaussian37.github.io/ml-concept-RandomForest/

Arroyo-Pérez, D. E., Álvarez-Canchila, O. I., Patiño-Saucedo, A., González, H. R., & Patiño-Vanegas, A. (2020, May). Automatic recognition of Colombian car license plates using convolutional neural networks and Chars74k database. *In Journal of Physics: Conference Series* (Vol. 1547, No. 1, p. 012024). IOP Publishing

Ewrobel3 (2019.). Chars74KDataAnalysis/kNN.py at master · ewrobel3/chars74KDataAnalysis. Retrieved from: https://github.com//ewrobel3/chars74KDataAnalysis/blob/master/kNN.py

Ioffe, S., & Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *In International conference on machine learning* (pp. 448-456). PMLR.

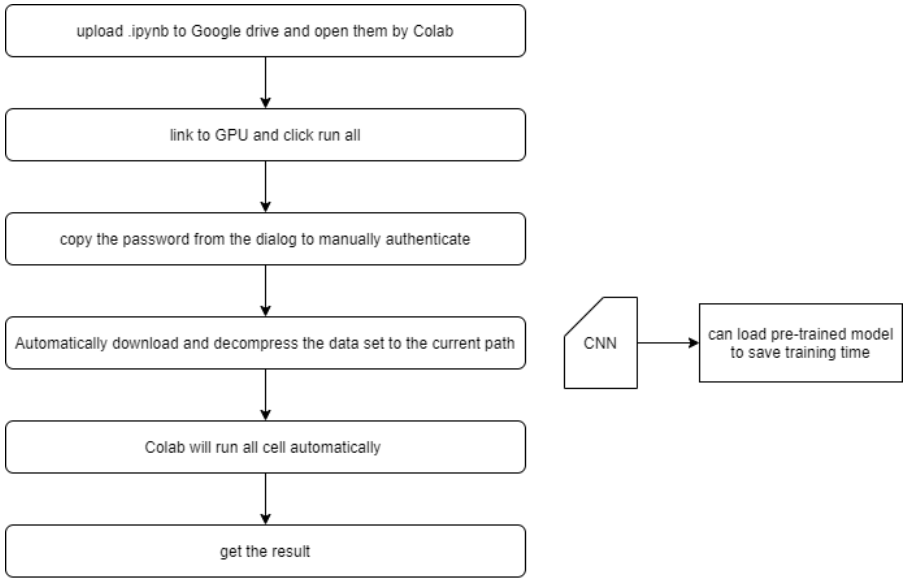
Kadir, N. H. M., Hidayah, S. N. S. M. N., & NorasiahMohammad, Z. I. (2019). Comparison of convolutional neural network and bag of features for multi-font digit recognition. Indonesian *Journal of Electrical Engineering and Computer* Science, 15(3), 1322-1328.

Yan, K., Huang, S., Song, Y., Liu, W., & Fan, N. (2017, July). Face recognition based on convolution neural network. *In 2017 36th Chinese Control Conference (CCC)* (pp. 4077-4081). IEEE.

Zero basic understanding of NN and CNN. (2018). Retrieved from https://blog.csdn.net/qq_31804159/article/details/88743875

Appendix:

1. Instructions



2. Operating environment

```
Sat Nov  6 08:24:55 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 495.44       Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla K80           Off  | 00000000:00:04.0 Off |                    0 |
| N/A   34C    P8    28W / 149W |      0MiB / 11441MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

3. open-source libraries

Numpy 1.21, Pandas1.34, Matplotlib3.43,
Sklearn 1.0 https://scikit-learn.org/stable/modules/classes.html#
Seaborn 0.11.2 https://seaborn.pydata.org/
TensorFlow 2.7.0 https://www.tensorflow.org/api_docs/python/tf

The Chars74K, classification, http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/