Assignment 2: Image Classification Based on Four Algorithms

Boning He, 480059136, Ruoqian Xu, 480324908, Ning Zhou, 450008188

***Abstract*—In recent years, with the innovation of Internet technology, machine learning methods have made a series of progresses in the field of image classification. Deep learning, as a branch of machine learning, drives the performance of machine learning algorithms to increase dramatically. Deep learning models, such as convolutional neural networks (CNN) and recurrent neural networks (RNN) have been widely used and focused in the fields of computer vision, speech recognition and processing, which have promoted the development of related fields. This report compares the performance of four different algorithms of machine learning and deep learning including KNN, random forest, decision tree and CNN in image classification. The data set used is fashion-mnist. For the evaluation part, ten fold cross-validation and confusion matrix have been applied to analyze the performance of each algorithm. The results show that CNN is the most efficient method with highest accuracy and shortest running time.**

***Index Terms*—machine learning, deep learning, CNN, random forest, decision tree, knn, ten fold cross validation.**

I. Introduction

ARTIFICIAL intelligence (AI) is one of the hot research fields of computer science. As a subset of AI, image recognition has been widely applied in a variety of environments from high technology to the mundane. More specifically, it provides powerful assistance and visual enhancement for human vision, offering a new approach of interacting with the external world.

According to Hubspot, in 2017, 150 more shares received for tweets with images than textual tweets [1]. Although visual communication has always been the essence way of human expression, images are a potentially more complicated and ambiguous method. For example, when the information is recorded in words, it is easy to find the desired content by searching for keywords. While the information is recorded by image, the content cannot be simply retrieved by applying the same manner. Although the picture provided an effective way to record and share information, but it reduces the efficiency of information retrieval. Under this circumstance, image recognition technology is particularly important.

By providing an automatic image recognition mechanism, it assists in reducing redundancy and improving the efficiency of information retrieval. In this study, four different effective algorithms are applied to analyse the Fashion-mnist dataset, including convolutional neural network, random forest algorithm, decision tree and k-nearest neighbors algorithm (KNN). After comparing the performance of four algorithms, it is shown that convolutional neural network with the highest accuracy rate, and is the most effective method to analysis this dataset.

II. Previous Work

According to the benchmarking system, which provided by the official Fashion-MNIST dataset website, support-vector clustering(SVC), gradient boosting classifier, and multilayer perceptron (MLP classifier) are the popular and successful methods that is used in this datasets [2].

Firstly, support-vector clustering. Different to support-vector machines, it is effective in unsupervised learning. The highest average accuracy rate for Fashion-MNIST dataset is 0.897, while the training time is one hour and twelve minutes [2]. The key concept of SVC is to map data points to high-dimensional feature space through Gaussian kernel, and to acquire sphere with the smallest radius, which contains most of the mapped data points in the characteristic space [3]. The main advantages of SVC algorithm included the detectability of cluster boundaries

of arbitrary shapes, and the use of soft boundary constants to handle outliers, so that spheres does not attach all points in the feature space [4]. However, the algorithm processes all data points that do not belong to one of the clusters, indicating it as an outlier. In particular, in the presence of decentralized data, this will be a major drawback, because not distinguishing outliers will lose important information [5].

Secondly, gradient boosting classifier. The highest average accuracy rate for Fashion-MNIST dataset is 0.888, while the training time is seventeen hours and thirty-eight minutes [2]. It constructs the model in a phased manner and builds an additive regression model by successively fitting a parameterized function to the current pseudo-residue with the least squares in each iteration [6]. The main advantage of gradient boosting classifier is the flexibility. It optimizes different pseudo-residue functions and offers several hyperparameter adjustment options to make the function very flexible. However, this method would be computationally expensive [7]. Many trees maybe required for the calculations, thereby it would be memory exhaustive and time consuming [7].

Thirdly, multilayer perceptron classifier (MLP). The highest average accuracy rate for Fashion-MNIST dataset is 0.877, while the training time is sixteen minutes [2]. MLP is an artificial neural network, which applied back propagation supervised learning techniques for training. One of the advantages of MLP is that it is a generalization of perceptron, which overcomes the weakness that perceptron cannot recognize linearly indivisible data [8]. For the main limitation, because it applies back propagation techniques, instead of identifying the global minima, the algorithm may recognise the local minima.

III. Method

A. Pre-processing

In image classification, differences in data sources usually lead to differences in the magnitudes of the data. In order to make these data comparable, standardized methods are needed to eliminate these differences. The most typical data standardization is the normalization of data, and the data is uniformly mapped to the interval [0,1]. The specific method is to divide all training set and test set data by 255. One of the advantages is that the accuracy of the model can be improved, and the other is that it can improve the convergence efficiency.

Random Forest Algorithm is also applied in the pre-processing process. This algorithm can produce values for the importance of each feature and a higher value means that the feature is more important for the final prediction. Since the prediction by the K-nearest neighbor algorithm is very time consuming, the features have been selected first based on the feature importance results produced by Random Forest algorithm.

B. K-Nearest Neighbors algorithm

The basic idea of KNN is that if majority of the instances of K that are most similar in an instance (ie, nearest neighbors in the feature space), which belong to a certain category, then the instance also belongs to that category. The selected neighbors are all instances that have been correctly classified. The nearest neighbor of an instance is defined according to the standard Euclidean distance(Eq. 1) :

$$d(x,y) = \sqrt{\sum_{k=1}^{N}(x_k - y_k)^2} \tag{1}$$

Equation(1): Euclidean distance

$$d(x,y) = \sqrt{\sum_{k=1}^{N}|x_k - y_k|} \tag{2}$$

Equation(2): Manhattan distance

The first K points with the smallest distance from the current point are selected, and the category with the most frequency among these K points is assigned to the prediction of the current point.

C. Decision Tree

The decision tree uses a tree-like structure to represent the division of the classes. The construction of a tree can be seen as the process of selecting

variables (attributes). The internal nodes represent the variables (trees) selected by the tree as partitions, each tree. The leaf nodes are represented as labels of the class, the topmost layer of the tree is the root node, and the data is classified by a series of rules. One of the decision trees is the CART decision tree, which is generated by recursively constructing a binary decision tree. The information gain maximization criterion is used for classification and pruning. The method of information gain is the Gini index formula (Eq. 3) [9] :

$$Gini(p) = \sum_{k=1}^{K} p_k(1-p_k) = 1 - \sum_{k=1}^{K} y_k^2 \quad (3)$$

Pk indicates the probability that the selected sample belongs to the k categoryAssuming there are K categories in the dataset, then (Eq. 4) :

$$Gini(p) = 1 - \sum_{k=1}^{K} \frac{(|C_k|)^2}{|D|)} \quad (4)$$

D. Random Forest

Random forests build a forest in a random way. There are many decision trees in the forest, and there is no correlation between them. When a new input sample comes in, let each decision tree in the forest make a separate judgment to see which class the sample should belong to (for the classification algorithm), and then see which class is selected the most, then this sample belongs to this class. The input is the sample set D=(x, y1), (x2, y2), ... (xm, ym), and the weak classifier iteration number T. The output is the final strong classifier f(x).

- For t=1, 2..., T: Perform the t-th random sampling on the training set, and collect a total of m times to obtain a sampling set Dt containing m samples. training the t-th decision tree model Gt(x) with the sample set Dt, and selecting a part of the sample features among all the sample features on the node when training the nodes of the decision tree model, among the randomly selected partial sample features Select an optimal feature to make the left and right subtree partitions of the decision tree.
- Perform the t-th random sampling on the training set, and collect a total of m times to obtain a sampling set Dt containing m samples. Training the t-th decision tree model Gt(x) with the sample set Dt, and selecting a part of the sample features among all the sample features on the node when training the nodes of the decision tree model, among the randomly selected partial sample features Select an optimal feature to make the left and right subtree partitions of the decision tree.If it is a classification algorithm prediction, one of the categories or categories in which the T weak learners cast the most votes is the final category. If it is a regression algorithm, the regression results obtained by the T weak learners are arithmetically averaged to obtain the final model output.

E. Convolutional Neural Networks

The convolutional neural network is a multi-layer supervised learning neural network. The convolutional layer and the pooling layer of the hidden layer are the core modules for implementing the feature extraction function of the convolutional neural network. The network model minimizes the loss function by using the gradient descent method to adjust the weight parameters in the network layer by layer, and improves the accuracy of the network through frequent iterative training.

The low hidden layer of the convolutional neural network is composed of a convolutional layer and a maximum pool sampling layer. The upper layer is the hidden layer and logistic regression classifier of the fully-connected layer corresponding to the traditional multi-layer perceptron.

The input of the first fully connected layer is a feature image obtained by feature extraction from the convolutional layer and the pooling layer layer. The last layer of the output layer is a classifier, which can be classified using logistic regression, Softmax regression, or even support vector machines.

The convolutional neural network structure includes: a convolutional layer, a pooling layer, and

a fully-connected layer. Each layer has a plurality of feature maps, each of which extracts a feature of the input through a convolution filter, each feature map having a plurality of neurons [10].

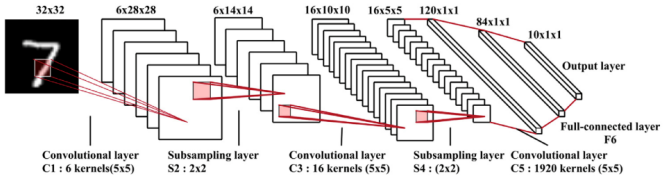

Fig.1. An example of CNN structure.

Keras is a highly integrated development framework for deep learning. This project uses Keras to implement CNN.

IV. Experiments and Discussion

In order to solve this problem, four algorithms were compared and evaluated during experiment, including Random Forest Algorithm, Decision Tree, K-nearest Neighbour and Convolutional Neural Network.

A. Dataset partition

For Random Forest Algorithm and Decision Tree, 54000 rows of data are used as training dataset, 6000 rows are used for validation, and 10000 rows for testing. While for K-nearest Neighbour, 18000 rows have been used for training, 2000 rows for validation, and 3000 rows for testing due to the long period of time for prediction. For the Convolutional Neural Network, 50000 rows of data are used as training dataset, 10000 rows are used for validation, and 10000 rows for testing.

Choosing an appropriate model and its complexity

B. Random Forest

Random Forest algorithm is under experiments and different hyperparameters have been tested for improving accuracy. For the adjustment of hyperparameter maxDepth in the RandomForestClassifier imported from sklearn, the accuracy result (88.1%) is the highest when maxDepth is 30. When maxDepth is set from 5 to 55, the accuracy results are shown in the figure below (Fig. 2). It seems that after maxDepth is over 40, the accuracy remains the same.

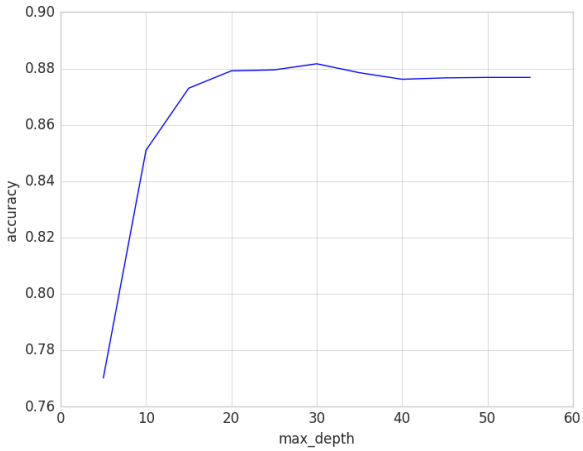

Fig.2. Random Forest-maxDepth.

For the adjustment of hyperparameter nEstimators in the RandomForestClassifier, the accuracy result is the highest (88.4%), when nEstimators is 70 and maxDepth is 30, which is shown below (Fig. 3). Therefore, in order to achieve the best performance, these hyperparameters have been chosen.

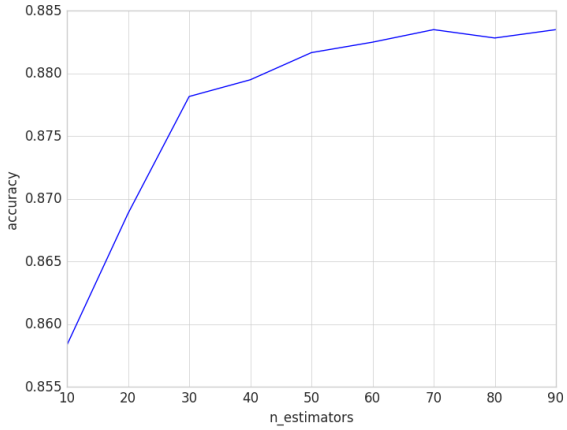

Fig.3. Random Forest-nEstimators.

For ten fold cross validation, when choosing the hyperparameters with the best performance,

the average accuracy result is 88.3%. The chart is shown below (Fig.4).

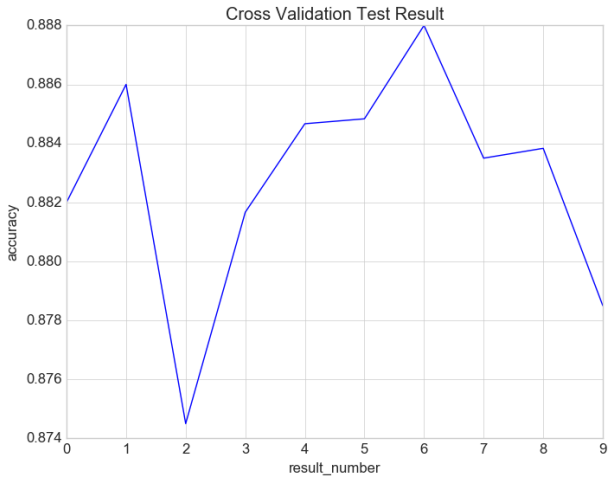

Fig.4. Random Forest-Ten Fold Cross Validation.

For the Precision, Recall and F-measure of test results, it can be seen that Trouser has the highest F1 score (98%), while shirt has the lowest F1 score (64%), the average F1 score for 10 classes is 88% for 10000 test data (Fig. 5).

	precision	recall	f1-score	support
T-shirt/top	0.81	0.86	0.83	1000
Trouser	1	0.96	0.98	1000
Pullover	0.77	0.8	0.79	1000
Dress	0.88	0.91	0.89	1000
Coat	0.77	0.82	0.79	1000
Sandal	0.97	0.96	0.97	1000
Shirt	0.71	0.59	0.64	1000
Sneaker	0.93	0.95	0.94	1000
Bag	0.96	0.97	0.97	1000
Ankle boot	0.95	0.95	0.95	1000
accuracy			0.88	10000
macro avg	0.87	0.88	0.87	10000
weighted avg	0.87	0.88	0.87	10000

Fig.5. Random Forest-result.

For the confusion matrix of test results (Fig. 6), it can be clearly seen that class 1 has the highest True Positive rate(96%) while class 6 has the lowest True Positive Rate(59%).

Fig.6. Random Forest-Confusion Matrix.

C. Decision Tree

Decision tree is applied to tackle this problem, and it achieves a much lower accuracy result than random forest algorithm. During the hyperparameter tuning process, maxDepth is adjusted and accuracy result is shown below. It can be seen that when maxDepth is set to 13, the algorithm achieves the highest accuracy (Fig. 7).

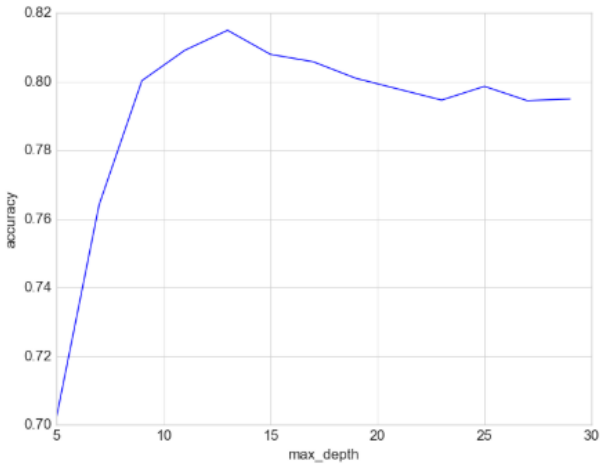

Fig.7. Decision Tree-maxDepth.

For ten fold cross validation, the average accuracy result is 82.0% when choosing the hyperparameters with the best performance. The chart is shown below (Fig. 8).

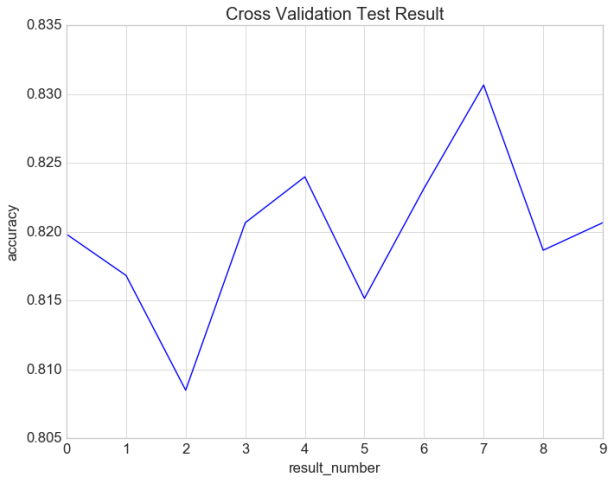

Fig.8. Decision Tree-TenFoldCrossValidation.

For the Precision Recall and F-measure of test results, it can be seen that Trouser has the highest F1 score (95%), while shirt has the lowest F1 score (56%), the average F1 score for 10 classes is 81% for 10000 test data (Fig. 9).

	precision	recall	f1-score	support
T-shirt/top	0.78	0.78	0.78	1000
Trouser	0.95	0.94	0.95	1000
Pullover	0.67	0.7	0.68	1000
Dress	0.83	0.79	0.81	1000
Coat	0.66	0.72	0.69	1000
Sandal	0.93	0.88	0.91	1000
Shirt	0.58	0.54	0.56	1000
Sneaker	0.88	0.92	0.9	1000
Bag	0.93	0.93	0.93	1000
Ankle boot	0.92	0.92	0.92	1000
accuracy			0.81	10000
macro avg	0.81	0.81	0.81	10000
weighted avg	0.81	0.81	0.81	10000

Fig.9. Decision Tree-result.

For the confusion matrix, it can be clearly seen that class 1 has the highest True Positive rate(94%) while class 6 has the lowest True Positive Rate(54%) (Fig. 10).

Fig.10. Decision Tree-Confusion Matrix.

D. KNN

When K-nearest Neighbor was applied to solve this problem, there was no better result achieved. In the hyperparameter tuning process, K which is the number of top K nearest neighbours is adjusted to achieve better performance. As it is time consuming to make prediction by KNN with a large scale of dataset, only part of data have been used for training, validation and test. Unfortunately, KNN shows only 83.8% accuracy result when K is 4 (Fig.11).

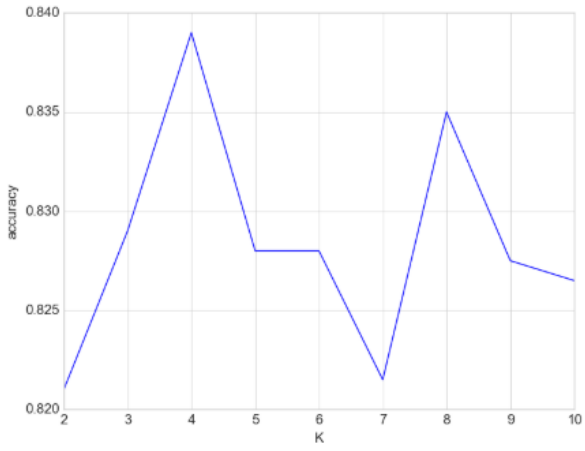

Fig.11. KNN - Value K.

For ten fold cross validation, the average accuracy result is 82.0% when choosing the hyperparameters

with the best performance. The chart is shown below (Fig.12).

Fig.12. KNN-Ten Fold Cross Validation.

For the Precision Recall and F-measure of test results, it can be seen that Trouser has the highest F1 score (97%), while shirt has the lowest F1 score (53%), the average F1 score for 10 classes is 83% for 3000 test data (Fig.13).

	precision	recall	f1-score	support
T-shirt/top	0.73	0.86	0.79	302
Trouser	0.98	0.95	0.97	308
Pullover	0.7	0.8	0.75	310
Dress	0.89	0.84	0.87	298
Coat	0.75	0.71	0.73	324
Sandal	1	0.82	0.9	285
Shirt	0.57	0.51	0.53	298
Sneaker	0.84	0.95	0.89	293
Bag	0.98	0.91	0.94	297
Ankle boot	0.9	0.94	0.92	285
accuracy			0.83	3000
macro avg	0.83	0.83	0.83	3000
weighted avg	0.83	0.83	0.83	3000

Fig.13. KNN-result.

For the confusion matrix, it can be clearly seen that class 1 has the highest True Positive rate(95%) while class 6 has the lowest True Positive Rate(51%) (Fig.14).

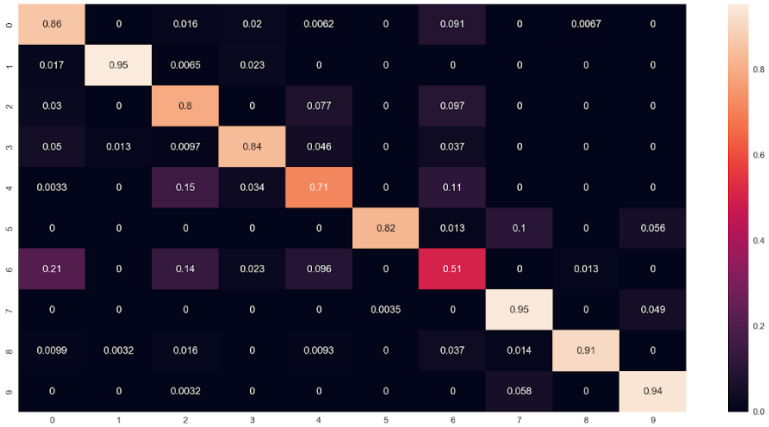

Fig.14. KNN-ConfusionMatrix.

E. Convolutional Neural Networks

The CNN structure consists of two convolutional layers, two pooling layers, two dropout layers, a flatten layer and two dense layers (fully-connected layers).

Dropout layer:
The Dropout layer can randomly break the input neurons by a certain probability, each time the parameters are updated during the training process, which can prevent overfitting.

Flatten layer:
The Flatten layer is used to ”flattening” the input, which can multidimensionalize the multidimensional input, it is often used in the transition from the convolutional layer to the fully connected layer. Flattening does not affect the size of the batch.

Through ten times epoch (Train ten times for the entire training set) on the training set, the final accuracy rate reached 95.74%, and the accuracy of the verification set is 91.23% (Fig.16). The total training time is 10 minutes and 2 seconds.

As can be seen from the line charts (Fig. 17 &Fig. 18), as the number of training increases, the training accuracy increases linearly, and the validation accuracy first increases sharply. When epoch equals to 1, the validation accuracy begins to remain stable with slightly increase. For loss,

Layer (type)	Output Shape	Param #
conv2d_5 (Conv2D)	(None, 28, 28, 32)	832
max_pooling2d_5 (MaxPooling2	(None, 14, 14, 32)	0
dropout_5 (Dropout)	(None, 14, 14, 32)	0
conv2d_6 (Conv2D)	(None, 14, 14, 64)	18496
max_pooling2d_6 (MaxPooling2	(None, 7, 7, 64)	0
dropout_6 (Dropout)	(None, 7, 7, 64)	0
flatten_3 (Flatten)	(None, 3136)	0
dense_5 (Dense)	(None, 256)	803072
dense_6 (Dense)	(None, 10)	2570

Total params: 824,970
Trainable params: 824,970
Non-trainable params: 0

Fig.15. CNN structure and params.

```
Train on 50000 samples, validate on 10000 samples
Epoch 1/10
50000/50000 [==============================] - 44s 889us/step - loss: 0.4467 - acc: 0.8406 - val_loss: 0.3506 - val_acc: 0.8712
Epoch 2/10
50000/50000 [==============================] - 45s 901us/step - loss: 0.2978 - acc: 0.8908 - val_loss: 0.2710 - val_acc: 0.8997
Epoch 3/10
50000/50000 [==============================] - 45s 891us/step - loss: 0.2536 - acc: 0.9053 - val_loss: 0.2636 - val_acc: 0.9049
Epoch 4/10
50000/50000 [==============================] - 45s 890us/step - loss: 0.2202 - acc: 0.9179 - val_loss: 0.2513 - val_acc: 0.9067
Epoch 5/10
50000/50000 [==============================] - 44s 890us/step - loss: 0.1922 - acc: 0.9275 - val_loss: 0.2465 - val_acc: 0.9071
Epoch 6/10
50000/50000 [==============================] - 209s 4ms/step - loss: 0.1734 - acc: 0.9337 - val_loss: 0.2543 - val_acc: 0.9152
Epoch 7/10
50000/50000 [==============================] - 46s 913us/step - loss: 0.1498 - acc: 0.9427 - val_loss: 0.2778 - val_acc: 0.9064
Epoch 8/10
50000/50000 [==============================] - 45s 890us/step - loss: 0.1357 - acc: 0.9484 - val_loss: 0.2669 - val_acc: 0.9106
Epoch 9/10
50000/50000 [==============================] - 45s 897us/step - loss: 0.1182 - acc: 0.9551 - val_loss: 0.2882 - val_acc: 0.9110
Epoch 10/10
50000/50000 [==============================] - 44s 878us/step - loss: 0.1109 - acc: 0.9574 - val_loss: 0.2777 - val_acc: 0.9123
```

Fig.16. CNN-Loss and accuracy of the model through 10 times epoch.

the training loss is substantially linearly reduced, while the validation loss decreases first. When epoch equals to 4, the validation loss begins to increase slowly.

For the Precision, Recall and F-measure of test results, it can be seen that Trouser has the highest F1 score (99%), while shirt has the lowest F1 score (74%), the average F1 score for 10 classes is 91% for 10000 test data (Fig.19).

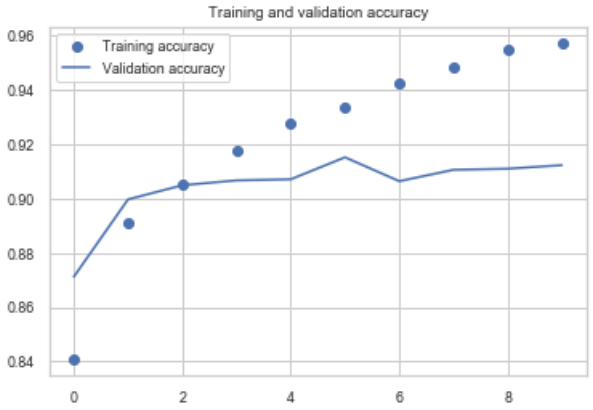

Fig.17. CNN-Accuracy and loss line chart 1.

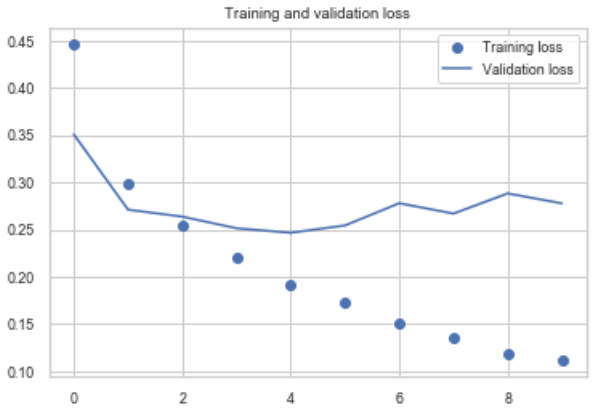

Fig.18. CNN-Accuracy and loss line chart 2.

	precision	recall	f1-score	support
T-shirt/top	0.82	0.89	0.86	1000
Trouser	0.99	0.98	0.99	1000
Pullover	0.88	0.84	0.86	1000
Dress	0.93	0.91	0.92	1000
Coat	0.79	0.93	0.85	1000
Sandal	0.97	0.99	0.98	1000
Shirt	0.83	0.67	0.74	1000
Sneaker	0.96	0.96	0.96	1000
Bag	0.97	0.98	0.98	1000
Ankle boot	0.98	0.96	0.97	1000
accuracy			0.91	10000
macro avg	0.91	0.91	0.91	10000
weighted avg	0.91	0.91	0.91	10000

Fig.19. CNN-result.

For the confusion matrix, it can be seen that class 5 has the highest number of TP (0.99), while class 6 has the lowest number of TP (0.73) (Fig.20).

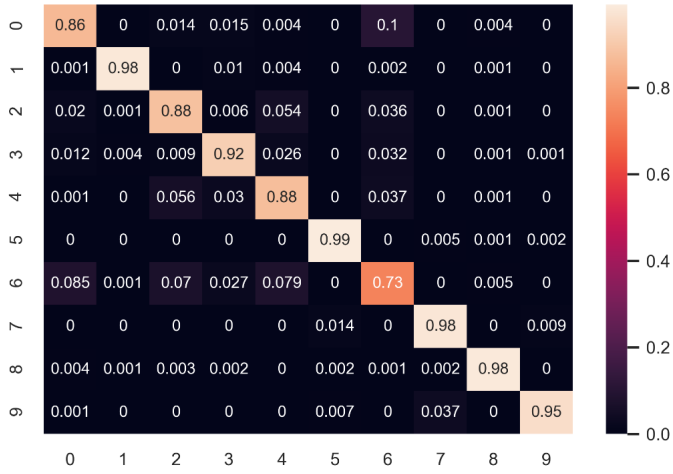

Fig.20. CNN-Confusion Matrix.

F. Comparison and Reflection

According to all the experiment results on four algorithms mentioned above, it can be seen that all the algorithms get better performance on class 1 and the lowest accuracy result on class 6. As for classification performance on all classes, CNN achieves the best accuracy result(91%) while Decision Tree(81%) and KNN(83%) have limited predictive power on this dataset. Random Forest shows a little better prediction result as it applies 70 decision trees and makes prediction based on all 70 results each time. So it is no wonder that it performs better than other machine learning algorithms. As CNN uses a large number of parameters and can extract important information from the dataset, it can achieve much higher accuracy results that traditional machine learning algorithms. Moreover, considering running time for each algorithm, KNN takes a longer period of time for prediction than other algorithms when a large scale of data are used. Due to all the performance results and running time comparison, CNN is selected for final prediction as it achieves the best performance result and time for training process is acceptable.

The convolutional neural network is characterized by layer-by-layer extraction of features. The features extracted by the first layer are relatively low-level, and the second layer continues to extract higher-level features based on the first layer. Similarly, the third layer is based on the second layer. The features extracted above are also more complicated. More advanced features can reflect the category attributes of the image, and the convolutional neural network extracts the excellent features of the image by layer-by-layer convolution.

There are three reasons to build CNN with Keras. The first reason is modularity: the various parts of the model, such as neural layer, cost function, optimizer, initialization, activation function, and normalization, are independent modules that can be combined to create a model. The second reason is scalability, and it's easy for Keras to add new modules and modify parameters. The third one is minimalism, so that each module is short and simple.

This experiment applied seaborn library to draw pictures. Seaborn has a more advanced API package based on matplotlib, which is mainly used for statistical charts. The advantage is that the code is simple, and the results are intuitive.

G. Optimization

Based on the Keras framework, this experiment has optimized the CNN model. First, using overlapping convolutions and pooling can alleviate overfitting problems and improve accuracy. For this data set, using the maximum pooling is better than the average pooling. This experiment performed two rounds of convolution and pooling operations, effectively alleviating overfitting. Second, Dropout technology has also used to improve the over-fitting problem for neural networks. Dropout refers to randomly discarding some neurons in a certain proportion during the training network and randomly selecting a part of the neurons in a layer to make the output value 0, which will make the selected neurons do not contribute to the next layer's connected neuron output and loses its effect. Therefore, each batch of data is trained in a different network structures, which is equivalent to training multiple networks, combining multiple networks of different structures, and integrating multiple training networks, which can effectively prevent a single Over-fitting of the

structural network. This experiment increased a Dropout ratio of 0.2 after each pooling. Finally, after several tests, the experiment set the epoch number to 10 and the batch size to 64, which can achieve a higher accuracy in a shorter training time.

Library numba is also applied to speed up the code during the hyperparameter tuning process, which takes a short period of time.

V. Conclusion and Future Work

With the rapid development of Internet and multimedia technologies, image data has experienced explosive growth. How to efficiently classify and retrieve massive images has become a new challenge. Image classification is the basis of many applications, such as image retrieval, object detection and image recognition, and is also a research hotspot in pattern recognition and machine learning.

In this report, four algorithms were compared during experiment, including Random Forest Algorithm, Decision Tree, K-nearest Neighbour and Convolutional Neural Network. Among all four algorithms, CNN has the highest accuracy (0.91), followed by random forest (0.88), KNN(0.83), and decision tree (0.81). Moreover, the algorithm with the shortest running time for processing the ten cross validations is random forest (8 minutes), followed by CNN (10.02 minutes), decision tree (11 minutes), and KNN(13 minutes). In summary, CNN and random forest are the two most effective algorithms used in this study to analyze the Fashion-MNIST data set.

In the future, it is recommended to use more algorithms, such as support-vector clustering, gradient boosting classifier, multilayer perceptron classifier, to compare performance with the existing algorithms for the Fashion-MNIST data set. In addition, applying the same four algorithms to other datasets, to compare and evaluate performance is also recommended in the future.

Appendix A
Hyperlink

- **Train-image:**http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-images-idx3-ubyte.gz
- **Train-labels:**http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-labels-idx1-ubyte.gz
- **Test-image:**http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-images-idx3-ubyte.gz
- **Test-labels:**http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-labels-idx1-ubyte.gz

Appendix B
External open-source libraries

- **S**eaborn 0.90
- **S**cikit-learn 0.21.1
- **P**andas 0.24.2
- **T**ensorFlow
- **K**eras
- **N**umba

Appendix C
Code Instructions

How to run the code:
Please run all the code one by one ,install idx2numpy first and put data in the input folder. 1.

1) How to run mainalgorithm.ipynb:

   To train model: add your local path of input folder as the parameter in the KerasCNN() function

   To see prediction results: add your local path of input folder as the parameter in the makeprediction() function.

   For example, if your input folder is in the path ”D:/assignment/”, add the path as the parameter.

   To load model: run function Loadmodel() and change filename to model name including your local path.

   For example, if the saved model ”mymodel.h5” is put in path ”D:/assignment/”, change filename to ”D:/assignment/mymodel.h5”.

2) How to run otherwork.ipynb:

To read data, add your local path of input folder as the parameter in the `read_data()` function.

For example, if your input folder is in the path "D:/assignment/", add the path as the parameter.

To tune algorithm, run functions with name including tune.

To complete cross validation, run function `model_cross_validation()` To make prediction on test and see test results, run function `model_test()`.

To load model and see prediction accuracy results, run function `load_model()` and change file name to model name including your local path.

For example, if the saved model "ranforestmodel.sav" is put in path "D:/assignment/", change file name to "D:/assignment/ranforestmodel.sav".

APPENDIX D
CONTRIBUTION

- **Boning He:** Mainly responsible for the design and implementation of CNN code. In the report, Boning is mainly responsible for the abstract, algorithm description, CNN result analysis and reflection part.
- **Ning Zhou:** Mainly responsible for the design and implementation of the KNN code. In the report, Ning is responsible for introduction, conclusion, and organizing references and applying the latex format.
- **Ruoqian Xu:** Mainly responsible for the design and implementation of the decision tree and random forest code. In the report, Ruoqian is mainly responsible for the result analysis and methods comparison.

REFERENCES

[1] Kula Partners, "17 fascinating stats about visual content marketing in 2017," 2017. [Online].Available:https://cdn2.hubspot.net/hubfs/211423/KULA%20Visual%20Content%20Infographic.pdf (accessed on May 22, 2019).

[2] Zalando Research, "Fashion mnist," 2019. [Online] .Available:http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/# (accessed on May 22, 2019).

[3] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik, "Support vector clustering," *Journal of machine learning research*, vol. 2, no. Dec, pp. 125–137, 2001.

[4] J. Lee and D. Lee, "An improved cluster labeling method for support vector clustering," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 3, pp. 461–464, 2005.

[5] R. Saltos and R. Weber, "A rough–fuzzy approach for support vector clustering," *information Sciences*, vol. 339, pp. 353–368, 2016.

[6] Q. Xu, Y. Xiong, H. Dai, K. M. Kumari, Q. Xu, H.-Y. Ou, and D.-Q. Wei, "Pdc-sgb: Prediction of effective drug combinations using a stochastic gradient boosting algorithm," *Journal of theoretical biology*, vol. 417, pp. 1–7, 2017.

[7] P. Li, Q. Wu, and C. J. Burges, "Mcrank: Learning to rank using multiple classification and gradient boosting," in *Advances in neural information processing systems*, pp. 897–904, 2008.

[8] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[9] L. Rutkowski, M. Jaworski, L. Pietruczuk, and P. Duda, "The cart decision tree for mining data streams," *Information Sciences*, vol. 266, pp. 1–15, 2014.

[10] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018.