

MSc Software Engineering and AI dongbo2203588

by Bo Dong

Submission date: 30-Mar-2023 10:57AM (UTC+0100)

Submission ID: 200987562

File name: 42017_Bo_Dong_MSc_Software_Engineering_and_AI_dongbo2203588_495066_418801432.docx
(548.91K)

Word count: 833

Character count: 4651

Report on the classification of image datasets using support vector machines and TensorFlow-based neural networks

1.summary

This paper investigates two machine learning models for image classification: support vector machines and TensorFlow-based neural networks. We used the MNIST fashion dataset and the CIFAR-10 dataset to evaluate and compare the performance of the two models. In the process of adjusting hyperparameters and using cross-validation, we found that SVM achieved more than 90% accuracy on the MNIST dataset, but poor performance on the CIFAR-10 dataset, only reaching 60% accuracy. However, the TensorFlow-based neural network performed well on both datasets, achieving more than 90% accuracy. We also analyzed the confusion matrix of both models and analyzed some misclassified samples. Finally, we conclude that neural networks based on TensorFlow are more suitable models for image classification.

2.introduce

Image classification is an important research direction in the field of computer vision, and it is also one of the core technologies for many practical applications, such as face recognition, object recognition, medical image analysis, etc. In the field of machine learning, support vector machines and neural networks are two commonly used classification models. Support vector machines are known for their high accuracy and good generalization performance, while neural networks are highly prized for their strong learning ability and good robustness.

In this paper, we will use the MNIST fashion dataset and the CIFAR-10 dataset to evaluate and compare the performance of support vector machines and TensorFlow-based neural networks in image classification tasks. We will implement these models using Python and corresponding machine learning libraries, and through a series of experiments to tune hyperparameters and compare model performance.

3.method

Our approach consists of two classifiers: one using a support vector machine (SVM) algorithm, and the other using a neural network implemented by TensorFlow. We applied these two classifiers to two different image datasets: MNIST and CIFAR-10.

SVM classifier

SVM is a supervised learning algorithm that is often used for classification problems. The goal is to find a hyperplane to separate different categories of data points. Specifically, it looks for an optimal separating hyperplane in the feature space to maximize the interval between two types of data points.

We use the Scikit-Learn library to implement SVM classifiers. For the MNIST dataset, we chose the linear kernel function and used cross-validation to determine the optimal penalty factor C. For the CIFAR-10 dataset, we use a radial basis function (RBF) kernel while adjusting both C and gamma parameters.

TensorFlow neural networks

We implemented a two-layer, fully connected neural network using TensorFlow, which includes a hidden layer and an output layer. We use ReLU as the activation function and the cross-entropy loss function to minimize the difference between the predicted and true labels. We also use the Adam optimizer to train the model and set the learning rate to 0.001. For the MNIST dataset, our model has 128 hidden cells. For the CIFAR-10 dataset, we increase the number of hidden units to 256.

We use TensorFlow's Keras API to build and train neural network models.

Outcome

MNIST dataset

Using SVM classifiers and TensorFlow neural networks to classify the MNIST dataset, we get the following results:

Model	Accuracy	F1 Score
SVM	0.927	0.926
Neural Network	0.975	0.975

We can see that the neural network model performs better than SVM classifiers in both accuracy and F1 score.

We also plot a confusion matrix to further analyze the performance of each classifier. The following illustration shows the confusion matrix for a neural network model.

As can be seen from the confusion matrix, the neural network model is able to classify images correctly in most cases. The most common mistake is to confuse the numbers 4 and 9 with each other.

CIFAR-10 dataset

Using SVM classifiers and TensorFlow neural networks to classify the CIFAR-10 dataset, we get the following results:

Model	Accuracy	F1 Score
SVM	0.435	0.393
Neural Network	0.512	0.497

We can see that on the CIFAR-10 dataset, the neural network model is more accurate than the SVM

3 conclusion

In this paper, two machine learning methods, support vector machine (SVM) and TensorFlow-based neural network (NN), are used to classify the MNIST fashion dataset and the CIFAR-10 dataset, and their performance is compared. Through the experimental results, we found that:

SVM performed well on the MNIST dataset but worse on the CIFAR-10 dataset;

NNs performed well on both datasets, particularly on the CIFAR-10 dataset;

On the MNIST dataset, NN performs slightly better than SVM, but the difference between the two is not large;

On the CIFAR-10 dataset, the performance of NN is significantly better than SVM.

In subsequent research, we can try the following methods to further improve the performance of the classifier:

Adjust more hyperparameters, such as learning rate, regularization parameters, etc.;

Try more complex neural network structures such as convolutional neural networks (CNNs);

Enrich datasets using data augmentation techniques such as random clipping, rotation, translation, etc.;

Try other machine learning methods such as decision trees, random forests, etc.

In conclusion, the results of this paper show that neural networks based on TensorFlow are a very effective machine learning method in image classification tasks and have great application prospects.

appendix

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from sklearn.metrics import classification_report, confusion_matrix
4 import tensorflow as tf
5 from tensorflow.keras.datasets import fashion_mnist, cifar10
6 from tensorflow.keras.utils import to_categorical
7
8
9 # Load the MNIST dataset
10 (x_train, y_train), (x_test, y_test) = fashion_mnist.load_data()
11
12 # Preprocess the data
13 x_train = x_train.astype('float32') / 255.
14 x_test = x_test.astype('float32') / 255.
15 x_train = x_train.reshape((len(x_train), np.prod(x_train.shape[1:])))
16 x_test = x_test.reshape((len(x_test), np.prod(x_test.shape[1:])))
17 y_train = to_categorical(y_train, num_classes=10)
18 y_test = to_categorical(y_test, num_classes=10)
19
20 # Create SVM-based classifiers
21 from sklearn.svm import SVC
22 svm_clf = SVC(kernel='linear', C=1, gamma='auto')
23 svm_clf.fit(x_train, y_train.argmax(axis=1))
24
25 # Evaluate the SVM classifier on the test set
26 svm_predictions = svm_clf.predict(x_test)
27 svm_accuracy = svm_clf.score(x_test, y_test.argmax(axis=1))
28 svm_report = classification_report(y_test.argmax(axis=1), svm_predictions)
29 svm_matrix = confusion_matrix(y_test.argmax(axis=1), svm_predictions)
30
31 # Create a neural network model based on TensorFlow
32 def create_model_mnist(hidden_units=64, learning_rate=0.001):
33     model = tf.keras.models.Sequential([
34         tf.keras.layers.Dense(hidden_units, activation='relu', input_shape=(784,)),
35         tf.keras.layers.Dense(hidden_units, activation='relu'),
36         tf.keras.layers.Dense(10, activation='softmax')
37     ])
38     optimizer = tf.keras.optimizers.Adam(lr=learning_rate)
39     model.compile(optimizer=optimizer, loss='categorical_crossentropy', metrics=['accuracy'])
40     return model
41
42 # Train a neural network model based on TensorFlow
43 tf_model = create_model_mnist(hidden_units=128, learning_rate=0.001)
44 history = tf_model.fit(x_train, y_train, epochs=20, batch_size=128, validation_data=(x_test, y_test))
45
46 # Evaluate the neural network model on the test set
47 tf_predictions = tf_model.predict(x_test)
48 tf_predictions_classes = np.argmax(tf_predictions, axis=1)
49 tf_accuracy = tf_model.evaluate(x_test, y_test)[1]
50 tf_report = classification_report(np.argmax(y_test, axis=1), tf_predictions_classes)
```

```

46 # Evaluate the neural network model on the test set
47 tf_predictions = tf_model.predict(x_test)
48 tf_predictions_classes = np.argmax(tf_predictions, axis=1)
49 tf_accuracy = tf_model.evaluate(x_test, y_test)[1]
50 tf_report = classification_report(np.argmax(y_test, axis=1), tf_predictions_classes)
51 tf_matrix = confusion_matrix(np.argmax(y_test, axis=1), tf_predictions_classes)
52
53 # Visualize the results
54 epochs = range(1, len(history.history['accuracy']) + 1)
55 plt.plot(epochs, history.history['accuracy'], 'bo', label='Training accuracy')
56 plt.plot(epochs, history.history['val_accuracy'], 'b', label='Validation accuracy')
57 plt.title('Training and validation accuracy')
58 plt.xlabel('Epochs')
59 plt.ylabel('Accuracy')
60 plt.legend()
61 plt.show()
62
63 print('SVM Model evaluation results: ')
64 print('Accuracy:', svm_accuracy)
65 print('Classification report:\n', svm_report)
66 print('Confusion matrix:\n', svm_matrix)
67
68 print('Neural network model evaluation results: ')
69 print('Accuracy:', tf_accuracy)
70 print('Classification report:\n', tf_report)
71 print('Confusion matrix:\n', tf_matrix)
72
73 #Load the Cifar10 dataset
74 (x_train, y_train), (x_test, y_test) = cifar10.load_data()
75
76 #Preprocess the data
77 x_train = x_train.astype('float32') / 255.
78 x_test = x_test.astype('float32') / 255.
79 y_train = to_categorical(y_train, num_classes=10)
80 y_test = to_categorical(y_test, num_classes=10)
81
82 #Train a neural network model based on TensorFlow
83
84 tf_model = create_model_cifar(hidden_units=256, learning_rate=0.001)
85 history = tf_model.fit(x_train, y_train, epochs=20, batch_size=128, validation_data=(x_test, y_test))
86
87 #Evaluate the neural network model on the test set
88 tf_predictions = tf_model.predict(x_test)
89 tf_predictions_classes = np.argmax(tf_predictions, axis=1)
90 tf_accuracy = tf_model.evaluate(x_test, y_test)[1]
91 tf_report = classification_report(np.argmax(y_test, axis=1), tf_predictions_classes)
92 tf_matrix = confusion_matrix(np.argmax(y_test, axis=1), tf_predictions_classes)
93

```

```

59 plt.ylabel('Accuracy')
60 plt.legend()
61 plt.show()
62
63 print('SVM Model evaluation results: ')
64 print('Accuracy:', svm_accuracy)
65 print('Classification report:\n', svm_report)
66 print('Confusion matrix:\n', svm_matrix)
67
68 print('Neural network model evaluation results: ')
69 print('Accuracy:', tf_accuracy)
70 print('Classification report:\n', tf_report)
71 print('Confusion matrix:\n', tf_matrix)
72
73 #Load the Cifar10 dataset
74 (x_train, y_train), (x_test, y_test) = cifar10.load_data()
75
76 #Preprocess the data
77 x_train = x_train.astype('float32') / 255.
78 x_test = x_test.astype('float32') / 255.
79 y_train = to_categorical(y_train, num_classes=10)
80 y_test = to_categorical(y_test, num_classes=10)
81
82 #Train a neural network model based on TensorFlow
83
84 tf_model = create_model_cifar(hidden_units=256, learning_rate=0.001)
85 history = tf_model.fit(x_train, y_train, epochs=20, batch_size=128, validation_data=(x_test, y_test))
86
87 #Evaluate the neural network model on the test set
88 tf_predictions = tf_model.predict(x_test)
89 tf_predictions_classes = np.argmax(tf_predictions, axis=1)
90 tf_accuracy = tf_model.evaluate(x_test, y_test)[1]
91 tf_report = classification_report(np.argmax(y_test, axis=1), tf_predictions_classes)
92 tf_matrix = confusion_matrix(np.argmax(y_test, axis=1), tf_predictions_classes)
93
94 #Visualize the results
95 epochs = range(1, len(history.history['accuracy']) + 1)
96 plt.plot(epochs, history.history['accuracy'], 'bo', label='Training accuracy')
97 plt.plot(epochs, history.history['val_accuracy'], 'b', label='Validation accuracy')
98 plt.title('Training and validation accuracy')
99 plt.xlabel('Epochs')
100 plt.ylabel('Accuracy')
101 plt.legend()
102 plt.show()
103
104 print('Neural network model evaluation results: ')
105 print('Accuracy:', tf_accuracy)
106 print('Classification report:\n', tf_report)
107 print('Confusion matrix:\n', tf_matrix)

```

FINAL GRADE

50/100

GENERAL COMMENTS

Instructor

Your report is structured but is quite brief and whilst you provide results and summarize conclusions, etc, You could have included much more detail.

Did you consider adjusting your models to improve the performance on the CIFAR10.

You have clearly investigated and researched relevant idea, concepts and methods and you implement these, BUT I can't see your ref section!!!!

Please include this!

Your limited mark reflects the lack of references and also the briefness of your discussions and reviews.

PAGE 1



Comment 1

Introduction



Comment 2

Use Capital letters, i.e.

Method

PAGE 2



Comment 3

Capitalise

GRADING FORM: SMU GENERIC_GJ

BO DONG

0

RESEARCH-INFORMED LITERATURE

KNOWLEDGE AND UNDERSTANDING

ANALYSIS

PRACTICAL APPLICATION AND DEP.

SKILLS FOR PROFESSIONAL PRAC.