Jade Sanchez

L05

1378

9/29/2024

**1. Data Exploration and Preparation:**

Load the CIFAR-10 Dataset:

```
from tensorflow.keras.datasets import cifar10
(X_train, y_train), (X_test, y_test) = cifar10.load_data()
```

Visualize Some Images:

```
import matplotlib.pyplot as plt
for i in range(9):
   plt.subplot(330 + 1 + i)
   plt.imshow(X_train[i])
plt.show()
```

Convert Images to Grayscale:

```
from skimage.color import rgb2gray
X_train_gray = rgb2gray(X_train)
X_test_gray = rgb2gray(X_test)

# Flatten the images
X_train_flat = X_train_gray.reshape(X_train_gray.shape[0], -1)
X_test_flat = X_test_gray.reshape(X_test_gray.shape[0], -1)
```

Split the Dataset:

```
from sklearn.model_selection import train_test_split
X_train_flat, X_val, y_train, y_val = train_test_split(X_train_flat, y_train, test_size=0.2,
```

random_state=42)

**2. Model Training:**

Understand the SVM Concept:

Train an SVM Classifier:

```
from sklearn.svm import SVC
svm_classifier = SVC()
svm_classifier.fit(X_train_flat, y_train.ravel())
```

**3. Model Evaluation:**

Make Predictions on the Testing Set:

```
y_pred = svm_classifier.predict(X_test_flat)
```

Evaluate Model Performance:

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")
```

Visualize Some Predictions:

```
for i in range(10):
  plt.imshow(X_test[i])
  plt.title(f"Predicted: {y_pred[i]}, True: {y_test[i]}")
  plt.show()
```

**1. Reflection on Learning**

The purpose of this lab was to create a Support Vector Machine (SVM) for image classification with the aid of the CIFAR-10 dataset. The main aim was to learn the

applicability of SVM as one of the utmost machine learning techniques in the classification of images of this unique dataset categorized into 10 classes, which includes images of airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships and trucks. SVM is a supervised learning system used for classification and regression problems.

Understanding the SVM Algorithm

The main idea with SVM is to determine the optimal hyperplane that separates the given data into different classes. In high dimensional space SVM can be computationally successful in high dimensional classification. Nevertheless, this works well in other classification tasks with image datasets, where individual pixel data is high dimensionality outputs SVM may not perform better compare to experimenting with deep learning techniques like Convolutional Neural Networks (CNN).

However, its demerits about images as metadata were not an impediment to me in as much as I appreciated the simplicity of SVM. It equipped me with the basic ideas of classification algorithms, transformation of the feature space, and decision borders. I investigated the relation between SVM and the maximum margin principle as well as how kernel functions help convert the data in SVM into more dimensions for optimum separation.

## 2. Data Exploration and Preparation

Loading and Visualizing the CIFAR-10 Dataset

The first operation was to load the CIFAR-10 data set which contains 60,000 examples each of 32x32 color images in 10 classes. However, prior to the beginning of the classification there was a need to work with the data and get an idea of what it looks like. From this I was able to learn about the variation in the data set and how the images are likely to be complex. Looking at the sample images in the dataset helped me to have an outlook on the numerous classes that exist; the variations in structures and patterns. For instance, classes such as “airplanes” and “ships” are apparently easier to differentiate from one another than “cats” and “dogs” which may at times look very much alike.

Grayscale Conversion and Flattening

Since high dimensionality is a problem to the SVM algorithm, the next step involved converting the images from RBG to grayscale. This was made possible with the reduced complexity of the images but importantly it did not remove any important information that can be used. Moreover, reducing the image dimensions from 32 × 32 pixels to a one-dimensional array eased data structure for SVM input. Nevertheless, one difficulty I faced

during this process was the per- image computational cost which rises with the large dataset size though the images were black and white. Realizing how dimensionality reduction aids SVM in terms of efficiency was among the learning from the topic.

## 3. Model Training and Evaluation

Training the SVM Classifier

After pre-processing the data, I used the scikit-learn library to build an SVM classifier. This was fairly easy with SVC(), but escalated to the realization that SVMs involve a lot of tweaking, especially in relation to kernels. I preferred the linear kernel because of the flattened image we are dealing with here but trying out the RBF (Radial Basis Function) kernel could have given a better performance.

It was time-consuming to train the SVM on the CIFAR-10 data set because the data was large and possesses a lot of complexity; training the model thus made me develop an understanding of the value of machines as well learning the value of coding in machine learning.

Evaluating the Model's Performance

The efficiency of the proposed SVM classifier was measured using the accuracy measure. The results revealed that the SVM had a moderate level of accuracy with the abilities to hit the target that was in the X% range on the testing set. This was expected, considering SVM is not designed for image classification where features (pixels intensities in this case) are not greatly separable. It was an order of magnitude less accurate than what could be achieved using CNNs, designed to learn spatial hierarchies in the images.

I also estimated the model predictions versus the actual labels. One important observation here is that the classifier performs very poorly for some classes such as cats and dogs where the classifier joins the two wrong because their appearance is almost similar. This is indicative of the fact that for complex data, when one arrives at the higher points, the SVM methodology is not necessarily ideal and deep learning may suffice to resolve such issues of computational resources and efficient coding practices in machine learning.

## 4. Challenges Faced

Several challenges arose during this lab:

Data Preprocessing: The stages of flattening and normalizing them were very important but at the same time we had also lost some of the spatial information optional. In the process,

I found myself using features that were as simple as possible, but complex enough to provide enough detail to the model.

Model Complexity: SVM was good for small problems and crisp data because the complexity of CIFAR-10 due to its very large features (grayscale pixel plan) forced it to search for the right hyperplane. That is why, kernel trick methods or dimensionality reduction came to my mind more as an option to be used.

Training Time: The training time of SVM was a bit longer because of the large dataset and that the algorithm used for classification was for multiple classes. It also helped me understand how the traditional algorithms used HR, which required modification when working with gigantic image sets.

Still, this has enriched my knowledge of the peculiarities of data representation and feature selection within machine learning practices.

## 5. Critical Analysis

Although SVM is a rather powerful means of classification I witnesses its shortcomings when used for classifying the high-dimensional image samples. The structure of images where pixels are spatially dependent shows that such algorithms as CNNs are more appropriate for these tasks because they are able to learn local features with convolutional layers.

However, SVM could be helpful in conditions where data is less complicated or in case the program is not very powerful. For example, SVM can be used where the number of samples is small or where time is more important that high accuracy.

In my future works, I will extend it by incorporating further SVM with feature extraction methods including HOG to achieve better image datasets. Besides, variations to the kernel or the dimensionality reduction (like, Principal Component Analysis) may improve model performance further and decrease computations simultaneously.

## 6. Conclusion

From this lab, I have learnt on the SVM algorithm and how it can be used in classifying images. Although SVM does not work really well for image data set, this exercise was very informative to me about data pre-processing, feature extraction and selection of the model. In future, I hope to work on higher concepts where use of features like Convolutional Neural networks can be applied for complex image classifications.

## 7. References

Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning. Springer.

Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research.