Comparison research on iot oriented image classification algorithms

Ke Du¹ and Kai-Yu Cai²

¹College of Computer, National University of Defense Technology, 410073ChangSha,Hunan,P.R.China
²College of Computer, National University of Defense Technology, 410073ChangSha,Hunan,P.R.China

Abstract. Image classification belongs to the machine learning and computer vision fields, it aims to recognize and classify objects in the image contents. How to apply image classification algorithms to large-scale data in the IoT framework is the focus of current research. Based on Anaconda, this article implements k-NN, SVM, Softmax and Neural Network algorithms by Python, performs data normalization, random search, HOG and color histogram feature extraction to enhance the algorithms, experiments on them in CIFAR-10 datasets, then conducts comparison from three aspects of training time, test time and classification accuracy. The experimental results show that: the vectorized implementation of the algorithms is more efficient than the loop implementation; the training time of k-NN is the shortest, SVM and Softmax spend more time, and the training time of Neural Network is the longest; The test time of SVM, Softmax and Neural Network are much shorter than that of k-NN; Neural Network gets the highest classification accuracy, SVM and Softmax get lower and approximate accuracies, and k-NN gets the lowest accuracy. The effects of three algorithm improvement methods are obvious.

1 Introduction

With the fast development of IoT and mobile Internet, the volumes of image data from different fields such as social networks and sensor networks are growing exponentially. How to process the large-scale image data and recognize objects from the image contents has become an important issue. Image classification is the way to handle this problem. In general, image classification algorithm is data-driven approaches. They usually get global descriptions of the images by manual featuring or learning methods, use the learned classifiers to determine whether there is a certain object in the image or not. Image classification algorithms have been applied to many fields, such as face recognition, car number recognition, image searching and so on.

Anaconda is an open source scientific computation and analysis platform released by Continuum Company. It installed many scientific computation libraries, such as NumPy, SciPy and Matplotlib. It also provides an interactive tool, Python Notebook, which combines experiment and document writing. Based on Anaconda, researchers and developers can use Python to implement and analysis different algorithms, and write document by markdown.

Image classification and object detection are active research areas in the computer vision field. After AlexNet[1] was proposed in 2012, those areas are developing rapidly. In ILSVRC 2015, ResNet[2] won by its 3.57% top5 error. In the same year, YOLO[3] provides a real-time object detection with a mAP of 63.4 and a FPS of 45. Based on Anaconda, this paper implements k-NN, SVM, Softmax and Neural Network algorithms and data normalization, random search, HOG and color histogram feature extraction algorithm improvement methods, and conducts comparison on them by the training time, test time and classification accuracy. Results show that the performance of the different algorithms vary a lot, improvement methods improve the classification accuracy. Algorithms should be deployed in the different parts of the IoT framework according to their performance.

2 Comparison on algorithms

2.1. k-NN algorithm

k-NN is short for k-Nearest Neighbour. Instead of finding the single closest image in the training set as Nearest Neighbor algorithm, k-NN finds the top k closest images, and makes them vote on the label of the test image. In particular, when k = 1, it recover to the Nearest Neighbor algorithm. Intuitively, higher values of k have a smoothing effect that makes the classifier more resistant to outliers is improved.

k-NN treats image as vector, using L2 distance, which has the geometric interpretation of computing the euclidean distance between two vectors, to compare the difference between two images. The distance formula takes the form:
The training process of k-NN is to store all training images in the memory, then the test process is to compare test image with stored images by L2 norm, finds the top k closest images, and makes them vote on the label of the test image.

\[ d_2(I_1, I_2) = \sqrt{\sum_{p} (I_{1p} - I_{2p})^2} \quad (1) \]

2.2. SVM algorithm

Support Vector Machine algorithm uses score function:

\[ f(x, W, b) = Wx + b (2) \]

to compute the class scores of image for different classes, then uses loss function:

\[ L_i = \sum_{j\neq i} \max(0, w_j^T x_i - w_i^T x_i + \Delta) \quad (3) \]

To compute the loss value, SVM loss function is often called the hinge loss. It measures how consistent the predictions on training data are with the ground truth labels. The loss value of entire dataset is composed of data loss and regularization loss:

\[ L = \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{m} \max(0, f(x_i; \theta) - f(x_j; \theta) + \Delta) + \lambda \sum_{i=1}^{m} w_i^2 \quad (4) \]

In the training process, SVM uses gradient descent method to find the optimal weights and biases, which make the loss value minimum, since making good predictions on the training set is equivalent to minimizing the loss. In the test process, algorithm uses the optimal weights and biases to classify test images.

2.3. Softmax algorithm

The score function of Softmax algorithm is the same with SVM algorithm. However, it replaces the hinge loss with cross-entropy loss for its loss function:

\[ L_i = -\log \left( \frac{e^{y_i f_j}}{\sum_j e^{y_j f_j}} \right) \quad (5) \]

The training and test processes of Softmax are similar with SVM. However, the core idea of Softmax is different. Unlike SVM, it is never fully happy with the scores it produces: the correct class could always have a higher probability and the incorrect classes always a lower probability and the loss would always get better.

2.4. Neural network algorithm

Neural Network models are often organized into distinct layers of neurons. For regular neural networks, the most common layer type is the fully-connected layer in which neurons between two adjacent layers are fully pairwise connected, but neurons within a single layer share no connections. With an appropriate loss function on the neuron's output, it can turn a single neuron into a linear classifier.

The training process consists of two parts: the forward pass and back propagation. In forward pass, it performs matrix multiplication and activation function to obtain image classification score by the current weights and biases. Take the two-layer neural network model in our experiment for example, the forward pass formula is:

\[ s = W_1 \text{max}(0, W_2 x) \quad (6) \]

and are the matrix combined weight and biases, is the class score.

The backpropagation is a process of computing gradients of expressions through recursive application of chain rule in networks. Algorithm then uses gradient signal to perform parameters update, till the value of loss function get small enough. In the test process, algorithm obtains class score by forward pass. It has been proved that Neural Network algorithm can approximate any continuous function[4].

3 Algorithm improvement methods

3.1. Mean subtraction and normalization

Mean subtraction and normalization are both data preprocessing methods. Mean subtraction involves subtracting the mean across every individual feature in the data. It has the geometric interpretation of centering the cloud of data around the origin along every dimension.

Data normalization refers to normalize the data dimensions, so that they are of approximately the same scale. It is the average for all data samples, the is the standard deviation of all the sample data.

Since in image processing, the relative scales of pixels are already approximately equal (in range from 0 to 255), so it is not strictly necessary to perform this normalization step.

3.2 Random search

Random search is a hyperparameter optimization method. Since hyperparameter tuning can be generally described as:

\[ \lambda^{(k)} = \arg\min_{\lambda \in \mathcal{A}} \mathbb{E}_{x \sim \mathcal{G}_x} \left[ \ell(x; \mathcal{A}_{\lambda}(x | \text{train})) \right] \quad (7) \]

represents hyperparameters, is a learning algorithm base on hyperparameter, is the loss function, samples are from a natural (grand truth) distribution, is the training set, a finite set of samples from. What we really need in practice is a way to choose so as to minimize generalization error. It has been proved that randomly chosen trials are more efficient for hyper-parameter optimization than trials on grid search[5].

3.3 HOG and colour histogram

HOG is short for Histogram of Oriented Gradients, this feature extraction method was proposed in 2005 by Dalai for pedestrian detection[6]. In our experiments, for each
image we will compute HOG as well as a colour histogram using the hue channel in HSV colour space. We form our final feature vector for each image by concatenating the HOG and colour histogram feature vectors.

HOG captures the texture of the image while ignoring colour information, and the colour histogram captures the colour of the input image while ignoring texture. As a result, we expect that using both together ought to work better than using either alone.

4 Experiments

4.1. Environments, datasets and measurements

The hardware environments are as follow: the CPU is Intel Core i5 2.8GHz, the capacity of memory is 8GB, and the capacity of disk is 500GB. The software environments are as follow: the operating system is OS X El Capitan, we use Python 2.7.11 and installed Anaconda 2.5.0 for Python 2.7 version. Additionally, since Anaconda installed many thirdparty libraries by default, we list some main libraries and their versions: numpy 1.10.4, scipy 0.17.0, matplotlib 1.5.1, pandas 0.17.1, ipython 4.0.3, ipython notebook 4.0.4.

We choose CIFAR-10 as our experimental dataset. CIFAR-10 is a subset of Tiny Images database, which contains 10 categories. It contains 60000 images, 50000 for training, 10000 for test. Its image size is 32 x 32 pixels. Although the image size is small, the amount of images for each category is large, so it is very suitable for training complex models such as deep learning model. Moreover, small image size means the request for computational capacity is not too high, which ease the pressure on our machine.

In experiments, algorithms are compared by three evaluation standards: classification accuracy, training time and test time.

The classification accuracy is defined as:

\[ \text{accuracy} = \frac{N_{y_i=y_j}}{M_{y_{test}}} \]

It evaluates the classification performance of the algorithms. The is the amount of category labels corresponding to test set. After algorithms finished classifying the test images, the number of prediction label which are consistent with real category is .

Use training time and test time to measure the computational efficiency of the algorithms, is used to measure how long it takes algorithm to learn optimal parameters in the training process. is used to measure how long it takes algorithm to classify the test image data in the test process.

4.2 Results

Since all algorithms need hyperparameter optimization, we divide CIFAR-10 dataset into three splits: the training set, validation set and test set. The training set contains 49000 images, the validation set and test set both contain 1000 images. k-NN algorithm divides training set into 5 subsets equally, uses cross validation to perform hyperparameter optimization. Other algorithms perform hyperparameter optimization on the validation set, and their initial hyper parameter tuning policy is grid search. The iteration numbers of SVM, Softmax and Neural Network algorithms are set to 1500. Other hyperparameters of algorithms are obtained in the tuning process.

4.2.1 Comparison on two kinds of implementations

Numpy library provides APIs that include efficient matrix computational operations and broadcasting mechanism, it can make algorithms implemented in vectorized computation. Compared with implementation by multiple loops, vectorized implementation improved the computation efficiency. The L2 distance computations in k-NN algorithm, and the loss functions in SVM, Softmax and Neural Network algorithms can all be implemented in loops and vectorized computations. In experiment, we use time and test time to compare the cooperation efficiency of the two kinds of implementation. We count the time in seconds. Details are in the following table:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>N_{train}</th>
<th>N_{test}</th>
</tr>
</thead>
<tbody>
<tr>
<td>k-NN</td>
<td>4288.6</td>
<td>9.1</td>
</tr>
<tr>
<td>SVM</td>
<td>0.47</td>
<td>0.044</td>
</tr>
<tr>
<td>Softmax</td>
<td>0.63</td>
<td>0.059</td>
</tr>
<tr>
<td>NN</td>
<td>0.60</td>
<td>0.065</td>
</tr>
</tbody>
</table>

Since the training process of k-NN algorithm simply stores the training data into memory with no matrix computational operation, it is not strange that k-NN takes so little time to finish training process. However, most computational operations of k-NN algorithm are in its test process. Unlike k-NN, Most computational operations of SVM, Softmax and Neural Network algorithms are in their training process for loss values and gradients calculation, and in their test process, the computational cost is low.

According to the table, the loop implementation form of k-NN algorithm costs nearly 420 times than the vectorized implementation. Moreover, the loop implementation forms of SVM, Softmax and Neural Network algorithms cost about 10 times than the loop implementation forms.

Therefore, it is obvious that the Numpy API can greatly improve the efficiency of algorithm. In the follow experiments, the algorithms are implemented in vectorized forms by default.
4.2.2 Comparison on algorithms

The computational efficiency and classification accuracy of the four algorithms are compared in the following table:

Table 2. Algorithms Comparison.

<table>
<thead>
<tr>
<th></th>
<th>Time (s)</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>k-NN</td>
<td>0.000066</td>
<td>32.7%</td>
</tr>
<tr>
<td>SVM</td>
<td>5.4</td>
<td>31.8%</td>
</tr>
<tr>
<td>Softmax</td>
<td>4.8</td>
<td>31.8%</td>
</tr>
<tr>
<td>NN</td>
<td>59.6</td>
<td>35.5%</td>
</tr>
</tbody>
</table>

The time of k-NN algorithm is much shorter than its. On the contrary, the time of SVM, Softmax and Neural Network algorithms are much shorter than their. Note that the time of Neural Network algorithm is about 10 times than of SVM and Softmax algorithms.

For now, the classification accuracies of four algorithms are on the same level, and the performance of Neural Network algorithm is slightly better. However, there was no significant difference among them. The reason is that the SVM, Softmax and Neural Network algorithms did not perform data preprocessing, feature extraction and hyperparameter tuning, so the potential capacities of these algorithms have not been further explored.

4.2.3 Data preprocessing method

In the experiment, input image data normalization was performed. The accuracy improvements of four algorithms are in the following table:

Table 3. Effect of Data Normalization.

<table>
<thead>
<tr>
<th></th>
<th>Grid Search</th>
<th>Random Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>k-NN</td>
<td>32.7%</td>
<td>32.7%</td>
</tr>
<tr>
<td>SVM</td>
<td>33.6%</td>
<td>33.2%</td>
</tr>
<tr>
<td>Softmax</td>
<td>34.5%</td>
<td>35.3%</td>
</tr>
<tr>
<td>NN</td>
<td>52.3%</td>
<td>53.3%</td>
</tr>
</tbody>
</table>

Through random search, the accuracy of Softmax improved by 0.8%, and the accuracy of Neural Network algorithm improved by 1%. It shows that the random search did find better parameters in the parameter space. However, the classification accuracy of SVM algorithm descends by 0.4%. It can be explain that in the grid search, the parameters are already close to the optimal parameters.

Corresponding to the accuracies, the hyperparameters of the algorithms are as follow: the learning rate of SVM is 5.0e-05, regularization strength is 9.0e+04, its iterations are 1500. The learning rate of Softmax is 4.866548e-07, regularization strength is 2.393255e+04, its iterations are 1500. The learning rate of Neural Network is 8.944667e-04, regularization strength is 9.493759e-01, its iterations are 1500, the number of layers are 2, and the number of neurons in the hidden layer are 360.

In general, the random search method is simple to implement and more efficient than grid search in hyperparameter optimization process.

4.2.5 Feature extraction method

We form our final feature vector for each image by concatenating the HOG and colour histogram feature vectors. Since k-NN algorithm simply compares the raw pixels, we did not perform the feature extraction process on it. For SVM, Softmax and Neural Network algorithms, take the feature vectors as the input data. The results show accuracy improvements in the following table:
Using feature vector as input data, the classification accuracy of SVM, Softmax and Neural Network all improved. The accuracy of SVM improved by 9.2%, the accuracy of Softmax improved by 6.5%, the accuracy of Neural Network algorithm improved by 6.1%. The capacity of our combined feature extraction method has been proved.

### 4.3. Conclusion

Based on Anaconda, this paper implemented k-NN, SVM, Softmax and Neural Network four classical algorithms by Python, then compared and analysed the performance of the algorithms from two respects: the image classification accuracy and the computational resource cost. According to the experimental results we can get the following conclusion:

- The vectorized implementations of algorithms are much more efficient than the loop implementations.
- Based on vectorized implementation, the training time length of k-NN algorithm is the shortest than that of any other algorithms. The training time length of Neural Network algorithm is the longest. The training time lengths of SVM and Softmax are approximately the same. However, the test time lengths of SVM, Softmax and Neural Network algorithms are much shorter than that of k-NN algorithm, which means the algorithms are suitable for the practical applications in the IoT framework except k-NN algorithm.
- The classification accuracy of Neural Network algorithm is the highest of all, which means it should be deployed to handle the most complex classification task. The classification accuracies of SVM and Softmax are lower and approximately the same, which means they should be deployed to handle some simple but real-time required classification tasks. And the classification accuracy of k-NN algorithm is the lowest.
- The data normalization, random search and feature extraction are all improvement tricks of the algorithm. After applying these tricks, the classification accuracy of algorithms have been improved by different levels. The random search gives a little improvement, the effect of data normalization is better, and the effect of HOG and colour histogram feature extraction is the best of all three tricks. All the tricks should be applied to algorithms in practical application.
- k-NN algorithm is not suitable for any practical applications in the IoT framework, since its low classification accuracy and high computational cost. The performances of SVM and Softmax algorithms are approximately the same, both on their classification accuracy and computational cost. They can be deployed to some real-time required tasks to handle some specific objects classification. The classification accuracy of Neural Network algorithm is the highest of all the four algorithms, which means it can handle some complex classification tasks. However, the computational cost of the Neural Network algorithm is huge, therefore, it should be deployed to the server and provide classification API to the remote devices.

In the future, we will implement the algorithms by using the TensorFlow framework, which is provided by Google, and deploy algorithms to different parts of the IoT system, measure the performance of the whole system and the algorithms respectively.

### References