Cifar-10 Image Classification with Statistical Learning Models

Liang Fang Student ID: XXX

Email: XXXX@sjtu.edu.cn

ABSTRACT

In this project, I implemented several statistical learning models in Python, e.g. K-Nearest Neighbors(KNN) and Linear Discriminant Analysis(LDA). Data preprocessing, of course, is necessary in statistical learning. With model selection analysis, we can see that SVM performs the best on Cifar-10 image classification task. Lastly, I compared statistical learning methods with convolutional neural networks based methods to illustrate the superiority of deep learning in image processing tasks.

Keywords: Statistical Learning, Image Classification, Model Selection and Inference

1. INTRODUCTION

This report shows the procedure and details of my work. Firstly, I choose HOG as the feature extracting method which can be used to reduce the input variable of model training. Secondly, 3 models in our Statistical Learning course are implemented in Python, and a CNN model as well. Model parameters tuning is essential yet labor-intensive work to achieve best accuracy on given dataset. Lastly, the evaluation analysis shows that SVM model is the preference model in cifar-10 image classification task.

2. DATA PREPROCESSING

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. In order to achieve high accuracy in image classification, we need to project the dataset into a lower dimension feature space since the original input space is too large while the sample size is relatively small. Hence, we ought to extract some representative features of every 32x32 image, which can be used as the input variable of our statistical learning model.

The histogram of oriented gradients (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection. HOG became widespread since being used by French researchers in their supplementary work on HOG descriptors at the Conference on Computer Vision and Pattern Recognition (CVPR) in 2005.[1] The technique counts occurrences of gradient orientation in localized portions of an image. This method is similar to that of edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy.

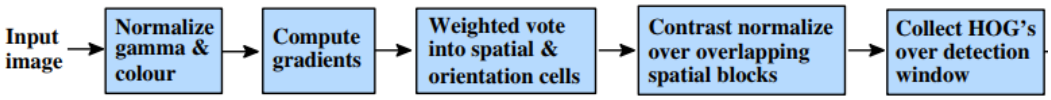

Figure 1. Computation procedures of HOG

It's reasonable to use HOG descriptors of every image as the input variables of my chosen statistical learning models. HOG descriptor can be obtained through procedures shown in Figure 1. It should be **noted** that the parameters, such as cell size, block size and orientations, will influence the output of HOG descriptor which will then influence the performance of estimator.

3. STATISTICAL LEARNING MODELS

I chose 3 representative models in this course to complete the image classification work. They are K Nearest Neighbors (KNN), Linear Discriminant Analysis (LDA) and Surpport Vector Machine (SVM). The theory of each model will be given in the following 3 subsections.

3.1 K-Nearest Neighbors

Neighbors-based classification is a type of instance-based learning or non-generalizing learning: it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the nearest neighbors of each point: a query point is assigned the data class which has the most representatives within the nearest neighbors of the point. Specifically, the k-nearest neighbor fit for $\hat{Y}$ is defined as follows:

$$\hat{Y}(x) = \frac{1}{k}\Sigma_{x_i \in N_k(x)} y_i \tag{1}$$

where $N_k(x)$ is the neighborhood of x defined by the k closest points x_i in the training sample. Closeness implies a metric,which we assume is Euclidean distance. So, in words, we find the k observations with x_i closest to x in input space, and average their response.

3.2 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis(QDA) are two classic classifiers, with, as their names suggest, a linear and a quadratic decision surface, respectively. Both LDA and QDA can be derived from simple probabilistic models which model the class conditional distribution of the data $P(X|y = k)$ for each class k. Predictions can then be obtained by using Bayes' rule:

$$P(y = k|X) = \frac{P(X|y = k)P(y = k)}{P(X)} = \frac{P(X|y = k)P(y = k)}{\Sigma_l P(X|y = l)P(y = l)} \tag{2}$$

and we select the class k which maximizes this conditional probability.

More specifically, for linear and quadratic discriminant analysis, $P(X|y)$ is modelled as a multivariate Gaussian distribution with density:

$$p(X|y=k) = \frac{1}{(2\pi)^n|\Sigma_k|^{1/2}} exp(-\frac{1}{2}(X-\mu_k)^t\Sigma_k^{-1}(X-\mu_k)) \tag{3}$$

To use this model as a classifier, we just need to estimate from the training data the class priors $P(y=k)$(by the proportion of the instances of class k), the class means μ_k and the covariance matrices(by the empirical smaple class covariance matrices).

In the case of LDA, the Gaussians for each class are assumed to share the same covariance matrix:$\Sigma_k = \Sigma$ for all k. This leads to a linear decision surfaces between, as can be seen by comparing the log-probability ratios $log[P(y=k|X)/P(y=l|X)]$:

$$(\mu_k - \mu_l)\Sigma^{-1}X = \frac{1}{2}(\mu_k^t\Sigma^{-1}\mu_k - \mu_l^t\Sigma^{-1}\mu_l) \tag{4}$$

In the case of QDA, there are no assumptions on the covariance matrices Σ_k of the Gaussians, leading to quadratic decision surfaces.

3.3 Surpport Vector Machine

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. The optimization problem of SVM can be wrote as follows:

$$\min_{\beta,\beta_0}\|\beta\| \quad s.t. \quad \begin{cases} y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \quad \forall i, \\ \xi_i \geq 0, \Sigma\xi_i \leq constant. \end{cases} \tag{5}$$

Computationally it's convenient to re-express the problem in the equivalent form:

$$\min_{\beta,\beta_0} \|\beta\|^2 + C\Sigma_{i=1}^N\xi_i, \quad s.t. \quad \xi_i \geq 0, y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \quad \forall i, \tag{6}$$

where C is the cost parameter, which can be tuned during the computation procedure.

3.4 CNN Based Methods

While traditional multilayer perceptron (MLP) models were successfully used for image recognition, due to the full connectivity between nodes they suffer from the curse of dimensionality, and thus do not scale well to higher resolution images. For example, in CIFAR-10, images are only of size 32x32x3 (32 wide, 32 high, 3 color channels), so a single fully connected neuron in a first hidden layer of a regular neural network would have 32*32*3 = 3,072 weights. A 200x200

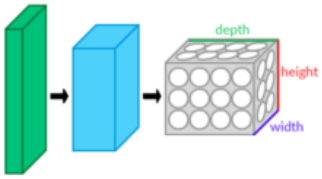

Figure 2. Layer of CNN

image, however, would lead to neurons that have 200*200*3 = 120,000 weights. Also, such network architecture does not take into account the spatial structure of data, treating input pixels which are far apart in the same way as pixels that are close together. Thus, full connectivity of neurons is wasteful for the purposes such as image recognition that are dominated by spatially local input patterns.

Convolutional neural networks are biologically inspired variants of multilayer perceptrons, designed to emulate the behaviour of a visual cortex. These models mitigate the challenges posed by the MLP architecture by exploiting the strong spatially local correlation present in natural images. As opposed to MLPs, CNNs have the following distinguishing features:

1. 3D volumes of neurons. As is shown in Figure 2, the layers of a CNN have neurons arranged in 3 dimensions: width, height and depth.

2. Local connectivity: following the concept of receptive fields, CNNs exploit spatial locality by enforcing a local connectivity pattern between neurons of adjacent layers.

3. Shared weights: In CNNs, each filter is replicated across the entire visual field. These replicated units share the same parameterization (weight vector and bias) and form a feature map.

Together, these properties allow CNNs to achieve better generalization on vision problems. Weight sharing dramatically reduces the number of free parameters learned, thus lowering the memory requirements for running the network and allowing the training of larger, more powerful networks.

4. IMPLEMENTATION AND EVALUATION

Three statistical learning models and a CNN model (ResNet18) are implemented in this project. Evaluation of the models will be given by accuracy or prediction error in this part, while taking HOG parameters into consideration.

4.1 Implementation Details

All code are written in Python, including HOG features extraction, model training and model performance test. HOG descriptors computation is based on skimage library. Cross validation methods (K-Fold) and model performance analysis (based on matplotlib) are also implemented. In order to achieve a high accuracy rate, a CNN model is trained, based on PyTorch, during this semester.

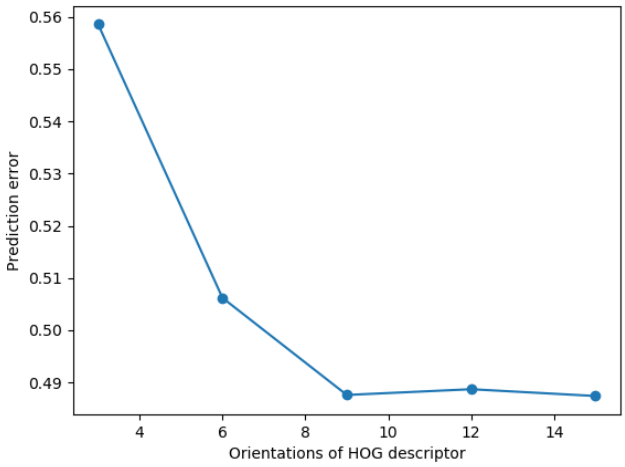

Figure 3. LDA model prediction error changes along with orientations of HOG descriptor

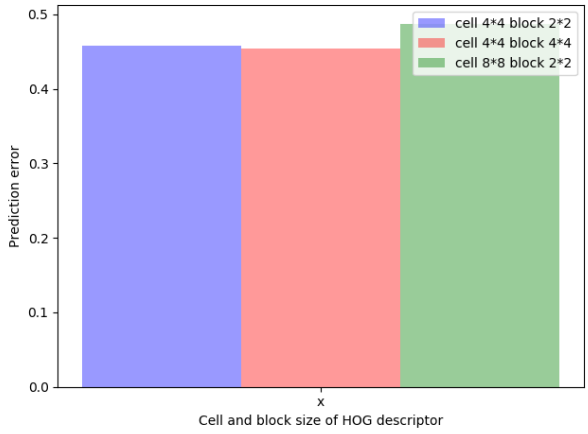

Figure 4. LDA model prediction error changes along with cell and block size of HOG descriptor

4.2 LDA Model

let's first see how the parameter of HOG descriptor influence the output of our LDA model. As mentioned before, HOG descriptors parameters consists of cell size, block size and orientations. Figure 3 shows how the orientation effect the prediction error of LDA model.

We can see that when orientations equals 9 model may achieve the minimum prediction error, which is about 48%.

Figure 4 shows the relationship between cell and block size of hog and the prediction error with our model.

The best performance of LDA model is that achieves about 46% prediction error under suitable HOG descriptor parameters.

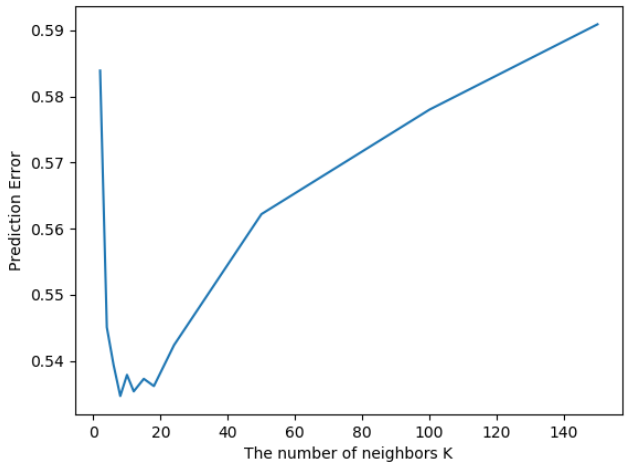

Figure 5. KNN model prediction error changes along with k value

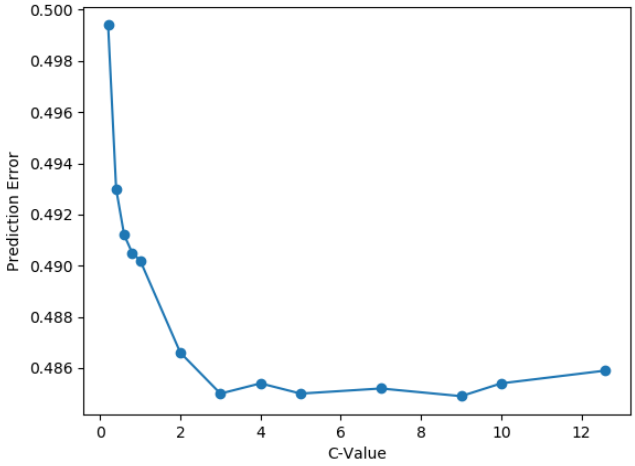

Figure 6. SVM model prediction error changes along with C-value

4.3 KNN Model

The results on KNN model is shown in Figure 5. We can see that with the increase of K, the prediction error decrease and then goes up again. The best K value in our case is 8, and the corresponding prediction error is 53.47%.

4.4 SVM Model

The tuning parameter of SVM model is the cost parameter C which is mentioned in 3.3. The accuracy of SVM model changes along with the parameter C, which is shown in Figure 6. The minimum prediction error is about 48.5% when C takes value 3.0.

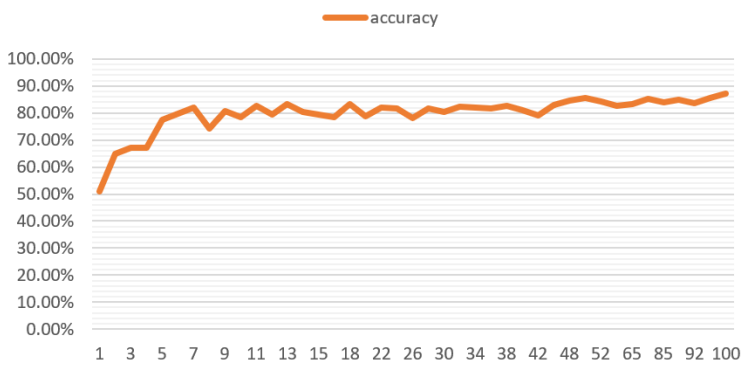

Figure 7. CNN model accuracy rate increases along with training epochs

4.5 CNN Model

CNN model is intuitively superior to traditional statistical learning model in computer vision field. Based on PyTorch, I trained a CNN model, called ResNet18, making use of the computing resource of my laboratory. CNN model can achieve up to 87% accuracy rate after training 100 epochs (every epoch takes about 2.5 hours), as shown in Figure 7.

5. CONCLUSION

Among KNN, LDA and SVM, the best accuracy could be 51.34%, which is achieved by SVM, while the KNN model got the lowest accuracy rate. By comaring the experiments' result we can see that it has a long gap to achieve the accuracy of CNN model using statistical learning models.

REFERENCES

[1] Dalal, N. and Triggs, B., "Histograms of oriented gradients for human detection," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* **IEEE Computer Society**, 886–893 (2005).