Kernel Methods

Authors:
Yunzhi Bai, Jie Zheng

The report just briefly describes the overall operation. The python code for this project is available for execution under the name of *codeToSubmit (*for more details, please read *README)*

The original images are color images stored in an RGB format, characterized by 3072 pixel values.

I. Preprocessing

The preprocessing stage extracts the gradient information of each target image. We used a simplified version of **HOG (Histogram of oriented gradients)** as a feature descriptor. Following are the main steps:

1. Reshape each original RGB image from (1,3072) to (32, 32, 3) representing 32 x 32 pixels, 3 channels;
2. Convert color images to grayscale ones using the weighted average 0.3R + 0.59G + 0.11B. Apply a Gamma correction with gamma equals to ½. Each image has one gray channel instead of three color channels;
3. Compute the gradients for the grayscale image on both direction x and y, with the following filter kernels $[-1,0,1]$, $[-1,0,1]^T$. The x- and y-directional gradients are noted gx and gy;
4. Compute the magnitude and orientation matrices from the gradients gx gy. The orientation angles are between 0 and 360 degree;
5. Divide the magnitude and orientation matrices into small cells (e.g. 6x6 pixels) using sliding windows along x and y of step=1;
6. For each cell compute a histogram of gradient directions for the pixels within the cell (e.g. nbins=9, the orientation angles are segmented into 9 bins of interval 360°/9=40°) . Each pixel within the cell casts a weighted vote equals to its magnitude.

After the stage of preprocessing, each image has a feature descriptor P of size 5625.

II. Kernel design

Here we implemented a **Multiple Kernel** strategy.

$$K(X,Y) = a * K1(X,Y) + b * K2(X,Y) + \sum ci * Ki(X,Y)$$

with a=4*1e-7 , b=1.0, HOG kernel K1(X, Y) = <P_X, P_Y>, Laplacian kernel K2(X, Y) = exp(-gamma ||X-Y||).
ci = 0 for Ki={ Gaussian kernel, Linear kernel, Polynomial kernel}.

III. Classification models

Support Vector Machines Classifiers The support vector machines (SVM) uses the strategy of **One-vs-One** [1] for multi-class classifications. It consists in constructing one SVM for each pair of classes. Thus, for our image classification problem with c=10 classes, c(c-1)/2=45 SVMs are trained to distinguish the samples of one class from the samples of another class. Following are the main steps:

1. Build a binary SVM classifier;
2. Add functions allowing a list of multiple kernels to be used in a classifier with their coefficients;
3. Write a general One-vs-One algorithm which enables any binary classifier to compute multi-class classification. Firstly, separate the samples according to their classes. Secondly, train a binary SVM classifier for each pair of classes A and B, so as to obtain, for each sample, its local posterior probabilities of belonging to class A or class B. Finally, express the global posterior probabilities as functions of the local posterior probabilities using (1). Now each sample has a list of 10 probabilities representing its probability of belonging to class 1 to 10;

$$\hat{P}(\omega_j \mid x) = \frac{\prod_{j'=1, j'\neq j}^{c} \hat{P}\left(\omega_j \mid f_{j,j'}(x)\right)}{\sum_{j''=1}^{c} \prod_{j'=1, j'\neq j''}^{c} \hat{P}\left(\omega_{j''} \mid f_{j'',j'}(x)\right)}. \qquad (1)$$

4. Implementation of the One-vs-One algorithm on SVM classifier for multi-class classifications.

IV. Result

Our best result is obtained by combining image feature descriptor, multiple kernels (0.0000004 *HOG + 1.0*Laplacian), and SVM One-vs-One strategy.

Besides, we experimented with several other approaches.

V. Other approaches

K-Nearest Neighbors An instance is classified by a majority vote of its k-nearest neighbor instances. The measurement is a distance metrics based on the weighted pairwise distances in RKHS.

One-layer Convolutional Kernel Network [2] The convolutional kernel network (CKN) is a new type of unsupervised convolutional neural network that is trained to approximate the kernel map. We initialized W with Gaussian random noise, implemented the Gaussian kernel of one layer.

VI. References

[1] Jonathan Milgram, Mohamed Cheriet, Robert Sabourin ”One Against One” or ”One Against All”: Which One is Better for Handwriting Recognition with SVMs? Section 4. *HAL Id: inria-00103955*

[2] Julien Mairal, Piotr Koniusz, Zaid Harchaoui, and Cordelia Schmid "Convolutional Kernel Networks" *arXiv:1406.3332*