Data Challenge / Kernel Methods for Machine Learning

Chia-Man Hung, Zhengying Liu
Master Data Science
firstname.lastname@polytechnique.edu

Mario Ynocente Castro
Master MVA
mario.ynocente-castro@polytechnique.edu

Abstract

Multi-class classification is a classical problem in machine learning and much progress has been observed in the literature in recent years. In this challenge, we experiment with various kernel methods to perform image classification. This short report aims at summarizing our approaches, focusing on feature extraction and classification.

1. Introduction

In this data challenge on image classification, we are given 5000 classified images as training data and 2000 images as test data. Each image is represented by a 32 (height) x 32 (width) x 3 (color) vector of values between -1 and 1. There are 10 classes. The performance is evaluated by the classification accuracy on the test data. A public leader board is available and the score is calculated upon approximately 50% of the test data. The final results will be based on the other 50%. The goal of this data challenge is to learn how to implement machine learning algorithms from scratch. For this reason, external machine learning libraries, as well as computer vision libraries are forbidden.

2. Feature Extraction

2.1. Local feature descriptor

We first introduce three classical algorithms in computer vision to detect and describe local features in images.

Histograms of Oriented Gradients (HOG) We follow the method introduced in [2], which consists of dividing the image into regions, for each region accumulating a local 1-D histogram of local gradient directions over the pixels of the region. A histogram corresponds to a local descriptor.

Scale-Invariant Feature Transform (SIFT) As described in [3], this method is composed of the following steps. First we build a pyramid of images convolved with Gaussian filters at different scales, and then the differences of successive Gaussian-blurred images. Potential keypoints are taken from the local extrema of the difference-of-Gaussians pyramid. Second, we adjust the keypoints to more accurate positions by interpolation using the quadratic Taylor expansion of the Difference-of-Gaussians function taking the candidate keypoints of the previous step as origins. At the same time, we discard keypoints that are either unstable, low-contrast or on edges. Third, we assign an orientation to each keypoint by computing a histogram of oriented gradients in the neighboring pixels. The orientation of the highest peak is assigned. If other peaks are within 80% of the highest peak, new keypoints at the same position with those orientations will be created. Last, we build a local feature descriptor for each keypoint by creating 4 histograms, taking the orientation of the previous step into account; each histogram is built upon the Gaussian-weighted oriented gradients of neighboring pixels. By concatenating 4 histograms, we obtain a local feature descriptor.

Kernel descriptors As stated in [1], this method is based on the insight that the inner product of the orientation histograms is a particular match kernel over image patches. We compute gradient and color match kernels for images. The kernel view of orientation histograms provides a simple and unified way to turn pixels into low-level descriptors.

2.2. Global feature descriptor

Besides simply concatenating local feature descriptors, we list two other ways below to construct a global feature descriptor to represent an image.

Bag of words We apply the K-means algorithm to cluster local feature descriptors. We consider each cluster as a bag and each local feature descriptor as a word. The number of words in each bag forms a vector.

Fisher Vector The idea behind Fisher vector is to measure the similarity between features using a Fisher kernel. Given a likelihood function u_λ with parameter λ, the score function of a given sample X is given by

$$G_\lambda^X = \nabla_\lambda \log u_\lambda(X). \tag{1}$$

The Fisher Information Matrix (FIM) is defined as

$$F_\lambda = E_{x\sim u_\lambda}\left[\nabla_\lambda \log u_\lambda(x) \nabla_\lambda \log u_\lambda(x)^T\right]. \tag{2}$$

The Fisher kernel is defined as

$$K(X,Y) = (G_\lambda^X)^T F_\lambda^{-1} G_\lambda^Y. \tag{3}$$

As the FIM is positive semi-definite, it can be decomposed as $F_\lambda^{-1} = L_\lambda^T L_\lambda$. Then, the Fisher kernel can be rewritten as a dot product between Fisher Vectors

$$\mathcal{G}_\lambda^X = L_\lambda G_\lambda^X. \tag{4}$$

To apply this to images, we consider $X = \{x_t, t = 1..T\}$ the set of T i.i.d. D-dimensional local descriptors.

$$G_\lambda^X = \frac{1}{T}\Sigma_{t=1}^T \nabla_\lambda \log u_\lambda(x_t). \tag{5}$$

$u_\lambda(x) = \Sigma_{i=1}^{K} w_i u_i(x)$ is a Gaussian Mixture Model (GMM) with parameters $\lambda = \{w_i, \mu_i, \Sigma_i, i = 1..N\}$ trained on a set of local descriptors, which can be regarded as a probabilistic visual vocabulary. We apply the K-means algorithm to initiate the centers in GMM and then perform the Expectation-Maximization (EM) algorithm for training.

Compared to the bag-of-words model, Fisher Vector contains higher-order statistics, up to order 2. More theoretical and implementation details can be found in [5].

3. Classification

Once we represent the image by local or global feature descriptors, we classify them using a classifier. In the case of Support Vector Machine (SVM), it is combined with a kernel.

3.1. Classifiers

We have the choice of classifiers among **cross entropy classifier**, a multi-class classifier with cross entropy loss, **kernel SVM one versus one classifier** and **kernel SVM one versus all classifier**. The kernel SVMs are built upon a kernel SVM binary classifier, which solves the dual problem by applying the Sequential Minimal Optimization (SMO) [4], an algorithm for solving the quadratic programming problem that arises in the training of an SVM.

3.2. Kernels

Below is a list of kernels we use in kernel descriptors for feature extraction, kernel SVMs for classification, or kernel PCA for dimension reduction.

- **Linear kernel** $K(x, y) = x^T y$
- **Gaussian kernel** $K(x, y) = \exp(\frac{-\|x-y\|_2^2}{2\sigma^2})$
- **Gaussian kernel for angle**

$$K(x, y) = \exp(\frac{-(\Sigma_i(\sin x_i - \sin y_i)^2 + \Sigma_i(\cos x_i - \cos y_i)^2)}{2\sigma^2})$$

- **Histogram intersection kernel** $K(x, y) = \Sigma_i \min(x_i^\beta, y_i^\beta)$
- **Laplacian RBF kernel** $K(x, y) = \exp(\frac{-\Sigma_i|x_i-y_i|}{\sigma^2})$
- **Sublinear RBF kernel** $K(x, y) = \exp(\frac{-\Sigma_i|x_i-y_i|^{0.5}}{\sigma^2})$
- **Hellinger kernel** $K(x, y) = \Sigma_i \sqrt{x_i y_i}$

4. Experiments and Results

Our implementation [1] is done in python. We only use the following libraries: numpy, scipy, random, pandas, matplotlib, os.

Data visualization We plot the images and identify the ten classes: (from 0 to 9) aircraft, car, bird, cat, deer, dog, frog, horse, boat, truck.

[1]The source code can be found at `https://github.com/zhengying-liu/Kernel-Methods-Data-Challenge`.

Computation complexity reduction A standard Principal Component Analysis (PCA) for dimension reduction is done to the local descriptors before handing them to compute the Fisher Vector. A kernel PCA with a Gaussian kernel is done to the global descriptors before performing the classification. In Fisher Vector, we only choose part of the local descriptors to feed the GMM. All of the above make our experiments computationally feasible. To give a rough idea, it takes approximately eight hours to extract SIFT descriptors for all images, fifteen minutes to extract HOG descriptors, four hours to perform one iteration of the EM algorithm in the GMM if considering all local descriptors, six minutes if considering only one local descriptor per image.

Parameter tuning In SIFT, since the images are relatively small, we reduce the contrast threshold to keep a reasonable amount of keypoints. In the detection of local extrema, we do not compare a pixel to the neighboring 26 pixels (in a cube) in the Difference-of-Gaussians pyramid, but rather the 6 neighboring ones (left, right, front, back, top, down). In the classification, we perform a 5-fold cross validation to choose classifiers, kernels, and the parameters of the kernels.

Ensemble learning Our final submission is done by voting among three predictions — HOG + SVM one versus one classifier with Laplacian RBF kernel ($\sigma = 3.4$), HOG + Fisher Vector + linear SVM one versus one classifier, and SIFT + Fisher Vector + linear SVM one versus one classifier. In case where all three predictions are different, we pick one at random. This gives a public score 0.691 (ranked 3rd) and a private score 0.697 (ranked 2nd).

5. Conclusion

In this data challenge, we have implemented several state-of-the-art feature extraction techniques, namely HOG, SIFT, kernel descriptors, and Fisher, and experimented with various kernel SVMs. To represent one image by a 1-D vector, we had the choice among raw data, the concatenation of local descriptors, and the construction of global descriptor on local descriptors. We performed multi-class prediction on test data by fitting a kernel SVM with training data. Some computation complexity reduction techniques are done prior to the classification training and parameters are tuned by cross validation. HOG worked well with SVM one versus one classifier with Laplacian RBF kernel in our use case. Fisher kernel is already data-adaptive so it is combined with a linear SVM. A final voting among several predictions further increased our accuracy.

References

[1] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. In *Advances in neural information processing systems*, pages 244–252, 2010.

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[3] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[4] J. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. 1998.

[5] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek. Image classification with the fisher vector: Theory and practice. *International journal of computer vision*, 105(3):222–245, 2013.