Non-Neighboring Rectangular Feature Selection using Particle Swarm Optimization

Akinori Hidaka
University of Tsukuba
hidaka.akinori($\alpha\tau$)aist.go.jp

Takio Kurita
National Institute of Advanced Industrial Science and Technology (AIST)

Abstract

Recently, Viola proposed a rectangular features (RFs) based classifier with high accuracy and rapid processing speed for object detection tasks [9]. In this paper, we propose Non-Neighboring RFs (NNRFs) as an extension of RFs, and a Particle Swarm Optimization (PSO) based feature selection algorithm for NNRFs. NNRFs are the pairs of arbitrary rectangular sub-regions in images, giving us huge number of candidate NNRFs for feature selection (e.g. 1.3 *billion NNRFs in* 19×19 *pixel image). We show that PSO can select the powerful subset of NNRFs efficiently from the various candidates, and the classification accuracy is improved with the same computational cost as compared with that of Viola's method.*

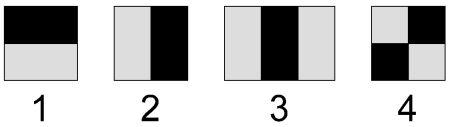

Figure 1. Configuration of elemental rectangles of usual RF.

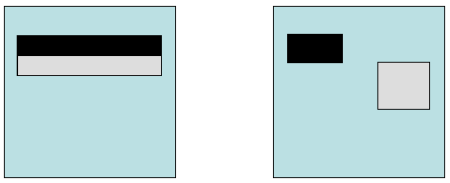

Figure 2. Examples of usual RF (left) and proposed NNRF (right).

1 Introduction

A rectangular feature (RF) based classifier proposed by Viola and Jones [9] is known to provide high classification accuracy and rapid processing performance for object detection tasks. In their work, a powerful subset of RFs is selected from many candidate RFs using Adaboost [1]. Adaboost is one of the most powerful algorithm among existing ensemble learning methods and it combines several base classifiers to construct a more powerful classifier.

It is known that the performance of ensemble classifiers are closely related to the diversity of components in the ensemble [6]. Although diverse (various) candidate RFs can yield powerful ensembles [8], the large number of candidates will cause also the increase in computation time for feature selection. Therefore, the training of the RF based classifiers are usually performed by using only limited (up to hundreds of thousands) candidate RFs.

To obtain the rich candidate RFs and achieve efficient feature selection in a consistent way, we propose Non-Neighboring RFs (NNRFs), and a NNRF selection algorithm based on Particle Swarm Optimization (PSO) [2][3]. NNRFs are the pairs of arbitrary rectangular sub-regions in images. Because there are about $36,000$ rectangular sub-regions in 19×19 pixel image, we can use about 1.3 billion ($36,000^2$) NNRFs potentially.

Considering the coordinates of NNRF's rectangular region to be parameters, PSO [5] performs effective search over the parameter space. In our experiments, we show PSO that can select the powerful subset of NNRFs from those abundant candidates efficiently, and a more powerful detector is obtained at the same computational cost as compared with that of Viola-Jones' method.

2 Rectangular Features Based Boosted Detector

2.1 Rectangular Features

For the object detection problem, Viola and Jones proposed rectangular features (RFs) which indicate dif-

978-1-4244-2175-6/08/$25.00 ©2008 IEEE

- Input N samples and its labels $\{I_i, y_i\}_{i=1}^N$.
- Initialize samples weights:
  if $y_i = 1$ then $w_i = \frac{1}{2p}$, otherwise $w_i = \frac{1}{2q}$.
  (p, q: # of face and non-face images, respectively)
- for $t = 1, \cdots, T$
  - Normalize weights: $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{i=1}^N w_{t,i}}$.
  - Optimize base-classifiers $\{b_c\}_{c=1}^C$:
    $\{b_t, e_t\} = Opt(\{b_c\}_{c=1}^C, \{I_i, w_i, y_i\}_{i=1}^N)$
  - Compute $\alpha_t = \log((1 - e_t)/e_t)$
  - Update samples weights:
    $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot \delta(y_i - b_t(I_i))]$.
    ($\delta(x) = 1$ (if $x = 0$), 0 (otherwise).)
- Final classification function is
  $H(I) = \begin{cases} 1 & \text{if } \sum_{t=1}^T \alpha_t b_t(I) \geq \Theta \sum_{t=1}^T \alpha_t. \\ 0 & \text{otherwise.} \end{cases}$
  (Θ: Threshold)

Figure 3. Common Adaboost algorithm. The function "Opt" is shown in Fig. 4.

ference of brightness of local rectangular regions, and they treated such features as simple classifiers [9]. RFs consists of from two to four elemental (small) neighboring rectangles which are of the same size. RFs have therefore five degrees of freedom, four for the position (x_s, y_s), (x_e, y_e) and one for the configuration $z \in \{1, 2, 3, 4\}$ of the elemental rectangles (Fig. 1).

The classification function of RFs can be written as follow:

$$b(I) = \begin{cases} 1 & \text{if } pf(I) > p\theta \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where I shows an input image, $f(I)$ implies a feature value of a rectangular feature at I, and $p \in \{1, -1\}$ and $\theta \in R$ are parameters determined by training (see [9]).

2.2 Adaboost for RF Selection

Adaboost [1] is the ensemble learning method that trains multiple base-classifiers and assembles these to create a more powerful classifier. The algorithms of Adaboost is shown in Fig. 3.

Viola and Jones used RFs as base-classifiers for the face detection [9]. By considering the five degrees of freedom in RF as kinds of base-classifier's parameters, Adaboost can be used for RF selection. For convenience, we call this algorithm Adaboost.VJ. There are two optimization phases in Adaboost.VJ:

1) optimization of $\{p, \theta\}$ for each candidate RFs,

2) selection of the best one from those candidates.

Details of optimization algorithm of Adaboost.VJ are discussed in [9].

2.3 Non-Neighboring RFs (NNRFs)

As can be seen in Fig. 1, usual RFs have the constraint that all elemental (small) rectangles are of the same size and are neighbors. In this paper, we propose NNRF by eliminating such a constraint from RF (Fig. 2).

There are many elemental rectangles in each image (e.g. $36,000$ rectangles in 19×19 pixel). For simplicity, the number of elemental rectangles used in NNRFs is fixed to two. This gives NNRF eight parameters given by $(x_s^1, y_s^1), (x_e^1, y_e^1)$ and $(x_s^2, y_s^2), (x_e^2, y_e^2)$ which are the diagonal apex of elemental rectangles r^1 and r^2, respectively. Like RF, the feature value of NNRF is the difference of the averaged intensity of elemental rectangular regions.

3 Particle Swarm Optimization for Fast Local Feature Selection

3.1 Particle Swarm Optimization (PSO)

PSO [5] is a swarm intelligence model describing behaviors of swarms of creatures, like ants looking for food. It is an optimization algorithm in multi-particle system, like genetic algorithm [4]. In PSO, hundreds or thousands of particles search the optimum while communicating with other particles. Each particle p has two state vectors: position x_τ^p and velocity v_τ^p. These state vectors are simply updated as follows:

$$\begin{aligned} v_{\tau+1}^p &= w v_\tau^p + c_s r_s (h_s^p - x_\tau^p) + c_g r_g (h_g - x_\tau^p), \\ x_{\tau+1}^p &= x_\tau^p + v_\tau^p, \end{aligned} \quad (2)$$

where w is the inertia term, c_s and c_g are parameters given manually, r_s and r_g are random values between 0 and 1, h_s^p and h_g show the best position in the history of p-th particle and all particles, respectively. All P particles search the best position up to T_{it} turns.

3.2 Fast NNRF Selection by PSO

As described in Section 2.3, NNRF has eight parameters; position $(x_s^1, y_s^1), (x_e^1, y_e^1)$ and $(x_s^2, y_s^2), (x_e^2, y_e^2)$ of elemental rectangles. The position vectors of PSO can now be constructed as $x_\tau^p = (x_s^1, y_s^1, x_e^1, y_e^1, x_s^2, y_s^2, x_e^2, y_e^2)_\tau^p$. Each position in the 8-dimensional space (i.e. a certain NNRF) is evaluated by the classification error rate for training

	Viola-Jones' RF	NNRF (PSO300-30)	NNRF (PSO-2000-25)
Accuracy ($T_{it} = 1$)	86.06%	84.32%	89.12%
Accuracy ($T_{it} = 50$)	95.61%	95.79%	96.39%
Accuracy ($T_{it} = 200$)	98.01%	97.95%	98.27%
# of Optimized RFs	$53,130$	$9,000$	$50,000$

Table 1. Classification accuracy for the test set and computational cost of each method.

- Receive arguments $\{\{b_c\}_{c=1}^{C}, \{I_i, w_i, y_i\}_{i=1}^{N}\}$.
- Set $c_s = c_g = 2, w_{min} = 0.2, w = w_{max} = 1.2$.
- Set state vector $x_\tau^p \in R^d$ and $v_\tau^p \in R^d$ at random.
- For $\tau = 1, \cdots, T_{it}$,
  - For $p = 1, \cdots, P$,
    - Set parameters $r_s, r_g \in [0, 1]$ at random.
    - Fit a classifier $b(x_\tau^p; I)$ to the training samples using weights w_i.
    - Compute $e_\tau^p = \frac{\sum_{i=1}^{N} w_i \cdot \delta(y_i - b(x_\tau^p; I_i))}{\sum_{i=1}^{N} w_i}$.
    - Update h_s^p or h_g if lower e_τ^p is obtained.
    - Update v_τ^p and x_τ^p by using Eq. (2).
  - Update momentum: $w \leftarrow w_{max} - \frac{\tau}{T_{it}}(w_{max} - w_{min})$
- Return $\{h_g, e_{h_g}\}$

Figure 4. Optimization function for Adaboost.PSO. Each position vector $x_\tau^p = (x_s^1, y_s^1, x_e^1, y_e^1, x_s^2, y_s^2, x_e^2, y_e^2)_\tau^p$ shows a certain NNRF where (x_s^k, y_s^k) and (x_e^k, y_e^k) are the diagonal apex of elemental rectangles r^k ($k = 1, 2$). h_s^p and h_g show the best position in searching histry of p-th particle and all particles, respectively. e_{h_g} implies weighted error of h_g for training samples.

samples. As a result, the best NNRF can be searched as the best point in the 8-D parameters space. The optimization algorithm is shown in Figs. 3 and 4.

The computational cost of Adaboost.PSO depends on the two hyperparameters: the number of particles P and the number of iterations T_{it}. $P \times T_{it}$ classifiers will be optimized in the each iteration of Adaboost.

4 Experiments

In this paper, we used MIT CBCL face database [7] for our experiments. The database consists of $2,901$ facial and $28,121$ background 19×19 pixel images. We used about $53,000$ candidate RFs in each image for Viola-Jones' method. There are about $36,000$ rectangular sub-regions in each image, so we can use about 1.3 billion ($36,000^2$) candidate NNRFs for our method. We divided all images into training and test set; $2,000$ face and $4,000$ background images are used for training, and the remaining images are used for testing. Three pairs of training and test sets were generated at random. Results presented in this section are the averaged values calculated from these three pairs. We performed two classification experiments described as follows.

First, we compared the hyperparameters of PSO for the NNRF selection. Denote Adaboost.PSO with 10 particles and 100 iterations as PSO10-100 for convenience. We then performed and compared the following conditions: PSO10-100, PSO100-10, PSO300-30, PSO1000-50, and PSO2000-25. As shown in Fig. 5, hyperparameters with much computational cost yielded good results, the best being PSO2000-25.

Next, we compared Viola's RF based classifier and our NNRF based classifier (Fig. 5 and Tab. 1). The second row of Tab. 1 is showing the potential of NNRFs: although the optimal RF obtained from $53,000$ candidates had 86.06% accuracy, our (sub-)optimal NNRF improved the accuracy to 89.12% by using just $50,000$ candidates out of 1.3 billion available NNRFs. As shown in Fig. 5 and Tab. 1, NNRFs selected by PSO2000-25 marked better performance than usual RFs with slightly less computation cost. Furthermore, NNRFs selected by PSO300-30 caught up with RFs near the 50-th step with about 1/6 of computational cost.

5 Conclusions

In this paper, we proposed Non-Neighboring Rectangular Features (NNRFs) and its fast feature selection algorithm based on Particle Swarm Optimization (PSO). In our experiments, we showed that the more powerful features than the best one of usual RFs are certainly included in our 1.3 billion of NNRFs. We also showed that the proposed feature selection based on PSO can efficiently find such powerful features from those extremely abundant candidates. As a result, we could obtain more accurate detector with the same computation cost as compared with the previous learning method.

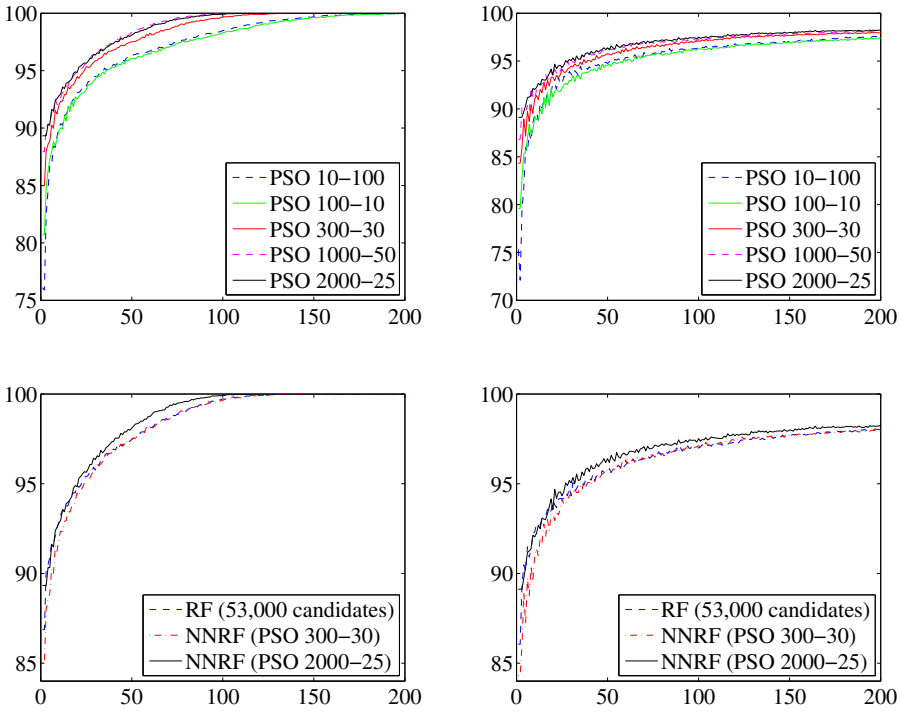

Figure 5. (Top) Comparison of PSO parameters for NNRFs training. (Bottom) Comparison of NNRF and RF. Horizontal axis implies the step of Adaboost.PSO (the number of selected RFs/NNRFs), and vertical axis shows classification accuracy (%) for the training (left) and test (right) samples, respectively.

Acknowledgement

This work was partily supported by KAKENHI (19500163).

References

[1] Y. Freund, R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," Journal of Computer and System Sciences, vol. 55, 1, pp. 119–139, 1997

[2] A. Hidaka, T. Kurita, "Fast Training Algorithm by Particle Swarm Optimization for Rectangular Feature Based Boosted Detector," In Proc. of FCV2008, pp.88-93, Jan. 2008.

[3] A. Hidaka and T. Kurita, "Fast Training Algorithm by Particle Swarm Optimization and Random Candidate Selection for Rectangular Feature Based Boosted Detector," In Proc. of IJCNN2008 on WCCI2008, Jun. 2008, *Accepted*.

[4] J. Holland, "Adaptation in Natural and Artificial Systems," University of Michigan Press, 1975.

[5] J. Kennedy, and R. Eberhart, "Particle Swarm Optimization," In Proc. of IEEE int. conf. on neural networks Vol. IV, pp. 1942-1948. IEEE service center, Piscataway, NJ, 1995.

[6] L. Kuncheva, "Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy," Machine Learning, vol. 51, 2, pp. 181–207, 2003

[7] MIT Center For Biological and Computation Learning, "CBCL Face Database #1," http://www.ai.mit.edu/projects/cbcl.

[8] R. Lienhart and J. Maydt, "An Extended Set of Haar-like Features for Rapid Object Detection," In Proc. of IEEE ICIP 2002.

[9] P. Viola and M. Jones, "Robust real time object detection," In Proc. of IEEE ICCV Workshop on Statistical and Computational Theories of Vision, July 2001.