Pattern Recognition Letters 69 (2016) 62–71

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Multi-feature Hashing Tracking☆

Chao Ma*, Chuancai Liu, Furong Peng, Jia Liu

School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

ARTICLE INFO

Article history:
Received 29 December 2014
Available online 23 October 2015

Keywords:
Hashing
Multi-feature
Tracking
Fusion

ABSTRACT

Visual tracking is a popular topic in computer vision due to its importance in surveillance, action recognition and event detection. The feature to describe the visual object is an essential element of the tracking model. But there does not exist such kind of feature to handle all situations. Based on this fact, researchers propose the fusion technique to capture robust representation of the object by integrating different features. However, general fusion methods are hard to be applied to tracking algorithm due to the reason of processing speed and online update. To solve this problem, an effective fusion-based hashing method is proposed. The hashing method fuses different features to generate compact binary feature, which could be efficiently processed. In addition, 2D manner and online update model are used to improve the tracker's performance. Experimental results demonstrate that our tracker out-performs the state-of-the-art trackers in tested sequences.

1. Introduction

Visual tracking, a hot topic in computer vision, has been widely used in many real world applications such as surveillance, action recognition and event detection. In the last decades, researchers have developed many successful trackers. They typically contain three main parts: appearance model, motion model (also called sampling model) and update model. As an important basis of good tracker, appearance model has been well studied in the literatures [2,4,6,9,21,23,24,28,32,33,35].

In these papers, researchers either propose a discriminative feature or develop an effective way of organizing features to construct the appearance model. Their experiments on various challenging sequences demonstrate the effectiveness of appearance-based trackers. Among these methods, there exists a data-driven manner for feature extraction, which assumes that the compact feature is learned from training data. This kind of method owns two advantages: (1) The learned feature can be suit for the current tracking situation; (2) The dimension of the feature is always lower than that of the original signals. Based on this idea, [23] propose the tracker IVT, which uses PCA to extract compact feature from original image samples. However, PCA cannot take the latent semantic information of training samples into consideration, which limits the tracker's performance. Following this work, [29] propose a tracking algorithm by using tensor manner to encode the structure information. The tracker achieves better performance than IVT but suffers the bottleneck in complex situations for the same reason as IVT. By the usage of semantic information, [16] use LDA to construct appearance model that makes the tracking algorithm more robust. However, the tracking results appear to be instable under severe situations due to the limitation of single-view. Recently, the tracker IDCT is proposed by [32]. Unlike other data-driven methods of feature extraction, the cosine function is used to reconstruct the original signal, and coefficients of wavelets can be treated as robust features to represent the object. To achieve robust tracking results, a work for combining different features is proposed by [36]. The tracker trains one classifier for each feature and integrates these classifiers by variance ratio to get final position. However, using variance ratio to integrate the feature is not an effective manner. With the development of hashing technique, [17] uses decision tree model to learn the binary code for visual tracking. The tracker is effective, but the update model is inefficient due to batch manner.

In this paper, to integrate multi-feature more effectively and keep efficiency of tracking algorithm, we propose **M**ulti-**f**eature **H**ashing **T**racking (MFHT). The proposed multi-feature hashing model aims to combine different features to get the compact binary code of the object in a natural way. The binary code can be processed efficiently while the original information is retained as much as possible. Moreover, 2D manner and incremental updating model are used to improve the training and updating speed during tracking. Experimental results demonstrate the importance of the techniques used in the algorithm and the superior of MFHT compared to the start-of-the-art methods.

2. Related work

The learning-based hashing technique has achieved good performance in visual application, since it is efficient and accurate in

☆ This paper has been recommended for acceptance by R. Davies.

* Corresponding author. Tel.: +86-159-5056-7833.
E-mail address: njustmachao@163.com (C. Ma).

http://dx.doi.org/10.1016/j.patrec.2015.09.019
0167-8655/

data organization. Compared to data-independent hashing methods [7,13,14,22], learning-based hashing [3,12,15,20,27,31] is able to learn more compact and discriminative binary code. Spectral hashing [31] utilizes the Laplacian graph decomposition method to learn hashing functions and achieves good performance. [20] propose hashing with graph, which encodes manifold structure of data into binary code. Local structure has also been used in other hashing methods [12,25]. Besides utilizing the data distribution, an alternative way is to fuse different features. Based on this idea, [3] propose data fusion hashing, which embeds the input samples from two feature spaces into the Hamming space by supervised learning. Cross view hashing [15] maximises the correlation of different features in single view and cross views, and the learned binary code appears to be more discriminative. [19] use Kernel CCA and neighborhood structure preserving method to fuse different features and get robust hashing code. With the success of these methods, the hashing technique is utilized to solve large-scale problem in more and more visual applications [26,34].

Due to the success of hashing methods, it is a good idea to apply them to visual tracking. However, hashing methods mentioned above are designed for off-line learning task and are inefficient for tracking problems. So in this paper, we attempt to propose an online hashing framework. In real application, training hashing functions are still time-consuming when conducted on a large number of data. In this work, an online learning and fusing framework for tracking, Multi-feature Hashing Tracking (MFHT) is proposed. In this framework, different features are fused to get robust binary code of each sample and the incremental learning model is used to update the tracker. Moreover, we use 2D technique to improve the low speed of training time. The main contributions of this paper are summarized as below:

1. We propose a multi-feature hashing method to extract compact and robust binary feature of object by embedding samples from different feature spaces.
2. 2D manner and efficient update model are proposed to apply the proposed hashing method to tracking problem.
3. Experiments are implemented to evaluate the performance of our tracking algorithm compared with other state-of-the-art trackers and the influence of different settings on our tracker.

3. Multi-feature Hashing

3.1. Notations

In this section, we introduce our multi-feature hashing method. Without loss of generality, we take two views as an example to introduce our method. Given M centered training samples $\{\mathbf{A}_i^1, \mathbf{A}_i^2, y_i\}_{i=1}^M$, where $\mathbf{A}_i^1, \mathbf{A}_i^2 \in \mathbb{R}^{D\times D}$ are two different features (views) of i-th sample and $y_i \in \{+1, -1\}$ denotes the label of i-th sample representing positive (foreground) or negative (background), we categorize them into pairwise training data $\{\mathbf{A}_i^1, \mathbf{A}_i^2, \mathbf{A}_j^1, \mathbf{A}_j^2, l_{ij}\}$, where $l_{ij} = 1$ denoting same-class pair ($y_i = y_j$) and $l_{ij} = -1$ denoting different-class pair ($y_i \neq y_j$). The goal is to learn the projection matrices $\mathbf{W}^1 = [\mathbf{w}_1^1, \ldots, \mathbf{w}_E^1] \in \mathbb{R}^{D\times E}$ and $\mathbf{W}^2 = [\mathbf{w}_1^2, \ldots, \mathbf{w}_E^2] \in \mathbb{R}^{D\times E}$ for two views, which give same bits for same-class pair $l_{ij} = 1$ and different bits for different-class pair $l_{ij} = -1$ as much as possible.

3.2. Problem formulation

To achieve the goal, the hashing method can be divided into two kinds of scenarios: single view and cross view. For single view, we can get optimal hash functions by minimizing:

$$\sum_{(i,j)} \mathbf{L}_{ij} \|\mathbf{h}^k(\mathbf{A}_i^k) - \mathbf{h}^k(\mathbf{A}_j^k)\|_H, \tag{1}$$

where $\mathbf{h}^k(\mathbf{A}) = sgn\left(\mathbf{W}^{k\top}\mathbf{A} + \mathbf{B}\right)$ denotes the hashing function for k-th view, $\|\cdot\|_H$ denotes Hamming distance and $\mathbf{L}$ is the matrix form of l_{ij}. As we know, Hamming distance could be treated as inner product $\langle\cdot,\cdot\rangle$ between binary codes. Then Eq. (1) could be rewritten as:

$$\sum_{+} \left\langle \mathbf{h}^k(\mathbf{A}_i^k), \mathbf{h}^k(\mathbf{A}_j^k) \right\rangle - \sum_{-} \left\langle \mathbf{h}^k(\mathbf{A}_i^k), \mathbf{h}^k(\mathbf{A}_j^k) \right\rangle, \tag{2}$$

where $+$ denotes $l_{ij} = 1$ and $-$ denotes $l_{ij} = -1$. In order to solve the above NP hard problem, we use a simple relaxation method to Eq. (2) by replacing sign function with its signed magnitude:

$$J(\mathbf{W}^k) = \sum_e \left\{ \sum_{+} \mathbf{w}_e^{k\top} \mathbf{A}_i^{k\top} \mathbf{A}_j^k \mathbf{w}_e^k - \sum_{-} \mathbf{w}_e^{k\top} \mathbf{A}_i^{k\top} \mathbf{A}_j^k \mathbf{w}_e^k \right\}.$$
$$\mathbf{W}^k = [\mathbf{w}_1^k, \mathbf{w}_2^k, \ldots, \mathbf{w}_E^k]. \tag{3}$$

Then, the objective function in two different views can be rewritten as follows:

$$J(\mathbf{W}^1) = \sum_e \left\{ \sum_{+} \mathbf{w}_e^{1\top} \mathbf{A}_i^{1\top} \mathbf{A}_j^1 \mathbf{w}_e^1 - \sum_{-} \mathbf{w}_e^{1\top} \mathbf{A}_i^{1\top} \mathbf{A}_j^1 \mathbf{w}_e^1 \right\},$$
$$J(\mathbf{W}^2) = \sum_e \left\{ \sum_{+} \mathbf{w}_e^{2\top} \mathbf{A}_i^{2\top} \mathbf{A}_j^2 \mathbf{w}_e^2 - \sum_{-} \mathbf{w}_e^{2\top} \mathbf{A}_i^{2\top} \mathbf{A}_j^2 \mathbf{w}_e^2 \right\}. \tag{4}$$

From above functions, we could see that the two projection matrices can be learned separately and the binary codes of samples can be obtained by Eq. (11).

Besides the single view, we aim to enable cross-view similarity search through binary codes, and we want the hashing functions to map similar objects to similar codewords over all the views. Intuitively, the learned binary code should preserve the consistency information of different views. By the pairwise defined in Section 3.1, the cross-view objective function can be written as:

$$\min \sum_{(i,j)} \mathbf{L}_{ij} \|\mathbf{h}^1(\mathbf{A}_i^1) - \mathbf{h}^2(\mathbf{A}_j^2)\|_H, \tag{5}$$

With the similar relaxation operation, Eq. (5)could become:

$$\begin{aligned} &\sum_e \sum_{(i,j)} \mathbf{L}_{ij} \mathbf{w}_e^{1\top} \mathbf{A}_i^{1\top} \mathbf{A}_j^2 \mathbf{w}_e^2 \\ &= \sum_e \left\{ \sum_{+} \mathbf{w}_e^{1\top} \mathbf{A}_i^{1\top} \mathbf{A}_j^2 \mathbf{w}_e^2 - \sum_{-} \mathbf{w}_e^{1\top} \mathbf{A}_i^{1\top} \mathbf{A}_j^2 \mathbf{w}_e^2 \right\} \\ &= \sum_e \mathbf{w}_e^{1\top} \left\{ \sum_{+} \mathbf{A}_i^{1\top} \mathbf{A}_j^2 - \sum_{-} \mathbf{A}_i^{1\top} \mathbf{A}_j^2 \right\} \mathbf{w}_e^2. \\ &\quad s.t. \quad \mathbf{w}_e^{1\top} \mathbf{w}_e^1 = 1, \mathbf{w}_e^{2\top} \mathbf{w}_e^2 = 1. \end{aligned} \tag{6}$$

We rewrite Eq. (6) into matrix form:

$$\mathbf{tr}\left\{ \mathbf{W}^{1\top} \left\{ \sum_{+} \mathbf{A}_i^{1\top} \mathbf{A}_j^2 - \sum_{-} \mathbf{A}_i^{1\top} \mathbf{A}_j^2 \right\} \mathbf{W}^2 \right\}. \tag{7}$$

To solve optimal directions $\mathbf{W}^1$ and $\mathbf{W}^2$ of cross-view objective function, Singular Value Decomposition (SVD) can be done on matrix $\sum_+ \mathbf{A}_i^{1\top} \mathbf{A}_i^2 - \sum_- \mathbf{A}_i^{1\top} \mathbf{A}_i^2$.

Now, combining Eqs. (7) and (4) together, we could get final objective function:

$$\begin{aligned} J(\mathbf{W}^k) = &\sum_{k=1}^{2} \mathbf{tr}\left\{ \mathbf{W}^{k\top} \left\{ \sum_{+} \mathbf{A}_i^{k\top} \mathbf{A}_j^k - \sum_{-} \mathbf{A}_i^{k\top} \mathbf{A}_j^k \right\} \mathbf{W}^k \right\} \\ &+ \eta \sum_{k=1}^{2} \sum_{l>k} \mathbf{tr}\left(\mathbf{W}^{k\top} \left\{ \sum_{+} \mathbf{A}_i^{k\top} \mathbf{A}_j^l - \sum_{-} \mathbf{A}_i^{k\top} \mathbf{A}_j^l \right\} \mathbf{W}^l \right), \\ &s.t. \quad \mathbf{W}^{k\top} \mathbf{W}^k = \mathbf{I}, k = 1, 2. \end{aligned} \tag{8}$$

where η is trade-off coefficient to balance single view term and cross view term.