DETECTING ANOMALY IN VIDEOS FROM TRAJECTORY SIMILARITY ANALYSIS

Yue Zhou, Shuicheng Yan, Thomas S. Huang

Beckman Institute, Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign, Urbana, IL, USA, 61801
{yuezhou,scyan,huang}@ifp.uiuc.edu

ABSTRACT

Trajectories of moving objects provide crucial clues for video event analysis especially in surveillance applications. In this paper, we study the problem of detecting anomalous events by analyzing the motion trajectories in videos. Different trajectories of the same category may have varying relative velocities, in addition to the variations and noises in location samples; hence the core of the problem is to provide a robust and accurate function for measuring the similarities of trajectory pairs. We propose a novel learning based algorithm for estimating the similarities of the multi-dimensional sequence pairs, and then an anomaly detection framework is presented to detect anomalous motion trajectories in surveillance videos. Our proposed algorithm offers several advantages over the traditional algorithms for dealing with the trajectories of moving objects. First, the similarity measurement is robust against data imperfections such as noise, algorithmic error and etc. Second, we introduce a learning algorithm which allows the similarity function to be adapted to the particular problems being solved. Third, the proposed anomaly detection framework is fully automatic and without parametric distribution assumption on the data. The experiments on both outdoor and indoor surveillance videos validate the effectiveness of our proposed framework in detecting anomalous trajectories.

1. INTRODUCTION

Detecting anomalous patterns from video sequence is useful for many applications such as surveillance, novelty extraction, automatic inspection and etc. The identification of anomalies can lead to the discovery of truly novel information from the video, e.g. suspicious object movements and/or rare trajectory patterns. With recent advances in visual tracking, it becomes feasible to obtain the trace of moving objects with reasonable accuracy from surveillance videos. This could be a very important source of features for event detection.

Previous research on abnormal activity detection can be roughly divided into two categories: parametric approaches and non-parametric approaches. For the first category [1], the explicit parametric model for normal and/or abnormal activities is constructed based on the features extracted from the observed data. Visual features such as position, speed, appearance and etc. can be extracted from object detection and tracking algorithms [2][3][4]. These models are either rule-based or obtained from supervised machine learning techniques - most by probabilistic graphical models [5][6][7]. The second category, on the other hand, does not explicitly predefine the models for the normal events, instead learns the normal and abnormal patterns from the statistical properties of the observed data. In [8], joint co-occurrence statistics of object trajectories over a codebook are accumulated and hierarchical classification algorithm is applied to identify activities. [9] uses feature descriptor vectors to represent an image and detect unusual activities in a co-embedding space. In [10], a video is considered regular if it can be composed from large chunks of spatial-temporal patches, and the irregular videos are detected by probabilistic inference in graphical model.

The rest of the paper is organized as follows. Section 2 introduces a similarity measurement of motion trajectory, and we develop a parameter learning algorithm for the similarity measurement in Section 3. Section 4 describes the system for detecting surveillance anomaly. The experimental results are demonstrated in Section 5 and we conclude this paper in Section 6.

2. MEASURING TRAJECTORY SIMILARITY

2.1. Requirements for trajectory similarity measurement

In video surveillance, the object trajectories are often obtained from certain tracking algorithm. The target of a tracking algorithm is to discover objects that move in a similar motion pattern or follow certain routine. A similarity measurement between trajectories of moving objects becomes necessary in dealing with such data. Due to the speciality of the trajectory data, the similarity measurement should meet three requirements. First, it should be able to compare trajectories with different lengths and different number of location samples. Second, the similarity measurement should be robust to sensor/algorithm noise and outliers. Third, it should be able to handle local temporal shifts, i.e. the shifts of sub-trajectories in time space. Local temporal shifts are usually caused by variance in sampling rate, object velocity and etc.

2.2. Edit distance and its generalization for real sequences

Edit distance [11] is widely used in text analysis, speech processing, and bio-informatics for comparing two strings. It can be extended to measure the similarity between two strings of real numbers. The edit distance of trajectories X^m and Y^n with lengths m and n is defined as follows:

$$ED(X^m, Y^n) = \begin{cases} \infty, & \text{if m = 0 and n = 0;} \\ n \cdot c_{ins}, & \text{if m = 0 and n} \neq 0; \\ m \cdot c_{del}, & \text{if m} \neq 0 \text{ and n = 0;} \\ \min\{ED(X^{m-1}, Y^n) + c_{del}, & \\ ED(X^{m-1}, Y^{n-1}) + c(x_m, y_n), & \\ ED(X^m, Y^{n-1}) + c_{ins}\}, & \text{else,} \end{cases} \quad (1)$$

where x_m and y_n are the m-th and n-th elements of the trajectories X^m and Y^n respectively, c is the cost function such that $c(x_m, y_n) = c_{match}$ if $dist(x_m, y_n) \leq \epsilon$; $c(x_m, y_n) = c_{mismatch}$ if $dist(x_m, y_n) > \epsilon$. c_{match}, $c_{mismatch}$, c_{ins}, c_{del} are the costs for match, mismatch, insertion, deletion respectively. By symmetry we have $C_{ins} = C_{del}$. ϵ is the matching threshold that quantizes the noise effect to distance c_{match} or $c_{mismatch}$. The edit-distance is essentially the minimum cost of operations required to transform one trajectory sequence to another one.

The edit distance can be efficiently computed by using dynamic programming with a computational complexity of $O(m \cdot n)$. The edit distance is non-metric since it does not obey the triangle inequality rule. An appropriate similarity measurement for trajectory pair can be obtained by normalizing the edit distance with the sum of the trajectory lengths,

$$\frac{ED(X^m, Y^n)}{m + n}. \quad (2)$$

3. LEARN TRAJECTORY SIMILARITY FUNCTION

Traditional edit-distance needs manually tune the parameters for specific applications. For instance, comparing the shape of trajectories does not require to penalize local temporal shift while comparing both the shape and velocity of trajectories requires to consider local temporal shift. This penalty determines the costs for insertion and deletion and is different for various applications.

We developed an supervised algorithm to automatically learn the cost function of edit-distance. The proposed algorithm is based on *Expectation-Maximization* (EM) [12] approach. To better present the algorithm, we first introduce a probabilistic model for the edit distance. Let E denote the set of edit-operations, i.e. $E = \{$match, mismatch, insertion, deletion$\}$. We define a probability function p such that:

$$p(e) \in [0, 1], \quad e \in E \cup \{*\}, \quad p(*) > 0, \quad \sum_{e \in E \cup \{*\}} p(e) = 1,$$

where $*$ is symbol for the end of the edit sequence.

Let $(e_1, e2, ..., e_l, *)_{X,Y}$ be the sequence of edit-operations that transform X into Y. The probability of the edit-operation sequence is:

$$p((e_1, e2, ..., e_l)_{X,Y}) = \prod_{i=1}^{l} p(e_i) \cdot p(*)$$

If we let c_e to be the negative logarithm of the edit-operation probability: $c_e = -\log p(e)$, the edit-distance is the negative logarithm of the maximum probability of edit-operation sequence that transform X to Y, which we denote as $P(X, Y)$. Therefore computing the edit-distance equals to computing the most likely edit-operation sequence or Viterbi sequence. It can be efficiently estimated by Viterbi algorithm using dynamic programming.

The edit distance parameter learning problem is formally defined as follows: for a given set of training data consisting of similar trajectory pairs $\{(X_1, Y_1), (X_2, Y_2), ..., (X_s, Y_s)\}$, we expect to find the probability function $p(e)$ or c_e, such that the probability product of the most-likely edit-operation sequences of them are maximized:

$$\{p(e)\}^* = \arg \max_{\{p(e)\}} \prod_{i=1}^{s} P(X_i, Y_i).$$

Ristad and Yianilos [13] developed a generative model for string edit-distance and utilized EM approach for training the model parameters, which is similar to the Baum-Welch algorithm for training HMM. In this paper, we extend the learning algorithm from sequences of discrete symbols to real-valued sequences. EM consists of two steps: E-step and M-step, and alternates between E-step and M-step until converged. The E-step and M-step are described as follows:

1. **E-step** Given current estimate of edit operation probability, compute the most likely edit sequence for each similar trajectory pairs by Viterbi algorithm. Compute the frequency of occurrence of each edit operation.

2. **M-step** Maximize its likelihood with respect to the probability of each edit operation using Eq 3. $\#(e_c)$ represents the frequency of occurrence for edit operation e_c. The frequency of occurrence of $*$ is manually set to be a small positive constant to ensure $p(*) > 0$.

$$p(e_c) = \frac{\#(e_c)}{\sum_{e_j \in E \cup \{*\}} \#(e_j)} \quad (3)$$

The EM algorithm in our algorithm differs from the algorithm in [13], and in the E-step, our algorithm computes the edit operation frequency using Viterbi algorithm instead of forward-backward algorithm as in [13]. With this modification, the traditional edit distance is directly connected with the probabilistic model of edit operations. The EM approach is guaranteed to converge to a local optimum on a given input. Our experiments suggest that such local optima can be effectively reduced with good initialization and random restart.

1. **Distance matrix computation:** Compute the distance matrix of the trajectories in the database.
2. **Clustering:** Cluster the trajectories and use example trajectories to represent each cluster.
3. **Outlier detection:** Given a new incoming trajectory, compare it with all the clusters to decide whether it belongs to a cluster or is a anomaly.

Fig. 1. The overview of an anomaly detection system

4. ANOMALY DETECTION

4.1. System overview

The definition of anomaly is context dependent. Supervised approaches develop a prediction model for normal and/or abnormal activities from labelled data. The incoming data is then matched against the model to find anomalies. Although this approach can be effective in situations with limited number of activities, it is usually very difficult to label and model a large amount of data under unconstrained environment. We therefore choose the unsupervised definition of anomalies: frequently occurred patterns are normal and the pattern dissimiliar to the majority of normal patterns is anomalous. Given the pairwise similarity matrix of the motion trajectories, the problem left is to identify unusual motion trajectories from frequent trajectory dataset. Fig 1 gives an overview of the proposed anomaly detection system.

4.2. Object detection and tracking

In this work, the object trajectories are obtained from a visual detection and tracking system. The whole system includes the following components: a Gaussian Mixture background model (GMM), motion detection from background subtraction, and the appearance manifold based tracking algorithm [14] to extract the trace of each object.

4.3. Clustering trajectories

Anomaly detection can be considered as a byproduct of clustering results. After clustering the training data, clusters with large number of data points are classified as normals while small clusters are classified as anomalies. For a new data, it will be compared against the large clusters and if none of them match, it will be considered as an anomaly.

4.3.1. Clustering by spectrum clustering

In our system, we use spectral clustering algorithm [15] for trajectory clustering. The advantages of spectral clustering are two-fold: first, it only requires pairwise distance of data points; second, the algorithm does not require the distance to be metric. To cluster the data into k subsets, the spectral clustering algorithm computes the largest k eigenvectors of the normalized Laplacian matrix from the affinity matrix, which represent the similarity between data points. Then it performs the popular k-means clustering algorithm on the resulting k-dimensional feature space.

4.3.2. Choosing representatives of the clusters

In many cases, the clusters are represented by the mean and variance of their data points. However, these parameters can not be directly obtained given that only pairwise distances are available. We instead select a subset of the atypical data points in the cluster as a representative set. To well approximate the atypical points of each cluster, we use a simple algorithm similar to [16]: the first point in the representative set is chosen as the point with the largest distance to all the other data points in the cluster. Then each time the data point with the largest distance to the current representative set is added until the representative set reaches a predefined size n_c.

4.4. Detecting anomaly from clusters

Given that the data points are associated with each other by pairwise similarity, distance based anomaly detection algorithms are preferred. The distance based algorithms do not require any apriori knowledge of data distributions that statistical algorithms often do. Moreover, the definition of anomaly can be generalized to model statistical test for normal, Poisson and other distributions. In our system we use the algorithm in [17], which is an extension of k-nearest neighbor outlier detection algorithm.

5. EXPERIMENTS

The anomaly detection algorithm was tested in both outdoor and outdoor surveillance videos. The indoor and outdoor videos are about 30 minutes long and they 24 and 11 normal trajectories respectively. We use 60% of the data to train the cost function. The videos are sampled at the frequency of 30 Hz. In all experiments ϵ is set to half of the object bounding box width and $p(*) = 0.01$.

Figure 2 demonstrates the experimental results of anomaly detection in traffic surveillance videos. The object with normal trajectory is highlighted using white bounding-box and the object with anomalous trajectory is highlighted using black bounding-box. In this video the normal trajectories are clustered into two classes. The upper row shows four typical frames of tracking result of a normal vehicle trajectory. The lower row shows a detected trajectory anomaly in which the car pulled over the road and stopped for a while. It is obvious that the anomalous trajectory differs from normal trajectories in both shape and velocity.

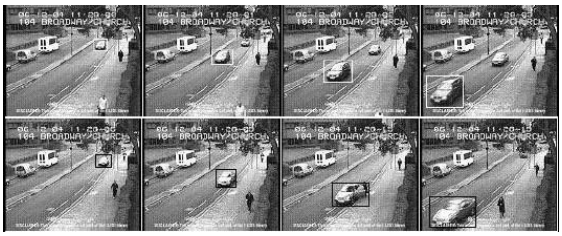

Fig. 2. Anomaly detection in road surveillance.

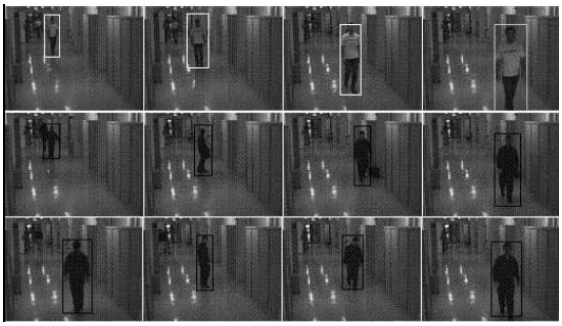

Fig. 3. Anomaly detection in indoor surveillance.

Figure 3 shows the anomaly detection results in indoor surveillance videos. The normal motion trajectories include two clusters in which people went down the hallway in two opposite directions. The first row demonstrates a typical normal trajectory. The second row shows an anomalous event in which the person loitered in the middle of the corridor for a while and dropped a bag. The third row gives another example of anomalous trajectory in which a person turned back and disappeared. In both cases our algorithm successfully detected the anomalies by the motion trajectory from the tracking algorithm.

6. CONCLUSION

In this paper, we presented a supervised algorithm to adaptively learn the parameters of edit distance for computing the similarity of motion trajectories. Then a fully automatic anomalous event detection system was proposed for surveillance videos. The anomaly detection was posed as general outlier detection problem in a non-parametric distance-based framework. Our future work will focus on: 1) extend the anomalous event detection algorithm to utilizing a larger feature space, and 2) propose novel machine learning algorithm for bridging the gap between original raw features and semantic events.

7. REFERENCES

[1] Yuri A. Ivanov and Aaron F. Bobick, "Recognition of visual activities and interactions by stochastic parsing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 852–872, 2000.

[2] C. R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780–785, 1997.

[3] Ismail Haritaoglu, Davis Harwood, and Larry S. David, "W4: Real-time surveillance of people and their activities," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 809–830, 2000.

[4] Paul Viola and Michael J. Jones, "Robust real-time face detection," *Int. J. Comput. Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[5] Gerard Medioni, Isaac Cohen, Francois Bremond, Somboon Hongeng, and Ramakant Nevatia, "Event detection and analysis from video streams," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 8, pp. 873–889, 2001.

[6] M. Brand, N. Oliver, and A. Pentland, "Coupled hidden markov models for complex action recognition," in *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, Washington, DC, USA, 1997, p. 994, IEEE Computer Society.

[7] Darnell J. Moore, Irfan A. Essa, and Monson H. Hayes, "Exploiting human actions and object context for recognition tasks.," in *ICCV*, 1999, pp. 80–86.

[8] Chris Stauffer and W. Eric L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 747–757, 2000.

[9] Hua Zhong, Jianbo Shi, and Mirkó Visontai, "Detecting unusual activity in video," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2004.

[10] Oren Boiman and Michal Irani, "Detecting irregularities in images and in video.," in *ICCV*, 2005, pp. 462–469.

[11] Vladimir I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, vol. 10, no. 8, pp. 707–710, 1966.

[12] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society series B*, vol. 39, pp. 1–38, 1977.

[13] Eric Sven Ristad and Peter N. Yianilos, "Learning string-edit distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 5, pp. 522–532, 1998.

[14] Kuang-Chih Lee, Jeffrey Ho, Ming-Hsuan Yang, and David Kriegman, "Visual tracking and recognition using probabilistic appearance manifolds," *Comput. Vis. Image Underst.*, vol. 99, no. 3, pp. 303–331, 2005.

[15] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," 2001.

[16] David W. Jacobs, Daphna Weinshall, and Yoram Gdalyahu, "Classification with nonmetric distances: Image retrieval and class representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 6, pp. 583–600, 2000.

[17] Edwin M. Knorr, Raymond T. Ng, and Vladimir Tucakov, "Distance-based outliers: algorithms and applications," *The VLDB Journal*, vol. 8, no. 3-4, pp. 237–253, 2000.