INCIDENT RETRIEVAL IN TRANSPORTATION SURVEILLANCE VIDEOS – AN INTERACTIVE FRAMEWORK

Xin Chen and Chengcui Zhang
Department of Computer and Information Sciences, the University of Alabama at Birmingham
Birmingham, Alabama, 35294 USA

ABASTRACT

Detecting and retrieving incidents from traffic surveillance videos is an important research topic in designing an Intelligent Transportation System (ITS). Most existing video analysis techniques focus on low level features of video data. A "semantic gap" exists between the machine-readable low level features and the high level human understanding of the video content. Aiming at this problem, we propose an interactive framework for semantic video retrieval. This framework is based on the spatio-temporal modeling of vehicle trajectories. With Relevance Feedback (RF), human interaction is involved in the learning and retrieval process. The retrieval mechanism is thus guided by the user's response to the retrieved results. Experiments show the effectiveness of the framework.

1. INTRODUCTION

The abundance of traffic surveillance videos makes it an urgent task to automate the process of retrieving scenes containing incidents. Such incidents of interest may include car crashes, illegal U-turns, speeding or anything that the user may be interested in. Most of the research in Intelligent Transportation Systems is focusing on vehicle tracking, which is only the preprocessing phase for studying the semantic meaning of videos. Currently, automatic traffic incident detection is drawing the attention of more and more researchers.

Various machine learning algorithms are explored for traffic incident detection: 1) Hidden Markov Model, as in Porikli et al. [8] to estimate traffic congestion without vehicle tracking; in Kamijo et al. [7] for traffic monitoring and accident detection at intersections. 2) Belief Networks. For examples, Huang et al. [9] propose a traffic scene analysis algorithm; Buxton and Gong [10] use Bayesian belief networks to model dynamic dependencies between parameters involved in visual interpretation. 3) Self-Organizing Map (SOM), as the hierarchical SOM in [11] and the fuzzy SOM in [6].

All these work use some traditional data mining methods, which still have the problem of "semantic gap" in mapping between the low level features and the high level semantic meanings. It is inherently difficult for the machine to understand the video content by only looking at pixels, frames or signals. Therefore, it is necessary that a human provides guidance to the machine. For such a purpose, Relevance Feedback (RF) [1] is adopted in our proposed framework. RF is a well-known technique in Content-Based Image Retrieval (CBIR). It is used to incorporate the user's subjective perception with the learning and retrieval process. As a supervised learning technique, Relevance Feedback has been shown to significantly increase the retrieval accuracy. We borrow this idea from CBIR and apply it to video data.

Another reason RF is adopted in the framework is that it can progressively gather training samples and customize the retrieval process. In the scenario of information retrieval, multiple "relevant" and "irrelevant" classes exist according to the different preferences of different users. In a large-scale multimedia database, it is difficult to pre-define a perfect set of training sets for all "relevant" classes before query, due to the uncertainty of users' interest. With RF, the initial query results are returned based on some heuristics i.e. the models of some generally categorized events. The training set for the user's specific query is built up gradually with the help of the user's feedback. Therefore, RF provides more flexibility in information retrieval as it customizes the search engine for the need of individual users.

Trajectories of moving objects in consecutive frames are a type of spatio-temporal data. The aim of semantic video retrieval is to extract semantic scenes by analyzing the spatio-temporal relations among moving and still objects in the video. In this study, we focus on the retrieval of incident scenes from transportation surveillance videos. The proposed framework first tracks and singles out each distinct vehicle in the video. The trajectories of all vehicles are recorded. In the learning and retrieval phase, Relevance Feedback is incorporated, with which the user provides feedback and the learning algorithm learns from it by penalizing the "irrelevant" scenes and encouraging the "relevant" scenes. In this paper we proposed a learning algorithm based on neural networks for time series data. Video data is a special kind of time series data as it consists of sequences of values or events changing over time. There are a large amount of literatures (e.g. [2]) on applying neural network in forecasting the behavior of real world time series data. It is popular in such applications as studying the fluctuations of stock market. However, relatively few works (e.g. [5]) have addressed the issue of event detection in time series data with neural network. We explore the spatio-temporal models of a neuron for semantic event mining and retrieving from video data.

Section 2 is the framework overview. Section 3 demonstrates the design details of each component. Experimental results are presented in Section 4. Section 5 concludes the paper.

2. FRAMEWORK OVERVIEW

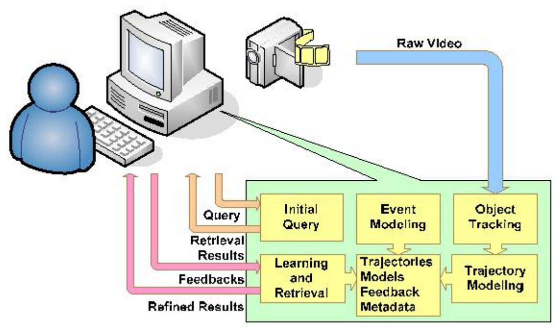

Figure 1. The System Overview

Figure 1 shows the architecture of the proposed framework. Raw videos are processed and the vehicle objects are extracted. After object tracking, the trajectories of vehicle objects are modeled with a curve fitting technique. When the user first submits a query, the system performs an initial search by some heuristics and returns the retrieval results to the user. The user responds to these results by giving his/her feedbacks. The learning mechanism in the system will then learn from these feedbacks and refine the retrieval results in the next iteration. The whole process goes through several iterations until a satisfactory result is obtained.

3. LEARNING AND RETRIEVAL

The first step is preprocessing, in which moving vehicles are automatically tracked and their trajectories are modeled. This component is based on our previous work [3]. With a series of recorded centroids on successive frames, we approximate the trajectory of each vehicle by the least-square curve fitting. The fitted curve represents a rough shape of the moving trajectory. The first derivative of a polynomial curve is a tangent vector, which represents the velocities of that vehicle at different time.

Following trajectory modeling is the semantic event modeling and then the learning and retrieval process. Details are depicted in the subsections below.

3.1 Semantic Event Modeling

With different event types, different properties of semantic objects need to be extracted to build the models for specific event types. In this study, a spatio-temporal model is built for car-crash events. Car crashes may involve one or multiple vehicles. In all cases, the focus shall be the sudden behavior pattern changes of individual vehicles.

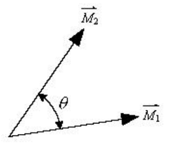

Figure 2. The Change of Motion Vector

Once the sampling rate is known, the change of velocity at each point can be easily calculated. A motion vector is a vector with its starting point being the centroid of some vehicle at the previous sampling point and the ending point being the centroid of the same vehicle at the current sampling point. As illustrated in Figure 2, the change of motion vector is denoted as the angle between the current motion vector and the previous one. $\overrightarrow{M_1}$ and $\overrightarrow{M_2}$ are two consecutive motion vectors. θ is the difference angle between them. Only its absolute value is needed. Another factor that needs to be taken into consideration is the distance between two vehicles. For each vehicle, we record its minimum distance from its nearest vehicle – *mdist* at each sampling point.

As mentioned earlier, some heuristics need to be established for the initial queries. This heuristic model is built based on the observation that the sudden change of velocity and driving direction may indicate an accident. Further, the closer the vehicle is to the other vehicles, the higher the chance of an accident. At the i^{th} sampling point, the property vector of a vehicle is $\alpha_i = [1/mdist_i, vdiff_i, \theta_i]$. A series of such vectors $\alpha = [\alpha_1, ..., \alpha_n]$ represent the trajectory of a vehicle. It is worth mentioning that this event model may also be adjusted to detect U-turns, speeding and other events that involves the abnormal behavior of a vehicle.

3.2 Learning and Retrieval Framework

The proposed learning and retrieval framework is based on neural networks. The most commonly used neural network model is to induce the function *f* in a standard Multilayer Perceptrons (MLP) or (Radial Basis Function) RBF architecture. By extending this model for temporal processing, the neural network can perform prediction in time series data. This method is often called sliding window technique as the N-tuple input slides over the entire training set. A time series is a sequence of vectors depending on time t: x_t, $t = 0, 1, \ldots$. Time series analysis using neural networks requires predicting or estimating x_k based on the preceding m observed data $x_{k-m}, \ldots, x_{k-2}, x_{k-1}$.

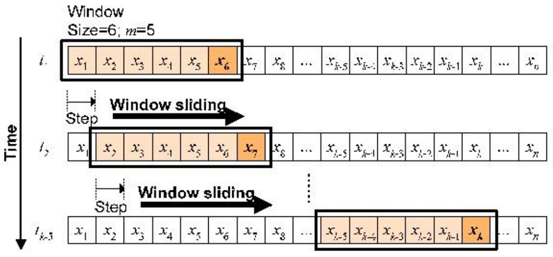
Window
Size=6; m=5
Time
t_1
t_2
t_{k-5}
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 … x_{k-5} x_{k-4} x_{k-3} x_{k-2} x_{k-1} x_k … x_n
Step
Window sliding

Figure 3. An Example of Sliding Window

Figure 3 shows an example of sliding window for time series training data. In this example, the input is a 5-tuple extracted consecutively from a time series. The window size is 6. The sixth data point in the window serves as the output. By sliding the window one step a time along the time t, a training set can be constructed. However, in the proposed framework, we do not need an exact prediction of x_k. Instead, only an indication of whether x_k is the event of interest is needed. The problem then turns into a classification problem i.e. mapping a sequence onto one of the two classes of either "relevant" or "irrelevant":

$$f_c : (x_{k-m}, \ldots, x_{k-2}, x_{k-1}) \rightarrow c_i \in C \quad (1)$$

where C is the set of all class labels. This can be regarded as a special case of function approximation, in which the targets are binary values. The use of classical linear autoregressive models in modeling time series data is rather limited, since they assume linear relationship among consecutive data series. MLP and RBFs offer an extension to the linear model by using a non-linear function.

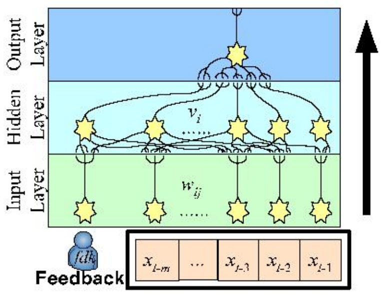

Figure 4. The Proposed Network

The proposed framework is based on a feedforward multilayer neural network. The structure of the network is shown in Figure 4. The user's feedback is incorporated as a node (*fdk*) in the input layer. The other input nodes are a sequence of vectors selected from $\alpha = [\alpha_1, ..., \alpha_n]$ by sliding the window, where α_i is the feature vector at the i^{th} sampling point. The number of input nodes is thus equal to the window size plus one *fdk* node. Instead of a binary value, the single output is a "score" that reflects the degree of relevance. It is used to rank all the sequences. The detailed design as to the window size, the number of hidden units and hidden layers, weights, and search algorithms are elaborated below.

3.3 Network Design

Window size and input nodes. The size of the sliding window determines the number of input nodes. If the window is too small, the sequence in the window is too "short" to capture the overall pattern of an event. On the other hand, if the window is too large, the sequence would be too "general" to reveal the core event characteristics with too many noise and/or irrelevant data around.

The advantage of video data over regular time series data is that it can be visualized. In our framework, the size of the window can be simply decided by the typical length of an event as it can be acquired by counting the number of frames. For a car crash event, the typical length is about 15 frames. Given a sampling rate of 5 frames/point, 3 sampling points are needed, making a window size of 3. Each input node, excluding the *fdk* node, is a feature vector α_i extracted according to the event model aforementioned.

Hidden layer. We adopt a two-layer neural network – one hidden layer having sigmoid transfer function and one output layer having linear transfer function. It has been shown that this network architecture can approximate virtually any functions of interest to any degree of accuracy, provided that sufficiently many hidden units are available. The optimal number of units in the hidden layer is hard to decide. A large number will reduce the convergence rate of learning and have a high generalization error, while a small number cannot guarantee the approximation accuracy. One rule of thumb [8] is that it shall not exceed twice the size of the input nodes. Suppose the size of input is *l*, we tested on the sizes of 0.5*l*, *l*, *1.5l* and 2*l* and found that *l* produces the minimum estimated generalization error.

Activation function. As mentioned above, the transfer function in the first layer (activation function) is sigmoid, which is used to introduce nonlinearity.

$$y = \tanh(\sum_i w_i x_i) \tag{2}$$

tanh is the tangent hyperbolic function, a conventional sigmoid function. w_i is the weight for the input x_i.

Initial weights. Most of the neural networks use random numbers as initial weights. However, since the backpropagation is a hill-climbing technique, the randomness of initial weights may result in local optimum. A multiple linear regression weight initialization method [4] is adopted in our framework. The initial weights in the first layer are still random. However, the weights in the second layer are obtained by multiple linear regression.

After the initial weights for the first layer are generated, input nodes and their corresponding weights are fed to the sigmoid function to get the output values for each neuron in the hidden layer. Suppose these outputs are $R_1, R_2, \ldots, R_l$. The second layer uses a linear transfer function so that

$$y = \sum_i v_i R_i \tag{3}$$

where v_i is the weight on the second layer and R_i's are regressors. This is a typical multiple linear regression model. It can be solved by the least square optimization.

Search algorithm. The basic form of the backpropagation is computationally expensive. Some existing numerical optimization techniques have been applied to the training of multilayer perceptrons. Among them are steepest descent, conjugate gradient and Newton's method. Steepest descent is the simplest, but is often slow in convergence. Newton's method is much faster, but requires the Hessian Matrix and its inverse be calculated. The conjugate gradient is a compromise. It does not need to calculate the second derivatives, yet it still has the quadratic convergence. We choose the conjugate gradient approach.

3.4 Event Learning and Retrieval Process

In the initial query, there is no training sample. We compute the relevance scores of all video sequences according to the heuristic model of that event. For car-crash events, the relevance score is computed as the square sum of all the three features in a feature vector $\alpha_i = [1/mdist_i, vdiff_i, \theta_i]$. The retrieval results are returned in descending order of their relevance scores. It is assumed that a big velocity change, a sudden change of driving direction and a short distance between two vehicles are indications for possible accidents.

After the initial query, a certain amount of results are presented to the user. The user identifies a returned sequence as "relevant" if it is of his/her interest; otherwise the user labels it "irrelevant". With this information, a set of training samples can be gathered. Each training sample is in the form of [α_{i-2}, α_{i-1}, α_i, *fdk*, *opt*]. *fdk* is zero if the user identifies it as "irrelevant", otherwise it is incremented by a small number ε. *opt* is the desired output having the value of one or zero with one being "relevant" and zero being "irrelevant". These training samples are then fed into the learning framework, which refines the retrieval results in the following iterations. After several iterations, the "relevant" sequences are promoted by incrementing their *fdk* values, while the "irrelevant" sequences are punished by keeping their *fdk* values at the lowest point. Our experiments show that the retrieval results are improved through iterations.

4. EXPERIMENTS

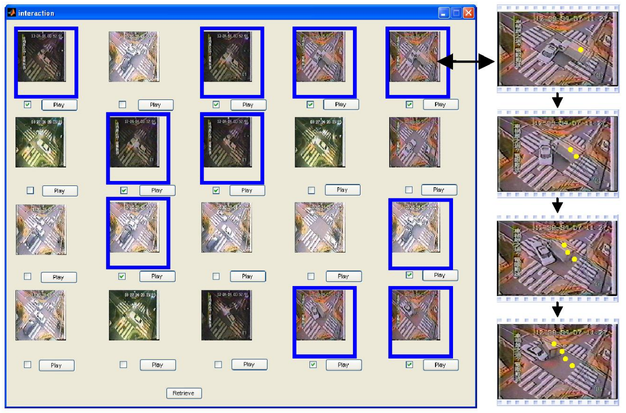

Figure 5. The User Interaction Interface

Figure 5 shows the interface for the user to provide feedback. Top 20 sequences are returned to the user, which can be played. Trajectories of possible problematic vehicles are traced. One example of the retrieved accident sequence is shown in Figure 5. The trajectory of the truck is found out to be problematic and is marked by yellow dots. If the user thinks the sequence contains a real incident, it will be marked 'relevant'. As shown in the interface, ten sequences (in blue rectangles) are labeled "relevant".

The proposed framework is tested on two video clips. The first one was taken in a tunnel and contains 2504 frames. The second one is a real-life traffic surveillance video taken at an intersection in Taiwan and contains 592 frames. The sampling rate is 5 frames/point and the window size is 3. After sampling and window sliding, there are 109 sequences (15 frames each) from the first clip and 168 sequences from the second clip.

The proposed framework is compared with the traditional weighted relevance feedback method, in which each feature in the feature vector α_i has a weight. The initial round of retrieval is the same as the proposed framework. With the user's feedback, the feature vectors of all relevant sequences are gathered. The inverse of the standard deviation of each feature is computed and used as the updated weight for this feature in the next round.

Five rounds of relevance feedback are performed - Initial (no feedback), First, Second, Third, and Fourth. The 'accuracy' is calculated, which is defined as the percentage of all the 'relevant' sequences within the top n (e.g. n=20) returned sequences. Figure 6 shows the retrieval accuracies over the top 20 sequences for the first clip after Initial, First, Third, and Fourth iterations.

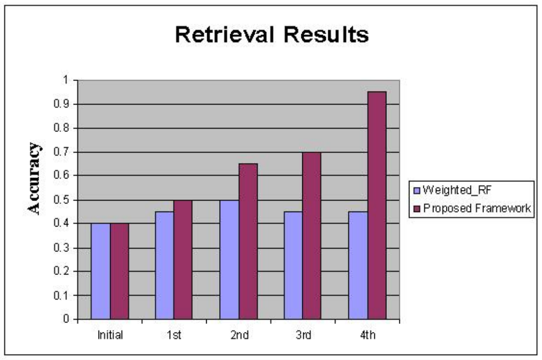

Figure 6. The Retrieval Accuracy for the 1st Clip

It can be gleaned from Figure 6 that the proposed framework performs much better in that the accuracy increases steadily from 40% to 95%. In "weighted_RF", the increase is not stable and the result of the 3rd round is even worse than the 2nd round.

Most of the accidents in the first clip only involve a single vehicle. In the second clip, the accidents often involve two or more vehicles. The retrieval results are compared with the weighted RF in Figure 7. Although the accuracy gains with the proposed framework is not as high as that in the first clip, it is far better than that of the weighted RF method.

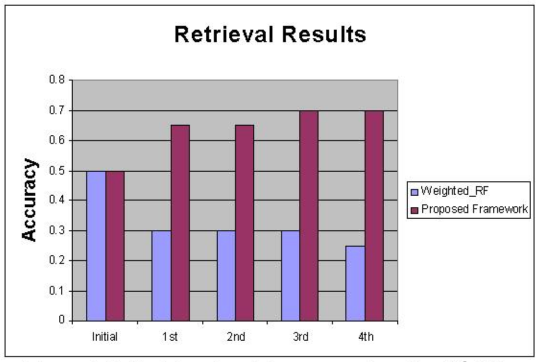

Figure 7. The Retrieval Accuracy for the 2nd Clip

5. CONCLUSION AND FUTURE WORK

We apply the neural network to process video sequences, a special type of time series data. A mapping between spatio-temporal trajectories and network input nodes is developed. Our goal is semantic event retrieval, which can be reduced to a "classification" problem. Therefore, the "prediction" neural network for time series data is adjusted to suit the needs of detecting spatio-temporal semantic events from video data.

The proposed work also distinguishes from other work by incorporating the relevance feedback in interactive video retrieval. As the Relevance Feedback techniques are widely used in CBIR, we adjust it to fit the needs of semantic video retrieval.

In future work, the event models for other general events such as U-turns will be constructed and tested. There is also a need for collecting more video data with the associated metadata (e.g. camera parameters) that is needed for the normalizing all the videos before the storage and retrieval. Currently, the framework only supports the user's query by the specified event type. This will be extended to include query by example, query by sketches, and customized combination of query types.

6. REFERENCES

[1] Y. Rui, T.S. Huang, and S. Mehrotra, "Content-based Image Retrieval with Relevance Feedback in MARS," In *Proc. of the International Conf. on Image Processing*, 1997, pp. 815-818.

[2] N. Davey, S.P. Hunt, and R.J. Frank, "Time Series Prediction and Neural Networks," *Journal of Intelligent and Robotic System,* Vol. 31, 2000.

[3] S.-C. Chen, M.-L. Shyu, S. Peeta, and C. Zhang, "Learning-Based Spatiotemporal Vehicle Tracking and Indexing for Transportation Multimedia Database Systems", *IEEE Trans. on Intelligent Transportation Systems*, Vol. 4, No. 3, pp. 154-167, Sept. 2003.

[4] M.-C. Chan, C.-C. Wong, and C.-C. Lam, "Financial Time Series Forecasting by Neural Network Using Conjugate Gradient Learning Algorithm and Multiple Linear Regression Weight Initialization," *Computing in Economics and Finance*, 2000.

[5] D. Gao, Y. Kinouchi, K. Ito, and X. Zhao, "Neural Networks for Event Extraction from Time Series: a Backpropagation Algorithm Approach," *Future Generation Computer Systems*, Vol. 21, pp.1096-1105, 2005.

[6] W. Hu, X. Xian, D. Xie, and T. Tan, "Traffic Accident Prediction Using 3-D Model-Based Vehicle Tracking," *IEEE Trans. on Vehicular Technology,* Vol. 53, No. 3, May 2004.

[7] S. Kamijo, Y. Matsushita, and I. Katsushi, "Traffic Monitoring and Accident Detection at Intersections," *IEEE Trans. on Intelligent Transportation Systems,* Vol. 1, No. 2, pp. 108-118, Jun. 2000.

[8] F. M. Porikli and X. Li., "Traffic Congestion Estimation Using HMM Models without Vehicle Tracking," In *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, pp. 188-193, Jun. 2004.

[9] T. Huang, D. Koller, J. Malik, G. Ogasawara, B. Rao, S. Russell, and J. Weber, "Automatic Symbolic Traffic Scene Analysis Using Belief Networks," In *Proc. of National Conference on Artificial Intelligence,* 1994.

[10] H. Buxton and S. Gong, "Visual Surveillance in a Dynamic and Uncertain World," *Artificial Intelligence*, Vol. 78, pp. 431-459, 1995.

[11] D. Xie, W. Hu, T. Tan, and J. Peng, "Semantic-based Traffic Video Retrieval Using Activity Pattern Analysis," In *Proc. of IEEE International Conference on Image Processing (ICIP)*, 2004.