AUTOMATIC RELEVANCE FEEDBACK FOR VIDEO RETRIEVAL

P. Muneesawang

Dept. of Elect. and Comp. Engineering
Naresuan University, Phisanulok, Thailand

L. Guan

Dept. of Elect. and Comp. Engineering
Ryerson University, Toronto, Canada

ABSTRACT

This paper presents an automatic relevance feedback method for improving retrieval accuracy in video database. We first demonstrate a representation based on a template-frequency model (TFM) that allows the full use of the temporal dimension. We then integrate the TFM with a self-training neural network structure to adaptively capture different degrees of visual importance in a video sequence. Forward and backward signal propagation is the key in this automatic relevance feedback method in order to enhance retrieval accuracy.

1. INTRODUCTION

Incorporating relevance feedback (RF) for improving retrieval accuracy is increasingly important for multimedia application [1] [2] [8]. While many RF models have been successfully developed for still-image applications, we have seen few for video database applications. The difficultly is that RF requires video representation to capture sequential information to allow analysis. While there are limited studies on relevance feedback for video retrieval [1][2], where the *audio-visual* information is utilized for characterizing spatio-temporal information within the video sequence, the application of RF to video files is, however, a time-consuming process since users have to play each of the retrieved video files, which are usually large, in order to provide relevance feedback. In practice, this is more difficult for retrieval on Internet databases. In this paper, we suggest implementing the RF in an automatic fashion. We first propose video representation based on template-frequency modeling (TFM) that emphasizes both spatial and temporal information to allow the RF to effectively analyze the dynamic content of the video. We then adopt the self-training neural network [3] to implement the automatic RF. Since neural network models perform well at matching given patterns against a large number of possible templates, we adopt this organization for similarity matching in video retrieval. We associate the TFM parameters to the network weights to reorganize the parameters through a signal propagation process within the network. This process allows the improvement of retrieval accuracy, while minimizing user interactions.

This paper is organized as follows: Section 2 describes video indexing using TFM. Section 3 presents automatic relevance feedback network for video retrieval. Section 4 shows the results of applying the method to the video database of CNN news.

2. VIDEO INDEXING USING TFM

Video database may be organized into three levels: shot, scene, and story. In this work, we apply retrieval algorithms to video database at the shot level. Characterizing video content within a shot has traditionally be done by the traditional method using key-frame based video indexing (KFVI) techniques, where a few representative frames are chosen for video representation, and for similarity matching between shots [5]. Although the KFVI method is relatively easy to implement, it produces a representation which may not be adequate to capture video content since it does not take into account temporal information. Instead, the similarity matching between videos is based on the spatial content of the predefined key-frames.

In view of this, we propose a video representation based on a template-frequency model (TFM) that takes into account spatio-temporal information. We view video data as a collection of visual templates, so that the video characterization is the analysis of the probability of the templates occurring in a video sequence. Compared to the KFVI technique, that relies on a few representative frames in its key-frame selection algorithms, the TFM differentiates the degrees of importance among frames by effectively incorporating temporal information.

Let $C = \{\vec{g}_r \in \Re^P | r = 1, 2, ..., R\}$ be a set of visual templates that have been generated by an optimization process, such as the learning vector quantization algorithms [6]. Also let video interval I_j be described by a set of descriptors $D_{I_j} = \{(\vec{x}_1, f_1), ...,$ $(\vec{x}_m, f_m), ..., (\vec{x}_M, f_M)\}$, where $\vec{x}_m \in \Re^P$ is a feature vector of the m-th video frame, f_m, e.g., the color histogram, which may be obtained during the shot segmentation process [7]. Our goal here is to analyze the degree of importance of each visual template $\vec{g}_r$ to the video I_j. So, if the template $\vec{g}_r$ presents many times in the video sequence I_j, it should be regard as important, and associated with a weight of high value. In this way, the video interval I_j can be described by a weight vector, $\vec{v}_j = \{w_{j1}^{\exists}, ..., w_{jr}^{\exists}, ..., w_{jR}^{\exists}\}$, where the weight w_{jr} is associated with the template $\vec{g}_r$. To obtain the weight vector, each video frame is first mapped through $\Re^P \rightarrow C$, such that each frame is represented by a set of template labels:

$$f_m \Rightarrow \rho^{(\vec{x}_m)} = \{l_{r^*,1}^{\vec{x}_m}, l_{r^*,2}^{\vec{x}_m}, ..., l_{r^*,\eta}^{\vec{x}_m}\}, \qquad (1)$$

where $l_{r^*,i}^{\vec{x}_m},\ i = 1, ..., \eta$ are the labels of the top η best match templates, e.g.,

$$l_{r^*,1}^{\vec{x}_m} = \arg\min_r(||\vec{x}_m - \vec{g}_r||) \qquad (2)$$

The resulting labels, $\{l_{r^*,1}^{\vec{x}_m}, l_{r^*,2}^{\vec{x}_m}, ..., l_{r^*,\eta}^{\vec{x}_m}\}, \forall m$, of all frames from the mapping of the entire video interval I_j are used to obtain the weight parameter: $w_{jr} = freq_{jr}$, where $freq_{jr}$ stands for the row frequency of template $\vec{g}_r$ in the video interval I_j (i.e., the

0-7803-7663-3/03/$17.00 ©2003 IEEE

number of times the template $\vec{g}_r$ is mentioned in the content of the video I_j). We also employ a weighting criterion demonstrated in [8] to improve the weight parameter by:

$$w_{jr} = \frac{freq_{jr}}{\max_r freq_{jr}} \times \log N/n_r \tag{3}$$

where N denotes the total number of videos in the systems, and n_r denotes the number of videos in which the index template $\vec{g}_r$ appears.

It suffices here to note that only a few from the large number of templates is used for indexing an input video sequence (i.e., if the template $\vec{g}_r$ does not appear in the video I_j then $w_{jr} = 0$), so that the weight vector $\bar{v}_j$ is very sparse and only non-zero elements are kept.

3. SELF-ORGANIZING RELEVANCE FEEDBACK

As we observed in the previous discussion, the TFM models a video by using numerical weight parameters, w_r, $r = 1, ..., R$ each of which characterizes a degree of importance of visual templates presented in the video. These weight parameters will be reorganized on a per query basis. At this point, a video cluster that maximizes the similarity within the cluster, while also maximizing the separation from the other clusters, can be formed based on content identifiers, to *initialize* the ranking for a incoming query. This ranking is then adopted to re-organize the degree of importance of the visual templates through the following process. First, the process identifies 'effective templates' that are the common templates among videos in a retrieved set. Then, those templates considered to be the most significant for re-weighting the existing templates of the initially submitted query are weighted, to improve the performance of ranking. In other words, we allow the templates that are not referred to by the initially submitted query (i.e., $w_r = 0$, $r \in [1, R]$), but are common among the top-ranked videos (i.e., the potentially relevant videos), to "expand". This results in reorganization of the degree of importance of the query's templates for better video similarity measuring.

This process is in the same spirit as the user-controlled relevance feedback techniques widely used in information retrieval applications, whereby a set of significant items specified by the user is added to the initial query, and used to re-weight the query components [8]. In this work, we similarly adopt this query reformulation scheme for the expanding of queries to improve ranking. However, our goal here is to minimize user involvement, by proposing the adoption of a self-learning model [3]. As neural networks perform well at matching given patterns against a large number of possible templates, we use this structure for selecting relevant videos. Fig. 1 shows a neural network architecture for automatic video ranking.

3.1. Signal Propagation Process

The network is composed of three layers: one for the query templates, one for the video templates, and the third for the videos themselves. Each node communicates to its neighbors via the linking connections. The query template nodes initiate the inference process by sending signals to the video template nodes. The video template nodes then themselves generate signals and send to the video nodes. Upon receiving this stimulus, the video nodes, in their turn, generate new signals directed back to the video template nodes. This process might repeat itself several times, between the second and the third layers, which allows the network to find templates that appear to be relevant on the basis of initial ranking, and use those templates to refine the video ranking.

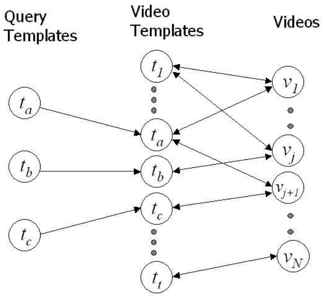


Fig. 1. Automatic relevance feedback network (ARFN)

To be precise, let $\vec{v}_q = \{w_{qr} | r = 1, ..., R\}$ denote the set of the query's weight components, obtained by converting the video query v_q into a set of templates. Let $mesg_{r^{(q)} \to r^{(t)}}$ denote the message sent along the connection $\{r^{(q)}, r^{(t)}\}$ from the r-th query node to the r-th video template node. Also, let $mesg_{r^{(t)} \to j^{(v)}}$ denote the message sent along the connection $\{r^{(t)}, j^{(v)}\}$ from the r-th video template node to the j-th video node, $j \in [1, N]$. Note that $mesg_{r^{(q)} \to r^{(t)}}$ is a one-to-one correspondence, while $mesg_{r^{(t)} \to j^{(v)}}$ is a one-to-many correspondence. First, each query template node is assigned a fixed activation level $a_r^{(q)} = 1$, $r \in [1, R]$. Then, its signal to the video template node is attended by normalized query template weights $\bar{w}_{qr}$, as follows:

$$mesg_{r^{(q)} \to r^{(t)}} = a_r^{(q)} \times \bar{w}_{qr} \tag{4}$$

$$\bar{w}_{qr} = \begin{cases} \frac{w_{qr}}{\sqrt{\sum_{r=1}^{R} w_{qr}^2}} & \text{if } \vec{g}_r \in v_q \\ 0 & \text{o.w.} \end{cases} \tag{5}$$

When a signal reaches the video template nodes, only the video template nodes connected to the query template nodes are activated. These nodes might send new signals out, directing towards the video nodes, which are again attenuated by normalized video template weights $\bar{w}_{jr}$ derived from the weights w_{jr}, as follows:

$$mesg_{r^{(t)} \to j^{(v)}} = mesg_{r^{(q)} \to r^{(t)}} \times \bar{w}_{jr} \tag{6}$$

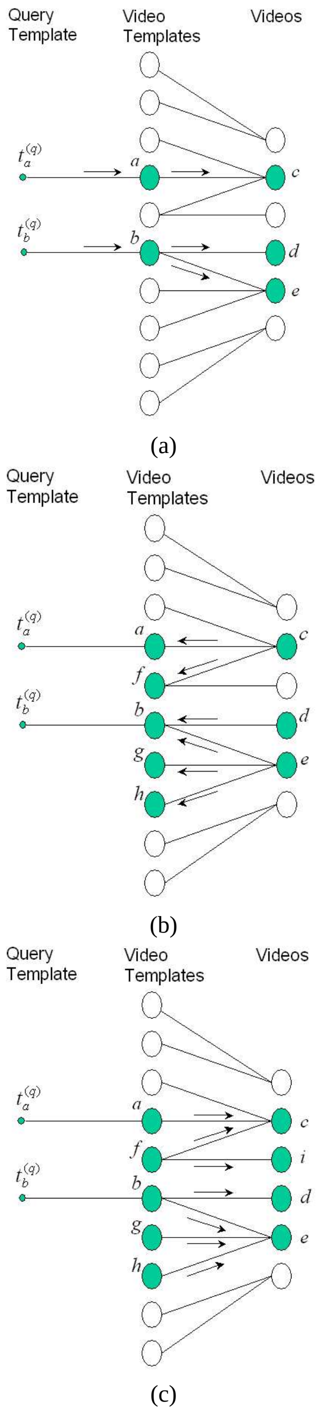
$$\bar{w}_{jr} = \begin{cases} \frac{w_{jr}}{\sqrt{\sum_{r=1}^{R} w_{jr}^2}} & \text{if } \vec{g}_r \in v_j \\ 0 & \text{o.w.} \end{cases} \tag{7}$$

As a result, some signals reach a video node, and the *activation level* of this video node (associated to the video v_j) is given by the sum of the signals (the standard cosine measure),

$$a_j^{(v)} = \sum\nolimits_{r=1}^{R} mesg_{r^{(t)} \to j^{(v)}} \tag{8}$$

$$= \sum\nolimits_{r=1}^{R} \bar{w}_{qr} \bar{w}_{jr} = \frac{\sum_{r=1}^{R} w_{qr} w_{jr}}{\sqrt{\sum_{r=1}^{R} w_{qr}^2} \sqrt{\sum_{r=1}^{R} w_{jr}^2}} \tag{9}$$

This finishes the first round of signal propagation. The network output (i.e., $a_j^{(v)}$, $j = 1, ..., N$) is the desired ranking of



(a)

(b)

(c)

Fig. 2. Signal propagation; (a) signals from the two query templates are sent to the video template nodes, and then three video nodes $\{c, d, e\}$ are activated; (b) the signal propagates back from the third layer to the second layer, resulting in more activated video temple nodes; (c) signal propagates back to the third layer. This results in the activation of new video nodes by expanding the original query template and the activated video nodes in (b)

the videos for retrieval. The process, however, does not stop here. The network continues the ever-spreading activation process after the first round of propagation. This time, however, a minimum activation threshold is defined such that the video nodes below this threshold send no signals out. Thus, the activation level at the r-th video template node is obtained from the input from the activating video nodes as:

$$a_r^{(t)} = \frac{1}{\left(\sum_{r=1}^{R} l_r^2\right)^{1/2}} \tag{10}$$

$$l_r = w_{qr} + \alpha \sum_{j \in Pos} a_j^{(v)} \bar{w}_{jr} + \beta \sum_{j \in Neg} a_j^{(v)} \bar{w}_{jr} \tag{11}$$

where $a_j^{(v)}$ is the activation of the j-th video, Pos is the set of j's such that $a_j^{(v)} > \tau$, and Neg is the set of j's such that $a_j^{(v)} < -\tau$, where τ is a threshold value. The activation process is allowed to continue flowing forwards and backwards between the video template nodes and the video nodes, inducing an order to the videos, based on the corresponding node activations at each stage.

In other words, we allow the network to automatically expand the query templates analogous to the relevance feedback model [8]. The signal propagation process is directly related to the derivation of new weights of query templates, thereby a new template appearing in the most highly activated videos, regardless of if they appeared in the original query, may become active and may activate other videos. This modifies the initial vector ranking in the retrieval process.

Fig. 2 graphically describes the spreading activation process. Fig. 2(a) shows two query templates sending signals to the video template nodes $\{a, b\}$. This means that the video nodes: $\{c, d, e\}$ are activated (the application of the threshold is omitted for the purpose of illustration.). Fig. 2(b) shows the signals propagating backward to the video template layers. At this time, f, g and h are the newly activated nodes. After re-calculation of the node activations, the video template nodes send signals forward to the video nodes as shown in Fig. 2(c). This results in a new ranking, which includes a new video node, i. We see that the network then utilizes new video template node f, to find more relevant video nodes.

4. EXPERIMENTAL RESULTS

In the experiments, the test video data was obtained from the Informedia Digital Video Library Project [9]. This is the CNN broadcast news, which includes full news stories, new headlines, and commercial break sections. This video results in 844 video shots, segmented by the color histogram based a shot boundary detection algorithm [7]. A 48-bin histogram computed on HSV color space is used for both shot segmentation and indexing algorithms.

We compared the TFM indexing technique with the KFVI, and applied the automatic relevance feedback network (ARFN) to improve retrieval accuracy. The KFVI uses a histogram vector generated from a middle frame of the video shot as a representative video shot. Similarity measure used is the normalized Euclidean metric. In the TFM case, a total of 5,000 templates are generated.[1] Each video shot is described by its associated weight vector. This

[1] Here, the template library was generated and obtimized from histogram vectors of the videos within the test set.

was generated by the template models, using neighborhood $\eta = 5$ [cf. (1)]. We then associated the weight vectors with ARFN. This video database results in a network with 5,844 nodes and 14,800 connections.

A total of 25 queries were made and the judgments on the relevance of each video to each query shot were evaluated. In general, the relevance judgment of videos is difficult because two video clips may be related in terms of the story context, and not just visual similarity. We were aware of this fact in this experiment, so we employed as a criterion a very subjective judgment of relevance: only retrieved video shots from the 'same' stories were judged to be relevant.

Table 1 shows precision results as a function of top matches, averaged over all 25 queries. The second column results were obtained by the KFVI, and the third column results were obtained by the TFM. In the ARFN case, we show the results of three tests: letting the activation spread for one, three, and twenty iterations, with $\tau = 0.1,\ \alpha = 0.95$ and $\beta = 0.05$, respectively [cf. (11)]. We observed that the TFM performed substantially better than the KFVI for every setting of the number of top matches (the average precision was higher by more than 18%). The following observations were also made from the results: First, the ARFN was very effective in improving retrieval performance—average precision increase by more than 11%, and is particularly significant in the top 10 to 16 retrievals. Second, it stabilized very quickly. Third, allowing many iterations meant that the performance deteriorates gradually. Finally, our results were achieved by simply allowing the activation flow automatically, with no user input.

# top matches	KFVI	TFM			
		0 RF	1 RF	3 RF	20 RF
1	100.0	100.0	0.00	0.00	0.00
2	98.0	100.0	0.00	0.00	0.00
3	93.33	98.67	+1.33	+1.33	-1.33
4	87.00	97.00	+1.00	+2.00	-2.00
5	78.40	96.00	+0.80	+1.60	-1.60
6	74.00	94.67	+0.67	+2.00	-1.33
7	71.43	90.29	+2.29	+3.43	+1.14
8	68.50	89.00	+3.00	+2.50	+1.50
9	67.11	86.67	+2.67	+3.11	+2.67
10	64.40	82.80	+5.60	+5.60	+4.80
11	61.45	80.36	+6.18	+6.18	+5.09
12	60.00	77.67	+7.33	+7.67	+7.00
13	57.85	74.77	+8.62	+10.15	+8.31
14	56.86	72.00	+9.43	+11.14	+9.71
15	54.93	69.33	+9.87	+11.20	+10.13
16	53.25	67.75	+9.50	+11.00	+10.00

Table 1. Average Precision Rate, APR (%) obtained by ARFN and KFVI, using 25 video shot queries. ARFN results are quoted relative to the APR observed with cosine measure at 0 RF.

5. CONCLUSION

Video database applications require suitable indexing techniques to capture the time-varying nature of video data, together with a high performance retrieval strategy, In this paper we proposed a template-frequency model (TFM) and its integration with a specialized neural network, which can satisfy these requirements. Unlike previous RF attempts, we incorporated a self-learning neural network to implement an automatic RF scheme, which requires no user input for its adaptation. Based on the simulation study, this adaptive system, utilizing the TFM and automatic-RF retrieval architecture, can be effectively applied to a video database, with promising results.

6. REFERENCES

[1] R. Wang, M. R. Naphade, and T. S. Huang, "Video retrieval and relevance feedback in the context of a post-integration model," *IEEE Int. Workshop on Multimedia Signal Processing*, Cannes, France, pp. 33-38, 2001.

[2] M. Naphade, R. Wang, and T. S. Huang, "Audio-visual query and retrieval: a system that uses dynamic programming and relevance feedback," *Journal of Electronic Imaging*, pp. 861-870 Oct. 2001.

[3] R. Wilkinson and P. Hingston, "Using the cosine measure in a neural network for document retrieval," *Proc. Of the ACM SIGIR Conf. on Research and Development in Information Retrieval*, pp. 202-210, USA, Oct 1991.

[4] H. S. Chang, S. Sull, and S. U. Lee, "Efficient video indexing scheme for content based retrieval," *IEEE Trans. On Circuits and Systems for Video Tech.*, vo. 9, pp. 1269-1279, 1999.

[5] A. K. Jain, A. Vailaya, and W. Xiong, "Query by video clip," *Multimedia Systems Journal*, vol. 7, pp. 369-384, 1999.

[6] S. Haykin, *Neural networks: a comprehensive foundation*, Prentice Hall, Upper Saddle River, New Jersey, 1999.

[7] U. Gargi, R. Kasturi, and S. H. Strayer, "Performance characterization of video shot-change detection methods," *IEEE Trans. On Circuits and Systems for Video Technology*, vol. 10, no. 1, pp. 1-13, Feb. 2000.

[8] G. Salton, E. A. Fox, and E. Voorheers, "Advance feedback methods in information retrieval," *Journal of the American Society for Information Science*, Vol. 36, pp. 200-210, 1985.

[9] Informedia Digital Video Library Project at Carnegie Mellon University, http://www.informedia.cs.cmu.edu, Sept 2001.