Text Based Video Retrieval among Video Clips

Chi Zhang
University of Science and Technology Beijing
Beijing, China
zhangchi2021@ia.ac.cn

Guixuan Zhang
Beijing Engineering Research Center of Digital Content
Beijing, China
guixuan.zhang@ia.ac.cn

Shuwu Zhang
Beijing Engineering Research Center of Digital Content
Beijing, China
shuwu.zhang@ia.ac.cn

***Abstract*—In recent years, the popularity of intelligent camera equipment has increased year by year, which has brought the number of videos growing rapidly. It is of practical significance to establish a high-quality video library for the reference of creators. In this paper we describe a video clip retrieval software, which can provide the corresponding video clips according to the needs of video creators such as directors and scriptwriters. The retrieval system is built around the traditional text based image retrieval (i.e. TBIR) method, and on the basis of which many improvements have been made, so that the retrieval speed and accuracy reach a high standard. From the practical experience, it is a successful attempt to apply TBIR to video retrieval.**

Keywords—Video Retrieval, Machine Learning, Image Retrieval

I. INTRODUCTION

With the rapid development of multimedia platforms in the video field such as network media, video portals and short video websites, in recent years, the majority of video production has gradually shifted from professional video creators to the public. This phenomenon makes it hard to collect originality for video creation because the videos made by public usually have low reference value.

Based on the above background, this design establishes a retrievable video clip library outlined in Fig. 1, so as to provide creative reference for video creators.

The framework of proposed method contains 2 stages: visual information annotation and video retrieval. The whole process begins after selecting a sufficient number of films as material. In annotation stage, the material is divided into shots that can be described by label pairs, and then each shot is labeled by the object detection model. In the second stage, the retrieval algorithm calculates the matching degree of each shot according to the labels from previous stage and the query input by the user. After ranking the scores, the shot with higher matching score is returned on the user interface.

II. THE PROPOSED METHOD

A. Preparation

Film is the portrayal of the world we live in. It can convey to us all kinds of historical events, national and regional cultural customs, as well as people's interaction in different scenes. In the film, the actors make various stage performances according to different plots, in changing shooting occasions. Because of the variety of film content, we choose film as the source of material. When users lack inspiration, they are provided with movie clips containing the required content.

We carefully selected 201 films, and tried to include a variety of themes in the selection process, such as love, action, science fiction, suspense, comedies, war, horror, etc. In addition, the time span of the selected films is large to include films of different times. The above is to improve the reference value of the collected films.

B. Segmentation & Annotation

After obtaining the film material, we need to understand the video content. Because the number of films in our material library is not large, we decide to design the whole retrieval system based on TBIR method. Although the text-based image retrieval method began in the 1970s, it is still used in many retrieval applications today because of its less difficulty in implementation, manual intervention in annotation and relatively high precision.

Segmentation. It's impossible to recommend a movie to the creators directly. The principle of TBIR also determines that we can't annotate the content of the whole movie. If we choose to segment the video in the form of scene, the software can meet the needs of video creation exactly, and provide the most valuable inspiration for users. But in the aspect of implementation, scene segmentation is a difficult problem which has not been solved well at present, and it is very difficult to realize. We choose to divide the films into shots. We use a simple but effective histogram-based segmentation algorithm similar to the one in [1]. A shot boundary is marked between frames if their histograms in HSV color space differ by more than 80%, and more than 80% of tracked SURF keypoints fail to match between the frames. We saved the time dividing point of the shot and the video of each shot. The former is used to annotate the information of each shot, while the latter is used to return it to the user for reference after retrieval.

Annotation. A shot consists of a considerable number of frames, each lasting several seconds. No matter what the majority of the shot information is, the main information contained in the shot cannot be separated from the place.

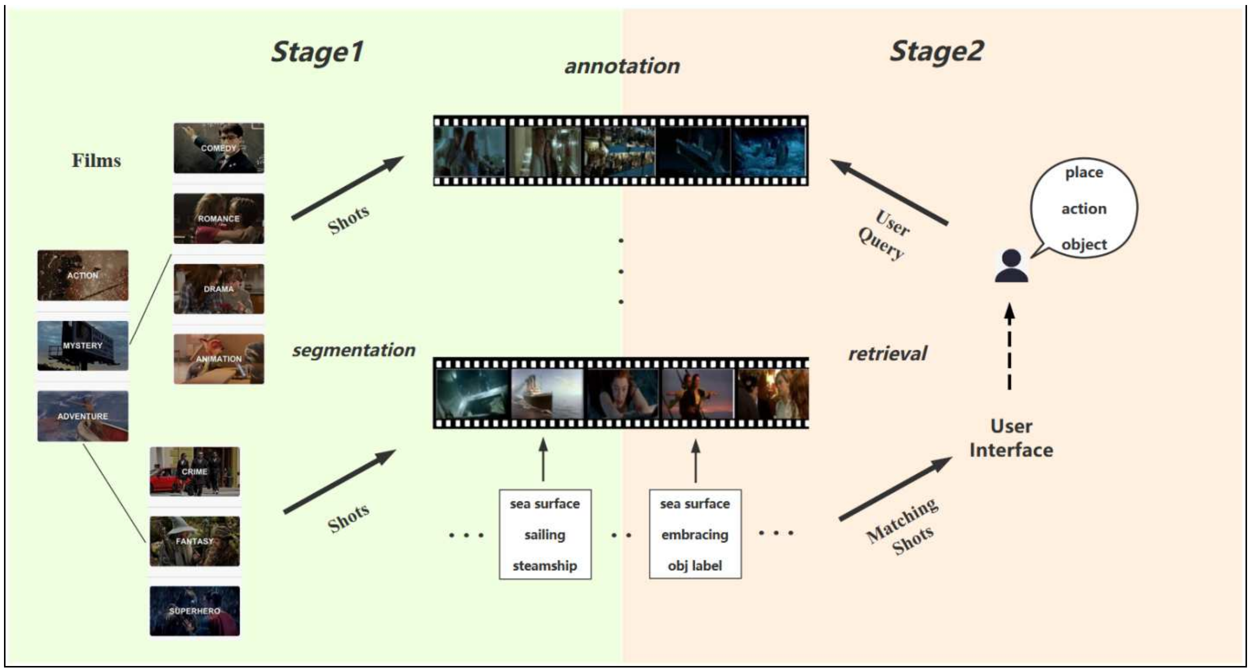

Fig. 1. The pipeline of text base video retrieval, which contains two stages being segmentation and retrieval.

This is the common sense of experience, "seaside", "court", "deck"... Any shot must contain place information. According to the different shooting objects, the shot will contain other main information. When the main shooting object is human, the first element of human description is action. Any shot of human description is directly related to action. The actor's own characteristics, such as hairstyle, height, skin color, etc., as well as the recessive characteristics, popularity, age, etc., are not the main information to be conveyed by the camera. Then, ignoring the secondary information of the characters, the information of the characters in the shot can be described by actions. When the main shooting object of the shot is not a person or more than a person, other information can be summarized as “objects”. We also observed that most of the film understanding datasets focus on two elements above, i.e. character[2-8], action[9-13].

In this way, the principal component of the shots includes three elements: place, action and object. For the video creators, when there is no inspiration for a shot creation, the retrieval results according to the above three shot elements can cover most of the information of the required shot and return the clips that match the user's needs.

In order to annotate accurately and conveniently, we use the *mmdetection* kit [14]. This is a pre-training model set based on various object detection datasets, and the structure of the models are also diverse. The pre-training models and their training datasets are recorded in Table 1.

TABLE I. PRE-TRAINING MODELS

Pre-training models	**Label**		
	Place	**Action**	**Object**
dataset	Places365	Kinetics600	ImageNet1000
structure	ResNet50	I3D	ResNet50

The accuracy of annotation is affected not only by the performance of the model, but also by the interference of the shot content itself. Since the shot usually lasts several seconds, the shot may contain information of multiple labels, so we use Top5 format to annotate each shot.

C. *Semantic matching*

When searching, user needs to input three keywords: place, action and object, so as to calculate the similarity with the corresponding labels annotated on shots. At present, the performance of embedding generated by BERT is impressive. Therefore, we use Google's Chinese BERT model to generate word embedding for user query because of the target users for our software are Chinese. It is known that the pre-training models we use in annotation are all based on English datasets (i.e. Places[15], Kinetics600[16] & ImageNet[17]), so, when calculating the matching degree between query and shot, we need to get the Chinese word embedding of labels on each shot.

Since the total number of all labels is not very large, we mainly use manual translation to translate English labels into Chinese. In detail, for each English label, three most popular translation software are used to translate it into Chinese. In this way, Places, Kinetics600 and ImageNet correspond to 1095, 1800 and 3000 Chinese words respectively. Translation software tend to give the most frequently used word in the corpus, so it is observed by us that there are a lot of repetitions in the translation results. In view of this problem, manual translation is adopted to improve the quality of translation.

we don't avoid the situation that several English words have the same Chinese translation. In addition, we hope that the co-occurrence information behind each English label can be fully mined in order to achieve better retrieval effect. E.g., for the place label "seaside", in addition to the literal translation "seaside", we also need to add words with high co-occurrence frequency, such as "beach", "offshore" and so on.

With the word embedding of label and query, we use cosine distance to measure the similarity between them. According to the definition of cosine similarity as shown in equation (1), we obtain these word vectors in advance and unite them, then save them in a binary file for retrieval.

$$\boldsymbol{Similarity(x,y)} = \frac{\mathrm{x}}{\sqrt{\sum_{\mathrm{i}=1}^{\mathrm{n}}(\mathrm{x}^{\mathrm{i}})^2}} \cdot \frac{\mathrm{y}}{\sqrt{\sum_{\mathrm{i}=1}^{\mathrm{n}}(\mathrm{y}^{\mathrm{i}})^2}} \tag{1}$$

Although the performance of Word2vec is not so good as BERT, the former generates word vectors that must contain information not considered by the latter because of different principles. But unlike BERT, Word2vec cannot generate word embedding for words that didn't appear in the corpus. In addition, it can't be guaranteed that user query will be in the range of Word2vec dictionary definitely. Our strategy is expressed as (2), which aims to calculate the similarity between user query and shot labels using BERT only if the user query do not have corresponding Word2vec embedding. Conversely, the cosine similarity calculated by Word2vec and BERT is linearly combined as the basis for sorting.

$$\boldsymbol{S_p^i} = \begin{cases} w_{w2v} * Sim\left(W_p^u, W_p^i\right) + w_B * Sim\left(B_p^u, B_p^i\right) & \boldsymbol{case\ \delta} \\ Sim\left(B_p^u, B_p^i\right) & \boldsymbol{other} \end{cases} \tag{2}$$

In (2), suppose $case\ \delta$ is when user place query and the place label for the shot *i* are both in the Word2vec dictionary range. The *Sim* was defined by (1), and the sum of Word2vec score weight w_{w2v} and BERT's w_B is 1. Word2vec is added to assist BERT in semantic matching, which contributes about 5% to retrieval precision and recall.

Thus, a single matrix multiplication yields all the scores for one class of labels. Take the action score as example, the process is shown in (3), where *n* denotes the number of action labels and *dim* denotes the dimension of word embedding. mark * indicates the user query.

$$\boldsymbol{S_{plc}^{1\times n}} = E_{plc^*}^{1\times dim} \times E_{plc}^{dim\times n} \tag{3}$$

D. Retrieval

The retrieval process is for all shots of 201 movies. From the first shot of the first movie to the last shot of the last movie, each shot is marked from 1 in ascending order. The matching score of each shot will be recorded under its number, and finally the shots with high matching degree are returned to the user. Fig.2 shows the operation of the numbering step.

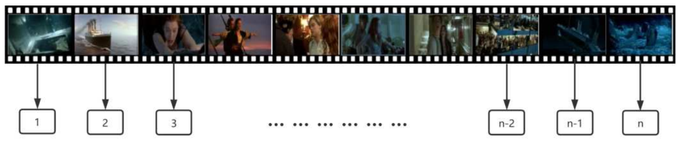

Fig.2. The Sketch of shot numbering process

The total score for a shot *i* is expressed as (4).

$$\boldsymbol{S_{total}^i} = w_p * S_{plc}^i + w_a * S_{act}^i + w_o * S_{obj}^i \tag{4}$$

Since the query that user input can't be predicted, the shortage of materials is inevitable. When the lack of material leads to the lack of matching shots, the retrieval system should try to return the satisfactory shots to the user. It is found by us that people tend to pay more attention to the characters and their behavior in the scene than to the place. Therefore, we hope that the weight w_p and w_a in (4) can be adjusted adaptively by a perceptron according to the query.

A training data looks like $(\left[x_p^j,\ x_a^j\right],\ y^j)$ is composed of the place and action vector which are concatenated, and *j* denotes the pair number. The value of *y* is defined as (5).

$$\boldsymbol{y}^{\mathrm{i}} = \begin{cases} 1.0 & Sim\left(B_p^j, B_a^j\right) > 0.8 \\ 0.5 & Sim\left(B_p^j, B_a^j\right) < 0.2 \end{cases} \tag{5}$$

Based on (5), 29704 training data were obtained from 1971,000 label pairs. The parameters of perceptron are derived from SGD optimizer with the loss function being MSELoss described as (6).

$$(\boldsymbol{w}^*, \boldsymbol{b}^*) = argmin_{(w,b)} MSE((w \cdot x + b), y) \tag{6}$$

Output of the perceptron is limited between 0.5 and 0.6 according to the range transformation formula (7).

$$\boldsymbol{\hat{y}} = 0.5 + \frac{y - 0.5}{1 - 0.5} * (0.6 - 0.5) \tag{7}$$

Replacing the original action score weight with $\hat{y}$ shown in (8) is the expansion form of (4). The 0.5 in front of object score aims to balance its weight among three scores.

$$\boldsymbol{S^i} = \hat{y} * S_{plc}^i + (1 - \hat{y}) * S_{act}^i + 0.5 * S_{obj}^i \tag{8}$$

E. User interface

The python package Tkinter [18] was chosen to develop user interface. We provide the top 25 shots on the user interface, which are divided in 5 pages, and each page has 5 shots. User can see the keyframe of each shot, the name of the movie and the time point of the shot in the movie. And they could watch a certain result through the default player by clicking the play button.

III. Experiment

We watched several classic movies with different themes, and in order to ensure the robustness of the experimental data, these movies are not included in our film library. We picked some shots in these movies, and described them using place, action and object words. These label combinations are used as test queries.

The performance of proposed method is evaluated by mean Average Precision (mAP) and precision-recall curves shown in Fig. 3. The range of P-R curve is from 5 results to 25 results because of the format of user interface. We manually observe the retrieval results to calculate the accuracy and recall, then obtain experimental data.

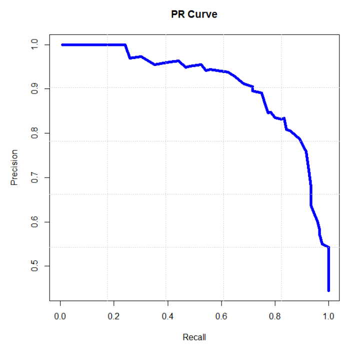

Fig.3. The PR curve of retrieval. The number of samples is range from 5 to 25.

The mAP of proposed method is 0.51.

IV. Conclusion

In this paper, we apply the traditional text base image retrieval method with several improvement measures to video retrieval and achieved good results finally. At the same time, the design itself has certain practical significance.

ACKNOWLEDGMENT

This work was supported by the National Key R&D Program of China (2018YFB1403900) and Technology Program of Beijing (Z201100001820002). It was also the research achievement of the Key Laboratory of Digital Rights Services.

References

[1] Chu, W. S. . "Video co-summarization: Video summarization by visual co-occurrence." IEEE Conference on Computer Vision & Pattern Recognition. IEEE, 2015, pp. 3584-3592.

[2] Arandjelovic, O. , and A. Zisserman . "Automatic face recognition for film character retrieval in feature-length films." IEEE Computer Society Conference on Computer Vision & Pattern Recognition. IEEE, 2005, pp. 860-867.

[3] Bauml, M. , M. Tapaswi , and R. Stiefelhagen . "Semi-supervised Learning with Constraints for Person Identification in Multimedia Data." Computer Vision and Pattern Recognition. IEEE, 2013, pp. 3602-3609.

[4] Haurilet, M. L. , et al. "Naming TV characters by watching and analyzing dialogs." IEEE Winter Conference on Applications of Computer Vision. IEEE, 2016, pp. 1-9.

[5] Nagrani, A. , and A. Zisserman . "From Benedict Cumberbatch to Sherlock Holmes: Character Identification in TV series without a Script.". British Machine Vision Conference .BMVC, 2017, pp. 111-117.

[6] Tapaswi, M. , M. Bauml , and R. Stiefelhagen . ""Knock! Knock! Who is it?" probabilistic person identification in TV-series." IEEE Conference on Computer Vision & Pattern Recognition. IEEE, 2012, pp. 2658-2665.

[7] Everingham, M. , J. Sivic , and A. Zisserman . "Hello! My name is. Buffy" -- Automatic Naming of Characters in TV Video.". British Machine Vision Conference. BMVC, 2006, pp. 617-622.

[8] Huang, Q. , Y. Xiong , and D. Lin . "Unifying Identification and Context Learning for Person Recognition." IEEE Conference on Computer Vision & Pattern Recognition. IEEE. 2018, pp. 2217-2225.

[9] Laptev, I. , et al. "Learning Realistic Human Actions from Movies." IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE, 2008, pp. 1-8.

[10] Duchenne, O. , et al. "Automatic Annotation of Human Actions in Video." IEEE International Conference on Computer Vision. IEEE, 2009, pp. 1491-1498.

[11] Marszalek, M. , I. Laptev , and C. Schmid . "Actions in context." IEEE Conference on Computer Vision & Pattern Recognition. IEEE, 2009, pp. 1191-1199.

[12] P. Bojanowski, F. Bach, I. Laptev, J. Ponce, C. Schmid and J. Sivic, "Finding Actors and Actions in Movies," IEEE International Conference on Computer Vision. IEEE, 2013, pp. 2280-2287.

[13] Bojanowski, P. , et al. "Weakly Supervised Action Labeling in Videos Under Ordering Constraints.". European Conference on Computer Vision. Springer, 2014, pp.628-643.

[14] https://github.com/open-mmlab/mmclassification

[15] Zhou, B. , et al. "Places: A 10 Million Image Database for Scene Recognition." IEEE Transactions on Pattern Analysis & Machine Intelligence. IEEE, 2018, pp. 1452-1464.

[16] Carreira, J. , et al. "A Short Note about Kinetics-600." arxiv, 2018, unpublished.

[17] Jia, D. , et al. "ImageNet: A large-scale hierarchical image database." IEEE Conference on Computer Vision & Pattern Recognition. IEEE, 2012, pp. 248-255

[18] https://docs.python.org/zh-cn/3/library/tk.html