UGC Approved Journal

IJIREEICE

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

International Journal of Innovative Research in
Electrical, Electronics, Instrumentation and Control Engineering

ISO 3297:2007 Certified

Vol. 5, Issue 6, June 2017

Video Retrieval using Tiny Video Kernels

Aditi Nain[1], Prof. K.S Bhagat[2], Dr. D.K Kirange[3]

Student, EXTC Dept, J.T M College of Engineering, Maharashtra, India[1]

Professor, EXTC Dept, J.T M College of Engineering, Maharashtra, India[2]

HOD, Computer Dept, J.T M College of Engineering, Maharashtra, India[3]

Abstract: Today humans live in the digital video age where everything we need, is available in terms of video information in the vast repositories online. There is a need to enable user devices and access existing vast store of video data in an easy manner. Although videos made day-to-day life easier and much more enjoyable due to the simplicity and flexibility that we get with the internet. In terms of video content retrieval all user must do, is to type in a search term and get back a relevant result. This process has its limitations in terms of time and speed of search tiny video is a search program that makes video retrieval more relevant in terms of the content within the video not regarding for the actual data tagged with it. It therefore promises greater accuracy in terms of relevant search information and is capable of cross-referencing an image or another sample video and not just text, to give a valid result to the user. This search algorithm is therefore an improvement on the current system and shows promise in terms of its accuracy. This paper examines the use of a unique search algorithm to improve video tagging and referencing given a large database of submitted content such as YouTube. We present our algorithm with 99% accuracy with database videos and speed of search increased 4 times than the existing search techniques.

Keywords: Tiny videos, Kernel, Video retrieval.

I. INTRODUCTION

Whenever the user searches for on-line content, we make the use of certain relevant keywords that are likely to give us the required content that they are looking for. If the users were to look deeper into this process, they would come to the realization that this is not as simple as picking a title and feeding it back. This employs a search algorithm known as semantic search.

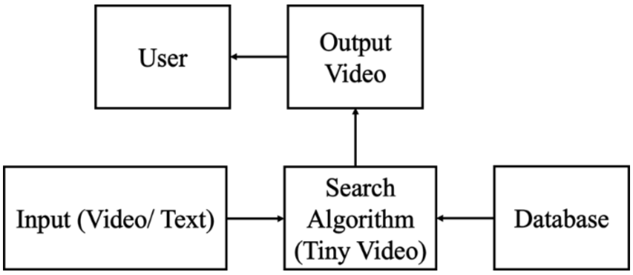

Fig. 1.Block diagram of tiny video kernel retrieval algorithm.

The algorithm references the name of the video along with metadata such as geotags, user comments and timestamps to create a database of clustered videos that will be called upon once the required term is searched for. The limitation of this process is its inability to reference the data present within the video itself which can account for more specific search terms involving objects and scenery present within a video. Here we require specific kernels such as tiny videos. The purpose of this project is the implementation of a new algorithm that can map the key objects and scenery within a video and give the user a relevant output based on a text or video query where the program will be able to reference a large dataset such as YouTube and retrieve relevant content to the user which will very much improve the search capability of any platform.

II. LITERATURE REVIEW

Similar work was done by Karpenko et al. where, they have taken video samples of 40 x 30 pixels. In our algorithm the resolution on both the images as well as the videos is much higher, showing that it can be applied to a more modern

DOI 10.17148/IJIREEICE.2017.5638

dataset.Their dataset is limited to the amount of time it takes for computation but as we will demonstrate in this paper, this limitation can be improved on vastly since scaling is not an issue for this algorithm if provided with enough computational power. There has also been expansion upon various other reports that showcase non-parametric search and key-frame based video retrieval. The accuracy of our method is much higher and given the right amount of resources can be made better and faster in its search times as well as the number of hits it will generate.

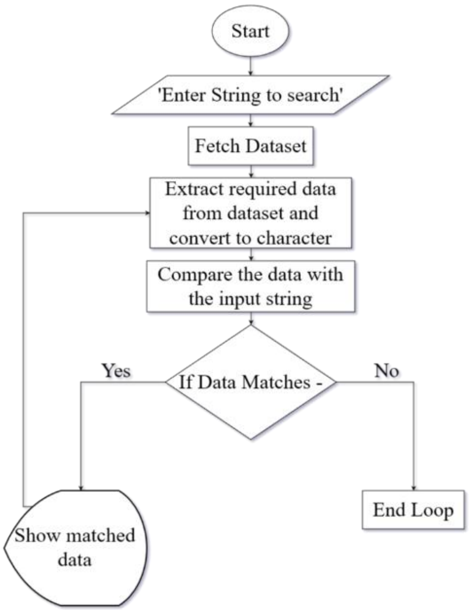
Start
'Enter String to search'
Fetch Dataset
Extract required data from dataset and convert to character
Compare the data with the input string
If Data Matches -
Yes
No
Show matched data
End Loop

Fig. 2 Semantic search algorithm

III. METHODOLOGY

Semantic search algorithm that we implemented is given in figure 2. Figure 3 shows both the earlier text based approach as well as the new feature based search approach, since our code uses both algorithms together. Our database consists of 400 videos for the current program and we tested datasets of up-to 5000 videos. Given the accuracy rates it can safely be said that it falls within an acceptable success rate (z ¡ 1.96) and hence is a viable search program for much larger datasets since it should scale in a similar fashion. Our tiny kernel search algorithm gave 99% accuracy on database videos and 4 times improvement with existing search algorithms.

A. Text-based search algorithm

This code is designed to reference pre generated search terms that are mapped using an excel database for all the videos. This is a one-time mapping and supervised process manually. The user takes the keywords that they find relate to the content within the video and maps them into an excel sheet for creation of database up to n (number of videos). Practically this will occur through user comments and other tags present under the video such as facebook or youtube. This map of all the keywords is given to the program which will refer to a new video for similar tags present in it and give the user appropriate search result. The program will extract three types of data from the database:

1. Number
2. Text
3. Raw data

This data is open for further expansion for raw and numbers data. For now, implementation of only the text is done. Numerical (Geo-tag) and raw data can be considered in future. The program will run through the rows and columns of the database map and read each term one by one. The strings it finds will be converted to characters. If the required text is found, it will set the flag. This will then check the row in which the specific keyword was present and will restore all the information of the similarly referenced videos to a database. It is done on a FIFO basis where it will play the first video (Highest priority) that it finds (In case only single video needed).

As dataset is increased, user can reference a higher number of videos since keywords repeat and like a dictionary it is limited to a set number of repeated words. Hence, once a dataset is made then ideally there is not much modification to it and the algorithm references the provided data and gives a relevant result. This is limited in scope due to its reliance

DOI 10.17148/IJIREEICE.2017.5638

on user input but in conjunction with a multi-modal search algorithm it becomes much more superior in referencing relevant content.

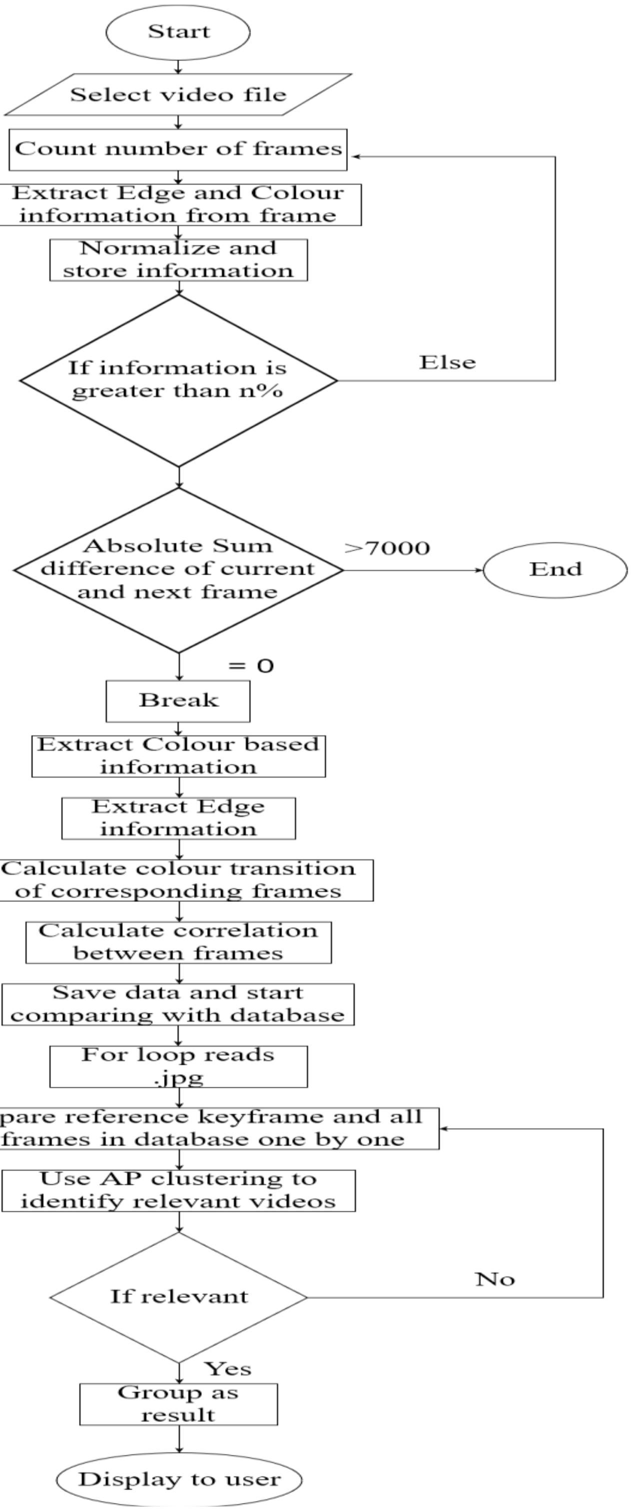
Start
Select video file
Count number of frames
Extract Edge and Colour information from frame
Normalize and store information
If information is greater than n%
Else
Absolute Sum difference of current and next frame
>7000
End
= 0
Break
Extract Colour based information
Extract Edge information
Calculate colour transition of corresponding frames
Calculate correlation between frames
Save data and start comparing with database
For loop reads .jpg
Compare reference keyframe and all keyframes in database one by one
Use AP clustering to identify relevant videos
If relevant
No
Yes
Group as result
Display to user

Fig. 3 Feature-based search algorithm

B. Feature-based search algorithm

This algorithm relies on the features available within a database of content to extract information in the form of key frames and use those as the references for fetching new videos that have also been tagged with similar key frame information. The algorithm first calls upon the UI to select a specific video file from the database and restricts the type of file to a .mpg since it is one of the most commonly prevalent types of files and for demonstration makes it less prone to error because of small file sizes and low data rates. This undergoes string concatenation and then the video file is read.

It then determines the number of frames present within the video. The number of frames can be unlimited if the user has sufficient computing power and memory to handle all the frames being loaded and extracted one by one from each video. For our use, we are limited to short videos of 5-10 seconds to prevent issues relating to memory allocation. For example, if we have a 320 x 240-pixel video consisting of say 300 frames and each frame consists of 3 colors of 8 bit each then we can multiply for ourselves and see the bit rate that it will take up per frame. To do this for a large database of videos can be a time-consuming process if user does not have enough computational resources. Hence, our study is limited to the suggested parameters for testing.

After this it searches each frame for relevant information that will serve as a keyframe for our algorithm. For this it compares current and next frame to estimate the motion within the frames. This can be streamlined by removing color information and converting the frame into a gray-scale for the program to use. Then the algorithm apply edge detection to the frame. The implementation used is Canny edge detection function within MATLAB for this task. This method of edge detection is the most accurate and it performs the function in the following manner: Apply a Gaussian filter to smooth the image to remove noise. Separate out intensity gradients present within the image. Apply non-maximum suppression to get rid of irrelevant response to edge detection. Use threshold values doubly to determine edges and eliminate spurious response. Track edges using hysteresis.

This will help the algorithm establish two parameters: 1. How much motion the frame contains 2. How much actual information is present to create a keyframe. It then normalises the average information and selects the frame. A parameter that specifies the presence of a minimum of 10% information within the frame to be taken as a keyframe has been implemented. This can be improved on by selecting a higher percentage but for experimentation this accuracy is satisfactory. Then the algorithm check for the absolute sum difference between the current and the next frame. This is done to ensure the detection of change of scene. If the difference is 0 then the scene hasnt changed and is stable. If there is a large difference, then the scene has changed and now the frame is not usable. For a zero difference frame the program uses break command to come out of the loop and the frame is selected. Otherwise an estimated value of difference of 7000 pixels between frames is to be the minimum value for finding a change of frame. If this loop does not end until the end of the video, program takes the last frame as the stable frame for its purpose.

This frame is the search frame for content based retrieval and the program now extracts relevant information from it. The first step to processing a key frame is extracting color information from the image. This information is taken and used to plot a histogram for the Red, Green and Blue colors that constitute an image. These are then concatenated into a singular matrix and stored.

Then conversion of the color image into a gray scale image is performed. Then as before, use of Canny edge detection on this image is done to establish the relevant edge information present.

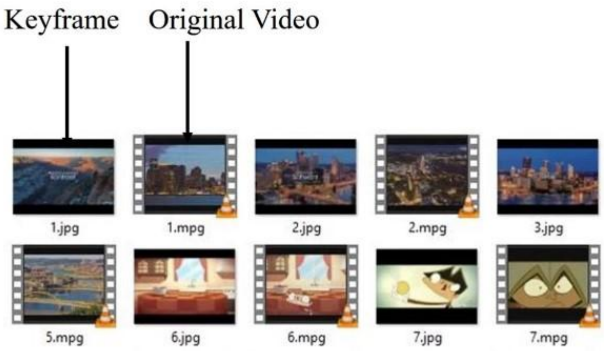

Fig. 4. For each video a keyframe in the form of an image has been extracted.

Use of color Moments command to retrieve information relevant to color transition in the given frames has been implemented. For example, a change between the shades of blue is calculated using first order differentiation to get the moments of the colour in the blue domain. Then second order and so on up-to the first four moments and the information is stored.

The next step is color correlation which uses the Correlogram command to correlate the color of the keyframe with the next frame in the color domain. It is a check to see if it matches the frame.

To provide this program a database user needs to specify a path that it can access and start running the algorithm on. To do this they need to save all the videos in a numeric order and call on them up-to a specified value. This manual numbering process simplifies the loop created for the program to access the database.

DOI 10.17148/IJIREEICE.2017.5638

This entire process exhibits an improvement in the existing semantic search since it will generate keyframes for a set of videos and for a large database will cross reference and there is no need for generation of keyframes for every video present in the database.

The next step takes the difference of the keyframe and the original video and stores it. This process is performed on every video and keyframe combination and the difference information is retrieved for all of them. If during searching there is minimum difference between a specific keyframe and the corresponding video, then it is added to the cluster in question.

Then the previous steps of color and edge detection information being retrieved from the content are repeated and differences are calculated. The difference between any of the histograms is stored and notified. The same frames could have a variation in color. Similarity is checked and then moments of color and correlation between frames is done and final values are stored.

Now finally a normalization is performed for all these individual values for each variable to fit them into a range of zero to one. For clustering to be functional it needs to adhere to this prescribed range. Normalization is done by calculating minimum and maximum value for all the values, subtracting it from all the values and dividing it by a difference of maxima and minima. This is performed for all values.

Then concatenation is done to put data into a single word and feed it to an Affinity Propagation clustering algorithm for final grouping. This is essentially a measure of how one value propagates to the next value. As an example, if user maps his hands and compare them with each other, then their AP will come out to be 1 due to similarity of edges. The algorithm checks how the value propagate and assigns it to a cluster. Eventually it starts out with an infinite number of clusters depending on dataset and eventually iterates down to 2 clusters. For our purposes this clustering can be performed within a few seconds due to the limited variables user provides to the program. To prevent infinite looping, few parameters have been provided:1. Size of dataset ¿= 6. 2. Algorithm is selected to be adaptive. 3. Maximum runs are 50000. 4. Maximum iterations are 2000. 5. Damping factor is 0.5. 6. A step size of 0.01 is considered. 7. Convergence condition of one is considered. After this the algorithm performs clustering operations depending on the parameters given to it. Now these finally clustered datasets have been assigned weights. The videos are therefore ranked per this system and the one that best conforms to the defined parameters is selected i.e. the lowest scored video. Once this is performed, the minimum value and required dataset is displayed and goes back to the main program. Then finally the required video is played.

IV. RESULTS

Here are the findings listed for various outcomes that can be concluded in the following graphs as shown:

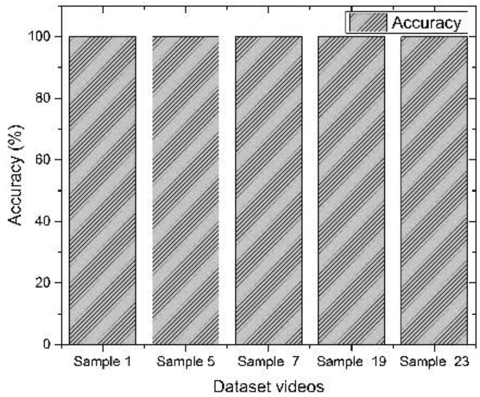

Fig. 5.Output of the algorithm given samples from within its database. As is expected a 100% accuracy of retrieval is observed.

V. CONCLUSIONS

This article presents a new way to search for content online that aims to minimize both the errors in tagging and the difficulty in storing all the relevant information through the concept of Tiny Video. The program extracts relevant information from the videos in the form of key frames and then cross references the information to the given search term and presents the user with the most relevant result. By using this algorithm, users can improve search efficiency thereby improving the functionality of the video platform in question. YouTube is the ideal target for such a program along with various other large content consumption platforms.

DOI 10.17148/IJIREEICE.2017.5638

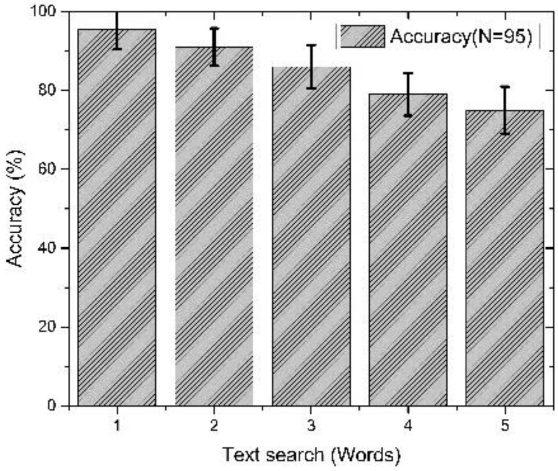

Fig. 6. Accuracy of the program given semantic search in the form of a set number of words. The user can see that as the number of words is increased, the accuracy decreases. This is due to the loss of correlation among words as the number of search terms increases, thereby reducing the number of results it can show

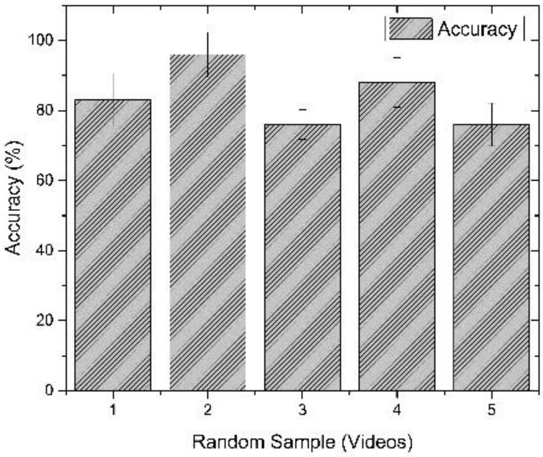

Fig. 7. Result of providing the algorithm random samples of videos to search from. As is seen there is no correlation among the results since the accuracy will vary depending on the relevance of content between the database and the search video.

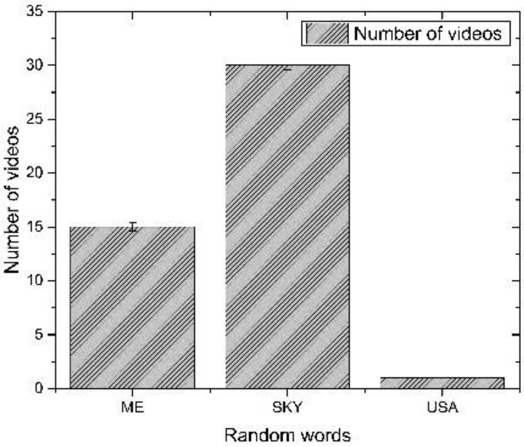

Fig. 8. Number of videos referenced for a specific keyword vs accuracy. For common words like Sky user can observe that many videos are returned, in this case 30. For USA which is an uncommon search word, a minimum return rate of 1-2 videos can be observed. Hence it will almost always give an output.

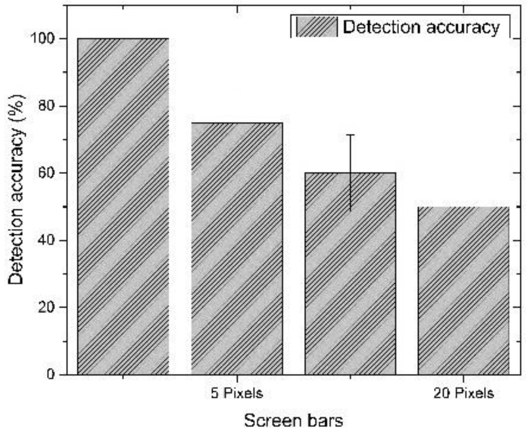

Fig. 9. The effect of screen bars on the detection of a video from the database itself. For 0 pixels, we see no loss in accuracy. Ideally there should be no change in this value but for 5 pixels to 20 pixels, we observe a drop in accuracy. Thus, addition of black bars to the scene changes how the algorithm perceives the same image.

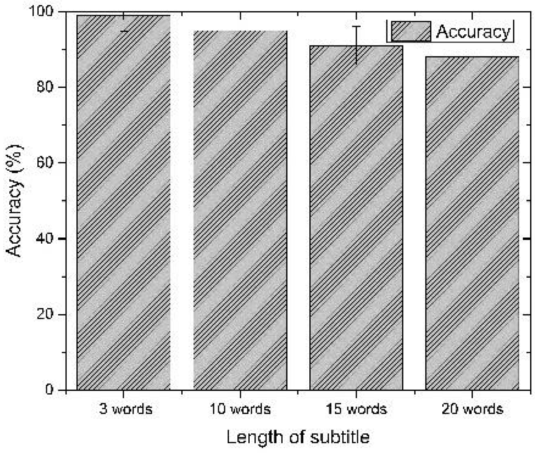

Fig. 10. The effect of adding subtitles into a frame as we can see is minimal. Here a sample video from the database has been taken and it is observed here that even though there is a loss in accuracy, it isnt that high. Therefore, the addition of subtitles does not affect the output in a significant manner.

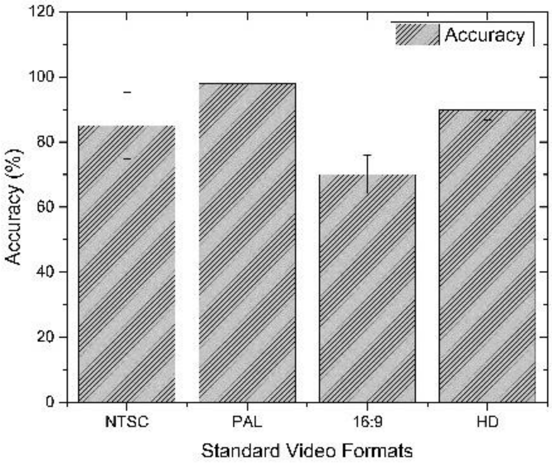

Fig. 11. Here user can observe the use of different formats for the same sample set and how it affects video detection. As for the PAL format, it is the standard for the database and hence shows a 100% return. The HD format is also standardized and hence shows similar results. The differences between PAL and NTSC cause a drop in accuracy as well as 16:9 ratio. Hence these search terms are undesirable.

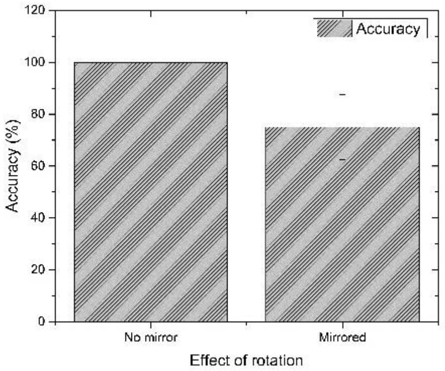

Fig. 12. This is the effect of mirroring the video image. For a non-mirrored image, i.e. a normal image, the output is perfect as expected. Mirroring introduces some error especially in a natural scene where the number of edges is high compared to a standard scene with well-defined edge detection values. Hence mirroring the image reduces accuracy.

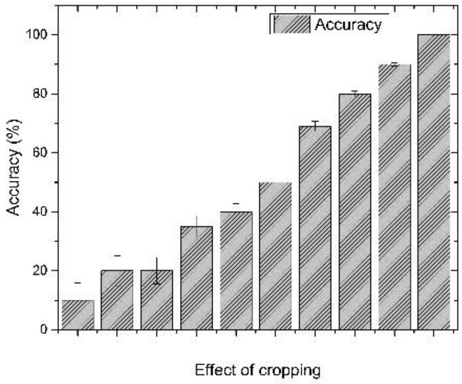

Fig13 If user crops out and provide the algorithm with only a fraction of the image, then we observe reduction in accuracy of the output. The above graph increments for values of 10% and goes on up-to 100%which is no cropping. It is observed that accuracy as well as error rate depending on the amount of cropping applied scene changes as it altered.

ACKNOWLEDGMENT

The authors would like to thank principal and all supporting staff of J.T.M. College of engineering. Authors would also like to thank North Maharashtra University.

REFERENCES

[1] H. Kopka and P. W. Daly, A Guide to L^{A}T$_{E}$X, 3rd ed. Harlow, England: Addison-Wesley, 1999.
[2] Karpenko, A., Aarabi, P. (2011). Tiny videos: a large data set for nonparametric video retrieval and frame classification. IEEE Transactions on Pattern Analysis and Machine Intelligence, 33(3), 618-630.
[3] Karpenko, A., Aarabi, P. (2008, December). Tiny videos: Non-parametric content-based video retrieval and recognition. In Multimedia, 2008. ISM 2008. Tenth IEEE International Symposium on (pp. 619-624). IEEE.
[4] Chaudhry, R., Ivanov, Y. (2010, September). Fast approximate nearest neighbor methods for non-Euclidean manifolds with applications to human activity analysis in videos. In European Conference on Computer Vision (pp. 735-748). Springer Berlin Heidelberg.
[5] Mehendale, N. D., Shah, S. A. (2015, April). Image fusion using adaptive thresholding and cross filtering. In Communications and Signal Processing (ICCSP), 2015 International Conference on (pp. 0144-0148).IEEE.
[6] Karpenko, A., Aarabi, P. (2009, December). Tiny videos: a large dataset for image and video frame categorization. In Multimedia, 2009. ISM'09. 11th IEEE International Symposium on (pp. 281-289). IEEE.
[7] Choros, K. (2010). Video structure analysis and content-based indexing in the Automatic Video Indexer AVI. In Advances in Multimedia and Network Information System Technologies (pp. 79-90). Springer Berlin Heidelberg.
[8] Park, D., Ramanan, D. (2015). Articulated pose estimation with tiny synthetic videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 58-66).
[9] Yoshida, K., Murabayashi, N. (2008, July). Tiny LSH for content-based copied video detection. In Applications and the Internet, 2008. SAINT 2008. International Symposium on (pp. 89-95). IEEE.
[10] Song, J., Yang, Y., Huang, Z., Shen, H. T., Hong, R. (2011, November). Multiple feature hashing for real-time large scale near-duplicate video retrieval. In Proceedings of the 19th ACM international conference on Multimedia (pp. 423-432). ACM.
[11] Revaud, J., Douze, M., Schmid, C., Jegou, H. (2013). Event retrieval in large video collections with circulant temporal encoding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2459-2466).