International Journal of Electrical and Computer Engineering (IJECE)
Vol. 8, No. 5, October 2018, pp. 3221~3231
ISSN: 2088-8708, DOI: 10.11591/ijece.v8i5.pp3221-3231

A Deterministic Eviction Model for Removing Redundancies in Video Corpus

Jyoti Malhotra, Jagdish Bakal
Department of Computer Science & Engineering, G. H. Raisoni College of Engineering, Nagpur, affiliated to RTM University Nagpur, India

Article Info

Article history:

Received Jun 15, 2017
Revised Jan 25, 2018
Accepted Aug 26, 2018

Keywords:

Archive
Comparison window
Deduplication
Hash codes
Multimedia
Similarity scores

ABSTRACT

The traditional storage approaches are being challenged by huge data volumes. In multimedia content, every file does not necessarily get tagged as an exact duplicate; rather they are prone to editing and resulting in similar copies of the same file. This paper proposes the similarity-based deduplication approach to evict similar duplicates from the archive storage, which compares the samples of binary hashes to identify the duplicates. This eviction is done by initially dividing the query video into dynamic key frames based on the video length. Binary hash codes of these frames are then compared with existing key frames to identify the differences. The similarity score is determined based on these differences, which decides the eradication strategy of duplicate copy. Duplicate elimination goes through two levels, namely removal of exact duplicates and similar duplicates. The proposed approach has shortened the comparison window by comparing only the candidate hash codes based on the dynamic key frames and aims the accurate lossless duplicate removals. The presented work is executed and tested on the produced synthetic video dataset. Results show the reduction in redundant data and increase in the storage space. Binary hashes and similarity scores contributed to achieving good deduplication ratio and overall performance.

Corresponding Author:

Jyoti Malhotra,
Department of Computer Science & Engineering,
G. H. Raisoni College of Engineering, Nagpur, affiliated to RTM University Nagpur, MS, India.
Email: jyoti.j.malhotra@gmail.com

1. INTRODUCTION

The multi-objective deduplication technique which removes the duplicate files is making its relevance and applicability in the storage world. In the view of digital data, its growth [1] and its storage challenges; the progress of deduplication can be seen for textual data [2], [3], [4], [5], [6] and relational data [7], [8]. Comparatively less research is observed for multimedia files. There is a huge multimedia universe within the data universe which is growing fast in the popularity because of social media, online courses, distance learning, presentations, self-study and growth of smartphone devices including mobile phones and tablets. With the growth of internet usage along with an increase in internet bandwidth, there has been a significant increase in online video data streaming along with video data downloading. Videos are becoming a powerful e-learning trend. From education viewpoint; students, researchers, professors, and individuals also prefer watching videos instead of reading manuscripts or books. The majority of them prefer downloading the videos, which results in a huge video basket resulting the need for data reduction in case of redundancy. Widely used methods for reduction are compression and deduplication. MPEG [9] addresses the compression of individual videos but not similar videos.

Inline deduplication [10] limits the storage of a duplicate file. The search for duplicates occur before the file is stored, but this is very time-consuming and resource-intensive process; particularly with the video files wherein CPU and memory resources are kept busy till the process of video streaming, and a duplicate

check is over. While in this work, we focus on archive storage and propose the post-process deduplication on the videos with the dynamic splitting of the videos and reduced comparison candidate key frames based on video length and duplicate removal window.

The prime objective of deduplication is to reduce multiple redundant copies. Deduplication performance depends on the factors such as- comparison window, space savings, metadata lookup and above all the accuracy percentage of duplicate content. The performance rises with more duplicates and more similarity. Unlike other text deduplication methods, duplicate removal ratio is improved in multimedia content by identifying similar duplicates. Video similarity can occur with respect to resolution, formatting, frame rate and content difference. The existing written work focuses on the progress of similar features among the videos. In [11], the author proposes the framework, aiming an efficient and fast duplicate removal process. Key factors are- hash table indexing, video similarity and temporal locality of the frames. The author creates multiple buckets and each bucket stores the extracted frames sharing the same hash code. Frames extracted in the bucket are queried for finding the similarity and removing the duplicates.

Near-duplicate removal based on shot-similarity is proposed in [12], to identify the duplicate scene shots. It focuses on the same event captured with different cameras, angles, and different temporal offsets. A similarity between a set of trajectories is calculated for finding the duplicates. The similarity is measured for strict near duplicates, object duplicates, and scene duplicates. The authors of [13], [14] have proposed a secure video deduplication framework based on H.264 compression and a block level codec used for high definition videos. A security parameter is focused on encryption, proof of storage, proof of ownership and retrieval. However, shot and scene similarity and security aspects are not within the scope of this paper.

Hashing based on multiple features for detecting near-duplicate videos is illustrated in [15]. Similarity and performance are evaluated by calculating Euclidean distance and mean average precision. The research presented in [16] introduces an efficient yet effective novel global descriptor; where non-geometric distortions and all other transformations for video format files result in detection of near duplicate video copy tasks. For processing videos, they are converted into image frames; authors of [17] investigated Euclidean distance and shape based similar image retrieval by means of Zernike moments. The validity of digital images is verified by comparing histogram and principal component analysis of two images by the authors of [18].

This paper focuses on removing duplicate and similar videos from the backup or archive storage. The archive is a collection of older and inactive data which is hardly ever accessed. It deals with long-term data preservation. We intend to reduce the multimedia data size of the archives by performing deduplication on the videos. Comparison window is shortened by an effective sampling of the required key frames based on length of the video. In addition, after finding the similarity, the copy which occupies less space is preserved in the storage. It is a two level approach, where the video is checked for the exact duplicates; this results in the removal of a duplicate copy if an exact content match is found. If the exact duplicate copy is not found, then the system searches for the near duplicate ones and they are removed by similarity ranking of the frames based on their binary hashes.

The order of the remaining sections in this article is structured as follows. Description of design and algorithm for removing near-duplicate videos is expressed in Section II. Besides it, the section also illustrates a model describing the data growth and data removal rate in an archive. Performance analysis and results stating the storage savings are illustrated in section III. The work is concluded in Section IV.

2. SYSTEM PROCEDURE

2.1. Video Redundancy Elimination

Deduplication is a partition dichotomy of the videos into two opposed classes={duplicates, non-duplicates} based on their fingerprint values. Videos are prone to editing trends; which results in its alteration and filtration. With respect to generic video editing trend; the duplicate class can be further alienated into two subclasses: {near-duplicates, no-duplicates} based on similarity percentage among the videos. Considering the scenario that there are V videos, n copies of the video and every video takes the storage space V_s, then the total storage space overhead $\mathrm{Total}_{\mathrm{space}}$ and the duplicate degree $\mathrm{Dup}_{\mathrm{degree}}$ can be computed as given in Equation (1) and (2):

$$Total_{space} = \sum_{i=1}^{V} n_i V_{s_i} \quad (1)$$

$$Dup_{degree} = \frac{Total_{space}}{space\ without\ duplicates} \quad (2)$$

The target is how to find these duplicate copies n to minimize the required storage space and duplicate degree. The system architecture for the application which aims to control the media crowd in the archive and lessen the number of copies n is as exposed in Figure 1. The application implements duplicate video elimination process in two phases namely, duplicate check based on the video signatures and similar frames. Description of these modules is given in Algorithm 1 and Algorithm 2. Table I brief various notations and methods used in these algorithms.

Table 1. Glossary of Notations and Methods

Notations		Methods	
Notation	Meaning	Method	Meaning
Q_v	Query Video	Video Dedup()	Find duplicate videos
H_a	Hash algorithm for video signature	Get Hash Fingerprints()	Load fingerprints available in the metadata
$Hash_{MD}$	Metadata hash fingerprints	Finger Print()	Calculate hash signature of video
Dup_{type}	Type of duplicate – near/no	Update Metadata()	Pass the reference link for the duplicate video
HC_{MD}	Collection of hash codes in metadata	Similarity Check()	Check similarity score between the videos
KF_{Qv}	Query video key frames	Get Frame Hash Codes()	Load hash codes available in the metadata
HCQ_{kf}	Collection of hash codes in query video key frames	Extract Frames()	Split query video into key frames
H_{kf}	Hash algorithm for key frames	Calculate Hash()	Calculate hash code of key frames
$CandHCQ_{kf}$	Query video candidate hash codes	Select Candidates()	Select comparison candidate key frames
$CandHC_{MD}$	Metadata video candidate hash codes		
$Dist_y^x$	Distance between hash codes x and y		
τ	Parameter for sampling candidate hash codes		
Sim_{score}	List of similarity score of candidate hash codes		
$finalSim_{score}$	Final similarity ranking		
ω	Decision parameter for near-duplicates		
γ	Parameter for selecting candidate key frame		

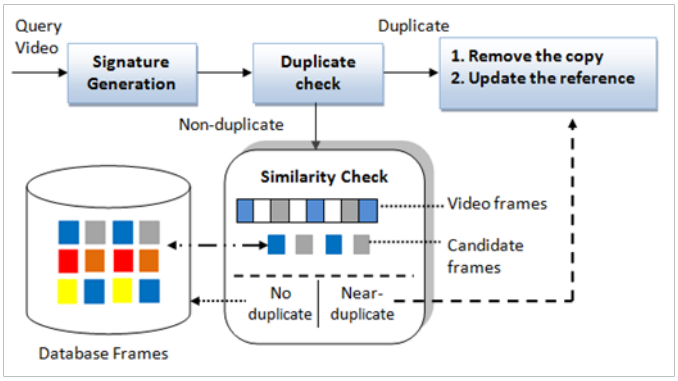

Figure 1. System architecture

Algorithms seeking the reduction of storage space are presented below. Algorithm 1 describes the high-level operational flow of the deduplication process. Input is the query video and hash algorithm [19] which uses avalanche effect. The fingerprint or hash signature of query video is matched against the existing fingerprints in the metadata. On the exact match; a duplicate copy is removed, and metadata is updated by passing the reference link for the duplicate copy. If a match is not found query video is given to the method described in Algorithm 2 for finding its similar copy. Algorithm 2 exhibits the process for finding similarity

scores of the videos. Algorithm loads the key frame hash codes available in the metadata corpus. The length of metadata corpus is as given in Equation (3), where v is number of videos and kf is number of key frames.

$$|HC_{MD}| = \sum_{i=i}^{v} \sum_{j=1}^{kf} HC_{MD_{ij}} \quad (3)$$

Query video is divided into KF_{Qv} key frames. Length of KF_{Qv} is dynamic and it differs as per the video length. Hash codes [20] are calculated for all these key frames. Understanding the length of metadata corpus$|HC_{MD}|$, candidate hash codes $CandHCQ_{kf}$ and $CandHC_{MD}$ are selected from query hashcodes and metadata hashcodes based on the parameter τ as illustrated in Algorithm 3. The value of τ is kept dynamic as per the value of KF_{Qv}.For each candidate hash codes, distance is calculated by comparing the codes as per Equation (4). Similarity score is calculated from these distances as shown in Equation (5), for each candidate hash codes l. After calculating the individual similarity score, the final similarity ranking of query video is calculated by taking an average of the individual scores. If this final score is bigger than the stated threshold ω , query video is measured as a near-duplicate one as described in Equation (6). The final score less than the threshold ω, means query video is not a duplicate and is considered as a new video file. This new video is stored in the storage and key frame details and their respective hash codes are stored in the metadata.

Considering x = $CandHCQ_{kf}$ and y = $CandHC_{MD}$we get; $Dist_y^x = \sum_{bit=1}^{k} |x_{bit} - y_{bit}|$ (4)

$$Sim_{score} \bigvee_{i=1}^{l} Dist_{CandHC_{MD_i}}^{CandHCQ_{kf_i}} \quad (5)$$

$$Q_v = \begin{cases} \text{near} - \text{duplicate}, & finalSim_{score} \geq \omega \\ \text{no} - \text{duplicate}, & otherwise \end{cases} \quad (6)$$

Algorithm 1: Video deduplication based on hash signatures
Input: Query video Q_v; Hash Algorithm H_a
Output: Duplicate type – duplicate or no-duplicate
Begin video Dedup
  $Hash_{MD}[\,] \leftarrow$ getHashFingerprints()
  $fp \leftarrow$ fingerPrint (H_a, Q_v);
  Search fp of the video in metadata $Hash_{MD}[\,]$
  if $fp = Hash_{MD}[\,]$ then
    Video is duplicate; remove the copy
    Update Metadata()
  else
    Dup_{type} = similarityCheck(Q_v)
    if Dup_{type} = Near-duplicate then
      Video is Unique; save the copy
    else // video is duplicate
      update Metadata()
  end if
End

Algorithm 2: Video deduplication based on hash similarities
Input: Query video Q_v
Output: Duplicate type{near-duplicate, no-duplicate}
Begin similarity Check
  $HC_{MD}[\,][\,] \leftarrow$ getFrameHashCodes()
  $KF_{Qv}[\,] \leftarrow$ extractFrames (Q_v); //Dynamic
  for each (key frame in $KF_{Qv}[\,]$)
    $HCQ_{kf}[\,] \leftarrow$ calculate Hash(KF_{Qv}, H_{kf})
  End for
  //select candidate hash codes
  $CandHCQ_{kf}[\,] \leftarrow$selectCandidates($HCQ_{kf}[\,], \tau$)
  $CandHC_{MD}[\,] \leftarrow$selectCandidates($HC_{MD}[\,], \tau$)
  for each(candidate in $CandHCQ_{kf}[\,]$ and $CandHC_{MD}[\,]$)
    Calculate $Sim_{score}[\,]$
  end for
  $finalSim_{score} \leftarrow$ average($Sim_{score}[\,]$)
  if ($finalSim_{score} \geq \omega$) then
    Dup_{type} = “Near-duplicate’
  else
    Dup_{type} = “No-duplicate’
End

Algorithm 3 illustrates the steps for selecting sample candidate key frames for the comparison.

Algorithm 3: Candidate key frame selection
Input: $HC[\,]$ Collection of hashcodes in metadata or query video and τ deciding the candidates
Output: $Cand_{HC}[\,]$
Begin select Candidates
for each (hash code k in $HC[\,]$)
if ($\tau == 1$) then
$Cand_{HC}[\,] \leftarrow HC[k]$
else
$Cand_{HC}[\,] \leftarrow HC[k+\gamma]$
End

2.1. Probability Model for Similar Video Deletion

This section illustrates the probabilistic and deterministic model describing the fundamental relationships between exact and near-duplicate videos, their storage space requirement and the rate of their growth and removal. The probabilistic experiment involves random variables namely, video hash signatures and similarity score. Given a sample space **Ω** of $\boldsymbol{V}$ videos; where the probability of duplicate video is p and non-duplicate is (1-p), then the probability of getting exactly k duplicate videos based on the hash signatures is as shown in Equation (7).

$$P(\text{k Duplicates}) = \binom{V}{k} p^k (1-p)^{V-k} \quad (7)$$

As videos are editing prone, the process of redundant video removal as near-duplicates is further processed based on its similarity score Sim_{score} and a threshold value ω. Threshold ω is the parameter which illustrates about the similarity score, and the value is less than 1. Probability that the Sim_{score} takes values in the interval from ω to 1 is same as the probability of all possible outcomes where this score can come. The Probability density and expected value of Sim_{score} for its numerical possibilities s are described in Equation (8) and (9).

$$P(\omega \leq Sim_{score} \leq 1) = \int_{\omega}^{1} f_{Sim_{score}}(s)ds \quad (8)$$

$$E(Sim_{score}) = \int_{\omega}^{1} s\, f_{\mathrm{Sim_{score}}}(s)\mathrm{ds} \quad (9)$$

A discrete time model illustrated in Figure 2 describes the evolution from video arrival for the backup to its storage process. The model has six possible states and seven possible transitions between the states annotated with their representative transition probabilities. At any known time; after the arrival, video can move to either of the states such as duplicate, similar or non-duplicate based on the video characteristics (fingerprint, similarity score). Once the video is identified as a duplicate or more similar, the system goes to the video reference state and stays in the same state. The system stays in the unique video state if there is no match on fingerprints and videos are not similar.

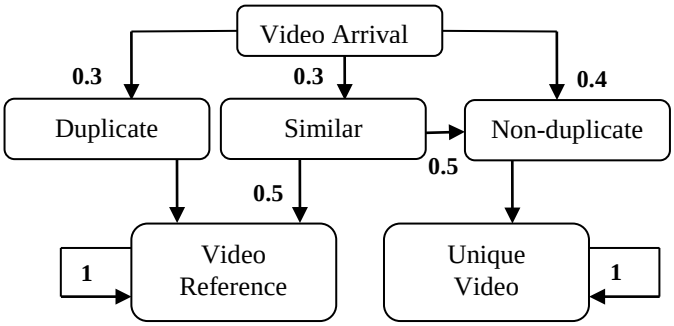

Figure 2. Probabilistic discrete time model for video storage

After n transitions the total probability of system $ב_{st}$ starting at state s and ending at state $t, \forall_{s,t}$ and k intermediate states is given in Equation (10).

$$ב_{st}(n) = \sum_{i=1}^{k} ב_{si}(n-1)P_{it} \quad (10)$$

Total probability of system starting in state 1 – "*video arrival"* and ending in the state 5 – "Reference" is as illustrated below in the Equations (11) to (13) {state 2- "*Duplicate"* and state 3- "*Similar"*}

$$ב_{15}(n) = ב_{12}(n-1)P_{25} + ב_{13}(n-1)P_{35} \quad (11)$$

$$ב_{15}(n) = 0.3 * 1 + 0.3 * 0.5 \quad (12)$$

$$ב_{15}(n) = 0.45 \quad (13)$$

Transition matrix for the model described in Figure 2 is as illustrated in Table 2 below. The probability of going from "video arrival" to "duplicate" is 30%. Initial state of system is [1,0,0,0,0,0] i.e. first state is "*video arrival"*. At one-time unit 30% videos are gone into "*duplicate"* state, 30% videos are gone into "*similarity"* state, and 40% of the videos are in "*non-duplicate"* state.

Table 2. Probability Transition Matrix

States*	VA	DUP	SIM	ND	UNI	REF
VA	0	0.3	0.3	0.4	0	0
DUP	0	0	0	0	0	1
SIM	0	0	0	0.5	0	0.5
ND	0	0	0	0	1	0
UNI	0	0	0	0	1	0
REF	0	0	0	0	0	1

*States(VA-Video arrival, DUP-Duplicate, SIM-Similar, ND-non-duplicate, UNI-Unique video, REF-Video reference)

Probability Distribution tells that after running the process for n videos (time units), the system travels to a "unique video" state with probability p_u and with probability p_r, system goes to a "reference video" state. The simulation was done on n=100 videos and total probability mentioned in the Equation (10) is illustrated below along with the probability distribution with an assumption that the system already had 30 duplicate videos.

P^n is given by: [0.00 0.00 0.00 0.00 0.55 0.45]
[0.00 0.00 0.00 0.00 0.00 1.00]
[0.00 0.00 0.00 0.00 0.50 0.50]
[0.00 0.00 0.00 0.00 1.00 0.00]
[0.00 0.00 0.00 0.00 1.00 0.00]
[0.00 0.00 0.00 0.00 0.00 1.00]

Probability distribution after running the process for 100 videos is - [0.0, 0.0, 0.0, 0.0, 0.55, 0.75]
i.e. $p_u = 0.55 \; and \; p_r = 0.75 \; i.e. (0.45 + 30)$
The deterministic model defines the formulation of inputs and outputs over the time where inputs are the number of duplicate arrivals and outputs are the number of duplicate removals. The rate of change of storage space depends on the rate of video arrivals and rate of video removals as stated in Equation (14).

$$R(storage_{space}) = R(video_{arrivals}) - R(video_{removals}) \quad (14)$$

Considering s_0 as the initial value of storage space, with duplicate video arrival and duplicate video removal rate per file as α and β, the storage size $S(t)$ at any time t can be obtained by rewriting Equation (14) as given in Equation (15).

$$R\left(storage_{space}\right) = S(t) = \alpha\, S(t) - \beta\, S(t) \tag{15}$$

$$\frac{dS}{dt} = \alpha S - \beta S \tag{16}$$

Let $r = \alpha - \beta$ in Equation 16, we obtain

$$\frac{dS}{dt} = rS \tag{17}$$

After solving this differential equation, we get -

$$S(t) = s_0 e^{rt} \tag{18}$$

According to Equation (18), the rate of change of storage space on the removal of duplicates depends on the value of r, and it should be greater than zero. This paper aim to keep the value of r (duplicate video arrivals-duplicate video removals) ≥ 0 by removing more duplicate copies.

3. RESULTS AND DISCUSSION

In this section, we firstly compare the compression ratio between the videos by comparing the hashes generated by avalanche effect and the binary hashes. Secondly, the performance of similarity score and precision-recall is observed. Experimentation was implemented on the combined dataset by taking a few videos from the dataset [21]. A synthetic dataset is generated according to the principles discussed in [22] on the videos from [23], YouTube videos and a personal video collection. Dataset videos are either exact duplicates or approximate similar, but different in resolution, encoding formats, content addition, length, etc. Videos used in the standard datasets are too small. Unlike [21], the dataset used for our application contains videos of all time durations from 1 minute to more than 10 minutes.

We first adopt the method of avalanche and it is compared against the binary hashes. Figure 3 shows the results of three various reduction methods (HE- Hash based exact duplicate removal, HS- Hash based similar duplicate removal, SS- Similarity-based similar duplicate removal) on two different datasets; dataset1 is a video collection with exact duplicates; dataset2 contains exact and similar duplicates. Reduction methods are HE- Hash based exact duplicate removal, HS - Hash based similar duplicate removal and SS - Similarity-based similar duplicate removal. HE is adopted on dataset1 and it is observed that data size got reduced to 12%, 28%, and 34% respectively with compression(C), deduplication (DD) and compression plus deduplication (CDD). HS and SS techniques are implemented on the common dataset2 with exact and similar duplicates. With HS the reduction percentage of C, DD, and CDD is observed as 5%, 19%, and 22% because hash based algorithm doesn't detect similarity features. Significant improvement in the reduction percentage is seen with SS approach as 5%, 46%, and 50% fulfilling the aim of data reduction and an increase in the storage space.

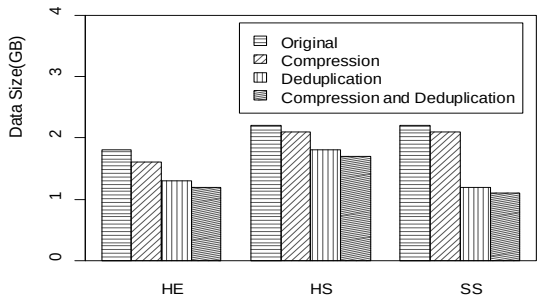

Figure 3. Compression vs deduplication performance

Figure 4 plots the graphical representation of the spread of all different video similarity scores among the dataset. Whiskers show the range of similarity scores for the entire dataset. Lowest score in this sample is 0 and maximum it goes to 1. Median similarity score in the dataset is approximately at 0.7. The first plot shows the representation of similarity score of all the test data. The second plot shows the result of the near duplicate ones when the threshold ω was kept 0.7. The rectangular box ranges from the 0.7 to 0.9 with whiskers extending up to maximum 1.0. The last plot is the result of no duplicate ones with scores below 0.7 and an outlier at score 0.0.

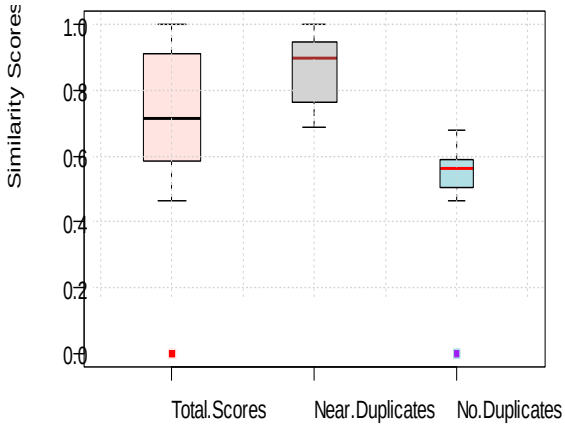

Figure 4. Distribution of similarity scores (Box plots depicting median and quartile values of similarity scores for entire dataset, near duplicates and no duplicates)

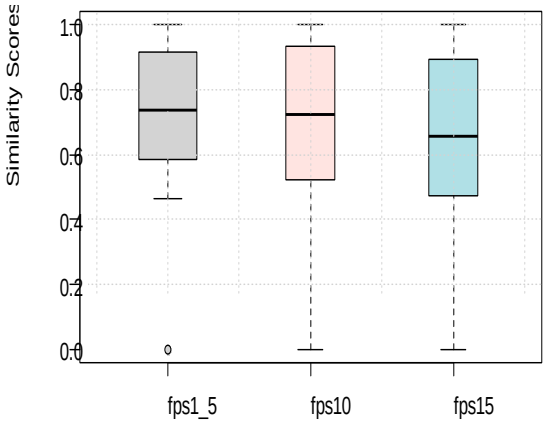

Figure 5. Comparison of similarity scores with different key frame sizes (Box and whisker plots depicting the comparison of similarity scores for different key frame sizes)

Figure 5 describes the comparison of similarity scores with different key frame sizes. As per algorithm 2 frame extraction is kept dynamic as per the video length. From the figure, we conclude that when our proposed system sets the fps (frames per second) ranging 1-5, we get the maximum similarity score from 0.45 to 0.95 with whiskers extending up to 1.0 along with an outlier towards 0, and median 0.75 is closer to the upper quartile. Whereas for fps 10, the similarity score varied from 0.55 to 0.9 with whiskers extending up to lower quartile 0.0 and upper quartile 1.0. In the last case, for fps 15 we got the rectangular box with a range of 0.5 to 0.9. The respective medians indicated from the graph are 0.75, 0.7, and 0.65.

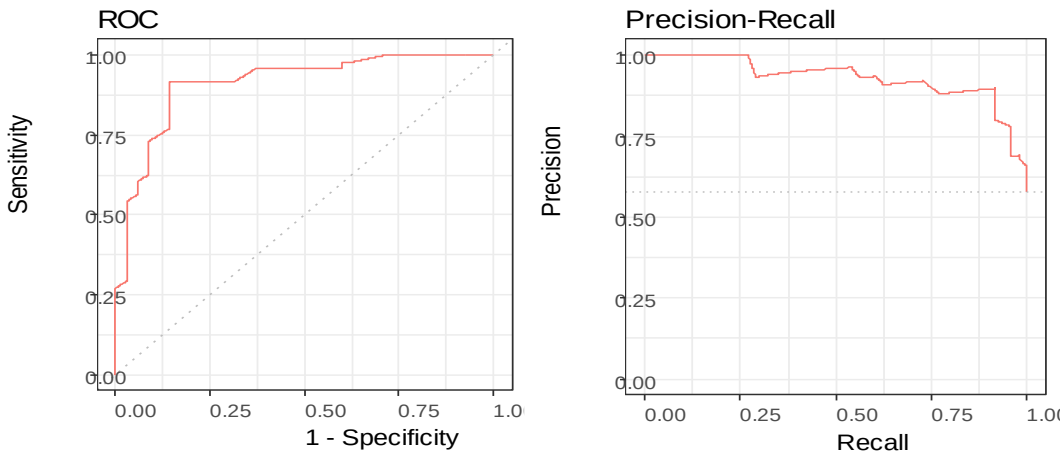

Figure 6. Performance of similar video removal process
(A precision-recall curve depicting the accuracy of similar video detection with ω =0.7. Receiver operating characteristic curve (ROC) viewing the performance of the near duplicates and no duplicates classification)

Figure 6 shows the accuracy and performance of the near similar and no duplicates with the threshold ω as 0.7. Precision is the portion of identified scores that are similar, while recall is the segment of similar scores that are identified. Precision-recall curves are used to identify the confident zone of the overall performance, where a larger area indicates the better performance. Accuracy of the experiment results is measured by the region below the ROC curve. The region towards 1.00 represents a perfect classification. As shown in the Figure 6, area of ROC curve is approximately 0.8-0.9 i.e. towards 1. Taking false positive and false negative into account, the weighted mean i.e. the F1 score is evaluated and the value is 0.89.

Figure 7 shows three ROC curves for threshold ω values 0.7, 0.75, 0.8 respectively. The excellent accuracy result is being depicted by red curve (ω=0.7) by calculating the area of ROC and it is observed towards 1. PR curve is thought to be more informative in finalizing the threshold value which gives the similarities as predicted. The larger curve area is been observed by red curve with ω=0.7. Hence, the similarity score threshold for detecting more similar videos is measured as 0.7. Videos with $Sim_{score} >= 0.7$ are considered as duplicates and are deleted from the corpus.

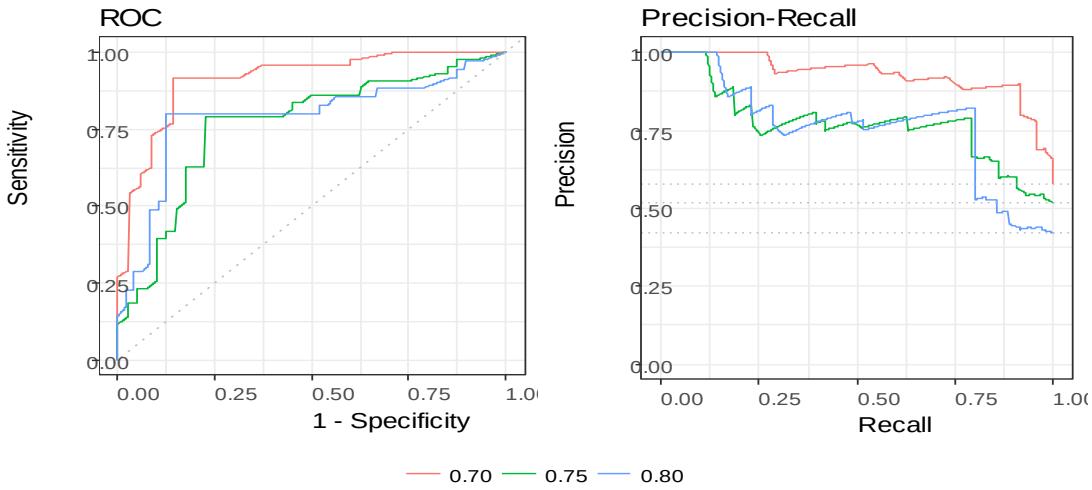

Figure 7. Performance using different similarity score threshold

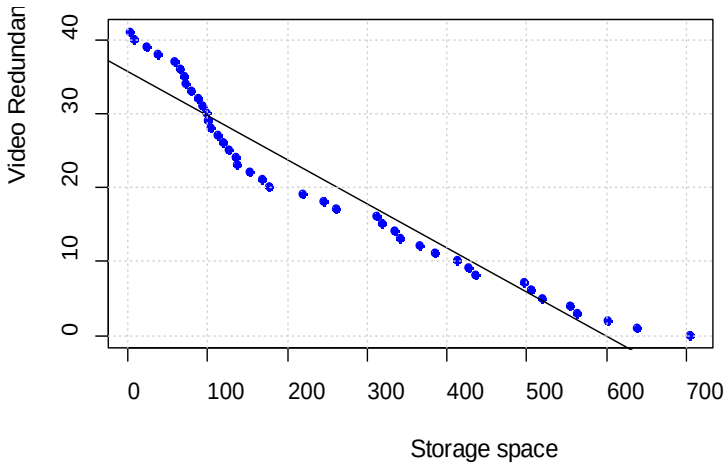

Figure 8. Rate of change of storage space

As per the deterministic model, discussed in Section II, the rate of change of storage space depends on the rate of duplicate video arrivals and rate of duplicate or similar video removals. Removing duplicates is inversely proportional to storage space. Figure 8 shows this rate of change and describes that as the video file redundancy count decreases i.e. as duplicate or similar video files are removed; there is an increase in the storage space.

4. CONCLUSION

This paper presents the efficient duplicate video removal process based on the video similarities. The reported work shows that deduplication with similarity scores perform well as compare to deduplication with hash signatures on the dataset where there are more similar contents. Original data size has significantly reduced with 50% deduplication ratio. The accuracy of similar video detection is observed with the precision-recall curve with larger curve area and ROC curve area towards 1. The accuracy of video classification as similar duplicates and no duplicates is measured with the F1 score (mean of precision and recall) of 89%.

ACKNOWLEDGEMENTS

We wish to express our sincere thanks to every anonymous person for their precious review, suggestions, and remarks to make this work the progress.

REFERENCES

[1] https://www.emc.com/collateral/analyst-reports/idc-the-digital-universe-in-2020.pdf
[2] A Agrawal, J Malhotra, "Clustered Outband Deduplication on Primary Data". *In Computing Communication Control and Automation (ICCUBEA)*, 2015 International Conference on 2015 Feb 26 (pp. 446-450). IEEE.
[3] Jin San Kong, *et.al*., "Two-Level Metadata Management for Data Deduplication System". IST 2013, ASTL Vol. 23, pp.299-303.
[4] D Kim, *et.al.*, "SAFE: Structure-aware file and email deduplication for cloud-based storage systems". In Data Deduplication for Data Optimization for Storage and Network Systems 2017 (pp. 97-115). *Springer International Publishing.*
[5] V. A. Narayana, P. Premchand, and A. Govardhan. "Performance and comparative analysis of the two contrary approaches for detecting near duplicate web documents in web crawling." *International Journal of Electrical and Computer Engineering (IJECE)*, 2.6, 2012, pp. 819-830.
[6] F Yinjin, *et.al.*, "Application-Aware Source Deduplication for Cloud Backup Services of Personal Storage". *IEEE Transactions on Parallel and Distributed Systems*. 2013.
[7] De Carvalho, Moises G., *et al.*, "A genetic programming approach to record deduplication," *IEEE Transactions on Knowledge and Data Engineering*, 24(3): 399-412, Mar 2012.
[8] A Culotta and A McCallum. "Joint deduplication of multiple record types in relational data." In Proceedings of the 14th *ACM international conference on Information and knowledge management* 2005 Oct 31 (pp. 257-258). ACM.
[9] Le Gall, Didier. "MPEG: A video compression standard for multimedia applications." Communications of the ACM 34.4 (1991): 46-58.
[10] N. Mandagere, *et.al*., "Demystifying data deduplication". In Proceedings of the *ACM/IFIP/USENIX Middleware'08 Conference Companion* 2008 Dec 1 (pp. 12-17). ACM.
[11] Li Y, Xia K. "Fast Video Deduplication via Locality Sensitive Hashing with Similarity Ranking". *In Proceedings of the International Conference on Internet Multimedia Computing and Service 2016* Aug 19 (pp. 94-98). ACM.
[12] Satoh SI, *et.al.*, "Scene duplicate detection from videos based on trajectories of feature points". In Proceedings of the *International workshop on Workshop on multimedia information retrieval 2007* Sep 24 (pp. 237-244). ACM.
[13] F Rashid, *et.al.*, "Proof of storage for video deduplication in the cloud". *In Big Data (BigData Congress), 2015 IEEE International Congress on 2015* Jun 27 (pp. 499-505). IEEE."
[14] F Rashid, *et.al*., "A Secure Video Deduplication Scheme in Cloud Storage Environments Using H. 264 Compression" *In Big Data Computing Service and Applications (BigDataService)*, 2015 IEEE First International Conference on 2015 Mar 30 (pp. 138-146). IEEE.
[15] J Song, *et.al.*,. "Multiple feature hashing for real-time large scale near-duplicate video retrieval". In Proceedings of the *19th ACM international conference on Multimedia 2011* Nov 28 (pp. 423-432). ACM.
[16] A Rouhi and James A. Thom. "A compressed-domain robust descriptor for near duplicate video copy detection." *In Proceedings of the 29th International Conference on Image and Vision Computing New Zealand*, 2014 Nov 19 (pp. 130-135). ACM .
[17] G. Sucharitha, and Ranjan K. Senapati. "Shape Based Image Retrieval using Lower Order Zernike Moments", *International Journal of Electrical and Computer Engineering (IJECE)*, 7(3), 2017, pp. 1144-1153.
[18] P. Amudhavalli, , N. Rajalakshmi, and K. S. Sindhu. "Deduplication Analysis of Products In Digital Marketing." *Indonesian Journal of Electrical Engineering and Computer Science(IJEECS)*, 10(1), 2018, pp. 392-399.
[19] RFC 3174 - US Secure Hash Algorithm 1, September 2001. https://tools.ietf.org/html/rfc3174
[20] http://www.phash.org/
[21] http://vireo.cs.cityu.edu.hk/webvideo/
[22] V Tarasov, *et.al.*, "Generating Realistic Datasets for Deduplication Analysis". *In USENIX Annual Technical Conference 2012* Jun 13 (pp. 261-272).
[23] http://geovid.org/dataset/videos/

BIOGRAPHIES OF AUTHORS

Jyoti Malhotra is M.E. in Computer Science and Engineering from the SPPU University, Pune in 2005. She is working as an Assistant Professor in Engineering College, Pune. She has 15+ years of teaching experience. She is pursuing her Ph.D. degree in Computer Science and Engineering; RTM Nagpur University under the guidance of Dr. J. W. Bakal. Her research interest lies in Data Storage patterns, Big Data, Software Testing and Theory of Computation. She has worked in C, Java, and Linux Programming. She is the life member of Computer Society of India. She has publications in National, International conferences and Journals like IEEE conference, Springer conferences, etc.

Dr. J. W. Bakal received M. Tech. (EDT), from Dr. Babasaheb Ambedkar Marathwada University, Aurangabad. Later, He completed his Ph.D. in the field of Computer Engineering from Bharati Vidyapeeth University, Pune. He is presently working as Principal at the S.S. Jondhale College of Engineering, Dombivali (East) Thane, India. In the University of Mumbai, he was on honorary assignment as a chairman, board of studies in Information Technology and Computer Engineering. He is also associated as chairman or member of Govt. committees, University faculty interview committees, for interviews, LIC or various approval works of institutes. He has more than 27 years of academics experience including HOD, Director in earlier Engineering Colleges in India. His research interests are Telecomm Networking, Mobile Computing, Information Security, Sensor Networks and Soft Computing. He has publications in journals, conference proceedings in his credit. During his academic tenure, he has attended, organized and conducted training programs in Computer, Electronics & Telecomm branches. He is a Professional member of IEEE. He is also a life member of professional societies such as IETE, ISTE INDIA, CSI INDIA. He has prominently contributed in the governing council of IETE, New Delhi India.