Near-Duplicate Video Retrieval by Aggregating Intermediate CNN Layers

Giorgos Kordopatis-Zilos[1,2(✉)], Symeon Papadopoulos[1], Ioannis Patras[2], and Yiannis Kompatsiaris[1]

[1] Information Technologies Institute, CERTH, Thessaloniki, Greece
{georgekordopatis,papadop,ikom}@iti.gr
[2] Queen Mary University of London, Mile end Campus, London E14NS, UK
i.patras@qmul.ac.uk

Abstract. The problem of Near-Duplicate Video Retrieval (NDVR) has attracted increasing interest due to the huge growth of video content on the Web, which is characterized by high degree of near duplicity. This calls for efficient NDVR approaches. Motivated by the outstanding performance of *Convolutional Neural Networks* (CNNs) over a wide variety of computer vision problems, we leverage intermediate CNN features in a novel global video representation by means of a layer-based feature aggregation scheme. We perform extensive experiments on the widely used CC_WEB_VIDEO dataset, evaluating three popular deep architectures (AlexNet, VGGNet, GoogLeNet) and demonstrating that the proposed approach exhibits superior performance over the state-of-the-art, achieving a mean Average Precision (mAP) score of 0.976.

Keywords: Near-duplicate · Video retrieval · CNNs · Bag of keyframes

1 Introduction

Near-duplicate video retrieval (NDVR) is a research topic of increasing interest in recent years. It is considered essential in a variety of applications that involve video retrieval, indexing and management, video recommendation and search, copy detection and copyright protection. The exponential growth of the Web is accompanied by a proportional increase of video content, typically posted and shared through social media platforms. At the moment, YouTube reports more than one billion users and approximately 500 h of video content is uploaded every minute[1]. This fact renders the NDVR problem extremely important.

NDVR is defined in various ways among the multimedia research community as pointed in [12]. Here, we adopt the definition of Wu et al. [21]: **near-duplicate videos are considered to be identical or close to exact duplicate of each other**, but different in terms of file format, encoding parameters, photometric variations (color, lighting changes), editing operations (caption, logo and border insertion), different lengths, and other modifications.

[1] https://www.youtube.com/yt/press/statistics.html (accessed on August 2016).

L. Amsaleg et al. (Eds.): MMM 2017, Part I, LNCS 10132, pp. 251–263, 2017.
DOI: 10.1007/978-3-319-51811-4_21

Motivated by the excellent performance of Convolutional Neural Networks (CNNs) on many computer vision problems, such as image classification, retrieval and object detection, in this paper we propose using intermediate convolutional layers to construct features for NDVR. Although CNN features have been recently used for video retrieval [14,22], it is the first time that **intermediate CNN layers are exploited for NDVR**. A first use of these layers was recently presented on the problem of image retrieval [13,23].

To make use of intermediate convolutional layers for NDVR, we extract layer-level feature descriptors by applying max pooling to the activations of each convolutional layer. In addition, **we propose two layer aggregation techniques**, a first by concatenating the layer vectors in a single vector, and a second by computing layer-specific codebooks and aggregating the resulting bag-of-words representations. Furthermore, **we evaluate three popular deep architectures** [10,16,19] in combination with both layer aggregation schemes by means of a thorough experimental study on an established NDVR dataset (CC_WEB_VIDEO [21]), and **we demonstrate the superior performance of the proposed approach over five state-of-the-art methods**. In particular, the best configuration of the proposed approach achieves a mean Average Precision (mAP) score of 0.976, i.e. a clear improvement over the already high mAP of 0.958 achieved by the Multiple Feature Hashing and Pattern-based approaches of Song et al. [18] and Chou et al. [3], respectively.

2 Related Work

NDVR is a very challenging task, which has attracted increasing research interest in recent years. Liu et al. [12] provide a survey with detailed overviews of the NDVR research problem and a number of recent approaches. These are typically classified based on the level of matching performed to determine the near-duplicate videos: video-level, frame-level and hybrid-level matching.

Video-Level Matching: Here, videos are represented with a global signature such as an aggregate feature vector, a fingerprint or a hash code. Huang et al. [6] proposed a video representation model called Bounded Coordinate System (BCS) which extends Principal Component Analysis (PCA). In [18], Song et al. present an approach for Multiple Feature Hashing (MFH) based on a supervised method that uses multiple image features and learns a group of hash functions that map the video keyframes into the Hamming space. The video signatures are generated by the combination of the keyframe hash codes and they constitute the video representation in the dataset.

Frame-Level Matching: Near-duplicate videos are determined by the comparison between individual frames or sequences of the candidate videos. Douze et al. [4] detect local points of interest, extract the SIFT [11] and CS-LBP [5] descriptors, and create a visual codebook for hamming embedding. Using post-filtering,

they verify retrieved matches with spatiotemporal constrains. In [15], Shang et al. introduce compact spatio-temporal features to represent videos and construct a modified inverted file index. The spatio-temporal features are extracted using a feature selection and w-shingling scheme. Cai et al. [2] presented a large-scale approach by applying a scalable K-means clustering technique to learn a visual vocabulary on the color correlograms of a training set of images and using inverted file indexing for fast retrieval of candidate videos.

Hybrid-Level Matching: A typical such approach is [21], where Wu et al. apply a hierarchical filter-and-refine scheme to cluster and filter out near-duplicate videos. When a video cannot be clearly classified as novel or near-duplicate, they apply an expensive local feature-based NDVR scheme. In a more recent approach [3], Chou et al. filter the non near-duplicate videos with a pattern-based indexing tree and rank candidate videos with m-pattern-based dynamic programming and time-shift m-pattern similarity.

The well-known TRECVID copy detection task [9] is also a specific case of NDVR. However, in the TRECVID copy detection task, the duplicates are artificially generated by applying standard transformations, whereas in case of NDVR duplicates correspond to real content.

3 Approach Overview

The proposed NDVR approach leverages features produced by the intermediate CNN layers of deep architectures (Subsect. 3.1) and introduces a layer-based aggregation scheme for deriving a bag-of-word representation for each video (Subsect. 3.2). The bag-of-words representations of videos are stored in an efficient inverted file index, while video retrieval is carried out based on cosine similarity between *tf-idf* weighted versions of the extracted vectors (Subsect. 3.3).

3.1 CNN Based Feature Extraction

In some recent research works [13,23], pre-trained CNN models are adopted to extract visual features from intermediate convolutional layers. These features are computed through the forward propagation of an image over the CNN network and the use of an aggregation function (e.g., VLAD encoding [7], max/average pooling) on every convolutional layer.

We experiment with three deep network architectures: AlexNet [10], VGGNet [16] and GoogLeNet [19]. All three architectures receive images of size 224×224 as input. For all experiments, input images are resized to fit these dimensions.

To extract frame descriptors, we are following the process of [23]. A pre-trained CNN network $\mathcal{C}$ is employed, with a total number of L convolutional layers, denoted as $\mathcal{L}^1, \mathcal{L}^2, ..., \mathcal{L}^L$. Forward propagating an image I through $\mathcal{C}$ generates a total of L feature maps, denoted as $\mathcal{M}^l \in \mathbb{R}^{n_d^l \times n_d^l \times c^l} (l = 1, ..., L)$, where $n_d^l \times n_d^l$ is the dimension of every channel for convolutional layer $\mathcal{L}^l$ (which depends on the size of the input image) and c^l is the total number of channels.

To extract a single descriptor vector from every layer, an aggregation function is applied on the above feature maps. In particular, we apply max pooling on every channel of feature map $\mathcal{M}^l$ to extract a single value. The extraction process is formulated in Eq. 1.

$$v^l(i) = \max \ \mathcal{M}^l(\cdot, \cdot, i), \quad i = \{1, 2, ..., c^l\} \tag{1}$$

where layer vector v^l is a c^l-dimensional vector that is derived from max pooling on every channel of feature map $\mathcal{M}^l$. The layer vectors are L2-normalized to unit length after their extraction.

Table 1. Total number of CNN channels per layer used by the proposed approach for the three selected deep architectures.

(a) AlexNet

Layer $\mathcal{L}^l$	c^l-dim
conv1	96
conv2	256
conv3	384
conv4	384
conv5	256
total	1376

(b) VGGNet

Layer $\mathcal{L}^l$	c^l-dim
conv2_1	128
conv2_2	128
conv3_1	256
conv3_2	256
conv3_3	256
conv4_1	512
conv4_2	512
conv4_3	512
conv5_1	512
conv5_2	512
conv5_3	512
total	4096

(c) GoogLeNet

Layer $\mathcal{L}^l$	c^l-dim
Inception 3a	256
Inception 3b	480
Inception 4a	512
Inception 4b	512
Inception 4c	512
Inception 4d	528
Inception 4e	832
Inception 5a	832
Inception 5b	1024
total	5488

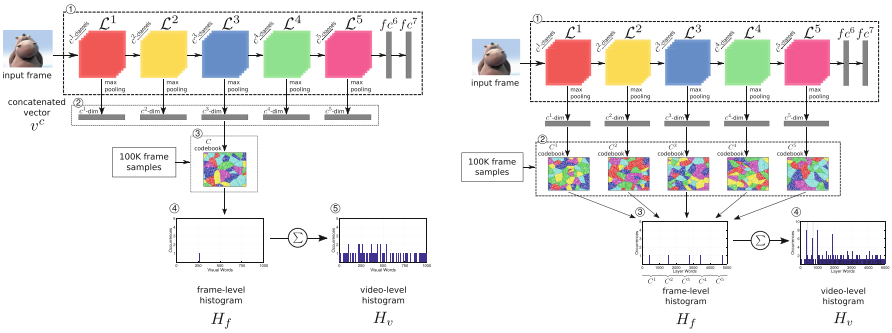
Table 1 depicts the employed CNN architectures and the number of channels in the respective convolutional layers. We extract image descriptors only from the activations in intermediate layers, since we aim to construct a visual representation that preserves local structure in different scales. The fully-connected layer activations are not used. A positive side-effect of this decision is that the resulting descriptor is compact, reducing the total processing time and storage requirements. For the VGGNet and GoogLeNet architectures, we do not use the initial layer activations as features, since those layers are expected to capture very primitive image features (e.g. edges, corners, etc.) that could lead to false matches. For the extraction of the above descriptors, we use the Caffe framework [8], which provides pre-trained models on ImageNet for all three CNN networks[2].

3.2 Feature Aggregation

We then follow two alternative feature aggregation schemes (i.e. ways of aggregating features from layers into a single descriptor for the whole frame): (a)

[2] https://github.com/BVLC/caffe/wiki/Model-Zoo.

vector aggregation and (b) **layer** aggregation. The outcome of both schemes is a frame-level histogram H_f that is considered as the representation of a frame. Finally, a video-level histogram H_v is derived from the respective keyframe representations by plain summing. Figure 1 gives an overview of the two schemes.



(a) Vector Aggregation (b) Layer Aggregation

Fig. 1. The two aggregation schemes and the final video representation.

Vector Aggregation. A bag-of-words scheme is applied on the vector v^c resulting from the concatenation of individual layer features to generate a codebook of K visual words, denoted as $C_K = \{t_1, t_2, ..., t_K\}$. The selection of K has critical impact on the performance of the approach and it is considered a system parameter, which is further explored in Sect. 5. Having generated the visual codebook, every video keyframe is assigned to the nearest visual word. Accordingly, every frame f with feature descriptor v_f^c is aggregated to the nearest visual word $t_f = NN(v_f^c)$, hence its H_f contains only a single visual word.

Layer Aggregation. To preserve the structural information of intermediate CNN layers $\mathcal{L}$, we generate L layer-specific codebooks of K words (denoted as $C_K^l = \{t_1^l, t_2^l, ..., t_K^l\}, l = 1, ..., L$), which we then use to extract separate bag-of-words representations (one per layer). The layer vectors v_f^l of frame f are mapped to the nearest layer words $t_f^l = NN(v_f^l)$, $(l = 1, 2, ..., L)$. In contrast to the previous scheme, every frame f is represented by a frame-level histogram H_f that results from the concatenation of the individual layer-specific histograms, therefore comprising L words instead of a single one.

In both schemes, the visual codebooks are generated using scalable K-Means++ [1] on a sample of 100 K randomly selected video frames. The Apache Spark[3] implementation of the algorithm is used for efficiency and scalability.

[3] http://spark.apache.org (accessed on August 2016).

Keyframe to Video Aggregation. The final video representation is generated using the bag-of-words histograms of its keyframes. Given a video d with $|F|$ keyframes, $F = \{f_1, f_2, ..., f_F\}$, its video-level histogram H_v is derived by summing the histogram vectors corresponding to its keyframes, i.e. $H_v = \sum_{f_i \in F} H_{f_i}$. Note that for the two aggregation schemes, histograms of different sizes are generated. In the first case, the total number of visual words is K, whereas in the second case it is $K \cdot L$.

3.3 Video Indexing and Querying

We use *tf-idf* weighting to calculate the similarity between two video histograms. The *tf-idf* weights are computed for every visual word in every video in collection D_b based on $w_{td} = n_{td} \cdot \log |D_b|/n_t$, where w_{td} is the weight of word t in video d, n_{td} and n_t are the number of occurrences of word t in video d and the entire collection respectively, while $|D_b|$ is the number of videos in the collection. The former factor of the equation is called *term frequency* (tf) and the latter *inverted document frequency* (idf). The calculation of weights take place in the offline part of the method, i.e. they are not recalculated for every new query.

The feature extraction and aggregation steps for a query video q are the same as the ones described above. Once the final histogram H_v^q is extracted from q, an inverted file indexing structure [17] is used for fast retrieval of videos that have at least a common visual word with the query video. Then, all these videos are ranked in descending order based on their cosine similarity with the query video, computed using the corresponding *tf-idf* representations.

4 Evaluation

4.1 Dataset

Experiments were performed on the CC_WEB_VIDEO dataset [21], which is available by the research groups of City University of Hong Kong and Carnegie Mellon University. The collection consists of a sample of videos retrieved by submitting 24 popular text queries to popular video sharing websites (i.e. YouTube, Google Video, and Yahoo! Video). For every query, a set of video clips was collected and the most popular video was considered to be the query video. Subsequently, all videos in the video set retrieved by the query were manually annotated based on their near-duplicate relation to the query video. Table 2 depicts the types of near-duplicate types and their annotation. In the present work, all videos annotated with any symbol but **X** are considered near-duplicates. The dataset contains a total of 13,129 videos consisting of 397,965 keyframes.

All experiments were carried out on a system with Intel(R) Core(TM) i7-4770K CPU at 3.50 GHz, 16 GB RAM, NVIDIA GTX 980 GPU and 64-bit Ubuntu 14.04 operating system.

Table 2. Type of transformation.

Annotation	Transformation
E	Exactly duplicate
S	Similar video
V	Different version
M	Major change
L	Long version
X	Dissimilar video

4.2 Evaluation Metrics

To measure detection accuracy, we employ the interpolated precision-recall (PR) curve. Precision is determined as the fraction of retrieved videos that are relevant to the query, while recall is the fraction of the total relevant videos that are retrieved. We further use mean average precision (mAP) as defined in [21] and in Eq. 2, where n is the number of relevant videos to the query video, and r_i is the rank of the i-th retrieved relevant video.

$$AP = \frac{1}{n} \sum_{i=0}^{n} \frac{i}{r_i} \tag{2}$$

4.3 Competing Approaches

In Sect. 5.4, we compare the proposed approach with five widely used content-based NDVR approaches.

Color Histograms (CH) - Wu et al. [21] generated a global video representation based on the color histograms of keyframes. The color histogram is a concatenation of 18 bins for Hue, 3 bins for Saturation, and 3 bins for Value, resulting in a 24-dimensional vector representation for every keyframe. The global video signature is the normalized color histogram over all keyframes in the video.

Auto Color Correlograms (ACC) - Cai et al. [2] used uniform sampling to extract one frame per second for the input video. The auto-color correlograms of each frame are computed and aggregated based on a visual codebook generated from a training set of video frames. The retrieval of near-duplicate videos is performed using tf-idf weighted cosine similarity over the visual word histograms of a query and a dataset video.

Local Structure (LS) - Wu et al. [21] combined global signatures and local features in a hierarchical method. Color signatures are employed to detect near-duplicate videos with high confidence and to filter out very dissimilar videos. For the rest of videos, a local feature based method was developed, which compares the keyframes in a sliding window using their local features (PCA-SIFT).

Multiple Feature Hashing (MFH) - Song et al. [18] exploited multiple image features to learn a group of hash functions that project the video keyframes into the Hamming space. The combination of the keyframe hash codes generates a video signature which constitutes the video representation in the dataset. Hamming distance is employed to determine similarity between candidate videos.

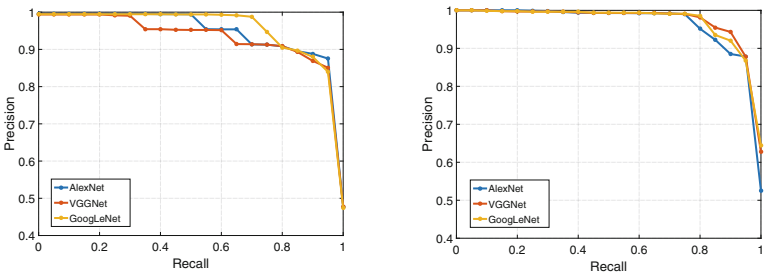
Pattern-based approach (PPT) - Chou et al. [3] built a pattern-based indexing tree (PI-tree) based on a sequence of symbols encoded from keyframes, which facilitates the efficient retrieval of candidate videos. They used m-pattern-based dynamic programming (mPDP) and time-shift m-pattern similarity (TPS) to determine video similarity.

5 Experiments

5.1 Impact of CNN Architecture and Vocabulary Size

In this section, we study the performance of the proposed approach in the CC_WEB_VIDEO dataset in relation to the underlying CNN architecture and the size of the visual vocabulary.

Regarding the first aspect, three CNN architectures are tested: AlexNet, VGGNet and GoogLeNet, with both aggregation schemes implemented using $K = 1000$ words. Figure 2 illustrates the PR curves of the different CNN architectures with the two aggregation schemes. Layer-based aggregation runs outperform vector-based ones for every architecture. GoogLeNet achieves the best results for the vector-based aggregation experiments with a precision close to 100% up to a 70% recall. For recall values in the range 80%–100%, all three architectures have similar results. For the layer-based aggregation scheme, all three architectures exhibit near perfect performance up to 75% recall.



(a) Vector Aggregation (b) Layer Aggregation

Fig. 2. Precision-recall curve of the proposed approach based on three CNN architectures and for the two aggregation schemes.

Similar conclusions are obtained from the analysis of mAP achieved using different CNN architectures, as depicted in Table 3. For the vector-based aggregation experiments, GoogLeNet achieved the best performance with a mAP of

0.958, and VGGNet the worst (mAP = 0.937). On the other hand, when using the layer-based aggregation scheme, the best mAP score (0.976) was based on VGG-Net. The lowest, yet competitive results in the case of layer-based aggregation, are obtained for AlexNet (mAP = 0.969).

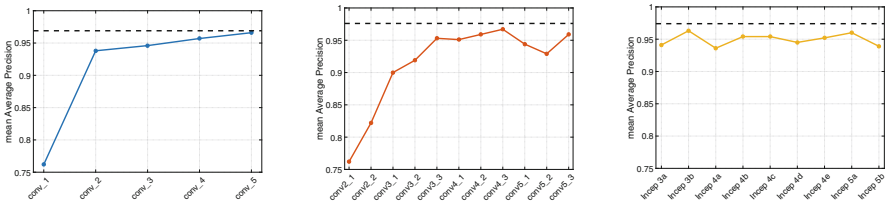
Table 3. mAP per CNN architecture and aggregation scheme.

Method	K	AlexNet	VGGNet	GoogLeNet
Vector aggregation	1000	0.951	0.937	**0.958**
	10,000	0.879	0.886	0.857
Layer aggregation	1000	0.969	**0.976**	0.974
	10,000	0.948	0.959	0.958

To study the impact of vocabulary size, we compare the two schemes when used with $K = 1000$ and $K = 10,000$ (Table 3). Results reveal that the performance of vector-based aggregation for $K = 10,000$ is significantly lower compared to the case when $K = 1000$ words are used. It appears that the vector-based aggregation suffers considerably more from the increase of K compared to the layer-based aggregation, which appears to be less sensitive to this parameter. Due to this fact, we did not consider to use the same amount of visual words for the vector-based and the layer-based aggregation, since the performance gap between the two types of aggregation with the same number of visual words would be much more pronounced.

5.2 Performance Using Individual Layers

We also assessed the retrieval capability of every layer for the three tested CNN architectures. Figure 3 depicts the mAP of the approach using only a selected layer vector. In the AlexNet and VGGNet architectures, the mAP of the first layers are quite low and as we are moving to deeper layers, the retrieval performance improves. In both cases, there are several layers that exceed the performance of the vector-based aggregation scheme. This indicates that it is better to extract



(a) AlexNet (b) VGGNet (c) GoogLeNet

Fig. 3. mAP of every layer for the three architectures.

the feature descriptors only from one layer than concatenating all layers in a single vector. However, no single layer overpasses the performance of the layer-based aggregation scheme, displayed with a dashed line. In GoogLeNet, the first layer (Inception 3a) is already deep enough to achieve competitive performance. In this case, the performance for all layers fluctuates between 0.935 and 0.960.

5.3 Performance per Query

Here, we analyze the performance of the best vector-based aggregation instance (GoogLeNet) with the best layer-based aggregation instance (VGGNet) on different queries. Figure 4 displays their Average Precision per query. Layer aggregation outperforms vector aggregation for every single query. However, both approaches fail in the difficult queries of the dataset, namely query 18 (`Bus uncle`) and query 22 (`Numa Gary`). The major factor leading to errors is that both videos have relatively low resolution/quality and the candidate videos are heavily edited, which leads to a significant number of relevant videos not to be retrieved at all (i.e. many false negatives). Nevertheless, CNN-L leads to considerably better results in both queries in comparison to CNN-V (Fig. 4).

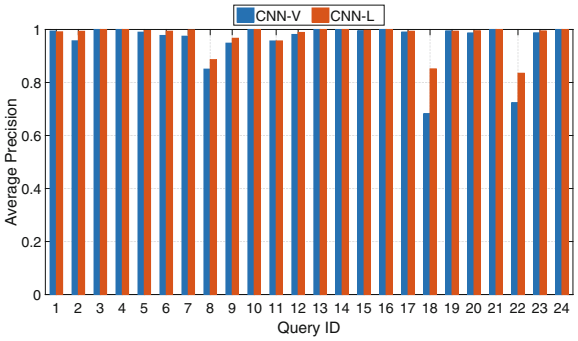


Fig. 4. Average precision/query for GoogLeNet (CNN-V) and VGGNet (CNN-L).

Table 4. Comparison between our approach and existing approaches.

Method	CH	ACC	LS	MFH	PPT	CNN-V	CNN-L
mAP	0.892	0.944	0.952	0.958	0.958	0.958	**0.976**

5.4 Comparison Against Existing NDVR Approaches

For comparing the performance of our approach with the five NDVR approaches from the literature, we select the same runs as in the previous section. The numeric data for the interpolated PR curves of the CH, LS and PPT methods

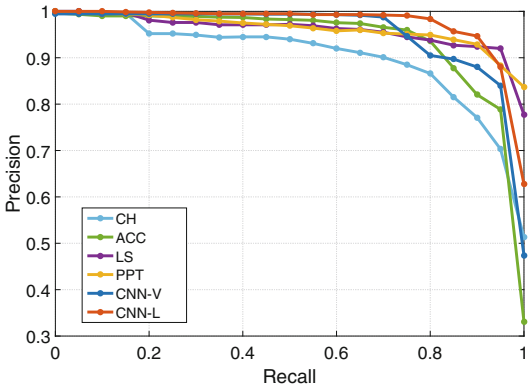


Fig. 5. Precision-recall curve comparison of two versions of the proposed approach with four state-of-the-art methods.

on the CC_WEB_VIDEO dataset were provided by the authors of [3,21], respectively. For the ACC method, we developed our own implementation, which was fine-tuned on the dataset.

Figure 5 illustrates the PR curves of the compared approaches. CNN-L outperforms all other methods up to 90% of recall, at which point the LS and PPT methods start outperforming it. Additionally, CNN-V is at the same level with CNN-L up to 70%, after which it starts performing worse. It is noteworthy that the approaches based on the bag-of-word scheme have low precision at high values of recall (>90%). In terms of mAP, both versions of the proposed approach are competitive in comparison to the state-of-the-art, as attested by Table 4. CNN-L achieves the best score (mAP = 0.976), followed by CNN-V, MFH and PPT (mAP = 0.958).

6 Conclusions and Future Work

We presented a new video-level representation for Near-Duplicate Video Retrieval, which leverages the effectiveness of CNN features and a newly introduced layer-based aggregation scheme that exhibited considerably improved performance over five popular approaches on the CC_WEB_VIDEO dataset in terms of Precision-Recall and mAP.

In the future, we plan to apply the necessary modifications to our method to exploit the use of generic C3D features [20]. Furthermore, we are going to conduct more comprehensive evaluations of the method using more challenging datasets, and we will also assess the applicability of the approach on the problem of Partial Duplicate Video Retrieval (PDVR).

Acknowledgement. This work is supported by the InVID project, partially funded by the European Commission under contract numbers 687786.

References

1. Bahmani, B., Moseley, B., Vattani, A., Kumar, R., Vassilvitskii, S.: Scalable k-means++. Proc. VLDB Endow. **5**(7), 622–633 (2012)
2. Cai, Y., Yang, L., Ping, W., Wang, F., Mei, T., Hua, X.S., Li, S.: Million-scale near-duplicate video retrieval system. In: Proceedings of the 19th ACM International Conference on Multimedia, pp. 837–838 (2011)
3. Chou, C.L., Chen, H.T., Lee, S.Y.: Pattern-based near-duplicate video retrieval and localization on web-scale videos. IEEE Trans. Multimed. **17**(3), 382–395 (2015)
4. Douze, M., Jegou, H., Schmid, C.: An image-based approach to video copy detection with spatio-temporal post-filtering. IEEE Trans. Multimed. **12**(4), 257–266 (2010)
5. Heikkila, M., Pietikainen, M., Schmid, C.: Description of interest regions with local binary patterns. Pattern Recogn. **42**(3), 425–436 (2009)
6. Huang, Z., Shen, H.T., Shao, J., Zhou, X., Cui, B.: Bounded coordinate system indexing for real-time video clip search. ACM Trans. Inf. Syst. **27**(3), 17 (2009)
7. Jègou, H., Douze, M., Schmid, C., Prez, P.: Aggregating local descriptors into a compact image representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3304–3311 (2010)
8. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM International Conference on Multimedia, pp. 675–678 (2014)
9. Kraaij, W., Awad, G.: TRECVID 2011 content-based copy detection: task overview. In: Online Proceedings of TRECVid 2010 (2011)
10. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
11. Lowe, D.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
12. Liu, J., Huang, Z., Cai, H., Shen, H.T., Ngo, C.W., Wang, W.: Near-duplicate video retrieval: current research and future trends. ACM Comput. Surv. **45**(4), 44 (2013)
13. Ng, J.Y.H., Yang, F., Davis, L.S.: Exploiting local features from deep networks for image retrieval. In: Proceedings of the IEEE CVPR Workshops, pp. 53–61 (2015)
14. Razavian, A.S., Azizpour, H., Sullivan, J., Carlsson, S.: CNN features off-the-shelf: an astounding baseline for recognition. In: Proceedings of the IEEE CVPR Workshops, pp. 806–813 (2014)
15. Shang, L., Yang, L., Wang, F., Chan, K.P., Hua, X.S.: Real-time large scale near-duplicate web video retrieval. In: Proceedings of the 18th ACM International Conference on Multimedia, pp. 531–540 (2010)
16. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014). arXiv preprint: arXiv:1409.1556
17. Sivic, J., Zisserman, A.: Video Google: a text retrieval approach to object matching in videos. In: Proceedings of Ninth IEEE International Conference on Computer Vision, pp. 1470–1477 (2003)
18. Song, J., Yang, Y., Huang, Z., Shen, H.T., Luo, J.: Effective multiple feature hashing for large-scale near-duplicate video retrieval. IEEE Trans. Multimed. **15**(8), 1997–2008 (2013)

19. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
20. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3D convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4489–4497 (2015)
21. Wu, X., Hauptmann, A.G., Ngo, C.W.: Practical elimination of near-duplicates from web video search. In: Proceedings of the 15th ACM International Conference on Multimedia, pp. 218–227 (2007)
22. Xu, Z., Yang, Y., Hauptmann, A.G.: A discriminative CNN video representation for event detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1798–1807 (2014)
23. Zheng, L., Zhao, Y., Wang, S., Wang, J., Tian, Q.: Good practice in CNN feature transfer (2016). arXiv preprint: arXiv:1604.00133