Video Retrieval for Everyday Scenes With Common Objects

Arun Zachariah
azachariah@mail.missouri.edu
University of Missouri, USA

Praveen Rao
praveen.rao@missouri.edu
University of Missouri, USA

ABSTRACT

We propose a video retrieval system for everyday scenes with common objects. Our system exploits the predictions made by deep neural networks for image understanding tasks using natural language processing (NLP). It aims to capture the relationships between objects in a video scene as well as the ordering of the matching scenes. For each video in the database, it identifies and generates a sequence of key scene images. For each such scene, it generates most probable captions using state-of-the-art models for image captioning. The captions are parsed and represented by tree structures using NLP techniques. These are then stored and indexed in a database system. When a user poses a query video, a sequence of key scenes are generated. For each scene, its caption is generated using deep learning and parsed into its corresponding tree structure. After that, optimized tree-pattern queries are constructed and executed on the database to retrieve a set of candidate videos. Finally, these candidate videos are ranked using a combination of longest common subsequence of scene matches and tree-edit distance between parse trees. We evaluated the performance of our system using the MSR-VTT dataset, which contained everyday scenes. We observed that our system achieved higher mean average precision (mAP) compared to two recent techniques, namely, CSQ and DnS.

KEYWORDS

Video retrieval, indexing, scene captioning, NLP, XML, ranking

ACM Reference Format:
Arun Zachariah and Praveen Rao. 2023. Video Retrieval for Everyday Scenes With Common Objects. In *International Conference on Multimedia Retrieval (ICMR '23), June 12–15, 2023, Thessaloniki, Greece.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3591106.3592239

1 INTRODUCTION

Video data possess a multitude of information with applications in e-commerce, healthcare, defense, education, social media, and entertainment. The commercial success of companies such as YouTube, Instagram, and TikTok, have resulted in enormous amount of video data on the Web. Hence, video retrieval continues to be challenged by the volume and variety of video data. In content-based video retrieval (CBVR), we aim to find videos in a given database that are similar to a query video [3]. Content may refer to colors, shapes, textures, objects, faces, and audio in the frames of a given video.

ICMR '23, June 12–15, 2023, Thessaloniki, Greece

ACM ISBN 979-8-4007-0178-8/23/06…$15.00
https://doi.org/10.1145/3591106.3592239

Coarse-grained approaches [7, 9, 12, 22, 23, 27, 32, 37, 42, 48] for CBVR aggregate frame-level features into a single vector representation or a hash code for each video, when are then indexed; during retrieval, a similarity metric is computed to obtain relevant videos. While coarse-grained approaches are efficient for retrieval, their performance is often limited. Fine-grained approaches [4, 6, 11, 18, 19, 28, 31, 34, 39, 41], on the other hand, extract representations to capture the spatio-temporal structure in the videos, use them during indexing, and perform similarity computation during retrieval. The sequence of frames is considered during similarity computation. While fine-grained approaches achieve higher retrieval performance than coarse-grained approaches, they are less efficient on large-scale datasets. Re-ranking approaches [9, 24, 26, 43, 46] combine the merits of coarse-grained and fine-grained approaches by first ranking via a coarse-grained approach to filter out irrelevant videos followed by re-ranking via a fine-grained approach.

Recently, a technique called QIK [49, 50] was proposed for content-based image retrieval on everyday scenes with common objects. Unlike prior approaches that relied on features constructed from CNNs, QIK showed that predictions made by deep neural networks for image captioning [40, 45] can be used for efficient image retrieval after exploiting NLP techniques on the image captions. QIK outperformed techniques such as CroW [21], FR-CNN [33], DIR [13], and DELF [30]. One may wonder if image captioning and NLP can be leveraged for effective video retrieval instead of the popular approach of indexing and similarity computation over spatio-temporal representations of videos (using CNN-based features).

Motivated by the above reason, we propose QIK+ for retrieval of videos with everyday scenes (containing common objects) by extending QIK in a novel way. By design, QIK+ captures both the relationship between objects in a scene as well as the ordering of the matched scenes in a video. Our key contributions are as follows:

- We propose a new way of indexing videos by identifying key scenes in them and generating the most probable captions for these scenes using state-of-the-art image captioning models. Using NLP techniques, the captions are transformed into parse trees and represented in XML (Extensible Markup Language). The ordering of the key scenes in a video is also preserved in the XML representation, which is then indexed by a database system.
- We propose a new filtering step to identify candidate videos. Given a query video, we first extract the key scenes of the video and generate the most probable captions as before. Optimized tree-structured queries are generated in XPath [5] for the query key scenes (based on their captions) and executed by the database system to identify the candidate videos.
- We propose a new ranking scheme to identify the top-k matches by synergistically combining the longest common subsequence (LCS) of scene matches (between the query key scenes and the candidate's key scenes) and tree-edit distance (TED) computed between

the parse tree of a query key scene's caption and the candidate key scene's caption that was considered a match.

• We compared the performance of QIK+ with state-of-the-art video retrieval techniques, namely, CSQ [48] and DnS [24] on the MSR-VTT dataset [44]. We observed that QIK+ achieved higher mAP compared to its competitors.

2 RELATED WORK

In this section, we briefly discuss relevant techniques that use CNN-based features for video retrieval.

MFH [38] learned a group of hash functions for extracting global and local features that mapped the vital video frames into a Hamming space. Cao *et al.* [8] proposed a hashing framework that combined hashing and feature pooling for powerful search. The hashes were obtained from heterogeneous hash codes and stored in a hash table to speed up the search. Revaud *et al.* [32] used properties of circulant matrices on features extracted using a CNN to encode frames. Ye *et al.* [47] developed a video hashing method to transform high-dimensional data into compact binary hash codes using the spatio-temporal information embedded in a video.

Recently, DVH [27] fused temporal information across different frames within a video to learn its feature representation. ViSil [22] calculated video-to-video similarity by considering fine-grained spatio-temporal relations between pairs of videos. On the other hand, DPC [15] followed a self-supervised approach that learned encoding video blocks of an equal number of frames. The video blocks were mapped to a latent space, and the next set of blocks were predicted using a predictive function. MemDPC [16] was later proposed to improve upon DPC's memory and architecture. CSQ [48] mapped video features to hash codes in a Hamming space and grouped similar data pairs to a common hash center by leveraging properties of a Hadamard matrix. Dissimilar pairs would converge to different centers improving learning efficiency and retrieval accuracy. More recently, DnS [24] used a Knowledge Distillation framework around ViSil to achieve high performance and efficiency using three different networks: a coarse-grained student network, a fine-grained student network, and a selection network. On the other hand, TCA [35] used higher-level video representation describing the long-range semantic dependencies of the temporal information among frame features. VCLR [25] proposed a self-supervised video-level contrastive learning framework to formulate positive and negative pairs using pairs of similar and augmented dissimilar images to learn high-level features with videos.

Unlike the aforementioned techniques, QIK+ aims to leverage scene captioning and NLP techniques for the indexing, filtering, and ranking steps involved during video retrieval.

3 BACKGROUND ON QIK

Due to space contraints, we briefly introduce the key features of QIK [49, 50] for image retrieval on everyday scenes with common objects. During indexing, for each image in the database, QIK generates its most probable caption using state-of-the-art image captioning models [40]. This enables QIK to capture the context of everyday scenes and learn the relationships between objects in them. For each caption, a parse tree [20] is constructed. A parse tree captures the syntactic structure of a caption using parts-of-speech (POS) tagging by identifying noun phrases/nouns, verb phrases/verbs, adjectives, etc. The collection of these ordered trees are represented as XML documents, which are indexed by an XML database system.

During query processing, QIK generates the most probable caption of a query image and the corresponding parse tree. The parse tree is processed to construct an optimized XPath query containing only essential keywords in the caption whilst preserving the ordering between these keywords and their relationships. (Prepositions, determiners, conjunctions, etc., in the caption are ignored.) The candidate images are retrieved by executing the optimized XPath query on the XML documents. During the ranking step, the tree edit distance between the parse tree of a candidate image's caption and the parse tree of the query image's caption is computed. The candidates are ranked by tree edit distance (low to high), and the top-k matches are output.

4 DESIGN OF QIK+

In this section, we introduce the design of QIK+ and highlight its novelty for video retrieval. We begin with the indexing strategy of QIK+ followed by its filtering and ranking strategies.

4.1 Index Construction

QIK+ processes each video in the database and extracts its key scenes. A key scene is *a representative frame* in a video identified by a scene dectection algorithm [2]. For each key scene, the most probable caption is generated using a state-of-the-art image captioning model (e.g., ClipCap [29]). The parse tree of the caption is then generated and represented as an XML document. As the ordering of the key scenes must be captured for a video, the final XML document for each key scene includes the scene ID and the video ID. The collection of XML documents for each video is then indexed by a database system. Figure 1(a) shows a scene of a video along with its most probable caption and its XML representation. Algorithm 1 summarizes the steps involved during indexing.

Key scene D_1

a man standing next to a parked car

```
<ROOT><NP><NP><DT>a</DT><NN>man</NN>
<NN>standing</NN></NP><NP><NP><JJ>next
</JJ></NP><PP><TO>to</TO></PP><NP><DT>
a</DT><JJ>parked</JJ><NN>car</NN></NP>
</NP></NP></ROOT>
```

Figure 1: A video scene, its caption and XML representation

Algorithm 1 IndexVideos(V)

Input: V denotes a list of videos in the database

1: **for** each video $v \in V$ **do**
2: Extract the key scenes in v; let $(s_1, ..., s_N)$ denote them
3: **for** each scene s_i **do**
4: Predict the most probable caption c of s_i
5: Generate the parse tree p of c
6: Represent p in XML, include scene ID i and video ID v, and index the final XML document

4.2 Video Retrieval

During video retrieval, QIK+ employs a filtering and ranking strategy. It first extracts the key scenes in a query video. For each key

scene, the most probable caption is generated using image captioning. An optimized XPath expression is generated for each caption using the strategy proposed by QIK [49]. Note that an XPath query consists of different axes that enable the traversal of a parse tree in a specific order to match nodes with specific labels. The XPath queries are executed on the XML database to retrieve a set of candidate scene IDs and their video IDs. A dictionary is created to group the matched scene IDs by their video ID. A (key, value) pair in the dictionary consists of a video ID as the key and a sorted list of scene IDs for that video as the value. The candidate videos are ranked to output the top-k matches. Algorithm 2 shows the main steps involved during video retrieval.

Algorithm 2 RetrieveVideos(k, q)

Input: k denotes the top-k videos to output; q denotes the query video

1: Let $(q_1, ..., q_n)$ denote the key scenes in q
2: **for** each q_i **do**
3: Predict the most probable caption C_i
4: $xp \leftarrow GenerateOptimizedXPath(C_i)$ // Using QIK [49]
5: Execute xp on the XML database to retrieve a set of candidate scene IDs and their video IDs
6: Create a dictionary D with key as a candidate video ID and value as a sorted list of matched scene IDs for that video (sorted by scene ID)
7: $O \leftarrow$ RankBasic(D) **OR** RankLCS(q, D) **OR** RankLCS+TED(q, D)
8: **return** top-k matches in O

Algorithm 3 RankBasic(D)

Input: D denotes the dictionary of candidate video IDs and their key scene matches

1: **for** each $(k, v) \in D$ **do**
2: Let $v = (s_1, ..., s_m)$
3: $scoreB_k \leftarrow m$
4: Let O denote the sorted list of candidate video IDs based on $scoreB_k$ (in descending order)
5: **return** O

Algorithm 4 RankLCS(q, D)

Input: q denotes the query video; D denotes the dictionary of candidates

1: Let $(q_1, ..., q_n)$ denote the IDs of key scenes in q
2: **for** each $(k, v) \in D$ **do**
3: Let $v = (s_1, ..., s_m)$
4: $scoreL_k \leftarrow length(LCS((q_1, ..., q_n), (s_1, ..., s_m))$
5: Let O denote the sorted list of candidate video IDs in D based on $scoreL_k$ (in descending order)
6: **return** O

Next, we discuss the ranking strategy of QIK+ by introducing three different schemes (Line 7 in Algorithm 2). The basic scheme shown in Algorithm 3 assigns the number of scenes matched for a video as its score and ranks videos based on these scores (high to low). The ordering of the scene matches is completely ignored. The next scheme shown in Algorithm 4 computes the LCS between the query key scenes and the candidate's key scenes that were matched for each candidate video. The intuition is that the ordering of key scenes in the query video should match the ordering of the matching key scenes in a candidate video for the best result. (Unlike LCS on strings where we match identical characters between two strings, we treat a query key scene q_i as a match for a candidate video's key scene s_j when the XPath query on q_i's caption returns s_j as a match.) The length of the LCS denotes a candidate video's score. Here, we aim to maximize the number of key scene matches. Also, the ordering between key scene matches is incorporated. The candidate videos are ranked based on this score (high to low). The final scheme (and the best one) shown in Algorithm 5 combines LCS and TED in an innovative way to account for the ordering of scene matches as well as the similarity between the matched scenes by comparing the parse trees of their captions. As a result, the relationship between objects in a matched scene is also considered by checking the ordering between essential keywords in captions and their relationships via TED. Specifically, the TED is computed only between the parse tree of a query scene's caption and the parse tree of candidate video scene's caption that appear in the LCS of scene matches (Lines 6-9 in Algorithm 5). The key idea is the provide a weighted score for the LCS of scene matches based on similarity of the matched scenes instead of just the length of the LCS. Ties can be broken during sorting using $scoreL_k$ or $scoreB_k$.

Algorithm 5 RankLCS+TED(q, D)

Input: q denotes the query video; D denotes the dictionary of candidates

1: Let $(q_1, ..., q_n)$ denote the IDs of key scenes in q
2: **for** each $(k, v) \in D$ **do**
3: Let $v = (s_1, ..., s_m)$
4: $(l_1, ..., l_o) \leftarrow LCS((q_1, ..., q_n), (s_1, ..., s_m))$
5: $scoreT_k \leftarrow 0$
6: **for** each l_i **do**
7: Let s_a denote the candidate video's scene that matched the query scene q_b in l_i
8: Compute tree edit distance $\mathbb{T}$ between the parse tree of s_a's caption and the parse tree q_b's caption
9: $scoreT_k \leftarrow scoreT_k + (1 + \mathbb{T})^{-1}$
10: Let O denote the sorted list of candidate video IDs in D based on $scoreT_k$ (in descending order)
11: **return** O

Example 4.1. Figure 2 shows two key scenes of a query video, their corresponding captions, and optimized XPath queries. Figure 3 shows how the two key scenes of a query video Q are matched to key scenes of three candidate videos V_1, V_2, and V_3. The matched scenes are shown as green and blue boxes. Algorithm 3 ranks them in the order $V_3 - V_1 - V_2$. Algorithm 4 ranks them in the order $V_1 - V_3 - V_2$ based on LCS assuming ties are broken using $scoreB_k$. Algorithm 5 ranks them in the order $V_1 - V_2 - V_3$ if $scoreT_k$ for the videos are 0.2, 0.2, and 0.1, assuming ties are broken via $scoreL_k$.

5 PERFORMANCE EVALUATION

We compared QIK+ with two different video retrieval techniques, namely, CSQ [48] and DnS [24]. CSQ is a supervised video hashing technique that uses Hamming distance for video ranking. DnS

Key scene S_1

a car is parked on the side of the road

```
/descendant::NN[text()='car']/followi
ng::VBZ[text()='is']/following::VBN[t
ext()='parked']/following::NN[text()=
'side']/following::NN[text()='road']
```

Key scene S_2

a woman standing next to a parked car

```
/descendant::NN[text()='woman']/follow
ing::VBG[text()='standing']/following:
:JJ[text()='next']/following::TO[text(
)='to']/following::VBN[text()='parked'
]/following-sibling::NN[text()='car']
```

Figure 2: Query video scenes and optimized XPath queries

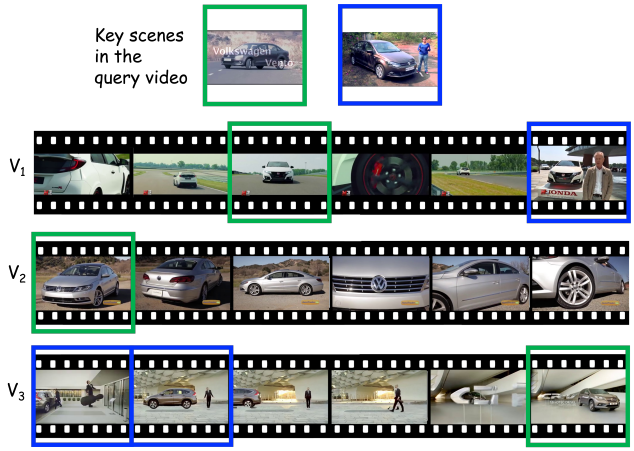

Figure 3: Example of scene matches after filtering

employs a Knowledge Distillation framework [17] to attain high retrieval performance and high computational efficiency. We used the original code published by the authors of CSQ and DnS.

5.1 Implementation and Experimental Setup

As QIK was written in Java, QIK+ was also implemented in Java and compiled using Java 1.8. BaseX (version 9.2) [1, 14], a high performance XML engine, was used to manage the XML data. For identifying and extracting the key scenes from a video, QIK+ used the content-aware detection in PySceneDetect [2] that detects jump cuts in a video. QIK+ used the pre-trained ClipCap [29] model for generating captions of images. The model was trained on the Conceptual Captions [36] dataset consisting of 3.3 million images and their descriptions extracted from the Web. We ran the experiments on CloudLab [10] and used hardware that had a 10-core Intel E5-2640v4 CPU (2.20 GHz) and 64 GB of RAM running Ubuntu 18.04.

5.2 Dataset & Queries

For evaluating the video retrieval performance, we used MSR-VTT [44], which contained 10K Web video clips totaling 41.2 hours. This dataset was curated by filtering the top 150 videos obtained after executing 257 queries (corresponding to 20 categories) on a commercial video search engine. The database of videos for indexing consisted of 7K videos in MSR-VTT that formed the training and validation set. For the query videos, we selected 1,031 videos from the test set with an average 12 scenes per video. All the videos that belonged to the query video category were considered as *true matches*. We computed the mAP value for different top-k matches.

5.3 Results

We first evaluated the retrieval performance of QIK+ for the different ranking schemes described in Section 4.2. Hereinafter, we denote the scheme based on Algorithm 3 as QIK+$_b$, Algorithm 4 as QIK+$_l$, and Algorithm 5 as QIK+$_t$. Our goal was to show that ranking candidate videos based on a combination of LCS and TED would provide the best performance as it can capture the relationships between scenes and between important objects in a scene. Table 1 show the average of the mAP values (computed over the query videos) for the various ranking procedures. (The winner is shown in bold.) QIK+$_l$ outperformed QIK+$_b$ validating the importance of maximizing the matching of scene ordering. QIK+$_t$ outperformed QIK+$_b$ and QIK+$_l$ validating our claim that by combining the scene ordering and captions (to capture object relationships in a scene) can provide the best video retrieval performance.

Table 1: Comparison of QIK+, CSQ, and DnS (avg. of mAP)

	k=2	k=4	k=8	k=16
QIK+$_b$	0.416	0.437	0.433	0.423
QIK+$_l$	0.428	0.448	0.447	0.434
QIK+$_t$	**0.434**	**0.456**	**0.452**	**0.437**
DnS	0.369	0.396	0.391	0.365
CSQ	0.402	0.416	0.411	0.403

Next, we compared the best retrieval performance of QIK+ (i.e., QIK+$_t$) with its competitors. For fair evaluation, we used the default parameters of CSQ and DnS. CSQ, a supervised learning algorithm, was re-trained on MSR-VTT to identify, hash, and cluster the videos. Table 1 reports the average of mAP values of QIK+ compared with its competitors for different values of k. (The winner is shown in bold.) Clearly, QIK+ was able to outperform its competitors by virtue of its design by capturing the relationships between objects in a scene and maximizing the matches based on scene ordering.

6 CONCLUSION

We presented QIK+ for video retrieval on everyday scenes with common objects. By design, QIK+ captures both the relationship between objects in a scene as well as the ordering of scenes in a video. Rather than using CNN-based features, QIK+ relies on the captions of key scenes and their parse trees for indexing, filtering, and ranking. It maps the parse trees into XML and leverages optimized XPath queries to find candidate videos and their scene matches. By considering the relationship between objects in different scenes as well as within the same scene, QIK+ achieved better mAP than its competitors (on MSR-VTT) demonstrating its effectiveness on everyday scenes with common objects. Note that QIK+ cannot capture the relationships between objects that span multiple key scenes in a video. In the future, we would like to optimize the execution time of queries in QIK+and investigate how it can be used for lifelog search. QIK+ is available at https://github.com/MU-Data-Science/QIK.

Acknowledgments. This work was supported in part by the National Science Foundation under Grant No. 1747751. We thank the anonymous reviewers for their valuable comments.

REFERENCES

[1] 2007. *BaseX | The XML Framework: Lightweight and High-Performance Data Processing.* Retrieved July 1, 2022 from https://basex.org

[2] 2014. *PySceneDetect.* Retrieved July 1, 2022 from http://scenedetect.com/en/latest/

[3] Aasif Ansari and Muzammil H Mohammed. 2015. Content Based Video Retrieval Systems-Methods, Techniques, Trends and Challenges. *International Journal of Computer Applications* 112, 7 (2015), 13–22.

[4] Lorenzo Baraldi, Matthijs Douze, Rita Cucchiara, and Hervé Jégou. 2018. LAMV: Learning to Align and Match Videos With Kernelized Temporal Layers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 7804–7813.

[5] Anders Berglund, Scott Boag, Don Chamberlin, Mary F. Fernandez, Michael Kay, Jonathan Robie, and Jerome Simeon. 2002. *XML Path Language (XPath) 2.0 W3C Working Draft 16.* Technical Report WD-xpath20-20020816. World Wide Web Consortium.

[6] Mina Bishay, Georgios Zoumpourlis, and Ioannis Patras. 2019. TARN: Temporal Attentive Relation Network for Few-Shot and Zero-Shot Action Recognition. In *British Machine Vision Conference*. 1–14.

[7] Yang Cai, Linjun Yang, Wei Ping, Fei Wang, Tao Mei, Xian-Sheng Hua, and Shipeng Li. 2011. Million-Scale near-Duplicate Video Retrieval System. In *Proceedings of the 19th ACM International Conference on Multimedia*. Scottsdale, Arizona, USA, 837–838.

[8] Liangliang Cao, Zhenguo Li, Yadong Mu, and Shih-Fu Chang. 2012. Submodular Video Hashing: A Unified Framework Towards Video Pooling and Indexing. In *Proc. of the 20th ACM International Conference on Multimedia*. Nara, Japan, 299–308.

[9] Chien-Li Chou, Hua-Tsung Chen, and Suh-Yin Lee. 2015. Pattern-Based Near-Duplicate Video Retrieval and Localization on Web-Scale Videos. *IEEE Transactions on Multimedia* 17, 3 (2015), 382–395.

[10] Dmitry Duplyakin, Robert Ricci, Aleksander Maricq, Gary Wong, Jonathon Duerig, Eric Eide, Leigh Stoller, Mike Hibler, David Johnson, Kirk Webb, Aditya Akella, Kuangching Wang, Glenn Ricart, Larry Landweber, Chip Elliott, Michael Zink, Emmanuel Cecchet, Snigdhaswin Kar, and Prabodh Mishra. 2019. The Design and Operation of CloudLab. In *2019 USENIX Annual Technical Conference*. 1–14.

[11] Yang Feng, Lin Ma, Wei Liu, Tong Zhang, and Jiebo Luo. 2018. Video Re-localization. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 1–16.

[12] Zhanning Gao, Gang Hua, Dongqing Zhang, Nebojsa Jojic, Le Wang, Jianru Xue, and Nanning Zheng. 2017. ER3: A Unified Framework for Event Retrieval, Recognition and Recounting. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*. 2253–2262.

[13] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. 2016. Deep Image Retrieval: Learning Global Representations for Image Search. In *Computer Vision - ECCV 2016*. 241–257.

[14] Christian Grün, Sebastian Gath, Alexander Holupirek, and Marc H. Scholl. 2009. XQuery Full Text Implementation in BaseX. In *Proc. of the 6th International XML Database Symposium on Database and XML Technologies*. 114–128.

[15] Tengda Han, Weidi Xie, and Andrew Zisserman. 2019. Video Representation Learning by Dense Predictive Coding. In *Proc. of the IEEE International Conference on Computer Vision Workshops*. 1–10.

[16] Tengda Han, Weidi Xie, and Andrew Zisserman. 2020. Memory-Augmented Dense Predictive Coding for Video Representation Learning. In *Computer Vision - ECCV 2020*. 312–329.

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531* (2015).

[18] Yu-Gang Jiang, Yudong Jiang, and Jiajun Wang. 2014. VCDB: A Large-Scale Database for Partial Copy Detection in Videos. In *Computer Vision – ECCV 2014*, David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (Eds.). Springer International Publishing, Cham, 357–371.

[19] Yu-Gang Jiang and Jiajun Wang. 2016. Partial Copy Detection in Videos: A Benchmark and an Evaluation of Popular Methods. *IEEE Transactions on Big Data* 2, 1 (2016), 32–42.

[20] Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (2nd ed.). Prentice Hall, USA.

[21] Yannis Kalantidis, Clayton Mellina, and Simon Osindero. 2016. Cross-Dimensional Weighting for Aggregated Deep Convolutional Features. In *Computer Vision - ECCV 2016 Workshops*. 685–701.

[22] Giorgos Kordopatis-Zilos, Symeon Papadopoulos, Ioannis Patras, and Ioannis Kompatsiaris. 2019. ViSiL: Fine-Grained Spatio-Temporal Video Similarity Learning. In *Proc. of the IEEE International Conference on Computer Vision*. 6351–6360.

[23] Giorgos Kordopatis-Zilos, Symeon Papadopoulos, Ioannis Patras, and Yiannis Kompatsiaris. 2017. Near-Duplicate Video Retrieval with Deep Metric Learning. In *Proc. of the IEEE International Conference on Computer Vision Workshops*. 347–356.

[24] Giorgos Kordopatis-Zilos, Christos Tzelepis, Symeon Papadopoulos, Ioannis Kompatsiaris, and Ioannis Patras. 2021. DnS: Distill-and-Select for Efficient and Accurate Video Indexing and Retrieval. *arXiv preprint arXiv:2106.13266* (2021).

[25] Haofei Kuang, Yi Zhu, Zhi Zhang, Xinyu Li, Joseph Tighe, Soren Schwertfeger, Cyrill Stachniss, and Mu Li. 2021. Video Contrastive Learning with Global Context. In *Proc. of the IEEE International Conference on Computer Vision*. 3195–3204.

[26] Siying Liang and Ping Wang. 2020. An Efficient Hierarchical Near-Duplicate Video Detection Algorithm Based on Deep Semantic Features. In *MultiMedia Modeling: 26th International Conference, MMM 2020, Daejeon, South Korea, January 5–8, 2020, Proceedings, Part I* (Daejeon, Korea (Republic of)). Springer-Verlag, 752–763.

[27] Venice Erin Liong, Jiwen Lu, Yap-Peng Tan, and Jie Zhou. 2017. Deep Video Hashing. *IEEE Transactions on Multimedia* 19, 6 (2017), 1209–1219.

[28] Hao Liu, Qingjie Zhao, Hao Wang, Peng Lv, and Yanming Chen. 2017. An Image-Based Near-Duplicate Video Retrieval and Localization Using Improved Edit Distance. *Multimedia Tools and Applications* 76, 22 (2017), 24435–24456.

[29] Ron Mokady, Amir Hertz, and Amit H Bermano. 2021. Clipcap: Clip Prefix for Image Captioning. *arXiv preprint arXiv:2111.09734* (2021).

[30] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. 2017. Large-Scale Image Retrieval with Attentive Deep Local Features. In *Proc. of 2017 IEEE International Conference on Computer Vision*. 1–10.

[31] Sébastien Poullot, Shunsuke Tsukatani, Phuong Anh Nguyen, Hervé Jégou, and Shin'ichi Satoh. 2015. Temporal Matching Kernel with Explicit Feature Maps. In *Proceedings of the 23rd ACM International Conference on Multimedia*. 381–390.

[32] Jérôme Revaud, Matthijs Douze, Cordelia Schmid, and Hervé Jégou. 2013. Event Retrieval in Large Video Collections with Circulant Temporal Encoding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*. 2459–2466.

[33] Amaia Salvador, Xavier Giro-i Nieto, Ferran Marques, and Shin'ichi Satoh. 2016. Faster R-CNN Features for Instance Search. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 1–8.

[34] J. Shao, X. Wen, B. Zhao, and X. Xue. 2021. Temporal Context Aggregation for Video Retrieval with Contrastive Learning. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE Computer Society, Los Alamitos, CA, USA, 3267–3277. https://doi.org/10.1109/WACV48630.2021.00331

[35] Jie Shao, Xin Wen, Bingchen Zhao, and Xiangyang Xue. 2021. Temporal Context Aggregation for Video Retrieval with Contrastive Learning. In *Proc. of the IEEE Winter Conference on Applications of Computer Vision*. 3268–3278.

[36] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset For Automatic Image Captioning. In *Proc. of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2556–2565.

[37] Jingkuan Song, Yi Yang, Zi Huang, Heng Tao Shen, and Richang Hong. 2011. Multiple Feature Hashing for Real-Time Large Scale near-Duplicate Video Retrieval. In *Proceedings of the 19th ACM International Conference on Multimedia*. Scottsdale, Arizona, 423–432.

[38] Jingkuan Song, Yi Yang, Zi Huang, Heng Tao Shen, and Richang Hong. 2011. Multiple Feature Hashing for Real-Time Large Scale Near-Duplicate Video Retrieval. In *Proc. of the 19th ACM International Conference on Multimedia*. 423–432.

[39] Hung-Khoon Tan, Chong-Wah Ngo, Richard Hong, and Tat-Seng Chua. 2009. Scalable Detection of Partial Near-Duplicate Videos by Visual-Temporal Consistency. In *Proc. of the 17th ACM International Conference on Multimedia*. 145–154.

[40] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2017. Show and Tell: Lessons Learned from the 2015 MS COCO Image Captioning Challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39, 4 (2017), 652–663.

[41] Kuan-Hsun Wang, Chia Chun Cheng, Yi-Ling Chen, Yale Song, and Shang-Hong Lai. 2020. Attention-Based Deep Metric Learning for Near-Duplicate Video Retrieval. In *Proc. of International Conference on Pattern Recognition (ICPR)*. 5360–5367.

[42] Xiao Wu, Alexander G Hauptmann, and Chong-Wah Ngo. 2007. Practical Elimination of Near-Duplicates from Web Video Search. In *Proc. of the 15th ACM International Conference on Multimedia*. 218–227.

[43] Xiao Wu, Alexander G. Hauptmann, and Chong-Wah Ngo. 2007. Practical Elimination of Near-Duplicates from Web Video Search. In *Proceedings of the 15th ACM International Conference on Multimedia*. Augsburg, Germany, 218–227.

[44] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. 2016. MSR-VTT: A Large Video Description Dataset for Bridging Video and Language. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*. 5288–5296.

[45] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *Proc. of the 32nd International Conference on Machine Learning*. 2048–2057.

[46] Yuanyuan Yang, Yonghong Tian, and Tiejun Huang. 2019. Multiscale video sequence matching for near-duplicate detection and retrieval. *Multimedia Tools and Applications* 78, 1 (2019), 311–336.

[47] Guangnan Ye, Dong Liu, Jun Wang, and Shih-Fu Chang. 2013. Large-Scale Video Hashing via Structure Learning. In *Proc. of the IEEE International Conference on*

Computer Vision. 2272–2279.

[48] Li Yuan, Tao Wang, Xiaopeng Zhang, Francis EH Tay, Zequn Jie, Wei Liu, and Jiashi Feng. 2020. Central Similarity Quantization for Efficient Image and Video Retrieval. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*. 3083–3092.

[49] Arun Zachariah, Mohamed Gharibi, and Praveen Rao. 2020. QIK: A System for Large-Scale Image Retrieval on Everyday Scenes With Common Objects. In *Proc. of the 2020 International Conference on Multimedia Retrieval*. 126–135.

[50] Arun Zachariah, Mohamed Gharibi, and Praveen Rao. 2021. A Large-Scale Image Retrieval System for Everyday Scenes. In *Proc. of the 2nd ACM International Conference on Multimedia in Asia*. Article 72, 3 pages.