CompoundEyes: Near-duplicate Detection in Large Scale Online Video Systems in the Cloud

Yixin Chen*, Wenbo He†, Yu Hua‡, Wen Wang*
*†School of Computer Science, McGill University
*Email: {yixin.chen, wen.wang4}@mail.mcgill.ca
†Email: wenbohe@cs.mcgill.ca
‡WNLO, School of Computer, Huazhong University of Science and Technology
Email: csyhua@hust.edu.cn

***Abstract*—At the present time, billions of videos are hosted and shared in the cloud of which a sizable portion consists of near-duplicate video copies. An efficient and accurate content-based online near-duplicate video detection method is a fundamental research goal; as it would benefit applications such as duplication-aware storage, pirate video detection, polluted video tag detection, searching result diversification. Despite the recent progress made in near-duplicate video detection, it remains challenging to develop a practical detection system for large-scale applications that has good efficiency and accuracy performance.**

In this paper, we shift the focus from feature representation design to system design, and develop a novel system, called CompoundEyes, accordingly. The improvement in accuracy is achieved via well-organized classifiers instead of advanced feature design. Meanwhile, by applying simple features with reduced dimensionality and exploiting the parallelism of the detection architecture, we accelerate the detection speed. Through extensive experiments we demonstrate that the proposed detection system is accurate and fast. It takes approximately 1.45 seconds to process a video clip from a large video dataset, CC_WEB_VIDEO, with a 89% detection accuracy.

I. INTRODUCTION

In recent years we have witnessed the proliferation of video content over the Internet. This growth was fueled by rapid advances in multimedia technologies, and also by the popularity of online video hosting and sharing services (e.g., YouTube, Yahoo! video). Videos have emerged as a dominant form of big data on the online world. According to Cisco Systems, Internet videos accounted for 78% of all U.S. Internet traffic in 2014, and is expected to rise to 84% in 2018 [1].

The expansion of video content is accompanied with ubiquitous duplications. Wu et al. [2] showed that among Internet 13,129 videos, around 27% are near-duplicate. Therefore, efficiently identifying near-duplicate videos (NDVs) on a large-scale is a fundamental research goal, which can benefit the performance of video sharing and hosting services in many aspects. For example, by identifying the NDV copies, bandwidth utilization and storage management in video content distribution systems can be further optimized. By comparing the metadata associated with NDVs, the pollution in metadata [3] can be detected as well. Furthermore, the detection of NDVs allows pirated copies to be identified (e.g., YouTube Content ID).

This research is jointly sponsored by NSERC RGPIN 418521-12, the National Basic Research Program of China (973) under Grant No. 2014CB340303, National Natural Science Foundation of China (NSFC) under Grant No.61173043, and State Key Laboratory of Computer Architecture under Grant No.CARCH201505.

Presently, a common way in practice to detect NDVs is based on metadata such as keywords, tags, or associated descriptions. However, metadata and descriptions are less reliable in detecting NDVs than visual content. It is very common that identical video clips have different sets of associated tags, and clips with identical set of tags can be significantly different. Therefore, a content-based NDVD (Near-Duplicate Video Detection) system [2], [4], [5] is more desirable than one based on metadata. These NDVD systems, however, tend to use high-dimensional feature representations and complicated algorithms to seek good detection accuracy, thereby sacrificing efficiency for accuracy. This approach is not practical for large-scale NDVD applications. As reported in YouTube statistics [6], 300 hours of video clips are uploaded every minute. If an NDVD system is not efficient enough, the detection speed cannot catch up to the video uploading speed. Thus, building a practical NDVD system is challenging due to the following two reasons.

- **The Complexity of Data**: Compared with other forms of big data such as records or logs, videos are more information-abundant, and complicated. Therefore, using features to profile a video is not as effective as it does in content-based duplicate document detection. In the cloud, there are numerous modifications on video content to produce NDVs, for example, variations in encoding format or parameters, photometric variations, or frame insertion or deletion. Every feature discovered hitherto has its own drawbacks, because certain information about video content has been discarded by this feature.
- **Detection Speed Requirement**: To cope with the sheer volume and increasing speed, a fast video detection system is necessary. However, this requirement is contradictory to the practice of using high-dimensional and composite feature representations to embody videos [4], [7], because the construction of these representations is exhaustive [4]. Consequently, it is generally conducted offline [4], [7].

In spite of time-consuming constructions, high-dimensional

and composite feature representations are intuitively more informative, and thus, discriminative. This is why recent research focuses on using high-dimensional feature design and feature fusion [4], [7] to detect NDVs. However, in this paper, using the information entropy concept, we demonstrate that composite feature representations are not necessarily more informative than a collection of simple representations. In addition, the increase of dimensionality may further reduce the informativeness. Accordingly, we shift the focus from advanced representation design to the system design. We design and implement an efficient yet accurate NDVD system, called CompoundEyes. Our idea was inspired by the compound eyes of insects, which are made up of numerous small optical systems. Although an individual small optical system is weak by itself, they together form a comprehensible eye sight, allowing for an incredibly wide viewing angle and the detection of fast movement.

The design of CompoundEyes follows the principles of systems approach. Although individual components are relatively weak in accuracy, together as a system they could achieve satisfactory performance improvement. Meanwhile the system efficiency is ensured because the individual components are simple and fast. We adopted CC_WEB_VIDEO [8] dataset to evaluate the performance of CompoundEyes. Compared with a similar work [9], the accuracy has been improved from 80% to 89%, with only 1.45 seconds average temporal cost for videos less than 10 minutes in length.

The contributions of our system can be further explained from the following aspects.

- **A Shifting of Detection Paradigm**: We apply a new design philosophy for NDVD systems, which employs multi-feature information fusion with well-coordinated classifiers instead of multi-feature fusion with a simple classifier. Based on the definition of the informativeness of video representation, we prove that theoretically a sophisticated representation combining multiple features does not provide more information than a collection of simple features, thus the latter approach does not guarantee higher accuracy then the former one.
- **Efficiency Improvement**: We use low-dimensional representations to achieve efficiency and scalability. Though the accuracy using individual features with reduced dimensionality is affected, we apply ensemble classifiers for information fusion and make the final detection result more accurate than state-of-the-art approaches. Moreover, we exploit the parallelism in our system to further accelerate the detection speed.
- **Implementation**: Our implementation of CompoundEyes along with the simplicity of input representations and native support of parallelism exhibits satisfactory performance in terms of both accuracy and detection efficiency. In CompoundEyes, LSH (Locality Sensitive Hashing) [10] is used to accelerate the video information search.

The rest of the paper is organized as follows. Background knowledge, the Feature-Centered detection paradigm, and related theoretical results are discussed in section II. In section III, the design of CompoundEyes is proposed. It is evaluated in section IV. Related work is reviewed in section V. Section VI concludes the paper.

II. PRELIMINARIES

In this paper, we adopt the most strict and least subjective [11] definition proposed by Wu et al. [2], in which NDVs are videos of similar visual content but have undergone various modifications such as illumination changes or caption insertion. Therefore the NDV detection is based on visual content rather than semantics.

A. Two-Stage NDVD Detection

The typical process of content-based NDVD systems is comprised of two stages: (1) feature extraction and description, (2) neighboring video retrieval.

1) Feature Extraction and Description: A video feature is a summary of information in visual content, which should preferably be stable and sufficiently distinguishable. A feature may span globally across the whole video, such as the color distribution, or be localized within a region, such as interest regions [12].

Extracting features from a video is conducted on a frame-by-frame basis. For instance, to calculate the color distribution of a video, the color distribution of each frame is calculated first, then the average of them is taken as the color distribution of the video.

Descriptors are constructed to quantitatively interpret extracted features. Among numerous descriptors, histograms are widely adopted, such as in color distribution, SIFT [12], and BoW (Bag of Words) methods.

2) Neighboring Video Retrieval: When the first stage ends, videos are summarized as multi-dimensional coordinates. Ideally, NDVs should be adjacent, whereas dissimilar ones should be distant in this feature space. With the coordinates of a given video and a distance measurement, we will be able to identify the near duplicate videos by its neighborhood.

Moreover, the execution of retrieving neighboring videos from large database is critical for detection speed improvement. To accelerate this execution, storage and retrieval assistance schemes such as, Hash table [4], inverted index file [7], or LSH (Locality Sensitive Hashing) [9], are introduced.

B. Feature-Centered Detection Paradigm

Conventionally, the feature representation construction in the first stage is the center of NDVD system design. In this part, we commence our discussion about this Feature-Centered detection paradigm with a theoretical model, in order to further investigate its drawbacks from with respect to dimensionality and informativeness.

1) Model: First, we define four relevant concepts in NDVD systems as follows.

Definition 1. The neighborhood of a video $v \in V$ is $U(v) = \{v' \in V | v' \in duplicate(v)\}$.

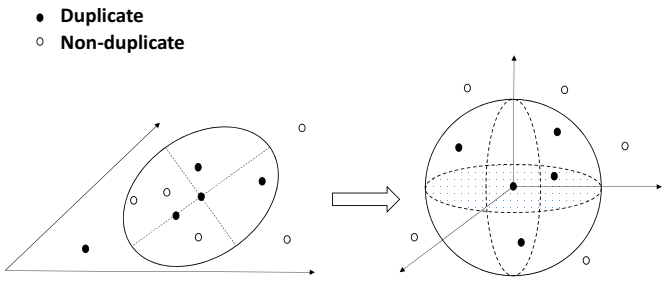


Fig. 1: Transformation of Feature Space

Definition 1 is independent of features.

Definition 2. The representation of a video $v \in V$ in feature $f \in F$ is defined as $X_f(v) \in R^n$ (i.e., Euclidean space).

Definition 3. The hypersphere neighborhood of a video $v \in V$ in feature $f \in F$ is defined as $S(X_f(v), \tau) = \{X_f(v') | v' \in V, |X_f(v') - X_f(v)| \leq \tau\}$, where $|.|$ is a distance measurement in the feature space.

Definition 4. The error set of $S(X_f(v), \tau)$ is defined as $E_f(v) = \{v' \in V | v' \in U(v), X_f(v') \notin S((X_f(v), \tau^*)\} \cup \{v' \in V | v' \notin U(v), X_f(v') \in S((X_f(v), \tau^*)\}$, where τ^* is the optimal value for $S(X_f(v), \tau)$.

By these definitions, after establishing the feature f, the detection, or classification task in this paradigm is as simple as testing whether $v' \in S(x_f(v), \tau^*)$, $v, v' \in V$. Its accuracy can be measured by the volume of $E_f = \{E_f(v) | v \in V\}$. The smaller it is, the better f is to embody videos.

As shown in the left part of Figure 1, the hypersphere neighborhood in a "raw" and simple feature f_1 may not be a satisfactory approximation , as $|E_f| = 4$. To increase the discriminative ability of feature representations, in the Feature-Centered paradigm, a higher-dimensional feature representation $X_f, f \in F$ is created by combining feature representations $X_{f_1}, X_{f_2}, \ldots, X_{f_n}$, $f_1, f_2, \ldots, f_n \in F$ [4], [7]. The hypersphere neighborhood in f should be more accurate as shown in the right part of Figure 1, as $|E_f| = 0$. However, this paradigm may encounter issues from the following perspectives of dimensionality and informativeness.

2) Dimensionality: The first potential issue of the Feature-Centered paradigm is the high dimensionality of representations. Typically, there are two manners of dimensionality growth; more features being integrated, or the vocabulary of visual words expanding. They can be explained by examples.

The LBP-based spatio-temporal feature [7] is an example of feature fusion. First, each frame is represented by a binary vector of 16 dimensions, thus there are $2^{16} = 65536$ possible patterns. Then the video representation, a histogram, is constructed by counting frames that fall into each pattern. In this way, the dimensionality of the representation is 65536.

In BoW methods, the dimensionality of representations is the number of visual words in the vocabulary, or $O(\sqrt{n})$ according to a rule of thumb, where n is the number of interest regions extracted from all videos. Given that there are 10^7 videos in a database, each of them has 10^2 frames and the average number of extracted regions is 10^3, the dimensionality of this representation, is $10^{\frac{7+2+3}{2}} = 10^6$.

Either the combinatorial explosion, or sublinear growth, could lead to the high-dimensionality of representations, which imposes heavy retrieval cost, thus reducing the detection speed of NDVD systems. On the other hand, the accuracy could also be negatively affected. When dimensionality increases, the maximum distance between two random representations becomes indiscernible compared to the minimum distance [13], as

$$\lim_{d \to \infty} E(\frac{dist_{max}(d) - dist_{min}(d)}{dist_{min}(d)}) = 0. \quad (1)$$

Thus the neighborhood defined on distance becomes less meaningful. In addition, when more irrelevant or noisy dimensions are involved, the accuracy of neighboring video retrieval will also drop [13].

3) Informativeness: The second potential issue of the paradigm comes from the reduction of informativeness, which is critical to the detection accuracy. We assume that feature representations emerge as histogram because it is widely adopted, such as in color distribution, SIFT, or BoW. The informativeness of representation is defined as entropy.

Definition 5. The informativeness of a video representation $X \in \{X_{f_1}, X_{f_2}, \ldots, X_{f_k}, \ldots\}$ is $H_v(X) = -\sum_i p_v(x_i) \log p_v(x_i)$; $f_1, f_2, \ldots, f_k, \ldots \in F$ are features; $p_v(x_i) = g_v(x_i) w_i$; $g_v : range(X) \to [0, 1]$ is the probability density function of X; $w_i = u_i - l_i$ is bin width, $x \in [l_i, u_i]$, $u_{i-1} = l_i$, $i = 2 \ldots n$, $\cup_{i=1}^{n} [l_i, u_i] = range(X)$, n is the dimensionality (number of bins) of X.

Two properties regarding information lost could be revealed under Definition 5.

Property 1. *$H_v(X) = 0$, if $n = 1$; $H_v(X) \to 0$, if $n \to \infty$.*

Proof. The first part is straightforward.

For the second part, as $n \to \infty$, $w_i \to 0$, thus $p_v(x_i) = g_v(x_i) w_i \to 0$. Additionally, according to the definition of entropy, $p(x) \log p(x) = 0$, when $p(x) = 0$. Therefore, $H_v(X) = -\sum_i p_v(x_i) \log p_v(x_i)_v \to 0$, as $n \to \infty$. □

According to Property 1, increasing the dimensionality of a representation does not necessarily make it more informative. On the contrary, as it becomes sparse, its informativeness is closer to 0. Essentially, it reveals the curse of dimensionality as Equation 1 does, from another perspective.

Property 2. $H(X_{f_1}, X_{f_2}, \ldots, X_{f_k}) \leq H(X_{f_1}) + H(X_{f_2}) + \ldots + H(X_{f_k})$.

Proof. Utilize the non-negativity [14] of the mutual information $I(X_{f_1}, X_{f_2}) = H(X_{f_1}) + H(X_{f_2}) - H(X_{f_1}, X_{f_2}) \geq 0$, and induction. □

From Property 2, constructing a sophisticated representation via feature fusion does not increase its informativeness compared with the collection of simpler representations. Therefore,

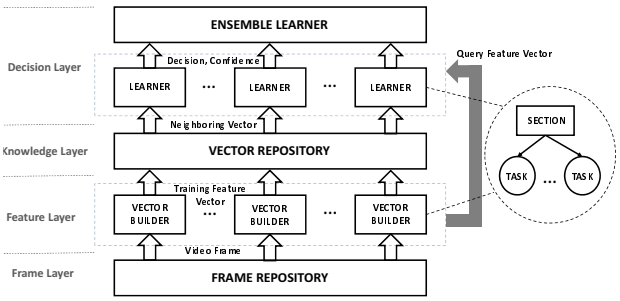


Fig. 2: The Architecture and Parallel Organization of CompoundEyes

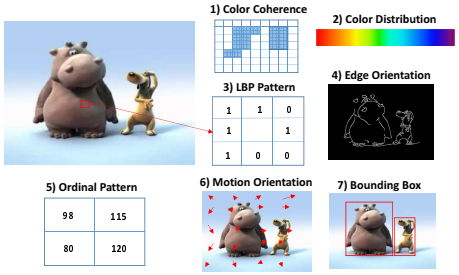


Fig. 3: The Seven Features in CompoundEyes

building a sophisticated classifier, and feeding it with simple feature representations, could be achieve higher accuracy than the combination of advanced representations and a simple classifier.

III. SYSTEM DESIGN

From Properties 2 and 1, we realize the gains in accuracy by shifting the focus away from building advanced feature representation towards advanced classifier. To achieve fast detection speed, our system is designed according to systems principles. Components are simple, efficient, and independent of each other. Parallelism provided by this autonomy is also exploited to further increase speed. In addition, efforts have been made to organize the feature extractors and classifiers to ensure a satisfactory performance both in accuracy and speed.

A. Architecture

CompoundEyes is designed by using an abstraction layer model. In this model, frames are sampled at the Frame layer, in which features are extracted and represented at the Feature layer. From these representations, patterns of NDVs rest in the Knowledge layer, which finally emerge in the Decision layer and are used to make predictions about videos being duplicated or not.

The system is divided into three subsystems: Feature Vector Builder, Vector Repository, and Ensemble Learner. These subsystems are located on the Feature, Knowledge and Decision layers, as shown in Figure 2. Furthermore, we divide the Feature Vector Builder subsystem into various Vector Builders, each of which uses a distinctive feature extraction and representation algorithm. For each Vector Builder, there is a weak Learner which uses its representations to form predictions. These predictions are collected by the Ensemble Learner, to make final predictions.

The parallel organization of CompoundEyes is hierarchical, as illustrated in Figure 2. The first level is the function parallelism among components, i.e., Vector Builders and weak Learners. They compete for parallel sections to perform their computations. The second level is the data parallelism within the computations of Vector Builders. Upon obtaining a parallel section, one or more parallel tasks are spawned, among which the computations of the Vector Builder are divided.

B. Data Flow

1) Feature Layer: In the Feature layer, we utilize seven feature extraction algorithms: color coherence, color distribution, LBP (Local Binary Pattern), edge orientation, ordinal pattern, motion orientation, and bounding boxes of objects, as explained in Figure 3. All of these algorithms are simple and efficient. Furthermore, feature diversity is positively correlated to the accuracy of the final prediction [15].

All the Vector Builders work on a frame-by-frame basis. Given the j-th Vector Builder deals with feature $f_j, j = 1, \ldots, 7$, it first extracts f_j from the i-th key-frame of video v and represents it as a histogram $m_i^{f_j}(v), i = 1, \ldots, N(v)$, where $N(v)$ is the number of key-frames in v. Then the video representation of v built by this Vector Builder is calculated as $M^{f_j}(v) = \frac{1}{N(v)} \sum_{i=1}^{N(v)} m_i^{f_j}(v)$. The frame-level data parallelism in this calculation is exploited by distributing the computations of $m_i^{f_j}(v), i = 1, \ldots, N(v)$ onto the tasks belonging to a parallel section obtained by this Vector Builder, as shown in Figure 2.

2) Knowledge Layer: To explain the neighboring video retrieval procedure in Vector Repository, we need the following definition.

Definition 6. The neighborhood of a video $v \in V$ in feature $f \in F$: $U_f(v, \tau) = \{M^f(v') | v' \in V, |M^f(v') - M^f(v)| \leq \tau\}$.

After videos are represented as $\{M^{f_j}(v) | j = 1, \ldots, 7\}, v \in V$, the representations of the videos in the training set V_t are stored and indexed in Vector Repository along with their ground-truth labels, separated by the features $f_j, j = 1, \ldots, 7$ into seven subspaces. This Vector Repository grants CompoundEyes the capability to act as both NDVD and NDVR (Near-Duplicate Video Retrieval) system.

When the representations of a query video v_q from the testing set V_q, $M^{f_j}(v_q), j = 1, \ldots, 7$ are issued to the Vector Repository, its neighborhoods in feature f_j, $U_{f_j}(v_q, \tau), j = 1, \ldots, 7$, are computed and returned to Learners in the Decision layer respectively. The main objective of the Video Repository is to make this neighboring video retrieval procedure more efficient.

We implement the Vector Repository as an LSH [16], [17] structure for two reasons. First, LSH is sensitive to locality thereby having the capacity of providing the neighboring video retrieval with more accurate results. Second, its retrieval

temporal cost is $O(1)$. In addition, the LSH structure is combined with Cuckoo Hashing [18]. As a result, the problems of unbalanced load among hash tables and of local similar sets are mitigated, further enhancing its retrieval performance.

3) Decision Layer: We model the NDVD task as a classification problem. A video $v \in V$ can belong to n possible classes $c_i, i = 1, \ldots, n$. For example, when $n = 2$, the classes are duplication and non-duplication. In CompoundEyes, $n = 7$, because the dataset we adopt divides videos into seven categories: Exactly Duplicate, Similar, Different Version, Major Change, Long Version, Dissimilar, and Do not Exist. Dissimilar and Do not Exist are treated as the same.

As in Figure 2, the learners (or classifiers) in the Decision layer are organized in a hierarchical manner. The prediction of a video being duplicate is made upon the hypotheses of the seven weak Learners.

The weak Learners are denoted as $L_j, j = 1, \ldots, N$, where $N = 7$ is equal to the number of features we adopt. The videos from both the training set V_t and testing set V_q are summarized as representations $\{M^{f_j}(v)|v \in V_t \cup V_q, j = 1, \ldots, 7\}$ in the Feature layer. $\{M^{f_j}(v)|v \in V_t, j = 1, \ldots, 7\}$ are stored in the Vector Repository along with their ground-truth labels $\{v = c_i|v \in V_t, i = 1, \ldots, 7\}$, while $\{M^{f_j}(v)|v \in V_q, j = 1, \ldots, 7\}$ are directed to Learners $L_j, j = 1, \ldots, N$, respectively, as shown in Figure 2. On L_j, the probabilities $p(v_q = c_i|L_j), i = 1, \ldots, 7$ are approximated with frequencies,

$$N(v_q = c_i|L_j) = \frac{|\{v = c_i|v \in V_t, v \in U_{f_j}(v_q, \tau)\}|}{|\{v|v \in V_t, v \in U_{f_j}(v_q, \tau)\}|},$$

$$i = 1, \ldots, 7.$$

The computation of $U_{f_j}(v_q, \tau)$ is performed by the Vector Repository, as mentioned above.

These frequencies are taken as input to the Ensemble Learner, which calculates the posterior probabilities $p(v_q = c_i|L_1, \ldots, L_7), i = 1, \ldots, 7$, utilizing the BKS (Behavior-Knowledge Space) method [19] as follows,

$$p(v_q = c_i|L_1, \ldots, L_7) \cong \hat{p}(v_q = c_i|L_1, \ldots, L_7),$$

$$\hat{p}(v_q = c_i|L_1, \ldots, L_7) = \frac{N(v_q = c_i|L_1, \ldots, L_7)}{\sum_i N(v_q = c_i|L_1, \ldots, L_7)}.$$

To make estimating $N(v_q = c_i|L_1, \ldots, L_7), i = 1, \ldots, 7$ easier, , we assume $L_j, j = 1, \ldots, 7$ are conditionally independent, because of the diversity of features, and with $p(v_q = c_i|L_j) \cong N(v_q = c_i|L_j), i = 1, \ldots, 7$, then we have,

$$\begin{aligned} p(v_q &= c_i|L_1, \ldots, L_7) \propto p(L_1, \ldots, L_7|v_q = c_i) \\ &= \prod_{j=1}^{7} p(L_j|v_q = c_i) \propto \prod_{j=1}^{7} p(v_q = c_i|L_j) \\ &\cong \prod_{j=1}^{7} N(v_q = c_i|L_j), i = 1, \ldots, 7. \end{aligned}$$

Therefore, with appropriate normalization, the probabilities are estimated as

$$p(v_q = c_i|L_1, \ldots, L_7) = \frac{\prod_{j=1}^{7} N(v_q = c_i|L_j)}{\sum_{i=1}^{7} \prod_{j=1}^{7} N(v_q = c_i|L_j)},$$

$$i = 1, \ldots, 7.$$

The class with the largest posterior probability would be the final prediction of the class of v_q.

The combination of the the Nearest Neighbor algorithm applied on the weak Learners and the BKS method on the Ensemble Learner appears satisfactory to the design of CompoundEyes. First, the Vector Repository directly provides an interface to efficiently compute $U_{f_j}(v_q, \tau)$, whose cost is $O(1)$. Second, the Nearest Neighbor algorithm is non-parametric, which is helpful to reduce the training cost to $O(1)$, fulfilling the in-situ requirement. Third, the Nearest Neighbor algorithm is sensitive to the variations of feature types [15], thus making it suitable in the scenario of multiple feature subspaces. Fourth, the BKS method is sufficiently accurate to be applied on the Ensemble Learner [19].

C. Advantages

The advantages of CompoundEyes can be illustrated from the following aspects.

a) Accuracy: The accuracy improvement is primarily achieved via the collective effort of learners. First, the coverage of feature space is broader. Not only are spatial and temporal information used, but also color, edge orientation, texture, and object sizes information is also included in learning. Second, the diversity of representations enhances the accuracy of learning.

b) Detection Speed: Primarily, two factors contribute to the increase of detection speed. The first one is the compactness of representations, which shortens the temporal cost of extracting feature vectors in the preprocessing stage and of neighboring representation retrieval in the processing stage. The second one is the exploiting of the function parallelism among the Vector Builders and Learners, and the frame-level data parallelism within the Vector Builders.

c) In-situ Updating: CompoundEyes has the capacity of constantly updating its classifiers when incorporates new knowledge (i.e., videos and corresponding ground-truth labels), because the cost of training classifiers is $O(1)$, and the changes in classifiers do not affect the construction of representations in the Feature Layer.

d) Modularity: The components in CompoundEyes are independent, and so can be changed without affecting others. For example, a new Vector Builder detecting a new type of features can be admitted if necessary, so is the case with weak learners implementing other algorithms, and the Vector Repository utilizing alternative indexing schemes. Therefore, the system could be easily upgraded.

IV. EVALUATION

A. Experimental Setup

We implement CompoundEyes in C++, C, and Matlab. Specifically, Vector Builders are coded in C++, using the OpenCV libraries. Weak Learners and NEST are implemented in C, and the Ensemble Learner is programmed in Matlab. The parallel parts of CompoundEyes are implemented by using OpenMP libraries.

Experiments are conducted on a 64-core Intel Xeon E5-4640 machine (2.4GHz, 12.5GB memory) with Ubuntu system. The cores are distributed equally into 4 NUMA nodes.

B. Dataset Description

We evaluate CompoundEyes on the CC_WEB_VIDEO dataset. There are four reasons for this selection.

- First, it was constructed from real online videos. All the videos were downloaded from YouTube, Google Video and Yahoo! Video.
- Second, various formats and editorial modifications are included.
- Third, it has been widely adopted, which facilitates us to compare the performance.
- Fourth, ground-truth labels are provided. These labels are obtained manually, which is laborious and makes the dataset precious for NDVD/NDVR research.

The CC_WEB_VIDEO dataset is comprised of 24 independent groups. In each group, a video is designated as the seed and others are compared with it and labeled accordingly.

C. NDVD/NDVR Systems to Compare with

To evaluate the performance of CompoundEyes, we compare it with existing state-of-the-art NDVD/NDVR systems, which have been evaluated on the CC_WEB_VIDEO dataset or on extended datasets. They are described briefly as follows.

Hierarchical detection system (HIER): Wu et al. [2] proposed a hierarchical NDVD system, which uses a global signature-based method to filter out duplicates with minor changes first, leaving more sophisticated ones to the local feature-based method.

Video Cuboid based detection system (VC): Zhou et al. [9] introduced the Video Cuboid signature, a n-gram based representation, to integrate the temporal and spatial information. Further optimizations include the use of the EMD distance, the incremental signature construction, and an LSH based matching scheme.

Spatial-Temporal feature based detection system (ST): Shang et al. [7] explored alternative approaches of combining the temporal and spatial information into signatures. Two approaches are proposed: Conditional Entropy (ST-CE) and Local Binary Pattern (ST-LBP). The retrieval process is accelerated by applying a fast intersection kernel and inverted file.

Multiple feature hashing based detection system (MFH): Song et al. [4] provided another combination of a global and a local feature of videos. A series of hash functions are learned from feature representations. The neighboring video searching is conducted in Hamming space of the hash codes.

TABLE I: The Comparison in MAP

SYSTEM	HIER	ST-CE	ST-LBP	MFH	**Ours**
MAP (%)	95.20	95.30	95.00	95.40	**99.75**

TABLE II: The Comparison in Peak Memory Usage and Time Complexity

SYSTEM	HIER	ST-CE	ST-LBP	MFH	**Ours**
Peak Memory Usage	$O(k)$	$O(n)$	$O(n)$	$O(k^3n^3)$	$O(k)$
Time Complexity	$O(kn^2)$	$O(kn)$	$O(kn)$	$O(k^3n^3)$	$O(kn)$

In these systems, VC provides us with the results of accuracy, while others are more concerned with mean average precision and average response time. Hence, we will compare CompoundEyes with VC in terms of accuracy, and with others in terms of mean average precision and average response time.

D. Experimental Results

In this subsection, extensive experiments are conducted to evaluate the performance of CompoundEyes. Datasets of various sizes are constructed by randomly selecting videos from the CC_WEB_VIDEO dataset. Unless stated otherwise, in each one of them, 50% are used as the training set and the other 50% as the testing set.

1) Accuracy:

a) Evaluation Metrics:

- **Accuracy**: It is computed as $AC = \frac{n}{N}$, the portion of correct predictions in total results.
- **Mean Average Precision**: The Mean Average Precision (MAP) is computed by averaging the Average Precision (AP) of each group g, as $MAP = \frac{1}{24}\sum_{g=1}^{24} AP_g$, $AP_g = \frac{1}{n}\sum_{i=1}^{n}\frac{i}{r_i}$, where n is the number of correct predictions , r_i is the rank of i-th correct prediction.

b) Results: CompoundEyes shows an improvement on detection accuracy. It achieves a higher Accuracy than the VC system, 89.28% vs. 80%, and outperforms other systems in Mean Average Precision, as shown in Table I.

2) Detection Speed:

a) The Definition of Temporal Cost: The detection speed of CompoundEyes is measured by the temporal cost, which is the sum of the preprocessing time and response time.

$$Temporal\,Cost = Preprocessing\,Time + Response\,Time.$$

b) Analysis of Preprocessing Time Cost: In literature, preprocessing is performed offline thus its temporal cost is not measured. However, the burden of preprocessing can be estimated from the fact that feature extraction of HIER, ST-CE or ST-LBP on a dataset of 132647 videos is practically impossible [4].

Suppose the number of videos is n, and the average number of key-frames in a video is k. The peak memory usage and worst case time complexity of the preprocessing of various systems is estimated in Table II.

TABLE III: The Comparison in Response Time (RT)

SYSTEM	HIER	ST-CE	ST-LBP	**Ours**
RT (ms)	9600	3.7	3.6	**0.2051**

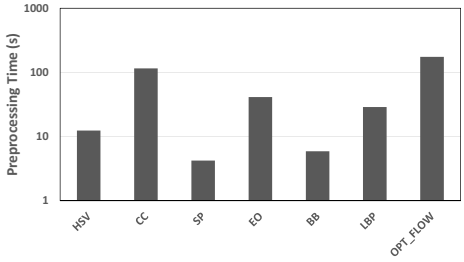


Fig. 4: The Sequential Preprocessing Time of All the Vector Builders.

According to Table II, CompoundEyes has advantages in both the peak memory usage and time complexity. It neither involves the computations and pairwise comparisons of SIFT descriptors as HIER, nor the computations of certain global variables, i.e. the entropy of ordinal relations in ST-CE, the correlation between LBP patterns in ST-LBP, and the transformation and bias matrices in MFH, which are both spatially and temporally exhaustive. In contrast, the two major operations of CompoundEyes in preprocessing, the construction of representations and inserting them into the NEST tables, are all spatially and temporally efficient. By experiment, the average temporal cost of preprocessing is 1.4537s.

c) Experimental Results of Response Time Cost: The advantage of CompoundEyes in detection speed can also be manifested from response time, as shown in Table III. The average response time of CompoundEyes only accounts to 5.70% of ST-LBP's.

Implementing the main part of CompoundEyes in C++, rather than Matlab may contribute to the reduction of response time. However, such a substantial reduction could not be explained merely by the efficiency of C++. In CompoundEyes, the dimensionality of representations could be 16, 32, or 64, all of which are much lower 65536 of ST-CE and ST-LBP [7]. This reduction in dimensionality is the main reason for the improvement on response time.

3) Parallel Speedup: Experiments in this subsection are performed on a 10% subset of CC_WEB_VIDEO, because it is time-consuming to use the whole dataset for all of them. To evaluate speedup, the temporal cost of sequential version and parallel version are compared.

The temporal cost of each Vector Builder is estimated in Figure 4 first, and used as a reference for workload distribution. On the horizontal axis are the abbreviations of the features they extract, which are color histogram (HSV), color coherence (CC), ordinal pattern (SP), edge orientation (EO), bounding boxes of objects (BB), local binary pattern (LBP), and motion orientation (OPT_FLOW).

a) Thread Allocation Strategies: Both of the parallel sections and tasks in Figure 2 are abstraction of thread.

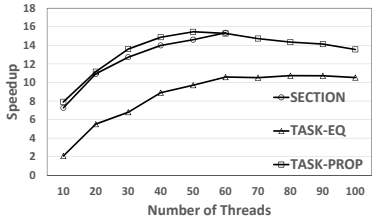
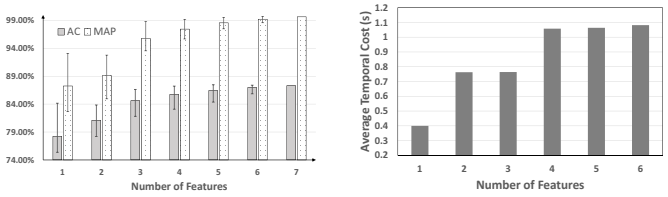


Fig. 5: The Speedup of CompoundEyes Under 3 Thread Allocation Strategies



(a) On AC and MAP (b) On Temporal Cost

Fig. 6: The Effect of Feature Information Fusion

Under different thread allocation strategies, the overall parallel speedup would be different. Therefore, we design and compare three allocation strategies as follows.

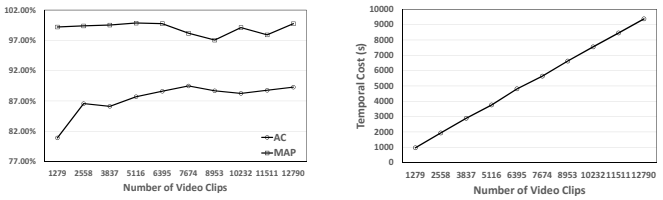
- **SECTION**: What varies in this strategy is the number of parallel section competed by Vector Builders, from 1 to 7. Once a parallel section obtained, a number of parallel tasks, proportional to the Vector Builder's sequential running time, will be allocated for computing.
- **TASK-EQ**: In this strategy, every Vector Builder acquires a parallel section. What varies is the number of tasks spawned by a section, which is equal for all the Vector Builders.
- **TASK-PROP**: In this strategy, not only does every Vector Builder obtain a parallel section, but also the number of tasks allocated to a Vector Builder is proportional to its sequential running time.

b) Results: As expected, from Figure 5, Strategy TASK-PROP achieves the best speedup, because it efficiently utilizes allocated threads. Moreover, we notice that when the thread number exceeds 60, the increase of speedup ceases. This value coincides with the number of cores in the machine. This phenomenon is a hint of resource contention.

We also notice that even under the best thread allocation strategy, the speedup is far from linear speedup. This is determined by the fact that in CompoundEyes, videos are processed sequentially, which limits the throughput of the system.

4) Feature Information Fusion: In this part, we assess the impact of the feature information fusion, mainly on detection accuracy. The experiments are conducted on a 10% subset. For the sake of fair comparison, the number of parallel sections is equal to the number of features to be used, and the number of tasks a section can spawn is equal for all the Vector Builders.

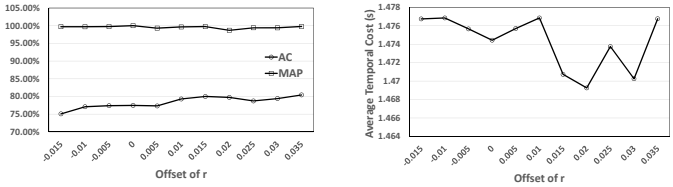
As shown in Figure 6(a), on average, the fusion increases



(a) On AC and MAP (b) On Temporal Cost

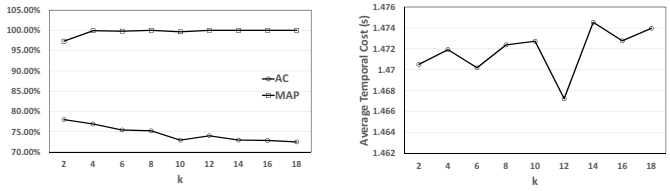
Fig. 7: The Effect of the Dataset Scale

Fig. 8: The Effect of the Portion of the Training Set on AC and MAP



(a) On AC and MAP (b) On Temporal Cost

Fig. 9: The Effect of r



(a) On AC and MAP (b) On Temporal Cost

Fig. 10: The Effect of k

detection accuracy, both in terms of Accuracy and Mean Average Precision. This advantage becomes smaller when measured by the best accuracy of fusion. For example, the accuracy difference between the best combination of three features and four is negligible. This suggests the importance of the selection of feature information to be fused.

For the best combinations except all-included, corresponding average temporal costs are shown in Figure 6(b). They are helpful when choosing the number of features. For example, fusing three is better than four, because it costs less time but achieves comparable detection accuracy.

5) Relevant Parameters:

a) The Scale of the Dataset: The first relevant parameter is the scale of the dataset. According to Figure 7(a), the Accuracy is satisfactory, above 80%, when the size is 1279. It also increases as the size of the dataset grows. Therefore, CompoundEyes is accurate when sufficient knowledge has been learned, and its discriminative capability develops as knowledge accumulates.

Figure 7(b) affirms that the total temporal cost increase linearly rather than exponentially with the growth of dataset. This linearity confirms that Vector Repository is capable of maintaining decent performance even if the size of the dataset becomes large.

b) The Portion of the Training Set: Because a system well-tuned on training set could behave poorly on testing set, it is necessary to evaluate the detection accuracy of CompoundEyes under different portions of the training set.

The effect of this portion on Accuracy and Mean Average Precision is recorded in Figure 8. The value of MAP stays stable, and the value of AC increases as the ratio increases. Both of them peak around $5:5$. Afterward, classifiers are over-trained.

c) NEST-related Parameters: Two NEST-related parameters, r and k, are of importance. Parameter r is used as τ in the definition of the neighborhood in feature. Parameter k is the number of hash tables in NEST. Generally speaking, a larger value of k increases detection accuracy, at the expense of longer response time.

Because the value of r is different for each type of feature representations, we set them by experience first, then change them with the same offset. The effect of r on Accuracy and Mean Average Precision is shown in Figure 9(a), and the effect on average temporal cost is shown in Figure 9(b).

Since k is the same for all feature spaces, we vary its value directly. From Figure 10(a), we observe that Accuracy and Mean Average Precision exhibit different trends, the former one goes down and the latter one goes up and stays around 100%. This is because as k increases, the recall of neighboring video retrieval grows, but the precision goes down. These changes reflect on Accuracy but not Mean Average Precision, for the number of correct results and their ranks are barely affected.

The effect of k on average temporal cost is shown in Figure 10(b), from which we know that 12 is the optimal value for detection speed.

V. RELATED WORK

The knowledge about the content of big data has manifested its importance to various applications, such as in-network image deduplication [20], and confidential data protection [21]. This knowledge is generally acquired by extracting features from data, for example, DoG and PCA-SIFT [20], or document fingerprint [21].

Feature combination is a common approach in constructing discriminative video feature representations, typical examples including global and local features [2], [4], [5], [22], or spatial and temporal features [7], [9]. The combination can be simply

concatenating representations [5], or using sophisticated mathematical transformations [4], [22]. Liu et al. [11] summarized the development of feature representations.

With the support of indexing structures such as hash tables [9], [4], [7], the response time of these systems is satisfactory. However, to be qualified as an online system, capable of identify NDVs in the cloud, the preprocessing procedure should not be conducted offline, and its temporal cost should be substantially reduced.

VI. CONCLUSION

In this paper, we proposed and developed an efficient NDVD cloud system, called CompoundEyes, by using a new detection paradigm. Instead of designing a sophisticated video representation, the focus has been shifted to the design of a well-organized system. Rather than feature design, we introduced improvements in accuracy through classifiers. Through use of reduced dimensionality and parallelism, we reduced the duration required for precise duplicate detection. Moreover, experiments and analysis corroborated that CompoundEyes outperforms contemporary NDVD and NDVR systems in accuracy. At the same time, CompoundEyes bested or matched its peers in both peak memory usage and time complexity. In conclusion, CompoundEyes is feasible and practical to perform large-scale NDVD tasks in the cloud. As other NDVD/NDVR systems, CompoundEyes needs a training set and ground-truth labels, the acquisition of which is beyond the scope of this paper. In the future, we will migrate this system to more cutting-edge cloud platforms such as Spark, to overcome the limitations of shared memory parallel computing architectures.

REFERENCES

[1] M. LOPES. Videos may make up 84 percent of internet traffic by 2018: Cisco. [Online]. Available: http://www.reuters.com/article/2014/06/10/us-internet-consumers-cisco-systems-idUSKBN0EL15E20140610

[2] X. Wu, A. G. Hauptmann, and C.-W. Ngo, "Practical elimination of near-duplicates from web video search," in *Proceedings of the 15th international conference on Multimedia*. ACM, 2007, pp. 218–227.

[3] J. S. Pedro, S. Siersdorfer, and M. Sanderson, "Content redundancy in youtube and its application to video tagging," *ACM Transactions on Information Systems (TOIS)*, vol. 29, no. 3, p. 13, 2011.

[4] J. Song, Y. Yang, Z. Huang, H. T. Shen, and R. Hong, "Multiple feature hashing for real-time large scale near-duplicate video retrieval," in *Proceedings of the 19th ACM international conference on Multimedia*. ACM, 2011, pp. 423–432.

[5] M. Hefeeda, T. ElGamal, K. Calagari, and A. Abdelsadek, "Cloud-based multimedia content protection system," 2013.

[6] YouTube. Statistics. [Online]. Available: http://www.youtube.com/yt/press/statistics.html

[7] L. Shang, L. Yang, F. Wang, K.-P. Chan, and X.-S. Hua, "Real-time large scale near-duplicate web video retrieval," in *Proceedings of the international conference on Multimedia*. ACM, 2010, pp. 531–540.

[8] A. G. H. Xiao Wu, Chong-Wah Ngo. Cc_web_video: Near-duplicate web video dataset. [Online]. Available: http://vireo.cs.cityu.edu.hk/webvideo/

[9] X. Zhou and L. Chen, "Monitoring near duplicates over video streams," in *Proceedings of the international conference on Multimedia*. ACM, 2010, pp. 521–530.

[10] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, ser. STOC '98. New York, NY, USA: ACM, 1998, pp. 604–613. [Online]. Available: http://doi.acm.org/10.1145/276698.276876

[11] J. Liu, Z. Huang, H. Cai, H. T. Shen, C. W. Ngo, and W. Wang, "Near-duplicate video retrieval: Current research and future trends," *ACM Computing Surveys (CSUR)*, vol. 45, no. 4, p. 44, 2013.

[12] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 10, pp. 1615–1630, 2005.

[13] WIKIPEDIA. Curse of dimensionality. [Online]. Available: https://en.wikipedia.org/wiki/Curse_of_dimensionality

[14] Wikipedia. Mutual information. [Online]. Available: https://en.wikipedia.org/wiki/Mutual_information

[15] C. Domeniconi and B. Yan, "Nearest neighbor ensemble," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 1. IEEE, 2004, pp. 228–231.

[16] Y. Hua, B. Xiao, and X. Liu, "Nest: Locality-aware approximate query service for cloud computing," in *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013, pp. 1303–1311.

[17] Z. Nie, Y. Hua, D. Feng, Q. Li, and Y. Sun, "Efficient storage support for real-time near-duplicate video retrieval," in *Algorithms and Architectures for Parallel Processing*. Springer, 2014, pp. 312–324.

[18] R. Pagh and F. F. Rodler, *Algorithms — ESA 2001: 9th Annual European Symposium Århus, Denmark, August 28–31, 2001 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, ch. Cuckoo Hashing, pp. 121–133. [Online]. Available: http://dx.doi.org/10.1007/3-540-44676-1_10

[19] S. Tulyakov, S. Jaeger, V. Govindaraju, and D. Doermann, "Review of classifier combination methods," in *Machine Learning in Document Analysis and Recognition*. Springer, 2008, pp. 361–386.

[20] Y. Hua, W. He, X. Liu, and D. Feng, "Smarteye: Real-time and efficient cloud image sharing for disaster environments," in *Proc. INFOCOM*, 2015.

[21] F. Hao, M. Kodialam, T. Lakshman, and K. P. Puttaswamy, "Protecting cloud data using dynamic inline fingerprint checks," in *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013, pp. 2877–2885.

[22] J. Liu, Z. Huang, H. T. Shen, and B. Cui, "Correlation-based retrieval for heavily changed near-duplicate videos," *ACM Transactions on Information Systems (TOIS)*, vol. 29, no. 4, p. 21, 2011.