Bandwidth and Energy Efficient Image Sharing for Situation Awareness in Disasters

Pengfei Zuo, *Student Member, IEEE*, Yu Hua, *Senior Member, IEEE*, Yuanyuan Sun, Xue Liu, *Member, IEEE*, Jie Wu, Yuncheng Guo, Wen Xia, Shunde Cao, and Dan Feng, *Member, IEEE*

Abstract—In order to save human lives and reduce injury and property loss, Situation Awareness (SA) information is essential and important for rescue workers to perform the effective and timely disaster relief. The information is generally derived from the shared images via widely used smartphones. However, conventional smartphone-based image sharing schemes fail to efficiently meet the needs of SA applications due to two main reasons, i.e., real-time transmission requirement and application-level image redundancy, which is exacerbated by limited bandwidth and energy availability. In order to provide efficient image sharing in disasters, we propose a bandwidth- and energy- efficient image sharing system, called BEES. The salient feature behind BEES is to propose the concept of Approximate Image Sharing (AIS), which explores and exploits approximate feature extraction, redundancy detection, and image uploading to trade the slightly low quality of computation results in content-based redundancy elimination for higher bandwidth and energy efficiency. Nevertheless, the boundaries of the tradeoffs between the quality of computation results and efficiency are generally subjective and qualitative. We hence propose the energy-aware adaptive schemes in AIS to leverage the physical energy availability to objectively and quantitatively determine the tradeoffs between the quality of computation results and efficiency. Moreover, unlike existing work only for cross-batch similar images, BEES further eliminates in-batch ones via a similarity-aware submodular maximization model. The response time of querying similar images is reduced via leveraging a geographic coordinate based index partitioning scheme. We have implemented the BEES prototype which is evaluated via three real-world image datasets. Extensive experimental results demonstrate the efficacy and efficiency of BEES. We have released the source codes of BEES at GitHub.

Index Terms—Image sharing system, content-based redundancy elimination, situation awareness, disaster environments

1 Introduction

During disaster events, Situation Awareness (SA) information, such as the surroundings and individuals, road conditions, resource information, etc., is essential and important, since the real-time responders and rescue workers rely on the SA information to perform the effective and timely disaster relief to save human lives and reduce injury and property loss [1], [2], [3].

Images are full of rich information (e.g., people, locations, and events) to present the real situations and provide vivid description of in-situ objects, which play an important role in the disaster relief [1], [2], [3], [4]. The images taken by smartphones usually include the current geographic coordinates that are recorded in the EXIF headers of generated JPEG photographs [5]. Due to the extensive use and easy access to Internet of smartphones, smartphones based crowdsourcing for sharing image-based information is important and helpful to support SA. For example, crowdsourcing has been applied in the Nepal earthquake to collect the latest information from earthquake-affected areas and create a dynamic map that shows the locations in which aid and relief are needed [6]. In the Typhoon Haiyan (2013), as part of the relief efforts, social medias using the shared images have been exploited and explored by volunteers to show where the most help is needed [7].

Although the image-based information is beneficial for SA, the image sharing via smartphones based crowdsourcing fails to efficiently support the SA in the disaster relief due to three main limitations. 1) Bandwidth Bottleneck. Due to the potential damages on communication infrastructure in disasters, network bandwidth possibly becomes very limited in capacity. Even though some schemes are utilized to remedy the network communication, such as, delay tolerant networks [8], [9] and mobile ad hoc networks [10], [11], the strict bandwidth constraint remains [12]. 2) Energy Constraint. The smartphones are used to take and upload massive images. It is well-known that smartphones have a low battery lifetime that is the main concern for users. For example, ChangeWave conducted a market study for smartphone dislikes, which shows 38 percent of the respondents listed battery lifetime as their biggest complaint [13]. More importantly, it is difficult for limited-energy smartphones to be re-charged in the context of disasters due to the infrastructure destruction [4]. 3) Real-time Transmission Inefficiency. Real-time data analytics are important for time-sensitive decision making in disaster relief [1], such as, the prediction

- *P. Zuo, Y. Hua, Y. Sun, J. Wu, Y. Guo, W. Xia, S. Cao, and D. Feng are with the Wuhan National Laboratory for Optoelectronics, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, Hubei 430074, China. E-mail: {pfzuo, csyhua, sunyuanyuan, wujie, ycguo, xia, csd, dfeng}@hust.edu.cn.*
- *X. Liu is with the School of Computer Science, McGill University, Montreal, Quebec H3A 0G4, Canada. E-mail: xueliu@cs.mcgill.ca.*

Manuscript received 25 Feb. 2018; revised 26 June 2018; accepted 22 July 2018. Date of publication 25 July 2018; date of current version 12 Dec. 2018. (Corresponding author: Yu Hua.)
Recommended for acceptance by Z. Chen.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPDS.2018.2859930

about the impact of the disasters and the selection of the suitable responses to the disasters. To support real-time data analytics, we need to timely deliver the images with SA information taken by smartphones to the servers.

A large number of images that users upload during the disaster events contain significant redundancies. For example, 53 and 22 percent similar images exist respectively in the San Diego fire (2007) and the Haiti earthquake (2010) imagesets [3]. Specifically, according to the observations of Facebook [14], users prefer to upload a batch of images (e.g., an album) instead of a single image. Thus the batch upload produces the cross-batch and in-batch similar images. In general, *cross-batch similar images* are the images that are similar to the images in the servers uploaded by other batches, which are produced by the cases that multiple users take pictures for the same objects or situations. *In-batch similar images* are the similar images among the images belonging to the same batch, which is also a common situation, such as, burst shooting and taking multiple pictures for identical objects. Image sharing in disasters mainly aims to provide the SA information and clues for disaster relief. Sharing redundant images consumes the limited resources but provides repetitive information [3], [4], [15], [16]. Hence, it is important to eliminate the transmission of redundant/similar images to obtain bandwidth and energy savings. However, to achieve the goals of the bandwidth and energy efficiency when eliminating the similar images, there are several challenges to address.

1) *Efficiently identify in-batch similar images*. Existing schemes [3], [4], [16] only eliminate the cross-batch similar images while overlooking in-batch similar images. The former is easy to be identified by querying the server index. For a queried image, if there exist similar images in the servers, the image does not need to be uploaded. Otherwise, it will be uploaded. However, identifying the latter is nontrivial. Only querying the server index cannot identify in-batch similar images since these images are not uploaded and hence their image features do not exist in the index. The key problem to identify in-batch similar images is how to select the retained unique images in a batch.
2) *Deal with the inefficiency of simply eliminating similar images*. The second challenge is that simply eliminating redundant images becomes inefficient to low redundancy. Existing schemes [2], [3], [4], [16] only aim to eliminate the image redundancy to improve the efficiency of image sharing, whose performance heavily relies on the percentage of redundant images to be uploaded. In the worst case, although not impossible, few redundant images exist in the uploaded images. These schemes become ineffective, and even consume more energy than directly uploading images due to requiring to extract image features for similarity detection.
3) *Obtain suitable tradeoffs in similar images elimination*. There exist a series of approximate computing processes during eliminating redundant images, such as image feature extraction and similar image detection. However, the boundaries of the tradeoffs between the quality of computation results and energy and bandwidth efficiency in these approximate computing processes are usually subjective and qualitative. For example, extracting high-quality image features results in high detection precision while consuming high energy. It is difficult to deal with the tradeoff between detection precision and energy efficiency for feature extraction.

To address these challenges, we propose a **B**andwidth- and **E**nergy-**E**fficient image **S**haring system [17], called BEES,[1] to offer real-time SA in the disaster environments. BEES aims to substantially reduce the consumption of bandwidth and energy during image sharing, and maintain the high efficiency even in the worst case where few redundant images exist. Moreover, by monitoring the energy availability, BEES adaptively adjusts its behaviors and carefully handles the tradeoffs between the quality of computation results and energy consumption to obtain energy savings and extend the battery life. To achieve these design goals, we have the following contributions.

1) To identify and eliminate in-batch similar images, we propose a similarity-aware submodular [18] maximization model (SSMM) in BEES to compute the unique image subset for each uploaded image batch.
2) To deal with the inefficiency of simply eliminating similar images, we propose the concept of Approximate Image Sharing (AIS), which leverages approximate feature extraction, approximate redundancy detection, and approximate image uploading to trade the slightly low quality of computation results in content-based redundancy elimination for higher bandwidth and energy efficiency.
3) To obtain suitable tradeoffs in approximate computing processes of AIS, we propose the energy-aware adaptive schemes to leverage the physical energy availability to determine the tradeoffs between the quality of computation results and efficiency, which provides an objective and quantitative tradeoff boundary.
4) To reduce the response time of querying similar images, we propose a geographic coordinate based index partitioning (GIP) scheme, which exploits the geographic coordinate properties of images. The proposed GIP partitions the entire feature index into grids based on the longitudes and latitudes of images. Thus to search the similar images of a given image, we only need to query limited grids that are covered by a small range of which center is the geographic coordinate of the queried image. Thus the response time of querying similar images is reduced via efficiently reducing the search range.
5) We have implemented the BEES prototype and evaluated its performance by using three real-world image datasets. Compared with the state-of-the-art schemes, including SmartEye [4] and MRC [16], experimental results demonstrate that BEES reduces more than 67.3 percent energy overhead, 77.4 percent bandwidth overhead, 70.4 percent average image

1. BEES collecting images in disaster areas via crowdsourcing looks like a number of bees gathering pollen in a flower field.

uploading delay, and extends the battery lifetime by 84.3 percent. Moreover, the energy-aware adaptive schemes in BEES further extend the battery lifetime by about 20 percent, compared with BEES without energy-aware adaptive schemes. The source codes of BEES have been released at GitHub.[2]

The rest of this paper is organized as follows. Section 2 presents the background of image features. Section 3 presents the system architecture of BEES. The design details of BEES are described in Section 4. We evaluate the performance in Section 5 and discuss several issues about the design of BEES. Section 6 presents the related work. We conclude this paper in Section 7.

2 Background and Motivation

In this section, we aim to explore which kind of image feature is suitable in the resource-constrained environments. We first present several state-of-the-art feature extraction algorithms. We then explore the performance characteristics of these feature extraction algorithms in smartphones.

2.1 Image Features

Both global features [19] and local features [20] of images can be used to detect similar images. Global features generalize the entire content of an image with a single feature vector, which can be computed by color histogram, texture values, shape parameters of images, etc. Local features are computed at multiple points in the images. Since local features have more robust and higher accuracy than global features for similarity detection [3], [16], [21], we focus on the local features of images in BEES.

We first present existing local feature extraction algorithms. Local feature extraction consists of two steps, i.e., feature point detection and feature descriptor. SIFT [20] is a widely used algorithm to detect and describe local features in images. Each feature in SIFT is a 128-dimension vector. SIFT has high accuracy, but causes high computation complexity. PCA-SIFT [22] improves the feature descriptor method of SIFT and reduces the dimensions of features from 128 to 36. SURF [23] aims to speed up the feature extraction by developing a scale- and rotation-invariant feature point detector and descriptor. The dimension of the SURF feature is configurable, i.e., 128 or 64 dimensions. ORB [24] uses FAST feature detector [25] and BRIEF descriptor [26]. The ORB feature is binary rather than vector-based computed by the above several algorithms, and described by 256 binary digits.

An image I_i can be represented as a set of local features S_i. The similarity of two images I_1 and I_2 can be computed as the Jaccard similarity of sets S_1 and S_2:

$$sim(I_1, I_2) = \frac{|S_1 \bigcap S_2|}{|S_1 \bigcup S_2|}. \quad (1)$$

2.2 Energy and Bandwidth-Constrained Feature Extraction in Smartphones

Previous schemes [24], [27], [28] mainly evaluate and compare feature extraction algorithms in terms of the precision of similarity detection, the time of feature extraction or the speed of similarity detection, failing to exploring the efficiency of their bandwidth and energy efficiency in smartphones. The efficiency is important in resource-constraint disaster environments. Hence, we study bandwidth and energy characteristics of image feature extraction algorithms via extensive smartphone-based experiments. The experimental setup is described in Section 5.1.

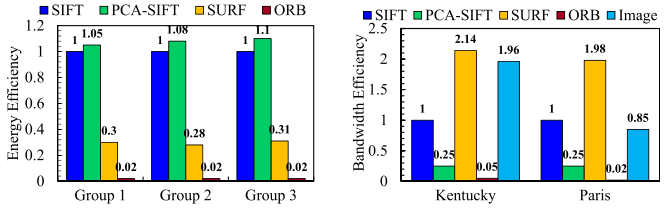


Fig. 1. The normalized energy and bandwidth efficiency.

• *Energy efficiency:* i.e., *the energy overhead of extracting image features in smartphones*. We select three-group images from the disaster imageset described in Section 5.1. Each group contains 100 images. We respectively extract their SIFT, PCA-SIFT, SURF and ORB features in smartphones, and capture the energy overheads. Their energy overheads are normalized to that of SIFT as shown in Fig. 1a.

• *Bandwidth efficiency:* i.e., *the bandwidth overhead of uploading image features, which is equal to the size of image features*. We respectively extract SIFT, PCA-SIFT, SURF and ORB features of all images in the Kentucky and disaster imagesets described in Section 5.1. Note that the numbers of SIFT, PCA-SIFT and SURF features for each image are related to image contents. 500 ORB features are extracted for each image as suggested by Rublee et al. [24]. Thus the size of ORB features for each image is only 16KB ($32B * 500$). The sizes of four kinds of features and the total size of images are normalized to the size of SIFT as shown in Fig. 1b.

As shown in Fig. 1, SIFT and PCA-SIFT consume huge energy to extract features with low energy efficiency. The SIFT and SURF are large-sized and even larger than the sizes of images themselves, with low bandwidth efficiency. The energy and bandwidth efficiencies of ORB are about two orders of magnitude better than that of other algorithms, due to lightweight computation and small-size binary features. The size of ORB features is much smaller than the image size.

In summary, ORB is more energy- and bandwidth-efficient than other feature extraction algorithms, and its produced image features have much smaller size than images. This observation motivates us to eliminate similar images in the source via uploading ORB features in advance for detecting similar images, which would significantly reduce the energy and bandwidth overheads of uploading images in real-time disaster SA.

3 The System Architecture of BEES

In this section, we first describe the traditional system architecture for smartphone-based image sharing and the system architecture of BEES. We then present the workflow of BEES system.

3.1 The System Framework

In the traditional system architecture [4], [16], as shown in Fig. 2, there are three key components, i.e., Image Feature

2. https://github.com/Pfzuo/BEES

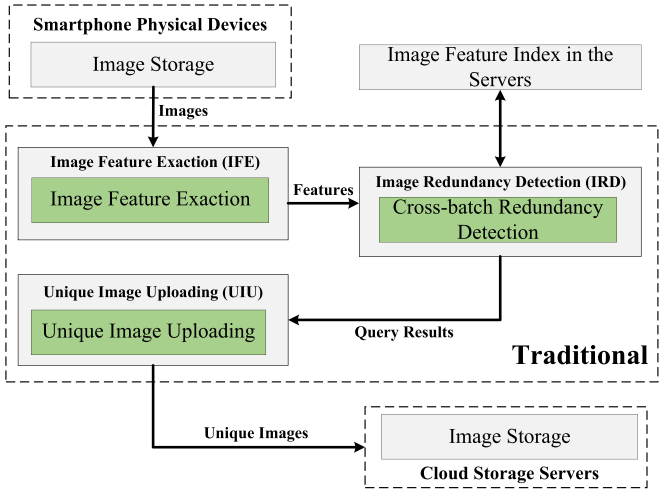


Fig. 2. The traditional system architecture.

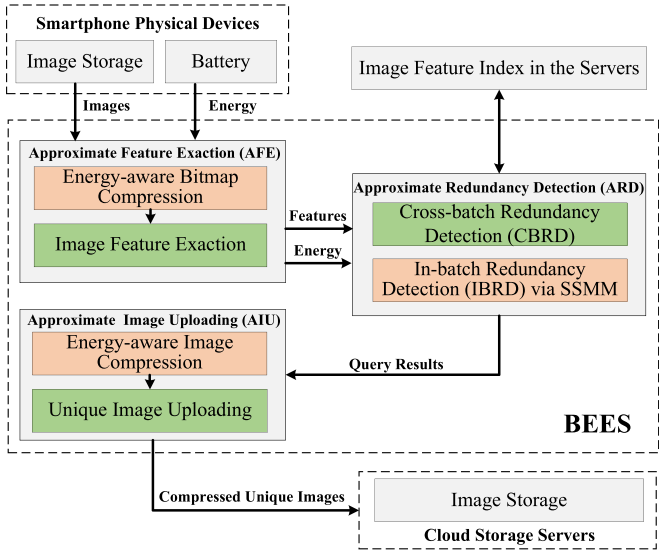


Fig. 3. The system architecture of BEES.

Extraction (IFE), Image Redundancy Detection (IRD) and Unique Image Uploading (UIU). IFE extracts the features (e.g., PCA-SIFT [22] and ORB [24]) of a batch of images in smartphones, and sends the image features to cloud servers. IRD queries the image features in the server index to determine whether there exist similar images in the cloud servers, and responds the query results to the smartphones. UIU only uploads the unique images in the image batch to the servers. The traditional framework is inefficient in terms of bandwidth and energy due to overlooking the several problems, i.e., failing to eliminate in-batch similar images and becoming inefficient when few similar images exist, as described in Section 1.

Therefore, we propose a new system architecture, i.e., BEES, for bandwidth- and energy-efficient image sharing, as shown in Fig. 3. BEES introduces three new key techniques. First, BEES proposes SSMM to detect in-batch similar images. Second, BEES proposes the concept of Approximate Image Sharing (AIS) to obtain bandwidth and energy savings via trading the quality of computation results. Specifically, in AFE, since extracting features consumes substantial energy, BEES explores and exploits bitmap compression to trade the feature computation quality for higher energy efficiency. In ARD, BEES adjusts the similarity detection threshold to select the images with the lower redundancy for saving bandwidth and energy. In AIU, BEES leverages image quality and resolution compression to trade image quality for higher bandwidth and energy efficiency. Third, it is a challenge to objectively and quantitatively handle the tradeoffs between the quality of computation results and efficiency. To address the challenge, three energy-aware adaptive schemes (EAAS) are respectively proposed in the three components (i.e., AFE, ARD and AIU) to objectively and quantitatively adjust the tradeoffs between the quality of computation results and efficiency by using the remaining energy of smartphones (E_{bat}). When the E_{bat} is sufficient, EAAS provides high-quality computation results; when the E_{bat} is insufficient, EAAS aims to save energy by slightly reducing the quality of computation results. Fourth, to reduce the response time of querying the server index, we propose a geographic coordinate based index partitioning (GIP) scheme to partition the entire feature index into grids. Thus a query request only needs to query the corresponding several grids, significantly reducing the search range.

3.2 The Workflow in BEES

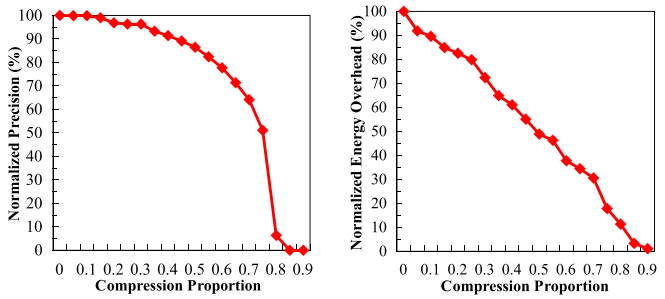
The workflow in BEES consists of the procedures in both smartphone clients and the servers as shown in Fig. 3. Before uploading a batch of images, a client first obtains the remaining energy of smartphone battery (E_{bat}) and then compresses the resolution of images based on E_{bat}, i.e., energy-aware adaptive image compression, as described in Section 4.1. The client further extracts the ORB features of the compressed images, and the geographic coordinate information. The image features, the parameter of E_{bat} and geographic coordinates are uploaded to the servers.

When receiving the data, the server queries the features of each image in the server index to obtain the maximum similarity of each queried image with the images in the server. The server then determines the similarity threshold T based on the parameter E_{bat}. The images, whose maximum similarities are larger than the threshold T, are considered as the redundant images and will not be uploaded according to the principle of Energy Defined Redundancy (EDR) described in Section 4.2.1. The remaining image batch, in which the maximum similarities of all images are smaller than T, are performed with submodular optimization and the greedy algorithm to eliminate the in-batch similar images, which is In-batch Redundancy Detection (IBRD) described in Section 4.2.2. The selected subset of images by IBRD is the images which will be finally uploaded. The server responds the redundancy detection result of each image to the clients. If the images are not redundant, the client compresses and then uploads them, which is described in Section 4.3. Otherwise, they will not be uploaded. After receiving the images, the server finally decompresses and stores the images.

4 The Design Details

4.1 Approximate Feature Extraction

To detect image similarity, image features are first extracted and uploaded to the servers. Even through being more energy-efficient than other feature extraction algorithms as described in Section 2, ORB still incurs much energy consumption [16]. It is necessary to reduce the energy overhead of feature extraction for energy-constraint smartphones. We observe



(a) Normalized precision (b) Normalized energy overhead

Fig. 4. The impact of bitmap compression proportion on precision and energy overhead (We also use 400, 800 images to examine precision and energy overhead when obtaining near-identical results).

that compressing the in-memory bitmaps of images before extracting their features can significantly decrease the computation (energy) overhead. However, computing the features from compressed image bitmaps also decreases the quality of image features. The low-quality features cause the low precision of similarity detection. Therefore, we need to obtain a suitable tradeoff between the energy overhead and the precision of similarity detection for image bitmap compression.

We first explore the relationships among the bitmap compression proportion, the precision of similarity detection, and the energy overhead of extracting features, via extensive experiments. The bitmap compression proportion is defined as the ratio of the decrement in the length or width of the compressed image bitmap to those of the original bitmap. We use a well-known public imageset (University of Kentucky imageset [29]) which contains 10,200 images in groups of 4 images from one scene. The 4 images in the same group are similar to each other in the imageset. We select one image from each group and 200 images in total as the queried images. The average number of the similar images in top-4 query results is used to measure the query precision (defined in Equation (3)) [16], [29]. We compress the bitmaps of queried images with the proportions from 0 to 0.9 with the interval of 0.05, and then extract their ORB features for similarity detection. In the meantime, we capture the energy overhead of extracting features.

Precision versus Compression Proportion. We normalize the query precision of compressed image bitmaps to that of original images as shown in Fig. 4a. With the increase of the compression proportion, the normalized precision decreases. We also observe that we can still ensure a high precision when significantly compressing image bitmaps. For example, when the compression proportion is 0.4, the normalized precision is higher than 0.9.

Energy Overhead versus Compression Proportion: We normalize the energy overhead of compressed images to that of original images, as shown in Fig. 4b. With the increase of the compression proportion, the energy overhead of extracting the ORB features from compressed image bitmaps decreases. We observe that there is an approximate linear relationship between the compression proportion and energy overhead.

Motivated by the observations, in order to obtain a suitable tradeoff between the energy overhead and detection precision, we present an energy-aware adaptive compression (EAC) scheme to dynamically adjust the bitmap compression proportion according to E_{bat}. When the energy is sufficient, EAC provides high detection precision; when the energy is insufficient, EAC is designed to save energy with a slight loss in detection precision.

In general, less than 10 percent errors for approximate computing processes are considered to be acceptable [30], [31]. In the EAC scheme, we design the relationship between the compression proportion (C) and the remaining energy (E_{bat}) as a linear function. Specifically, in order to ensure the compromising precision smaller than 10 percent, we set the function as $C = 0.4 - 0.4E_{bat}$ based on the statistic analysis of the practical measured data. The function can ensure a high precision while significantly saving energy in the case of low energy. For example, when E_{bat} is 5 percent, C is set to 0.38 according to the function, which can save about 40 percent energy of extracting features while ensuring higher than 90 percent precision, as shown in Fig. 4.

4.2 Approximate Redundancy Detection

For a given image batch in a smartphone, redundant image detection in BEES includes two parts, i.e., cross-batch redundancy detection (CBRD) and in-batch redundancy detection (IBRD). CBRD detects the similarity between the images in the given image batch and the images in the servers which are previously uploaded by other batches/smartphones. CBRD eliminates cross-batch similar images by querying the server index. However, only querying the index cannot eliminate in-batch similar images since these images are not uploaded and hence their image features do not exist in the index. Therefore, we propose IBRD to detect the redundancy among the images in a batch.

4.2.1 *Cross-Batch Redundancy Detection*

In the context of this paper, a redundant image is determined by the maximum similarity which is defined as the similarity between the queried image and its most similar image (i.e., the image that has the highest similarity score) in the servers. If the maximum similarity is more than a similarity threshold T, the queried image is considered to be redundant and will not be uploaded. Otherwise, the image is unique. Note that the similarity score is defined in Euqation (1).

A reasonable similarity threshold relies on the subjective viewpoints from users and the objective similarity scores computed by using ORB features. In order to explore the similarity property of similar and dissimilar image pairs, we respectively select 5,000-pair similar and dissimilar images from the Kentucky imageset (described in Section 5.1). In the imageset, if two images of an image pair belong to the same group, the image pair is considered as a similar image pair. Otherwise, it is a dissimilar image pair. We extract the features of these images and compute their similarity (defined in Equation (1)). The similarity distributions of similar and dissimilar image pairs are shown in Fig. 5.

Fig. 5 shows the true positive rates (similar images are accurately detected) and false positive rates (dissimilar images are detected to be similar) for similarity detection given a similarity threshold. For example, the similarity of 95.4 percent similar images is larger than 0.01 and the similarity of 26.2 percent dissimilar images is larger than 0.01. Thus, if the similarity threshold is set to 0.01, the true positive rate is 95.4 percent and the false positive rate is 26.2 percent. We can also observe that both true and false

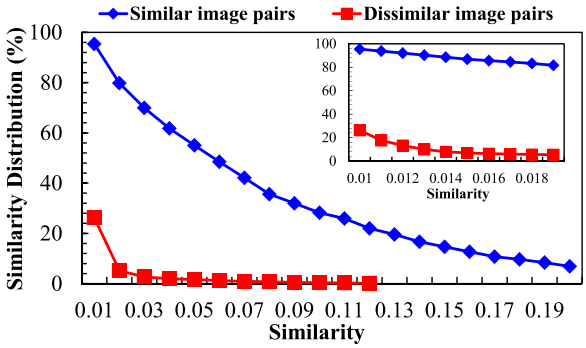


Fig. 5. The similarity distribution.

positive rates decrease with the increase of the threshold. Hence, a reasonable threshold is a tradeoff between the high true positive and the low false positive rates.

In BEES, the similarity threshold T is not fixed. We propose the Energy Defined Redundancy (EDR) which uses the remaining energy (E_{bat}) to dynamically adjust T to determine whether an image is redundant. EDR aims to eliminate the higher-similarity images when the energy is sufficient, and eliminate more images by reducing T when the energy is insufficient. Based on the experimental results shown in Fig. 5, we argue that the similarity threshold should be larger than 0.013, which leads to 90 percent true positive rate and 10 percent false positive rate. In order to ensure the false positive ratio not larger than 10 percent [30], [31], EDR defines the relationship between the threshold T and E_{bat} as $T = 0.013 + k * E_{bat}$ $(k = 0.006)$.

EDR is more important when the smartphones are in low battery status and fail to upload all images. Smartphones only need to upload the images which have low/no similarity with the images in the servers using the limited energy, since EDR reduces T in low battery status.

4.2.2 In-Batch Redundancy Detection

In a batch of images, there may exist multiple images similar to each other. The key problem is how to select the retained unique images in the uploaded batch. A simple solution is to enumerate all image subsets, and then sort them based on distance-based metric and select the top one, which unfortunately results in high computation and time overheads. In order to address this problem, we propose a similarity-aware submodular maximization model (SSMM).

We first formulate the problem and define an image batch as a weighted graph $G = (V, E, w)$. V is the set of images. E is the set of edges that connect two images in set V. Each edge $(i, j) \in E$ has a non-negative weight $w_{i,j}$ that is scored by the similarity between images i and j. Given a batch of images $V = \{v_1, v_2, \ldots, v_n\}$, we aim to find a subset $S \subseteq V$, which best summarizes the batch and represents the images using the smallest number. We leverage a scoring function ($F : 2^V \rightarrow \mathrm{R}$) to quantitatively represent the quality of a summary. The image subset can be computed:

$$S^* \in \arg\max_{S \subseteq V} F(S) \quad s.t. \quad |S| \leq b. \tag{2}$$

Given a constraint $|S| \leq b$, Equation (2) can be modeled as the form of submodular maximization subject to knapsack constraints which is NP-complete [32]. Moreover, if the function F is the monotone submodular function, a simple greedy algorithm can efficiently and near-optimally address Equation (2) with the worst-case guarantee of $F(\hat{S}) \geq (1 - 1/e)F(S_{op}) \approx 0.632F(S_{op})$, where S_{op} is the optimal subset and $\hat{S}$ is the subset obtained by a greedy algorithm [32], [33].

Equation (2) needs a constraint, and otherwise, it is NP-hard. In the constraint $|S| \leq b$, $|S|$ is the number of images in S and b is the budget. In the existing work [33], [34], the budget is fixed and assigned by users. For example, a user wants to select the maximum of 9 images to post on Facebook from an image collection taken in a holiday, and thus the budget b is 9. However, the fixed budget is inefficient in our application situation, since the budget should be the number of non-redundant images which is different from batch to batch.

Hence, we propose the SSMM to adaptively determine the budget b based on the similarities among the images in V, which aims to achieve that the higher the similarities among the images in V are, the lower the budget b is. In the weighted graph $G = \{V, E, w\}$, SSMM cuts the edges of which the weight w is smaller than a threshold T_w. Thus the graph G is partitioned into multiple subgraphs. There are higher similarities among the images within the subgraph. SSMM takes the number of the partitioned subgraphs as the budget b. Thus the higher the similarities among the images in V are, the smaller the number of the partitioned subgraphs is, which results the lower budget. Moreover, the number of the partitioned subgraphs, i.e., the budget b, not only depends on the similarities among the images in V but also the threshold T_w. It is obvious that the larger the threshold T_w is, the more the partitioned subgraphs is, the larger the budget b is. We also dynamically adjust the threshold T_w based on E_{bat}. Specifically, we design $T_w = 0.013 + k * E_{bat}$ $(k = 0.006)$ referring to the parameters in EDR.

In the following, we first present the submodular [18], and then describe submodular component functions and the simliarity-aware greedy algorithm used in SSMM.

Definition 1 (Submodular). *Given a finite set V, a function $f : 2^V \rightarrow \mathrm{R}$ is submodular if for any set $A \subseteq B \subset V$, and any element $v \in V \setminus B$, f satisfies: $f(A \bigcup \{v\}) - f(A) \geq f(B \bigcup \{v\}) - f(B)$.*

This means that the benefit of adding v to set A is more than the benefit of adding v to a larger set $B \supseteq A$.

Submodular Component Functions. If a series of functions $f_i (i = 1, 2, \ldots, m)$ are submodular, their weighted sum $\sum_{i=1}^{m} \lambda_i f_i$ is also submodular where λ_i is non-negative. In BEES, we design $F(S)$ as the weighted sum of multiple submodular component functions, i.e., $F(S) = \sum_{i=1}^{m} \lambda_i f_i(S)$, $\lambda_i \geq 0$. Nevertheless, good image batch summaries can be characterized by two general properties, i.e, coverage and diversity [33], [34]. Thus we build the coverage and diversity component functions.

Coverage Function. A summary with good coverage allows all distinct contents in the batch to have their corresponding representatives in the summary. The summary coverage can be quantified by the sum of the similarity between image i in V and the most similar image j in S which is formulated into $f_{cov} = \sum_{i \in V} \max_{j \in S} w_{i,j}$.

Diversity Function. A summary with good diversity does not contain multiple images that are similar to each other. As mentioned above, we use SSMM to partition the graph

G into b subgraphs: $g_1, g_2, \ldots, g_b$. I_i is the set of images in the subgraph g_i. A better summary covers more subgraphs and contains fewer images in the same subgraph. We define the diversity function $f_{div} = \sum_{i=1}^{b} N(S, I_i)$. If S and I_i share no element, $N(S, I_i) = 0$. Otherwise, $N(S, I_i) = 1$.

Similarity-Aware Greedy Algorithm. We show how to determine the submodular function $F(S)$ and the budget b above. We can generate the unique image subset using the greedy algorithm as shown in Algorithm 1.

Algorithm 1. The Similarity-Aware Greedy Algorithm

Input: Submodular function: $F()$, The weighted graph $G = \{V, E, w\}$, the remaining energy E_{bat}.
Output: S_k where k is the number of iterations.

```
Determine the threshold T_w based on E_bat;
Partition graph G using T_w;
Get the number of partitioned subgraphs b;
Choose v_1 arbitrarily;
S_1 ← v_1;
while |S_i| ≤ b do
    Choose v_i ∈ arg max_{v_i ∈ V\S_i} F(S_i ⋃ {v_i});
    S_{i+1} ← S_i ⋃ {v_i};
    i ← i + 1;
end while
```

4.3 Approximate Image Uploading

By carrying out the redundancy detection, redundant images are eliminated and the unique images need to be uploaded. Nevertheless, the images taken by smartphones are typically large-size. The average size of high-quality images taken by modern smartphones can be more than 2 MB [16]. Uploading the large-size images consumes too much bandwidth and energy. We argue that the high resolution and quality of images are not necessary for such disaster environments due to the constrained energy and real-time transmission requirements [30]. We hence explore how to compress the images to reduce their file size before uploading. There are two kinds of image compression methods, as described in the following.

Quality Compression. Quality compression uses mathematical operations to convert pixels of an image from the spatial domain into the frequency domain for reducing the required storage space of an image, which does not change the resolution of an image. Our experiments explore the relationship between the file size and compression proportion in quality compression. The compression proportion in quality compression denotes the compression degree on image quality. A larger compression proportion means compressing an image more aggressive. There are many quality image compression standards, such as, JPEG [35], PNG [36] and WebP [37]. We use JPEG as a concrete example due to its widespread use. JPEG is a lossy compression method. We argue that the slight loss in image quality is acceptable in such disaster environments [30]. In JPEG, the range of compression proportion is from 0 to 1. When the compression proportion is 0, the image is not compressed; when the compression proportion is 1, the image is compressed to the utmost extent.

Resolution Compression. Resolution compression aims to directly reduce the resolutions of images to reduce the file size. We also perform experiments to explore the relationship between the file size and resolution compression proportion. The resolution compression proportion is defined as the ratio of the decrement in the length or width of the resolution of the compressed image to those of the original resolution. For example, an image with the resolution $1000 * 500$ is compressed to $800 * 400$, where the compression proportion is 0.2.

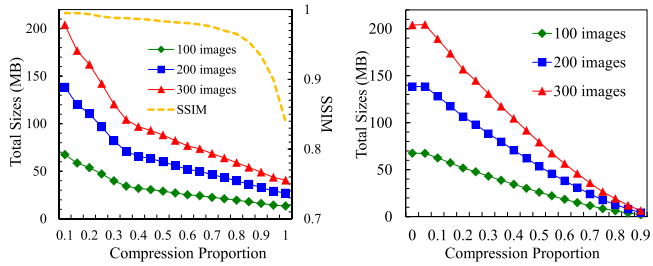


Fig. 6. The influences of quality compression and resolution compression to bandwidth overheads.

We respectively compress 100,200,300 images with different compression proportions using JPEG and resolution compressions, and then upload them from smartphones to the servers. Their bandwidth overheads are shown in Fig. 6. We use the SSIM (Structural SIMilarity) index [38], which is a well-known method for image quality assessment, to evaluate the influence of different quality compression proportions on image quality as shown in Fig. 6a. We observe that quality compression can significantly reduce bandwidth overheads while also causes the decrease of image quality. Due to causing the slight loss in image quality, we suggest to compress the image quality with a fixed compression proportion, i.e., 0.85. If the compression proportion is larger than 0.85, the image quality will be significantly decreased, as shown in Fig. 6a.

As shown in Fig. 6b, resolution compression can also obtain significant bandwidth savings. The reduced resolutions are unrecoverable. Therefore, it is a tradeoff between the bandwidth overhead and the image resolution. In order to obtain a suitable tradeoff, we propose an energy-aware adaptive uploading (EAU) scheme to adaptively adjust the resolution compression proportion based on E_{bat}. When the energy is sufficient, BEES aims to upload higher-resolution images; Otherwise, to save energy, BEES uploads the lower-resolution images, which also results in more images to be uploaded regardless of the low resolution. In the EAU scheme, we describe the relationship between the resolution compression proportion (C_r) and E_{bat} as a linear function. Specifically, to ensure a relatively high resolution even in the case of low energy, we design the function as $C_r = 0.8 - 0.8 * E_{bat}$. For example, when E_{bat} is 5 percent, C_r is 0.76. For a smartphone with 8 million-pixels camera taking $2448 * 3264px$ photos, the resolutions of the compressed photos are still $588 * 783px$ while reducing about 87 percent file size compared with $E_{bat} = 100$ percent.

4.4 Geographic Coordinate Based Index Partition

A cloud server can receive a large number of images uploaded by volunteers in disasters. To support the similar image detection, the cloud server needs to index the features of these images. Nevertheless, with the continuous increase of the number of received images, the size of the

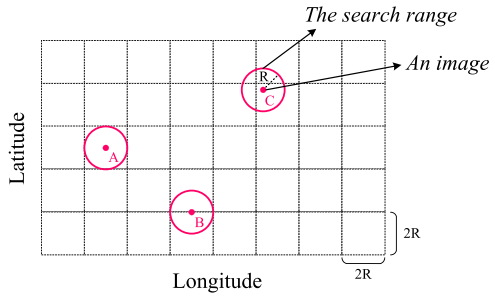


Fig. 7. The geographic coordinate based index partition. *(The red points are the geographic coordinates of the queried images, and the red circles are the corresponding search ranges.)*

server index significantly increases, thus increasing the response time of querying similar images.

To improve the query performance of the server index, we propose a geographic coordinate based index partitioning (GIP) scheme. Specifically:

1) In the server side, when building the feature index, we first partition the map of the disaster area into the grids with the same size based on the longitude and latitude, as shown in Fig. 7. Thus each grid has a range in longitude and latitude, e.g., a grid covers the area from 2.311 to 2.312 degrees longitude and from 48.855 to 48.856 degrees latitude. The server then stores the image features of each image into its corresponding grid based on the longitude and latitude of the image. In each grid, the server respectively creates an independent feature index by using any existing approximate query index techniques, e.g., locality sensitive hashing (LSH) [39]. Thus the full feature index is partitioned into many independent small indexes each in a grid.
2) In the smartphone side, when taking pictures, the current geographic coordinates of images, i.e, latitudes and longitudes, are usually recorded in the EXIF headers of generated JPEG photographs [5]. We can simply extract the latitude and longitude data from the headers of images. When uploading the image features of an image for redundancy detection, we also upload the geographic coordinate of the image. After receiving the image features along with the geographic coordinate, the server only needs to query several grids instead of the full index. The queried grids are determined by the geographic coordinate of the queried image and a distance R.

The distance R is called effective distance, which means two images cannot be similar if the actual distance of their geographic coordinates is larger than R [40]. In practice, for any two photographers, if the actual distance between two photographers is larger than R, they cannot take pictures for the same objects. Therefore, to search the similar images of an image, we only need to query the grids that are covered by a circle of which center is the geographic coordinate of the image and radius is R, without affecting the query precision. To achieve the goals of searching at most four grids for any queried images and minimizing the area of each grid, it is easy to obtain that the length and width of each grid should be $2R$. For example, as shown in Fig. 7, if a query image locates at the center of a grid (e.g., the point A), we only need to search one grid; if a query image locates at the midpoint of a boundary (e.g., the point B), we need to search the two adjacent grids; if a query image locates other points (e.g., the point C), we need to search four grids.

The GIP scheme is able to improve the query performance in terms of the following points:

- *Reducing the search range.* In the traditional full index schemes [4], [16], the sever needs to query similar images in the full feature set. By using the proposed GIP scheme, we only need to search at most four partitioned grids, thus reducing the query time.
- *Reducing the false positive.* A real case possibly exists that an object is visually similar to another object but they are in two different geographic locations. Taking pictures for the two objects in different geographic locations produces the false similar images which indicate two or multiple images are considered to be similar in similarity detection but do not really contain the same objects. By using the proposed GIP scheme, we can isolate the similar objects in different geographic coordinates to avoid producing false similar images.
- *No affecting the query precision.* Even though the search range is reduced in the proposed GIP scheme, the query precision is not reduced. the reason is that the similar images of the queried image must be in the search range.

5 Performance Evaluation

5.1 Experimental Setup

The BEES prototype consists of two parts, i.e., the client application and cloud server. The client application is programmed in Java and native C++ (JNI) with about 1,000 source lines of code (SLoC), and linked with the openCV library [41] for feature extraction. We reduce the size of the client APP, which is only 593 KB in the latest version. The size of the APP is smaller than that of an image, and users can download it with very low bandwidth overhead in disasters. We install the client APP into the Android-based smartphones for evaluation. The smartphone is equipped with Helio X10 8-core CPU at 2.2 GHz, a 32 GB ROM and a 3 GB RAM, in which the battery capacity is 3150 mAh with a voltage of 3.8 Volts. The server is programmed in C++ with about 1,600 SLoC, and also uses openCV library to extract image features. The server is implemented in the Ubuntu 14.04 operating system running on a 16-core CPU each at 3.40 GHz, with a 32 GB RAM and a 2 TB hard disk.

Network bandwidth is very limited in capacity in the disaster environments. Existing schemes [2], [42], [43] limit their bandwidth to several hundred Kbps to simulate the low bandwidth. Hence, in our experiments, we connect smartphones into the WiFi of a computer, and setup the Charles tool [44] in the computer to control the network bandwidth that smartphones use. The transmission bandwidth of each smartphone fluctuates from 0Kbps to 512Kbps to emulate the low-bandwidth network. Moreover, we setup the PowerTutor tool [45] in smartphones to evaluate energy consumptions of applications.

Three real-world imagesets are used in our experimental evaluation.

- *The Kentucky imageset* [29]: The imageset contains 10,200 images in 2,550 groups. In each group, four images are taken from the same object or scene which can be considered to be similar to each other. Since the imageset is widely used for evaluating the precision of similarity detection [16], [29], we also use it to evaluate precision in Section 5.2.1.
- *The disaster imageset*: We use the Google and Bing image search services to collect 1,000 images taken in Nepal earthquake in 2015 [6]. The imageset is used for general tests in Sections 5.2.2, 5.2.3, and 5.2.4.
- *The Paris imageset* [46]: The imageset contains 501,356 geotagged images, which is collected from Flickr and Panoramio using a geographic bounding box around the inner city of Paris. The geographical positions of images in the Paris imageset have a real-world distribution. Due to the large number of images, the imageset is used for large-scale tests in Sections 5.2.2 and 5.2.5.

Note that for the following experiments, to simulate real conditions, all used images are resized to about 700KB which reflects the average size of normal-quality images taken by smartphones [16].

For evaluating the precision of similarity detection, we compare BEES with the state-of-the-art algorithms for similarity detection, i.e., SIFT and PCA-SIFT. For evaluating the energy overhead, delay and bandwidth overhead, we examine a baseline scheme, i.e., directly uploading images. We also compare BEES with the state-of-the-art schemes for image-based SA in disasters, i.e., SmartEye [4] and MRC [16]. Due to our lack of the source code of MRC, we implement the MRC based on the scheme described in its paper [16] for evaluation.

5.2 Results and Analysis

5.2.1 Precision of Similarity Detection

To evaluate the effectiveness of similarity detection, we use the measure of *precision* (also called positive predictive value), which is the fraction of retrieved instances that are relevant. In the image similarity detection, we can define *precision* as:

$$precision = \frac{|\{similar\ images\}| \bigcap |\{retrieved\ images\}|}{|\{retrieved\ images\}|}. \quad (3)$$

In the Kentucky imageset, it is easy to determine if a queried image is similar to existing ones. Thus we evaluate the precision using the Kentucky imageset that was widely used for evaluating the precision [16], [29]. We select one image from each group as the queried image. Without loss of generality, we respectively execute 500, 1000, and 1500 queries to compute the average precisions. As the baseline comparison, we also evaluate the precisions of SIFT and PCA-SIFT. Moreover, due to the energy-aware adaptive bitmap compression (presented in Section 4.1) in BEES, E_{bat} is also related with the precision. We also evaluate the precisions of BEES under the conditions of different E_{bat}. Precisions of all schemes are normalized to that of SIFT in Fig. 8.

As shown in Fig. 8, SIFT obtains the highest precision and the precision of SURF is close to that of SIFT. Compared with SIFT, the precision in BEES(100) is higher than 90.3 percent, which is close to PCA-SIFT. Moreover, we observe that the precision of BEES slightly decreases with the decrease of E_{bat}, since BEES improves the energy efficiency with a slight decrease in precision in the low battery status. BEES(10) obtains over 84.9 percent precision compared with SIFT. SIFT, SURF and PCA-SIFT obtain a little higher precision than BEES that uses ORB, which however are not suitable in the disaster environments to be uploaded and used to detect similarity due to their huge bandwidth and energy overheads as shown in Section 2.

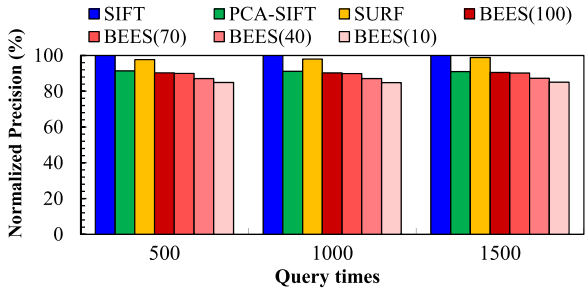


Fig. 8. The normalized precision (BEES(X) means the BEES under the condition of $X\%$ E_{bat}).

5.2.2 Energy Overhead

1) Energy Overhead. We investigate the impact of different schemes on energy overheads. SmartEye, MRC and BEES consume extra energy to compute and upload image features for similarity detection while saving energy by reducing redundant images to be transmitted, compared with Direct Upload. Thus different redundancy ratios of uploaded images produce different energy overheads. Therefore, we capture the energy overheads when the uploaded images are at different redundancy ratios. The redundancy ratio is defined as the ratio of the number of redundant images in the uploaded images to the total number of uploaded images.

We select an image batch with 100 images from the disaster imageset as the uploaded images and store the images in the smartphone. We set different cross-batch redundancy ratios 0, 25, 50, and 75 percent, by adding and removing the redundant images (similar to the uploaded images) into the servers. Note that the redundant images added in the servers have the high similarity (i.e., more than 0.3 computed by Equation (1)) with the uploaded images, which ensures all redundant images can be detected by three different schemes for fair comparisons. Moreover, 10 in-batch similar images exist in the 100 images and do not have similar images in the servers, thus clearly showing the benefit of in-batch redundancy elimination in BEES. We respectively upload the 100 images using the four schemes and capture their energy overheads.

As shown in Fig. 9, the higher the cross-batch redundancy ratio is, the less energy SmartEye, MRC and BEES consume due to eliminating redundant images. The energy overhead of SmartEye is more than that of MRC, since SmartEye extracts image features using PCA-SIFT that consumes more energy than MRC to extract ORB features. BEES produces much lower energy overhead than SmartEye and MRC, since in-batch redundancy reduction and using approximate image sharing in BEES decrease the amounts of the uploaded data, thus obtaining significant

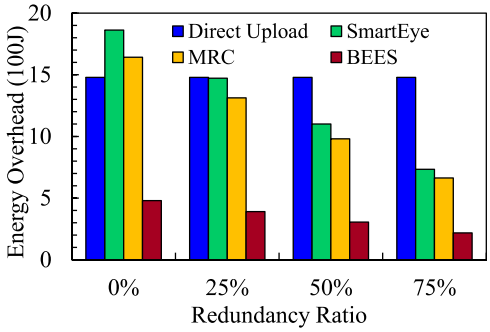


Fig. 9. Energy overhead.

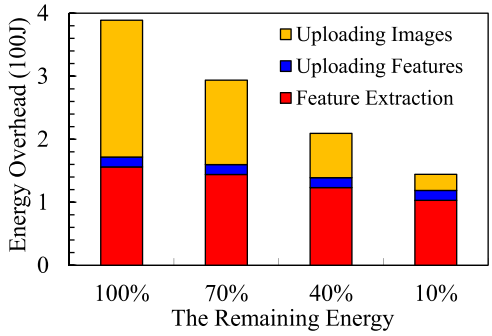


Fig. 10. Energy-aware adaptation.

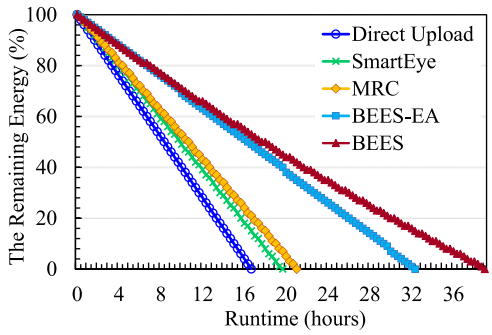


Fig. 11. Battery lifetime.

energy savings. Compared with MRC, BEES reduces 67.3–70.8 percent energy overheads. Compared with Direct Upload, BEES reduces 67.6–85.3 percent energy overheads. Even in the worst case with no cross-batch redundancy, BEES also obtains 67.6 percent energy saving while SmartEye and MRC consume more energy than Direct Upload.

2) Energy Savings from Energy-aware Adaptation. Energy-aware adaptation aims to save energy and extend the battery lifetime when smartphones are in the low battery status. To verify the energy benefits of energy-aware adaptive schemes in BEES, we examine the energy overheads when smartphones contain different amounts of remaining energy. We use the same 100 image collection (used in Section 5.2.2(1)) with 10 in-batch similar images. We set the same cross-batch redundancy ratio to be 25 percent for each uploading. When the remaining energy of the smartphone is 100, 70, 40, and 10 percent, we respectively upload the 100 images using BEES and examine their energy overheads of feature extraction, uploading features and images.

As shown in Fig. 10, the total energy overhead, the energy overheads of feature extraction and image uploading decrease with the decrease of E_{bat}, due to energy-aware adaptation. The energy overhead of uploading features is small, due to the lightweight ORB features.

3) Battery Lifetime. We investigate the impact of different schemes on the battery lifetime of smartphones. Moreover, in order to demonstrate the efficiency of energy-aware adaptive schemes in BEES, we also examine the battery lifetime using BEES-EA. BEES-EA represents BEES without energy-aware adaptive schemes in which BEES does not adjust its behaviors based on the remaining energy. For keeping the same conditions in each scheme, the initial energy of battery is full. During uploading images in each scheme, all applications in the smartphone, except BEES App and the system-related programs, are always closed and the screen is always bright. We select 150-group images from the Paris imageset, and store them in the smartphone in advance. Each group contains 40 images. We set the cross-batch redundancy ratio of each group to about 50 percent by adjusting the server index. There are almost no in-batch similar images in each group. We upload one group every 20 minutes, until the smartphone battery is exhausted. We record the remaining energy of battery every 20 minutes. Thus the battery lifetime of a smartphone is evaluated by computing the time from the smartphone starts up to its battery is exhausted.

As shown in Fig. 11, with the increase of the runtime of the smartphone, the remaining energy linearly/near-linearly decreases in Direct Upload, SmartEye, MRC, and BEES-EA. The relationship of the runtime and E_{bat} in BEES is a curve instead of a straight line. The slope of the relationship curve in BEES slowly decreases with the decrease of E_{bat}, since energy-aware adaptive schemes in BEES adaptively adjusts the behaviors based on E_{bat} to save energy and slow down the speed of energy consumption. SmartEye, MRC, BEES-EA, and BEES, respectively extend 18.0, 25.7, 93.4, and 133.1 percent battery lifetime, compared with Direct Upload. Compared with MRC, BEES extends 84.3 percent battery lifetime. Compared with BEES-EA, BEES extends 19.8 percent battery lifetime due to energy-aware adaptive schemes.

5.2.3 Bandwidth Overhead

When examining the energy overheads of the four schemes in Section 5.2.2(1), we record the bandwidth overhead of each scheme. As shown in Fig. 12, the higher the cross-batch redundancy ratio is, the lower bandwidth overheads SmartEye, MRC, and BEES consume. MRC consumes a little more bandwidth overhead than SmartEye due to requiring thumbnail feedback. BEES is superior to both SmartEye and MRC due to not only further reducing in-batch redundancy but also leveraging approximate image sharing, thus reducing much more bandwidth overheads. Compared with SmartEye, BEES reduces 77.4–79.2 percent bandwidth overheads.

5.2.4 Delay

We compare the delays of different schemes in this subsection. We use the same image collection with 100 images used in Section 5.2.2(1). There are 10 in-batch similar images in the 100 images. We set the same cross-batch redundancy ratio (50 percent) for each scheme. The network bitrate, under which the smartphone communicates with the servers, affects the uploading delay. Thus we also capture the delay under different network bitrates with the medians

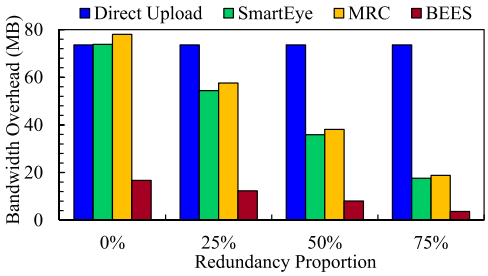


Fig. 12. Network bandwidth overhead.

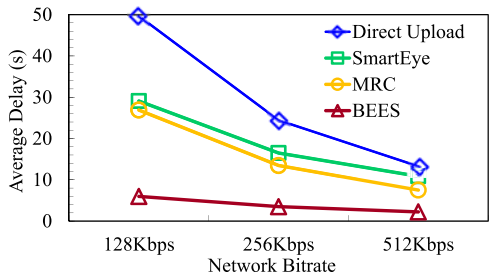


Fig. 13. The average delay of uploading an image.

128 Kbps and 512 Kbps, besides 256 Kbps. The delay consists of the time of extracting image features, uploading features and images.

As shown in Fig. 13, we observe that Direct Upload produces the highest delay, and SmartEye, MRC, and BEES reduce the delay in different degrees via reducing redundancy. The average delay of SmartEye is more than that of MRC, since SmartEye extracts image features using PCA-SIFT which consumes more time than MRC using ORB. BEES reduces the image uploading time by further reducing in-batch redundant images and reduces the feature extraction time by using energy-aware feature extraction, and also obtains much more time saving by using energy-aware image compression before uploading images, thus being superior to SmartEye and MRC. As shown in Fig. 13, BEES reduces 83.3–88.0 percent average delay compared with Direct Upload, and reduces 70.4–77.8 percent average delay compared with MRC. In general, the extremely low delay of BEES meets the needs of disaster environments in terms of real-time transmission.

5.2.5 Coverage

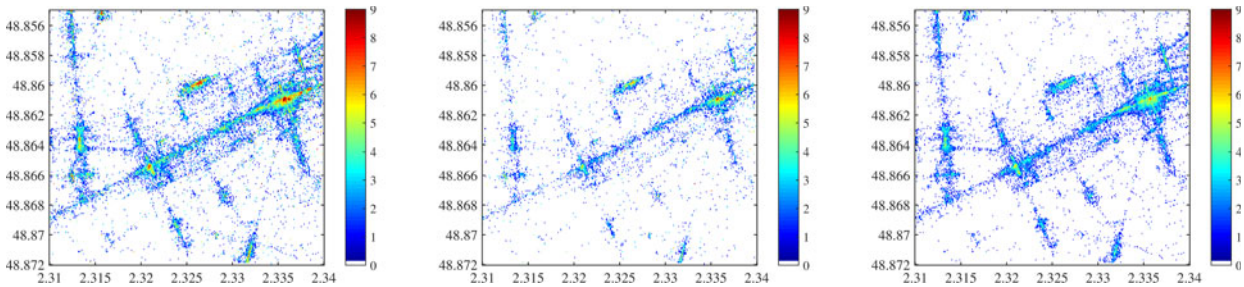
When a disaster occurs, the images uploaded by smartphones are used for SA. However, the energy of smartphone battery is limited. It is important for the energy-constrained smartphones to use the limited energy to collect more information. We use the region area of the situation awareness (i.e., the coverage of uploaded images) to quantify the amount of information obtained by the uploaded images, and evaluate the coverage of BEES.

We use the Paris imageset to evaluate the coverage, since each image in the imageset is geotagged to facilitate its mapping in the real map. Since the complete set of the Paris imageset is too large, we select a subset as the test imageset covering the area from 2.31 to 2.34 degrees east longitude and from 48.855 to 48.872 degrees north latitude. The test imageset consists of 165,539 images which have 58,818 unique locations (i.e., longitudes and latitudes) in the map. The densest location has 5,399 images. The real distribution is shown in Fig. 14a. We equally divide the 165,539 images into 25 groups and respectively store them in 25 smartphones. The initial energy of all 25 smartphone batteries is full. 40 images are considered as a group in the smartphones. The 25 smartphones respectively upload an image group every 20 minutes. The servers add the features of the uploaded images into the index for redundancy detection once receiving the images from BEES clients. After the batteries of all the smartphones are exhausted, we map all the images that the servers receive in the map based on their geotags.

Using Direct Upload, the smartphones upload 49,437 images in total. The uploaded images have 23,399 unique locations in the map. Fig. 14b shows the coverage of the images uploaded using Direct Upload in the map. In BEES, the smartphones upload 58,750 images which have 46,122 unique locations in the map. Fig. 14c shows the coverage of the images uploaded using BEES in the map. BEES uploads 18.8 percent more images while has 97.1 percent larger coverage (i.e., the number of unique locations covered) than Direct Upload, since BEES reduces the redundant images and uses submodular maximum model to efficiently summarize the uploaded image batch.

5.2.6 The Query Performance of Feature Indexes

To show the benefits of our proposed geographic coordinate based index partitioning (GIP) scheme, we compare the query performance of the feature indexes with and without our proposed GIP scheme. For the feature index without the GIP scheme, we build an index for the full feature set using DLSH [47], a state-of-the-art LSH scheme. For the feature index with the GIP scheme, we first partition the full feature set into grids using the GIP scheme and then build the index in each grid using DLSH. We use the same subset of the Paris imageset as that used in Section 5.2.5. Since the



(a) The test imageset (b) Direct Upload (c) BEES

Fig. 14. Coverage. (The x axis is east longitude and the y axis is northern latitude. The values beside the color table are the index of 2. For instance, the locations colored by the color corresponding to 6 have $2^6(64)$ images.)

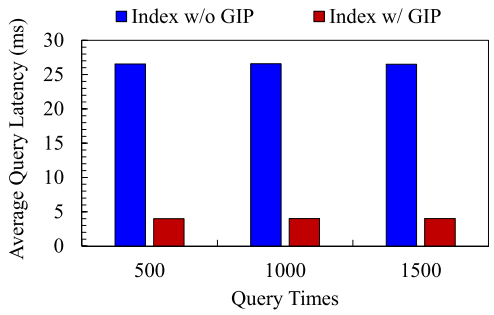


Fig. 15. The average query time in the indexes with and without our proposed GIP scheme.

maximum distance between two similar images in this imageset is not larger than 200 meters, we set the effective distance R to 200 meters. Thus the evaluated image set is partitioned into $8 * 16$ grids. We respectively do 500, 1000, and 1500 queries to the two indexes, and evaluate the average query latency. For a queried image, we aim to find its most similar image that is used for the cross-batch redundancy detection as presented in Section 4.2.1, via querying the feature indexes. The experimental results are shown in Fig. 15.

By comparing the average query latency in the indexes with and without our proposed GIP scheme, we observe that using the GIP scheme reduces the query latency by 85 percent on average, i.e., speeding up the query by $6.6\times$. The reason is that the GIP scheme efficiently reduces the search range of each query and the search range (four grids) using the GIP scheme is about $1/32(= 8 * 16/4)$ of the full set. Moreover, the GIP scheme also reduces the false similar images as presented in Section 4.4, which produces fewer query results compared with querying the full set via DLSH. In the set with fewer query results, computing the most similar image consumes less time.

6 Related Work

Data Deduplication. Data deduplication is widely used in data storage systems, such as backup and archive storage [48], [49], [50], primary storage [51], [52] and cloud storage [53], [54], [55], to save storage space via eliminating exact-matching redundancy. Deduplication is also used in network environments [56], [57] to save network bandwidth. Unfortunately, deduplication is inefficient to detect similar images, since deduplication detects redundancy in the byte level while images are similar in the content level. A small difference in the content may cause significantly different byte-level encoding [3]. BEES shares the similar design goals but at the content level and focuses on identifying redundant images.

Content-Based Redundancy Elimination in Disaster Environments. Several schemes have been proposed to eliminate the image redundancy in disaster environments, which can be divided into two categories. One focuses on eliminating redundant images in delay tolerant networks (DTNs). PhotoNet [2] presents a content-based redundancy elimination routing scheme that uses image metadata, i.e., geotags and color histograms of images, to approximately evaluate and eliminate similar images in DTNs. Wu et al. [15] propose a resource-aware framework using image metadata to eliminate redundant images in the DTNs. CARE [3] uses image features to perform more accurate similarity detection than PhotoNet in DTNs. The other aims to eliminate redundant images in the source (i.e., smartphones) by uploading image features, which avoids redundant images passing into the bandwidth-constrained networks. SmartEye [4] proposes in-network deduplication based on software defined network (SDN) to eliminate redundant images in the source. MRC [16] proposes a framework combining global image features and local image features to detect and eliminate redundant images in the source. Both SmartEye and MRC detect similar images by querying the server index that can only eliminate the cross-batch redundancy. Besides eliminating the cross-batch redundancy, BEES builds the submodular maximum model to eliminate the in-batch redundant images. More importantly, unlike all existing work, BEES is a complete system which proposes the approximate image sharing and energy-aware adaptation to obtain higher bandwidth and energy efficiency.

7 Conclusion

In this paper, we propose a bandwidth- and energy- efficient image sharing system, called BEES, for real-time SA in disasters. BEES reduces not only the cross-batch redundant images but also in-batch redundant images in the source, and further leverages approximate image sharing to trade the the quality of computation results in content-based redundancy elimination for higher bandwidth and energy efficiency. Moreover, the energy-aware adaptive schemes are introduced in BEES to offer an objective and quantitative tradeoff between computation quality and efficiency based on the remaining energy. Extensive experimental results demonstrate that BEES reduces more than 67.3 percent energy overhead, 77.4 percent bandwidth overhead, 70.4 percent average image uploading delay, and extends 84.3 percent battery lifetime, compared with the state-of-the-art work.

Acknowledgements

This work is supported by National Key Research and Development Program of China under Grant 2016YFB1000202, and National Natural Science Foundation of China (NSFC) under Grant No. 61772212 and 61502190. The preliminary version appears in the Proceedings of the 37th International Conference on Distributed Computing Systems (ICDCS), 2017, pages: 1510–1520.

References

[1] Big Data and Disaster Management: A Report from the JST/NSF Joint Workshop, 2015. [Online]. Available: https://goo.gl/0gath8
[2] M. Y. S. Uddin, H. Wang, F. Saremi, G.-J. Qi, T. Abdelzaher, and T. Huang, "PhotoNet: A similarity-aware picture delivery service for situation awareness," in *Proc. IEEE 32nd Real-Time Syst. Symp.*, 2011, 317–326.
[3] U. Weinsberg, Q. Li, N. Taft, A. Balachandran, V. Sekar, G. Iannaccone, and S. Seshan, "CARE: Content aware redundancy elimination for challenged networks," in *Proc. ACM Workshop Hot Topics Netw. (HotNets)*, 2012, pp. 1–6.
[4] Y. Hua, W. He, X. Liu, and D. Feng, "SmartEye: Real-time and efficient cloud image sharing for disaster environments," in *Proc. IEEE Conf. Comput. Commun.*, 2015, pp. 1616–1624.
[5] K. Toyama, R. Logan, and A. Roseway, "Geographic location tags on digital images," in *Proc. 11th ACM Int. Conf. Multimedia*, 2003, pp. 156–166.
[6] How Social Media Is Helping Nepal Rebuild After Two Big Earthquakes, 2015. [Online]. Available: https://goo.gl/CbIMzT

[7] [New Scientist] Social Media Helps Aid Efforts after Typhoon Haiyan, [Online]. Available: https://goo.gl/yJMtns

[8] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. ACM Conf. Appl. Technol. Architectures Protocols Comput. Commun.*, 2003, pp. 27–34.

[9] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," in *Proc. ACM Conf. Appl. Technol. Architectures Protocols Comput. Commun.*, 2004, pp. 145–158.

[10] S. M. George, W. Zhou, H. Chenji, M. Won, Y. O. Lee, A. Pazarloglou, R. Stoleru, and P. Barooah, "DistressNet: A wireless ad hoc and sensor network architecture for situation management in disaster response," *IEEE Commun. Mag.*, vol. 48, no. 3, pp. 128–136, Mar. 2010.

[11] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proc. ACM 6th Annu. Int. Conf. Mobile Comput. Netw.*, 2000, pp. 255–265.

[12] Y. Wang, W. Hu, Y. Wu, and G. Cao, "SmartPhoto: A resource-aware crowdsourcing approach for image sensing with smartphones," in *Proc. ACM 15th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2014, pp. 113–122.

[13] "ChangeWave research," 2011. [Online]. Available: http://www.changewaveresearch.com

[14] D. Beaver, S. Kumar, H. C. Li, J. Sobel, P. Vajgel, et al., "Finding a needle in haystack: Facebook's photo storage" in *Proc. Symp. Operating Syst. Des. Implementation*, 2010.

[15] Y. Wu, Y. Wang, W. Hu, X. Zhang, and G. Cao, "Resource-aware photo crowdsourcing through disruption tolerant networks," in *Proc. IEEE 36th Int. Conf. Distrib. Comput. Syst.*, 2016, pp. 374–383.

[16] T. Dao, A. K. Roy-Chowdhury, H. V. Madhyastha, S. V. Krishnamurthy, and T. La Porta, "Managing redundant content in bandwidth constrained Wireless networks," in *Proc. ACM 10th Int. Conf. Emerging Netw. Experiments Technol.*, 2014, pp. 349–362.

[17] P. Zuo, Y. Hua, X. Liu, D. Feng, W. Xia, S. Cao, J. Wu, Y. Sun, and Y. Guo, "BEES: Bandwidth-and energy-efficient image sharing for real-time situation awareness," in *Proc. 37th Int. Conf. Distrib. Comput. Syst.*, 2017, pp. 1510–1520.

[18] J. Edmonds, *Combinatorial Structures and their Applications*, New York, NY, USA: Gordon and Breach, 1970, pp. 69–87.

[19] M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, and C. Schmid, "Evaluation of gist descriptors for web-scale image search," in *Proc. Int. Conf. Image Video Retrieval*, 2009, Art. no. 19.

[20] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[21] D. A. Lisin, M. A. Mattar, M. B. Blaschko, E. G. Learned-Miller, and M. C. Benfield, "Combining local and global image features for object class recognition," in *Proc. IEEE Comput. Society Conf. Comput. Vis. Pattern Recognit. Workshops*, 2005, pp. 47–47.

[22] Y. Ke and R. Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors," in *Proc. IEEE Comput. Society Conf. Comput. Vis. Pattern Recognit.*, 2004, pp. II-506–II-513.

[23] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Proc. Eur. Conf. Comput. Vis.*, 2006, pp. 404–417.

[24] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 2564–2571.

[25] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proc. 9th Eur. Conf. Comput. Vis.*, 2006, pp. 430–443.

[26] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 778–792.

[27] O. Miksik and K. Mikolajczyk, "Evaluation of local detectors and descriptors for fast feature matching," in *Proc. IEEE 21st Int. Conf. Pattern Recognit.*, 2012, pp. 2681–2684.

[28] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, Oct. 2005.

[29] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proc. IEEE Comput. Society Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 2161–2168.

[30] S. Mittal, "A survey of techniques for approximate computing," *ACM Comput. Surveys*, vol. 48, no. 4, 2016, Art. no. 62.

[31] A. Rahimi, L. Benini, and R. K. Gupta, "Spatial memoization: Concurrent instruction reuse to correct timing errors in SIMD architectures," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 60, no. 12, pp. 847–851, Dec. 2013.

[32] H. Lin and J. Bilmes, "Learning mixtures of submodular shells with application to document summarization," in *Proc. 28th Conf. Uncertainty Artif. Intell.*, 2012, pp. 479–490.

[33] S. Tschiatschek, R. K. Iyer, H. Wei, and J. A. Bilmes, "Learning mixtures of submodular functions for image collection summarization," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 1413–1421.

[34] I. Simon, N. Snavely, and S. M. Seitz, "Scene summarization for online image collections," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, 2007, pp. 1–8.

[35] G. K. Wallace, "The JPEG still picture compression standard," *Commun. ACM*, vol. 34, no. 4, pp. 30–44, 1991.

[36] PNG (Portable Network Graphics) Specification Version 1.0, 1997. [Online]. Available: http://tools.ietf.org/html/rfc2083

[37] WebP: A new image format for the Web, 2015. [Online]. Available: https://goo.gl/5pSOZ7

[38] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[39] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proc. 30th Annu. ACM Symp. Theory Comput.*, 1998, pp. 604–613.

[40] Y. Wu and G. Cao, "Videomec: A metadata-enhanced crowdsourcing system for mobile videos," in *Proc. 16th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, 2017, pp. 143–154.

[41] OpenCV Library, [Online]. Available: https://www.opencv.org/

[42] J. Pyke, M. Hart, V. Popov, R. D. Harris, and S. McGrath, "A teleultrasound system for real-time medical imaging in resource-limited settings," in *Prcoc. IEEE 29th Annu. Int. Conf. Eng. Med. Biol. Society*, 2007, pp. 3094–3097.

[43] V. Popov, D. Popov, I. Kacar, and R. D. Harris, "The feasibility of real-time transmission of sonographic images from a remote location over low-bandwidth internet links: A pilot study," *Amer. J. Roentgenology*, vol. 188, no. 3, pp. 219–222, 2007.

[44] Web Debugging Proxy Application, 2017. [Online]. Available: https://www.charlesproxy.com/

[45] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in *Proc. 8th IEEE/ACM/IFIP Int. Conf. Hardware/Software Codesign Syst. Synthesis*, 2010, pp. 105–114.

[46] T. Weyand, J. Hosang, and B. Leibe, "An evaluation of two automatic landmark building discovery algorithms for city reconstruction," in *Trends and Topics in Computer Vision*. Berlin, Germany: Springer, 2012, pp. 310–323.

[47] Y. Sun, Y. Hua, X. Liu, S. Cao, and P. Zuo, "DLSH: a distribution-aware LSH scheme for approximate nearest neighbor query in cloud computing," in *Proc. Symp. Cloud Comput.*, 2017, pp. 242–255.

[48] B. Zhu, K. Li, and R. H. Patterson, "Avoiding the disk bottleneck in the data domain deduplication file system," in *Proc. USENIX Conf. File Storage Technol.*, 2008, Art. no. 18.

[49] S. Quinlan and S. Dorward, "Venti: A new approach to archival storage," in *Proc. USENIX Conf. File Storage Technol.*, 2002.

[50] G. Wallace, F. Douglis, H. Qian, P. Shilane, S. Smaldone, M. Chamness, and W. Hsu, "Characteristics of backup workloads in production systems," in *Proc. 10th USENIX Conf. File Storage Technol.*, 2012, pp. 4–4.

[51] F. Chen, T. Luo, and X. Zhang, "CAFTL: A content-aware flash translation layer enhancing the lifespan of flash memory based solid state drives," in *Proc. 9th USENIX Conf. File Stroage Technol.*, 2011, pp. 77–90.

[52] A. El-Shimi, R. Kalach, A. Kumar, A. Ottean, J. Li, and S. Sengupta, "Primary data deduplication-large scale study and system design," in *Proc. USENIX Annu. Tech. Conf.*, 2012, vol. 2012, pp. 285–296.

[53] I. Drago, M. Mellia, M. M Munafo, A. Sperotto, R. Sadre, and A. Pras, "Inside dropbox: understanding personal cloud storage services," in *Proc. Internet Meas. Conf.*, 2012, pp. 481–494.

[54] P. Zuo, Y. Hua, C. Wang, W. Xia, S. Cao, Y. Zhou, and Y. Sun, "Mitigating traffic-based side channel attacks in bandwidth-efficient cloud storage," in *Proc. IEEE 26th Int. Parallel Distrib. Process. Symp. Workshops*, 2018, pp. 761–761.

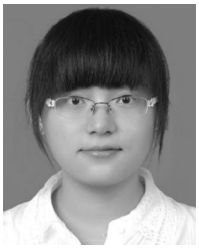
[55] M. Vrable, S. Savage, and G. M. Voelker, "Cumulus: Filesystem backup to the cloud," *ACM Trans. Storage*, vol. 5, no. 4, 2009, Art. no. 14.

[56] A. Anand, C. Muthukrishnan, A. Akella, and R. Ramjee, "Redundancy in network traffic: findings and implications," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 37, no. 1, pp. 37–48, 2009.

[57] H. Pucha, D. G. Andersen, and M. Kaminsky, "Exploiting similarity for multi-source downloads using file handprints," in *Proc. 4th USENIX Conf. Netw. Syst. Design Implementation*, 2007, pp. 2–2.

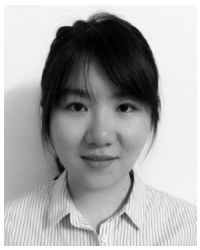
Pengfei Zuo received the BE degree in computer science and technology from the Huazhong University of Science and Technology (HUST), China, in 2014. He is currently working toward the PhD degree majoring in computer science and technology at the Huazhong University of Science and Technology. His current research interests include data deduplication, non-volatile memory, and key-value store. He published several papers in major journals and conferences including the *IEEE Transactions on Parallel and Distributed Systems*, USENIX OSDI, MICRO, USENIX ATC, SoCC, ICDCS, IPDPS, MSST, DATE, HotStorage, etc. He is a student member of the IEEE.

Yu Hua received the BE and PhD degrees in computer science from Wuhan University, China, in 2001 and 2005, respectively. He is a professor with the Huazhong University of Science and Technology, China. His research interests include computer architecture, cloud computing, and network storage. He has more than 100 papers to his credit in major journals and international conferences including the *IEEE Transactions on Computers*, the *IEEE Transactions on Parallel and Distributed Systems*, USENIX ATC, USENIX FAST, INFOCOM, SC, ICDCS and MSST. He has been on the program committees of multiple international conferences, including USENIX ATC, RTSS, INFOCOM, ICDCS, MSST, ICNP and IPDPS. He is the distinguished member of CCF, senior member of ACM and IEEE and a member of USENIX.

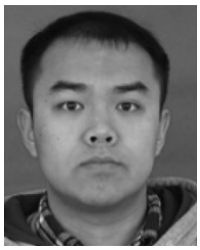


Yuanyuan Sun received the BE degree in computer science and technology from the Huazhong University of Science and Technology (HUST), China, in 2014, and the PhD graduate degree majoring in computer science and technology from the Huazhong University of Science and Technology. Her current research interests include queries in storage systems, semantic hashing and metadata management. She has published several papers in major journals and conferences including the *IEEE Transactions on Parallel and Distributed Systems*, USENIX ATC, SoCC, MSST, ICDCS, IPDPS.

Xue Liu received the BS degree in mathematics and MS degree in automatic control from Tsinghua University, Beijing, China, and the PhD degree in computer science from the University of Illinois at UrbanaChampaign, IL, in 2006. He is currently an associate professor with the School of Computer Science at McGill University, Montreal, QC, Canada. His research interests include computer networks and communications, smart grid, real-time and embedded systems, cyber-physical systems, data centers, and software reliability. His work has received the Year 2008 Best Paper Award from the *IEEE Transactions on Industrial Informatics*, and the First Place Best Paper Award of WiSec 2011. He serves as an associate editor of the *IEEE Transactions on Parallel and Distributed Systems* and editor of the *IEEE Communications Surveys and Tutorials*. He is a member of the IEEE and the ACM.

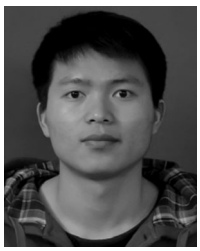


Jie Wu received the BE degree in 2015 and Master degree in 2018 from the Huazhong University of Science and Technology (HUST), China. Her current research interests include cloud storage and data deduplication.



Yuncheng Guo received the BE degree in 2015 and Master degree in 2018 from the Huazhong University of Science and Technology (HUST), China. His current research interests include non-volatile memory, algorithms of hashing, and data analytics.

Wen Xia received the PhD degree in computer science from the Huazhong University of Science and Technology, China, in 2014. His research interests include data reduction, storage systems, and cloud computing. He has published more than 30 papers in major journals and conferences including Proceedings of the IEEE (PIEEE), the *IEEE Transactions on Computers* (TC), the *IEEE Transactions on Parallel and Distributed Systems* (TPDS), USENIX ATC, FAST, INFOCOM, MSST, Performance, IPDPS, HotStorage, etc.



Shunde Cao received the BE degree in computer science and technology from the Wuhan University of Science and Technology, China, in 2014, and the ME degree in computer science and technology from Huazhong University of Science and Technology, China, in 2017. His research interests include data deduplication, content-based similarity detection, and key-value store.

Dan Feng received the BE, ME, and PhD degrees in computer science and technology from the Huazhong University of Science and Technology (HUST), China, in 1991, 1994, and 1997, respectively. She is a professor and vice dean of the School of Computer Science and Technology, Huazhong University of Science and Technology. Her research interests include computer architecture, massive storage systems, and parallel file systems. She is a member of the IEEE and the ACM.