2017 IEEE 37th International Conference on Distributed Computing Systems

BEES: Bandwidth- and Energy- Efficient Image Sharing for Real-time Situation Awareness

Pengfei Zuo, Yu Hua✉, Xue Liu†, Dan Feng, Wen Xia, Shunde Cao, Jie Wu, Yuanyuan Sun, Yuncheng Guo
Wuhan National Laboratory for Optoelectronics
School of Computer, Huazhong University of Science and Technology, Wuhan, China
† McGill University, Montreal, Canada
✉ Corresponding Author: csyhua@hust.edu.cn

***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. 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.**

I. Introduction

According to the most recent World Disasters Report [1], a total of over 6,000 disasters worldwide occurred between 2006 and 2015, which resulted in more than 0.77 million people casualties, affected other nearly 2 billion people, and caused the total amount of estimated damage of over 1.4 trillion US dollars. 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 [2]–[4].

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 [2]–[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% 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 [5]. 3) Real-time Transmission Inefficiency. Real-time data analytics are important for time-sensitive decision making in disaster relief [2], such as, the prediction 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% similar images exist respectively in the San Diego fire (2007) and the Haiti earthquake (2010) imagesets [4]. 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

1063-6927/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDCS.2017.36

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 [4], [5], [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 [4], [5], [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 [3]–[5], [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, 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.

- To identify and eliminate in-batch similar images, we propose a similarity-aware submodular [17] maximization model (SSMM) in BEES to compute the unique image subset for each uploaded image batch.
- 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.
- 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.
- 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 [5] and MRC [16], experimental results demonstrate that BEES reduces more than 67.3% energy overhead, 77.4% bandwidth overhead, 70.4% average image uploading delay, and extends the battery lifetime by 84.3%. Moreover, the energy-aware adaptive schemes in BEES further extend the battery lifetime by about 20%, compared with BEES without energy-aware adaptive schemes.

The rest of this paper is organized as follows. Section II presents the system architecture of BEES. The design details of BEES are described in Section III. We evaluate the performance in Section IV. Section V discusses the related work and Section VI concludes this paper.

II. The System Architecture of BEES

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

In the traditional system architecture [5], [16], as shown in Figure 1, there are three key components, i.e., Image Feature Extraction (IFE), Image Redundancy Detection (IRD) and Unique Image Uploading (UIU). IFE extracts the features (e.g., PCA-SIFT [18] and ORB [19]) 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

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

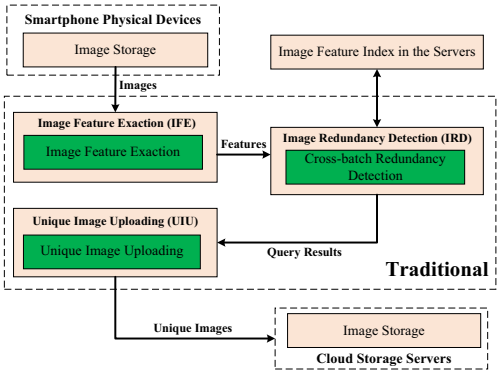


Fig. 1. The traditional system architecture.

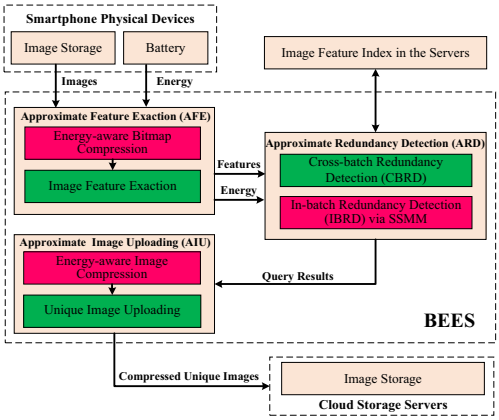


Fig. 2. The system architecture of BEES.

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 I.

Therefore, we propose a new system architecture, i.e., BEES, for bandwidth- and energy- efficient image sharing, as shown in Figure 2. 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.

III. The Design Details

In this section, we present the design details of BEES, including approximate feature extraction, redundancy detection and image uploading. We also present how we select the image feature extraction algorithm used in BEES.

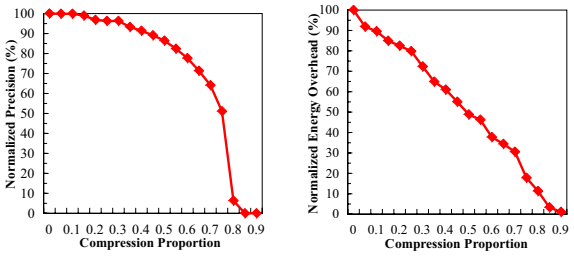
A. Approximate Feature Extraction

To detect image similarity, image features are first extracted and uploaded to the servers. Even though we select a relatively optimal extraction algorithm (described in Section III-D), i.e., ORB [19], the energy overhead to extract image features is still high when the image size is too large. We observe 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 [20]) 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], [20]. 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 vs. Compression proportion: We normalize the query precision of compressed image bitmaps to that of original images as shown in Figure 3(a). 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 vs. Compression proportion: We



(a) Normalized precision (b) Normalized energy overhead

Fig. 3. 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).

normalize the energy overhead of compressed images to that of original images, as shown in Figure 3(b). 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 the remaining energy (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% errors for approximate computing processes are considered to be acceptable [21], [22]. 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%, 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%, C is set to 0.38 according to the function, which can save about 40% energy of extracting features while ensuring higher than 90% precision, as shown in Figure 3.

B. 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.

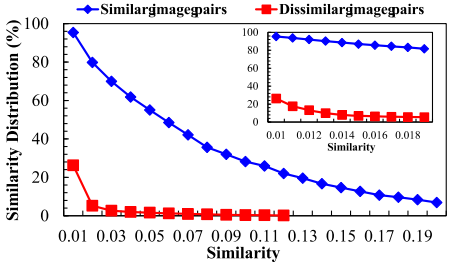


Fig. 4. The similarity distribution.

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 2.

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 IV-A). 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 2). The similarity distributions of similar and dissimilar image pairs are shown in Figure 4.

Figure 4 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% similar images is larger than 0.01 and the similarity of 26.2% dissimilar images is larger than 0.01. Thus, if the similarity threshold is set to 0.01, the true positive rate is 95.4% and the false positive rate is 26.2%. We can also observe that both true and false 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 Figure 4, we argue that the similarity threshold should be larger than 0.013, which leads to 90% true positive rate and 10% false positive rate. In order to ensure the false positive ratio not larger than 10% [21], [22], 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.

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 $V = \{v_1, v_2, ..., v_n\}$, 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 ($F : 2^V \rightarrow$ R) to quantitatively represent the quality of a summary. The image subset can be computed:

$$S^* \in \arg\max_{S \subseteq V} F(S) \quad s.t. \quad |S| \leq b \tag{1}$$

Given a constraint $|S| \leq b$, Equation 1 can be modeled as the form of submodular maximization subject to knapsack constraints which is NP-complete [23]. Moreover, if the function F is the monotone submodular function, a simple greedy algorithm can efficiently and near-optimally address Equation 1 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 [23], [24].

Equation 1 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 [24], [25], 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 [17], 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 $f : 2^V \rightarrow$ 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 $f_i (i = 1, 2, ..., m)$ 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 [24], [25]. 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: $g_1, g_2, ..., g_b$. 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.
1: Determine the threshold T_w based on E_{bat};
2: Partition graph G using T_w;
3: Get the number of partitioned subgraphs b;
4: Choose v_1 arbitrarily;
5: $S_1 \leftarrow v_1$;
6: **while** $|S_i| \leq b$ **do**
7: Choose $v_i \in \arg\max_{v_i \in V \setminus S_i} F(S_i \bigcup \{v_i\})$;
8: $S_{i+1} \leftarrow S_i \bigcup \{v_i\}$;
9: $i \leftarrow i + 1$;
10: **end while**

C. 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 2MB [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 [21]. 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.

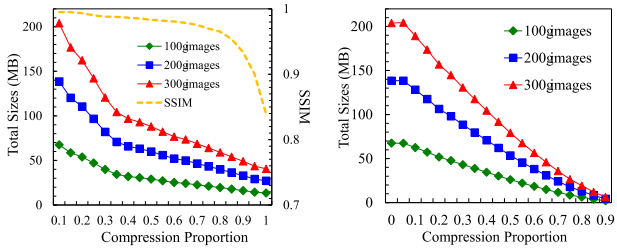
Quality Compression. Quality compression uses mathematical operations to convert pixels of an image from the spatial domain into the frequency domain for reducing the required storage space of an image, which does not change the resolution of an image. Our experiments explore the relationship between the file size and compression proportion in quality compression. The compression proportion in quality compression is defined as the ratio of the number of compressed pixels to the number of initial pixels. There are many quality image compression standards, such as, JPEG [26], PNG [27] and WebP [28]. 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 [21].

Resolution Compression. Resolution compression aims to directly reduce the resolutions of images to reduce the file size, e.g., compressing an image with the resolution $1200 * 800$ to $600 * 400$. We also perform experiments to explore the relationship between the file size and compression proportion in resolution compression. The compression proportion in resolution compression 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.

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 Figure 5.

We use the SSIM (Structural SIMilarity) index [29], 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 Figure 5(a). 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 Figure 5(a).

As shown in Figure 5(b), resolution compression can also



(a) Quality compression (b) Resolution compression

Fig. 5. The influences of quality compression and resolution compression to bandwidth overheads.

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 the remaining energy (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%, 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% file size compared with $E_{bat} = 100\%$.

D. Image Features

Both global features and local features 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 small patches in the images. Since local features have more robust and higher accuracy than global features for similarity detection [4], [16], we focus on the local features in BEES.

There are several common local feature algorithms. SIFT [30] is a widely used algorithm to find 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 [18] reduces the dimensions of features from 128 to 36 while increasing the time of computing features. ORB [19] computes binary visual descriptors rather than the vector-based descriptors computed by the above several algorithms. Each ORB feature is described by 256 binary digits. Existing work [19] shows that ORB is about two orders faster than SIFT while obtaining approximate accuracy with SIFT in similarity detection. Thus ORB has lower computation and time overheads than other algorithms, which closely matches our cost-efficient design

goal. Hence, we choose ORB to extract image features in BEES.

An image I_i can be represented as a set of ORB 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:

$$sim(I_1, I_2) = \frac{|S_1 \bigcap S_2|}{|S_1 \bigcup S_2|} \quad (2)$$

IV. Performance Evaluation

We have implemented the BEES prototype. In this section, we present the performance evaluation based on the prototype.

A. 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) and linked with the openCV library [31] 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++ 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 [3], [32], [33] limit their bandwidth to several hundred Kbps to simulate the low bandwidth. Hence, in our experiments, the smartphones communicate with the servers via the WiFi network, in which the transmission bandwidth of each smartphone fluctuates from 0Kbps to 512Kbps to emulate the low-bandwidth network.

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

- **The Kentucky imageset** [20]: 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], [20], we also use it to evaluate precision in Section IV-B1.
- **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 IV-B3, IV-B4, and IV-B5.
- **The Paris imageset** [34]: 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 IV-B3 and IV-B6.

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 [5] 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.

B. Results and Analysis

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], [20]. 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 III-A) 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 Figure 6.

As shown in Figure 6, SIFT obtains the highest precision. Compared with SIFT, the precision in BEES(100) is higher than 90.3%, which is close to PCA-SIFT. We observe that the precision of BEES slightly decreases with the decrease of E_{bat}, since BEES improves the energy efficiency with

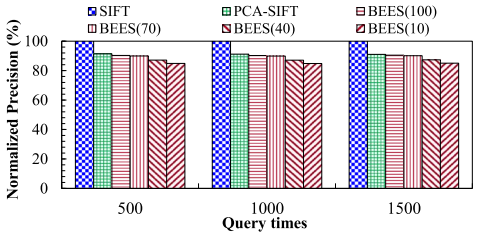


Fig. 6. The normalized precision (BEES(X) means the BEES under the condition of $X\%$ E_{bat}).

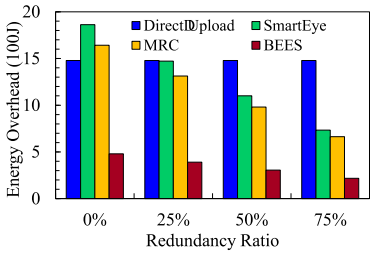


Fig. 7. Energy overhead.

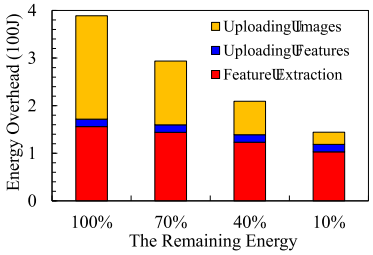


Fig. 8. Energy-aware adaptation.

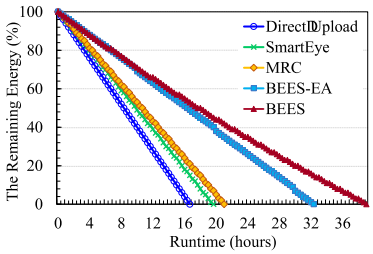


Fig. 9. Battery lifetime.

a slight decrease in precision in the low battery status. Moreover, BEES(10) obtains over 84.9% precision compared with SIFT. SIFT and PCA-SIFT obtain a little higher precision than BEES, which however are not suitable in the disaster environments to be uploaded and used to detect similarity due to their huge space overheads as shown in the following subsection.

2) Space Overhead of Image Features: We respectively extract SIFT, PCA-SIFT and ORB features of images in Kentucky and Paris imagesets. For clarity, we also normalize their space overheads to that of SIFT features as shown in Table I. SIFT consumes the largest space overhead, which may be even larger than the space overhead of images themselves. The space overhead of BEES using ORB is about one order smaller than that of PCA-SIFT, and about two orders smaller than that of SIFT. Hence, BEES requires small space and bandwidth overheads to store and upload image features, thus meeting the needs of disaster environments.

TABLE I
SPACE OVERHEADS

Imagesets	Image size	SIFT	PCA-SIFT	BEES
Kentucky	6.67 GB	3.40 GB (*100%*)	956 MB (*25%*)	155.6 MB (*4.46%*)
Paris	361.5 GB	424.3 GB (*100%*)	119.3 GB (*25%*)	7.47 GB (*1.76%*)

3) 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%, 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 2) 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 Figure 7, 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 energy savings. Compared with MRC, BEES reduces $67.3\% - 70.8\%$ energy overheads. Compared with Direct Upload, BEES reduces $67.6\% - 85.3\%$ energy overheads. Even in the worst case with no cross-batch redundancy, BEES also obtains 67.6% 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 IV-B3(1)) with 10 in-batch similar images. We set the same cross-batch redundancy ratio to be 25% for each uploading. When the remaining energy of the smartphone is $100\%, 70\%, 40\%$, and 10%, we respectively upload the 100 images using BEES and examine their energy overheads of feature extraction, uploading features and images.

As shown in Figure 8, 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% 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 battery is exhausted. We record the remaining energy of battery every 20 minutes.

As shown in Figure 9, 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% battery lifetime, compared with Direct Upload. Compared with MRC, BEES extends 84.3% battery lifetime. Compared with BEES-EA, BEES extends 19.8% battery lifetime due to energy-aware adaptive schemes.

4) Bandwidth Overhead: When examining the energy overheads of the four schemes in Section IV-B3(1), we record the bandwidth overhead of each scheme. As shown in Figure 10, 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 $77.4\% - 79.2\%$ bandwidth overheads.

5) Delay: We compare the delays of different schemes in this subsection. We use the same image collection with 100 images used in Section IV-B3(1). There are 10 in-batch

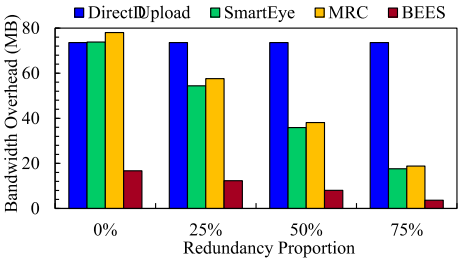


Fig. 10. Network bandwidth overhead.

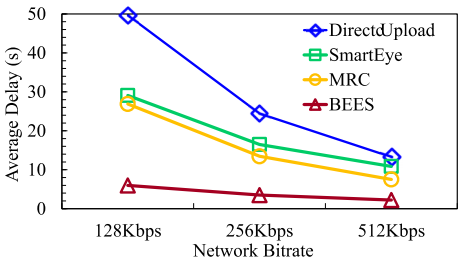


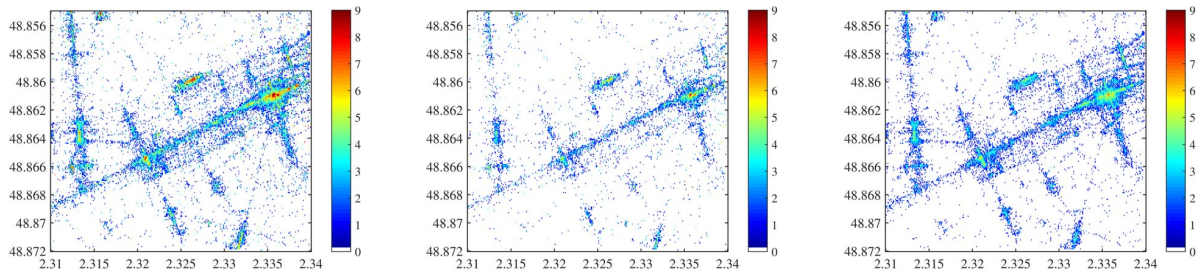
Fig. 11. The average delay of uploading an image.

similar images in the 100 images. We set the same cross-batch redundancy ratio (50%) 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 128Kbps and 512Kbps, besides 256Kbps. The delay consists of the time of extracting image features, uploading features and images, without the querying time of the servers, since we focus on the resource-constrained smartphones and the network instead of the cloud servers with sufficient resources in disaster environments.

As shown in Figure 11, 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 Figure 11, BEES reduces $83.3\% - 88.0\%$ average delay compared with Direct Upload, and reduces $70.4\% - 77.8\%$ 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.

6) 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



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

Fig. 12. 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.)

shown in Figure 12(a). 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. Figure 12(b) 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. Figure 12(c) shows the coverage of the images uploaded using BEES in the map. BEES uploads 18.8% more images while has 97.1% 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.

V. Related Work

Data Deduplication in Networks. Deduplication [35], [36] detects and eliminates exact-matching redundancy to save bandwidth and storage space. 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 [4]. 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 [3] 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 [4] 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 [5] 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.

VI. 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% energy overhead, 77.4% bandwidth overhead, 70.4% average image uploading delay, and extends 84.3% battery lifetime, compared with the state-of-the-art work.

Acknowledgement

This work was supported by National Key Research and Development Program of China under Grant 2016YFB1000202; National Natural Science Foundation of China (NSFC) under Grant 61502190.

REFERENCES

[1] "World Disasters Report 2016: Resilience: saving lives today, investing for tomorrow." *International Federation of Red Cross and Red Crescent Societies*.

[2] "Big Data and Disaster Management: A Report from the JST/NSF Joint Workshop," https://goo.gl/0gath8, 2015.

[3] 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 RTSS*, 2011.

[4] 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 Hotnets*, 2012.

[5] Y. Hua, W. He, X. Liu, and D. Feng, "SmartEye: Real-time and Efficient Cloud Image Sharing for Disaster Environments," in *Proc. IEEE INFOCOM*, 2015.

[6] "How Social Media Is Helping Nepal Rebuild After Two Big Earthquakes," https://goo.gl/CbIMzT, 2015.

[7] "[New Scientist] Social media helps aid efforts after typhoon Haiyan," https://goo.gl/yJMtns.

[8] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. ACM SIGCOMM*, 2003.

[9] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," in *Proc. ACM SIGCOMM*, 2004.

[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 Communications Magazine*, vol. 48, no. 3, pp. 128–136, 2010.

[11] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proc. ACM MobiCom*, 2000.

[12] Y. Wang, W. Hu, Y. Wu, and G. Cao, "SmartPhoto: a resource-aware crowdsourcing approach for image sensing with smartphones," in *Proc. ACM MobiHoc*, 2014.

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

[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. OSDI*, 2010.

[15] Y. Wu, Y. Wang, W. Hu, X. Zhang, and G. Cao, "Resource-Aware Photo Crowdsourcing Through Disruption Tolerant Networks," in *Proc. IEEE ICDCS*, 2016.

[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 CoNEXT*, 2014.

[17] J. Edmonds, *Combinatorial Structures and their Applications*. chapter Submodular functions, matroids and certain polyhedra, pages 69–87. Gordon and Breach, 1970.

[18] Y. Ke and R. Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors," in *Proc. IEEE CVPR*, 2004.

[19] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *Proc. IEEE ICCV*, 2011.

[20] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proc. IEEE CVPR*, 2006.

[21] S. Mittal, "A survey of techniques for approximate computing," *ACM Computing Surveys (CSUR)*, vol. 48, no. 4, p. 62, 2016.

[22] A. Rahimi, L. Benini, and R. K. Gupta, "Spatial memoization: Concurrent instruction reuse to correct timing errors in simd architectures," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 60, no. 12, pp. 847–851, 2013.

[23] H. Lin and J. Bilmes, "Learning mixtures of submodular shells with application to document summarization," in *Proc. UAI*, 2012.

[24] S. Tschiatschek, R. K. Iyer, H. Wei, and J. A. Bilmes, "Learning mixtures of submodular functions for image collection summarization," in *Proc. NIPS*, 2014.

[25] I. Simon, N. Snavely, and S. M. Seitz, "Scene summarization for online image collections," in *Proc. IEEE ICCV*, 2007.

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

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

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

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

[30] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[31] "OpenCV library," https://www.opencv.org/.

[32] J. Pyke, M. Hart, V. Popov, R. D. Harris, and S. McGrath, "A tele-ultrasound system for real-time medical imaging in resource-limited settings," in *Prcoc. IEEE EMBS*, 2007.

[33] 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," *American Journal of Roentgenology*, vol. 188, no. 3, pp. 219–222, 2007.

[34] 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*. Springer, 2012, pp. 310–323.

[35] A. Muthitacharoen, B. Chen, and D. Mazieres, "A low-bandwidth network file system," in *Proc. ACM SOSP*, 2001.

[36] B. Zhu, K. Li, and R. H. Patterson, "Avoiding the Disk Bottleneck in the Data Domain Deduplication File System." in *Proc. USENIX FAST*, 2008.