Pers Ubiquit Comput (2016) 20:325–335
DOI 10.1007/s00779-016-0924-x

ORIGINAL ARTICLE

PicPick: a generic data selection framework for mobile crowd photography

Bin Guo[1] · Huihui Chen[1] · Zhiwen Yu[1] · Xing Xie[2] · Daqing Zhang[3]

Received: 29 September 2015 / Accepted: 30 March 2016 / Published online: 29 April 2016

Abstract Mobile crowd photography (MCP) is a widely used technique in crowd sensing. In MCP, a picture stream is generated when delivering intermittently to the backend server by participants. Pictures contributed later in the stream may be semantically or visually relevant to previous ones, which can result in data redundancy. To meet diverse constraints (e.g., spatiotemporal contexts, single or multiple shooting angles) on the data to be collected in MCP tasks, a data selection process is needed to eliminate data redundancy and reduce network overhead. This issue has little been investigated in existing studies. To address this requirement, we propose a generic data collection framework called PicPick. It first presents a multifaceted task model that allows for varied MCP task specification. A pyramid tree (PTree) method is further proposed to select an optimal set of pictures from picture streams based on multi-dimensional constraints. Experimental results on two real-world datasets indicate that PTree can effectively reduce data redundancy while maintaining the coverage requests, and the overall framework is flexible.

Keywords Data selection · Mobile phone sensing · Crowd sensing · Pyramid tree clustering · Multi-dimensional coverage

✉ Bin Guo
guob@nwpu.edu.cn

[1] Northwestern Polytechnical University, Xi'an 710072, China

[2] Microsoft Research, Beijing, China

[3] Peking University, Beijing, China

1 Introduction

With the development of smartphone sensing, wearable computing, and mobile social networks, a new sensing paradigm called mobile crowd sensing and computing (MCSC) [1], which leverages the power of ordinary users for large-scale sensing, has become popular in recent years. Data collected onsite in the real world, combined with the support of the *backend server* where data fusion takes place, makes MCS a versatile platform that can often replace static sensing infrastructures.

Picture taking is a widely used sensing technique of MCSC, referred to as mobile crowd photography (MCP) in this paper. MCP has benefited a variety of application areas, such as pollution monitoring [2], disaster relief [3, 4], scientific data collection [5], public sensing [6], traffic planning [7]. Each MCP task differs in their sensing targets and sampling contexts (e.g., place, time, shooting angle). Nevertheless, existing MCP systems mainly aim at one or one kind of specific task, and there is not a unified platform for managing varied MCP tasks. The motivation for building a generic framework for MCP is inspired by MTurk (http://www.mturk.com/) and has the following merits. First, it facilitates the rapid specification of MCP tasks with different constraints, without the need to develop individual or proprietary systems. Second, it lowers the barrier for ordinary users to publish MCP tasks. Third, it provides mobile users with a unique way to access MCP tasks, which can simplify worker recruitment and information sharing. Despite the aforementioned benefits, there are several challenges to building such a framework.

1. *Unified MCP task modeling*. Different MCP tasks have distinct needs and contextual constraints. For example, some tasks allow for a long sampling interval [5],

while others need short intervals [7]; some tasks may need pictures from different shooting angles [3, 4], while others may need only one snapshot [2, 7]. In order to build a generic MCP framework, we should conduct a thorough analysis of MCP concepts and constraints, and find a way to model different tasks.

2. *Selective picture collection*. In participatory collection, data is collected in a distributed manner and the data delivered by participants arrive at the backend server intermittently, forming a picture stream. In terms of the MCP task requirements, later pictures might be similar to previous ones (e.g., pictures of the same target at the same location taken from slightly different angles) or semantically redundant (e.g., when only one snapshot is needed for a scene but more are captured) and have low utility. Therefore, data selection should be conducted to reduce redundancy and minimize the network overhead of picture streams. In other words, a representative subset that offers roughly the same "coverage" of the target but with fewer pictures should be selected. The computation cost on selection should also be optimized, particularly considering that the picture set at the server side is increasing with the data stream size.

To tackle the above challenges, we investigate the requirements and constraints of various MCP tasks, and propose a generic and optimized MCP data collection framework called PicPick. Specifically, our contributions include:

1. *Proposing a generic picture collection framework for MCP, which is applicable to tasks of varying themes and constraints*. Based on a multifaceted task model, the framework enables the specification of tasks with multi-dimensional constraints. It also presents a method for coverage-based minimum data selection.
2. *Developing a pyramid tree (PTree) method that can cluster the data stream and enable efficient picture selection*. With the proposed PTree model and associated tree generation rules (e.g., branching and layering), PicPick can intelligently cluster the dynamic pictures and facilitate decision making on data selection.
3. *Performance validation*. By recruiting 50 more participants, we have collected two real-world MCP datasets with different themes. Experiments on them indicate that our approach is highly effective and efficient compared to related studies.

The rest of this paper is organized as follows. In Sect. 2, we present the related studies of our work. Conceptual modeling and detailed problem analysis is given in Sect. 3, followed by the PicPick framework in Sect. 4. The PTree data selection method is presented in Sect. 5, and the performance of it is evaluated in Sect. 6. We conclude our paper in Sect. 7.

2 Related work

There are two closely related research areas in our work: *mobile crowd photography* and *data selection in crowd sensing*.

2.1 Mobile crowd photography

MCSC has become a novel paradigm to achieve large-scale sensing. In MCSC, average users are allowed to share local knowledge acquired by their smartphones. It has been successfully used in numerous application areas, including environment monitoring [8], urban profiling [9], indoor floor planning [10], public safety protection [11]. Regarding the various application needs, MCSC can involve different modalities of sensing, such as numeric readings [9, 10], audio recording [8], online posts [11], and picture taking [2–7]. Ma et al. [12] investigated the opportunistic characteristics of human mobility from the perspectives of both sensing and data transmission, and presented approaches to effectively collecting MCSC data. Zhang et al. [13] investigated the generic four-stage lifecycle of MCSC systems and studied the core issues in each stage. Our previous work [1] characterized the unique features of MCSC (e.g., grassroots-powered sensing, human–machine intelligence fusion [14], opportunistic/transient networking, and cross-space sensing) and presented the technical challenges of it, such as participant selection, incentive mechanisms, data quality maintenance, and cross-space data mining [15].

Mobile crowd photography (MCP), large-scale visual sensing via built-in cameras of smartphones, has become a dominant form of crowd sensing. Typical examples of MCP include traffic dynamics detection [7], creek pollution monitoring [2], on-the-board poster reposting and sharing [6], and emergency scene recording [3, 4]. In essence, these MCP applications can be viewed as tasks with different sensing targets and constraints. Though MCP is widely used, little research has been undertaken to develop a general MCP framework to meet diverse requirements. PicPick is, to the best of our knowledge, the first attempt to develop such a framework and support optimized data collection for various MCP tasks.

2.2 Data selection in crowd sensing

Data contributed by a crowd can be redundant for task needs. Therefore, data selection is necessary to eliminate

data redundancy and lower transmission cost. Data selection can be conducted in two distinct ways, i.e., *offline* and *online*. In offline selection, the server selects pictures when the entire picture set becomes available. The information retrieval community has worked at length on offline redundancy elimination for the retrieved information [16, 17]. In online selection, however, the decision on whether a picture should be selected is made instantly upon its arrival. Since image transmission incurs a high cost on network traffic, the online model has been adopted in many MCP systems, such as PhotoNet [3] and CARE [18]. As a result, we will address challenges pertaining to online data selection in MCP.

Data selection is application specific, and varied requirements of MCP tasks should be considered. For example, the targets of tasks might be either local (e.g., PhotoNet [3]) or global (e.g., Creekwatch [2]), the shooting angles could be either single (e.g., FlierMeet [6]) or multiple (e.g., SmartPhoto [4], PhotoCity [19]), and the status of sensed objects might change slowly (e.g., FlierMeet [6]) or quickly (e.g., SignalGuru [7]). Therefore, the generic data selection framework for MCP should adapt to various task requirements. PicPick is a generic MCP framework that supports efficient online redundancy elimination based on multi-dimensional task constraints.

2.3 Data stream clustering

Compared to entire datasets, clustering over data streams is challenging due to its dynamic nature and online processing requirements. Guha et al. [20] presented a one-pass clustering algorithm; however, the quality of the clusters was poor when the data evolved considerably over time. Aggarwal et al. [21] proposed an online–offline approach that explored the nature of the evolution of the clusters over different time periods. Chen et al. [22] presented a density-based approach to cluster real-time streaming data. Gaber et al. [23] gave a review of the problems and methods on data stream mining. They also made a summary of the typical applications on data stream mining. Different from previous studies, we have presented a hierarchical clustering method called PTree, where every level represents a resolution or granularity of clusters. This method can meet the multi-dimensional constraints of MCP tasks and can efficiently process real-time crowdsourced picture streams.

3 Conceptual modeling and problem analysis

We first present two use cases of PicPick, based on which a generic MCP task model is presented. The problem of online data selection under task constraints is then analyzed.

3.1 Use cases

We use two different use cases about MCP to demonstrate the motivation and usage of PicPick.

1. *Disaster Relief*. After a disaster, the rescue center needs to gain enough knowledge about the damage status to design rescue plans. It asks survivors and first responders to send pictures about the disaster area through their smartphones to the rescue center. To obtain a complete picture about the disaster damage, the rescue center prefers the delivery of pictures from different places of the disaster area while not the delivery of many pictures from only populated places [3]. On the other hand, it wants to obtain the temporal dynamics of the disaster area. Since the network bandwidth is limited after the disaster, it prevents delivery of all pictures to the rescue center. PicPick thus needs to select pictures to maximize the spatiotemporal coverage. At least two constraints are needed to be considered in data selection: the geo-distance of pictures (e.g., 20 m) and the time interval (e.g., 1 h).
2. *Object Imagery*. Object imagery (e.g., 3D modeling of buildings, Google's street view) has recently developed as a hot research area. It is valuable in a number of application areas, ranging from augmented reality to urban planning. In object imagery, we want to reach a rapid and even coverage of different aspects, usually pre-defined, of the target object. To achieve this, online data processing and visualization is often needed at the backend server [19], which allows participants to share in real-time the coverage status and steer them to uncovered or poorly-covered aspects. To make people understand the real-time coverage status, PicPick can set multi-dimensional constraints to determine the semantic redundancy of two submissions (i.e., they cover the same aspect). For example, photographs taken over 100 m apart or with shooting angles differing by over 30° are thought to have captured different aspects.

From the above scenarios, we can derive three requirements for the MCP framework.

- *Unified task model*. We find that different MCP tasks have different data collection requests. In order to adapt to various MCP tasks, PicPick should present a unified model that allows task requesters to characterize their different needs.
- *Redundant data grouping*. With the data redundancy issue, methods should be studied to group similar pictures (at the content- or semantic-level) over the picture stream in a real-time manner.

- *Data selection with task constraints*. Given the predefined task constraints, a data selection method should be developed to select appropriate data from the data streams. This can improve the data transmission performance.

3.2 Task modeling

Based on the analysis of existing MCP applications, we propose a 7-tuple task model: *Task* = <*whn*, *whr*, *itv*, *loc*, *ori*, *imgs, cont*>. Here, *whn* is a valid period for performing the task, including the start time (*ST*) and the end time (*ET*); *whr* refers to the targeted sensing area; *loc* is the geo-distance and pictures within this might be semantically redundant; *itv* refers to how often new pictures should be collected; *ori* denotes the minimum orientation gap to the pictures of the same scene; *imgs* refers to the threshold over the distance among pictures using certain visual features. These six elements can be used as *constraints/features* for data selection at the server side, while the last element *cont* is a description of the task for a participant (or mobile client) to easily understand and execute tasks. It should be noted that the task model is flexible and we can insert new constraints in it for task specification.

Once a picture is taken, the mobile client saves the image file and records the associated contexts, forming a *picture record*. An MCP picture record p is modeled as p = <*wid*, *img*, *t*, *l*, *ang*>. Here, *wid* refers to the *id* of the participant; *img* refers to the picture file; t and l denote when and where the picture is taken; *ang* represents the shooting angle of a picture in the form: <*azimuth, pitch, roll*>, which can be obtained based on accelerometer and magnetic field sensor readings [24]. Pictures are contributed in a distributed manner and finally form a picture stream P. It consists of a sequence of picture records p_1,…, p_k,… arriving at timestamps t_1,…, t_k…

According to different task requirements, redundancy can have distinct meanings, roughly categorized as the following two types.

- *Content-redundancy(ConR)*. This refers to the visual similarity among pictures, using visual features such as SIFT [6], color histogram [3].
- *Semantic-redundancy(SemR)*. The similarity is defined at the semantic or contextual level, using features such as location [3], shooting angle [4]. For example, different buildings may look alike in pictures, but if their locations are different, there is no *SemR* because they carry distinct information.

An example to illustrate the above definitions is shown in Fig. 1. For a task, there are three targets Tg_1–Tg_3. Eight participants w_1–w_8 contribute a picture stream $P = p_1$–p_8 under different contexts. The identification of redundancy among pairs of pictures is given in the figure. For example, though the time interval between p_2 and p_3 is above r, the orientation gap between them is lower than v, so *SemR* is not true. Similarly, for p_4 and p_5, the *SemR* is true, while *ConR* (measured by the visual feature *ImgF*) is not true. p_6–p_8 are both semantic and content relevant, and p_6 is finally selected because it arrives earlier.

3.3 Problem analysis

Generally, our work can be viewed as an online min-selection problem, i.e., given the multi-dimensional coverage requirements and a picture stream, dynamically selecting a minimum set of pictures (to eliminate redundancy) that can satisfy these constraints. Clustering is an often-used approach for redundancy elimination [25]. With clustering, semantically similar objects can be grouped and only representative ones from each cluster are selected for the final dataset. The original problem thus becomes a data stream clustering issue [26].

Traditional clustering algorithms generally adopt the distance over selected visual features as a metric to measure their similarity and cluster pictures. The problem is different in MCP as we need to address multi-dimensional, heterogeneous constraints (space, time, shooting angle, etc.). It is difficult to integrate these constraints in a single

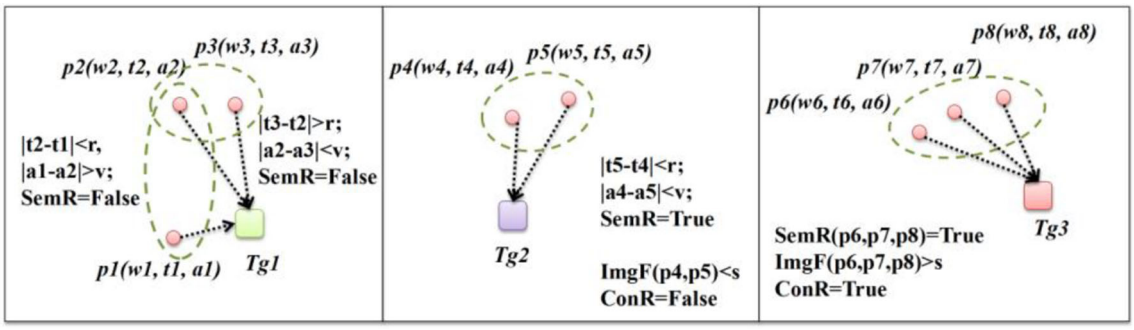

Fig. 1 Task, picture stream, and data selection; $Task.itv = r$, $ori = v$, $imgs = s$; p_1–p_5 are kept, while p_6 is selected from among p_6–p_8 based on its arrival time

distance measurement metric, so we employ a hierarchical manner (by considering different constraints at different levels). Existing hierarchical clustering methods are usually based on the entire dataset with a specific feature, which demonstrates the granularity of clusters in terms of the feature. We have different features and need to use them at different levels, similar to a decision tree. Though the decision tree model is generally used for classification, it has been explored in an unsupervised manner for hierarchical clustering [27]. However, [27] 's method is based on the entire dataset and it cannot deal with data streams.

To address the above issues, we have developed Pyramid Tree (PTree), a hierarchical, multi-constraint-based clustering algorithm for selecting representative pictures from crowdsourced data streams. This takes advantage of the unsupervised decision tree model for interpretable, constraint-based grouping, and proposes branching and layering strategies for efficient clustering of picture streams.

4 The PicPick framework

Based on the identified requirements, we have developed the PicPick framework, as shown in Fig. 2. The framework consists of several important components. The multifaceted task model is responsible for defining tasks with different types of requests and constraints. The *task publishing & allocation* component allows task requesters to specify and publish new tasks. It can also allocate tasks to a group of qualified participants, as discussed in our previous work [28]. In *task execution*, pictures and their associated contexts are recorded and intermittently transmitted to the backend server. The generated data stream is fed into the *data selection* module, which selects data from the picture stream in view of pre-defined task constraints. Three core components are involved: *PTree layering, branching* and *distance measurement*. To reduce network costs, a thumbnail of an incoming picture and associated contexts (extremely light weight compared to the original image) will first be sent to the backend server. The thumbnail will then be used as an input for a dynamically generated *PTree* and clustered with existing pictures. The clustering result is used to judge whether the full-size picture should be submitted or not.

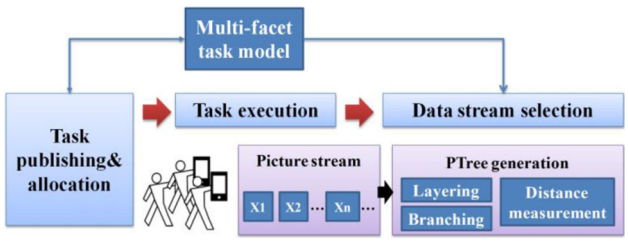

Fig. 2 PicPick framework

5 Pyramid tree-based picture selection

We describe the PTree model and use examples to explain its working mechanism.

5.1 The pyramid tree model

PTree is a tree structure for data stream clustering with multi-dimensional features or constraints. A PTree has $d + 2$ layers if the task has d features. The root node exists at layer-0, and the other layers are defined as layer-l ($1 \leq l \leq d + 1$). *Leaf nodes* (LNs) only exist at the bottom layer, consisting of picture record instances. Each *non-leaf node* (NLN) n corresponds to a *micro-cluster*, composed of the LNs whose ancestors involve n. Each node in a PTree has an *id* and an *index*. *id* is the serial number of a node among its siblings. We define the path from the root node to this node as the *index* of the node, composing of a sequence of *ids* of nodes in the path.

Assuming that the picture stream is $\{p_1,\ldots,p_{12}\}$ and the task has three constraints, a 5-layer PTree can be generated, as shown in Fig. 3a. Each node is represented by its *index* (e.g., N_{12}). Records p_1, p_3, and p_5 belong to the micro-cluster of N_{111}, and p_1–p_6 belong to the micro-cluster of N_{11}.

The generation of a PTree consists of two collaborative operations: *branching* and *layering*. Branching decides which cluster a new picture should be placed in (at the horizontal level), while layering refers to the one-to-one mapping between the feature set and the layer set (at the vertical level).

5.2 Branching and picture selection

In data stream clustering, we must decide whether an incoming picture should be grouped in an existing cluster. To do this in PTree, we first calculate the semantic/content distance between the picture and already generated micro-clusters. Since we are conducting hierarchical clustering with PTree, it needs to be matched from the top to the bottom layers with the micro-clusters at each layer. If a picture record has a distance less than *th_t* (a threshold) to a micro-cluster at layer-*t*, we call the associated NLN an m-NLN (matched-NLN). The distance measurement method will be described in Sect. 5.3. A picture can have more than one m-NLN on the same layer. To choose a suitable one, we use two methods: Fast-match (fastM) and Min-match (minM). The former method chooses the first-

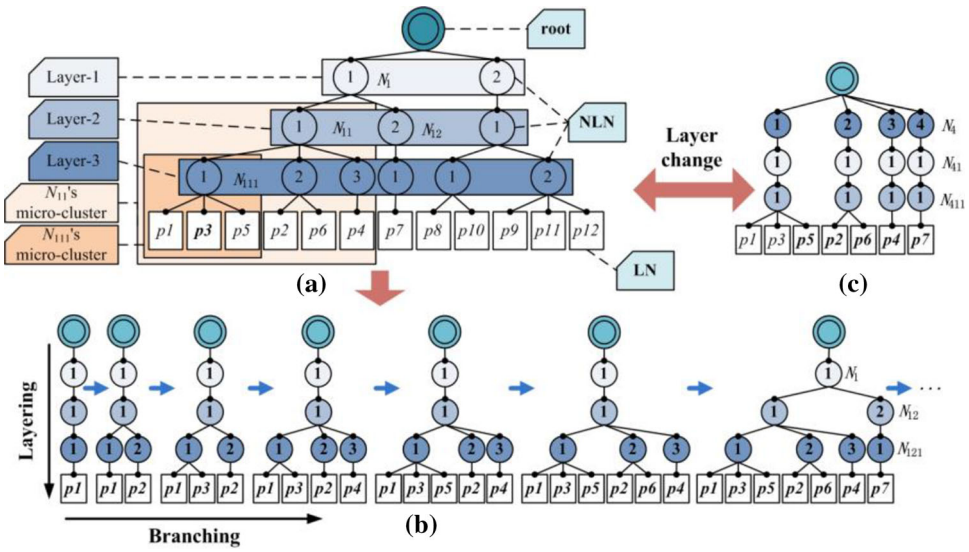

Fig. 3 Pyramid tree example

matched m-NLN, while the latter chooses the m-NLN with the shortest distance value. The branching process is elaborated in detail below.

- If m-NLN N_j is chosen as a match for picture record p, then only the child nodes of N_j will be searched in the next layer. If the next layer is the bottom layer, an LN (with the picture record p) will be created as the child node of N_j.
- If the m-NLN of p is not found in layer-t, a new branch from the m-NLN N_j in layer (t-1) to the bottom layer will be created.

An example of the branching process is shown in Fig. 3b, and the PTree grows with the arrival of records p_1–p_7 by using fastM. For example, when p_7 arrives, N_1 is its m-NLN in layer-1 and no m-NLN is found in the next layer, so a new branch from N_1 to N_{121} is created.

To ensure constraint coverage and reduce the network cost, the *time-priority* method is used for picture selection, where only the first-arriving item of a micro-cluster needs the full record. For example, p_1, p_3 and p_5 have the same ancestor N_{111}. Since p_1 arrives first, the full-size picture is submitted. For p_3 and p_5, we only keep their thumbnails and contexts. With this strategy, the final selection result for Fig. 3a is $\{p_1, p_2, p_4, p_7, p_8, p_9\}$.

5.3 Distance measurement

The distance measurement in clustering is measured with the *center* of a micro-cluster (or NLN) and the picture record. The center of NLN N_{111} in Fig. 3a is calculated based on the value of items p_1, p_3, and p_5. The method of calculating the center of an NLN is relevant to the feature used in that layer. For location or shooting angle, the mean value can be used as the value of the center. For other features such as visual features, it is somewhat difficult to obtain an average value. For simplicity's sake, we use either *first-as-center* (FaC) or *last-as-center* (LaC) to set the center of an NLN. As shown in Fig. 3a, the *center* of N_{111} with FaC and LaC is p_1 and p_5, respectively. The method for calculating distance varies for different features. Some of them used in this work are given below.

- *Location/time*. We use the Euclidean distance to measure them.
- *Shooting angle distance*. It refers to the angle between the shooting directions of two pictures in the 3D space.
- *Visual features*. We use the SIFT feature to identify the matched points of two pictures. The visual distance refers to the ratio of unmatched points to the entire point set.

5.4 Layering strategy

Layering is associated with the one-to-one mapping of the attribute set with the layers of a PTree. Given the task constraints, we can estimate the number of potential branches (NPB) for each constraint. For example, for task t, the NPB to the time constraint can be estimated as $(\mathrm{TE} - \mathrm{TS})/t.itv$.

During task execution, if the crowd contributions adequately meet the constraints (e.g., data is distributed with the right time- and geo-distances), a near-complete tree with all potential branches to each constraint can be generated. However, due to uncertain crowd behavior patterns and the dynamics of targets to be sensed, in many cases people tend to contribute more data in certain contexts (e.g., time slots, places), while few or none in others. The resulting PTree might be incomplete in these cases. In such situations, the PTree generation process is similar to the Trie-tree construction problem [29], where the computational cost is greatly affected by the feature used at each layer.

Its optimization is difficult since it is hard to anticipate the dynamics of the picture stream. However, the computational cost can be lowered when generating a gradually expanding tree (we call this an *A*-shape) for Trie-trees [29]. To achieve this, one strategy is to have features with smaller NPBs in the upper layers. Due to uneven coverage, if a feature with a small NPB is placed at lower layers, few branches will be generated, which may destroy the *A*-shape. An example is given in Fig. 3c, which moves layer-3 in Fig. 3a to layer-1 so that the *A*-shape tree cannot be generated.

6 Evaluation

The aim of PicPick is to enable optimized and efficient picture selection over diverse MCP task constraints. We first present the measurement metrics and then the results over two real-world datasets.

6.1 Metrics, baseline, and datasets

1. *Metrics*. We define the following metrics for evaluation.
   - *Effectiveness*. Given a picture set P, if two pictures in P are identified as similar under certain constraint settings, we build a link/edge between them. This generates a graph G (each picture is a vertex in G), and finding the full-coverage subset of P equals the retrieval of the maximum independent set (MIS) of G. Each member of MIS is called a facet of P. We use SP to denote the selected subset of P using PTree. The precision and recall of the performance on data selection is formulated as Eqs. (1) and (2). The former equation can be explained as the redundancy-elimination ratio, while the latter is the facet-coverage ratio.

$$\begin{aligned}\text{Precision} &= \frac{\text{TP}}{\text{TP}+\text{FP}} \\ &= \frac{|\text{MIS}(\text{SP})|}{|\text{MIS}(\text{SP})| + (|\text{SP}| - |\text{MIS}(\text{SP})|)} \\ &= \frac{|\text{MIS}(\text{SP})|}{|\text{SP}|}\end{aligned} \tag{1}$$

$$\begin{aligned}\text{Recall} &= \frac{\text{TP}}{\text{TP}+\text{FN}} \\ &= \frac{|\text{MIS}(\text{SP})|}{|\text{MIS}(\text{SP})| + (|\text{MIS}(P)| - |\text{MIS}(\text{SP})|)} \\ &= \frac{|\text{MIS}(\text{SP})|}{|\text{MIS}(P)|}\end{aligned} \tag{2}$$

   - *Efficiency*. We compute the computation cost (*ComC*) as the number of pairwise matches when generating the PTree over a picture set. The flexibility of PTree on different picture stream sizes is tested as well. We also measure the performance difference on different layering strategies.
2. *Baseline and ground truth*. The content-based prioritization scheme (CAP) by PhotoNet [3], which aims to maximize event coverage, is used as the baseline. It is based on linear pairwise comparisons, while PTree uses hierarchical clustering. The computation overhead is thus different. To validate the selection approach, we should obtain the ground truth, i.e., the MIS of the picture set. Finding the MIS is an NP-hard problem, and we employ the method from [30] and use content similarity (by SIFT, similarity threshold $th\_s = 0.1$) to compute the MIS.
3. *Datasets*. We used two real-world datasets for our experiments. The first dataset was the FlierMeet [6] dataset. It is an MCS app that allows people to repost and share fliers distributed on urban surfaces. Thirty-eight students in our university were recruited to use the application to capture fliers on university campus, and over 2000 pictures were collected within 8 weeks. The second dataset refers to another kind of MCS app, used for social event sensing (SESense for short). The data was collected from an academic forum at our university. The attendees were asked to record interesting or important moments during the event, and a total of 155 pictures were collected. All the participants were rewarded in data contribution.

6.2 Performance measurement

To evaluate the performance, we tested the approach under different constraint settings. Two constraints were used: location and visual features. We used three pairs of constraint thresholds (*th_l, th_v*), namely (5.0,0.1), (5.5,0.1), and (6.0,0.1). Two layering approaches, LV ({(location, l_1), (visual, l_2)}) and VL, were used to validate the impact of layering. Furthermore, the performance difference of fastM and minM in clustering was also tested.

The results show that the selection performance was affected by constraint settings. The redundancy ratio was greatly reduced by PTree, while the facet-coverage ratio was maintained (precision and recall are around 90 and 85 % under different settings). As shown in Fig. 4, under the same setting, the performance of mimM had slight differences with fastM. However, its *ComC* was much higher. This reveals that we can choose fastM in clustering. Compared to the baseline approach CAP, PTree performs better for all metrics, i.e., it has better selection quality with low computational costs. There are two possible reasons.

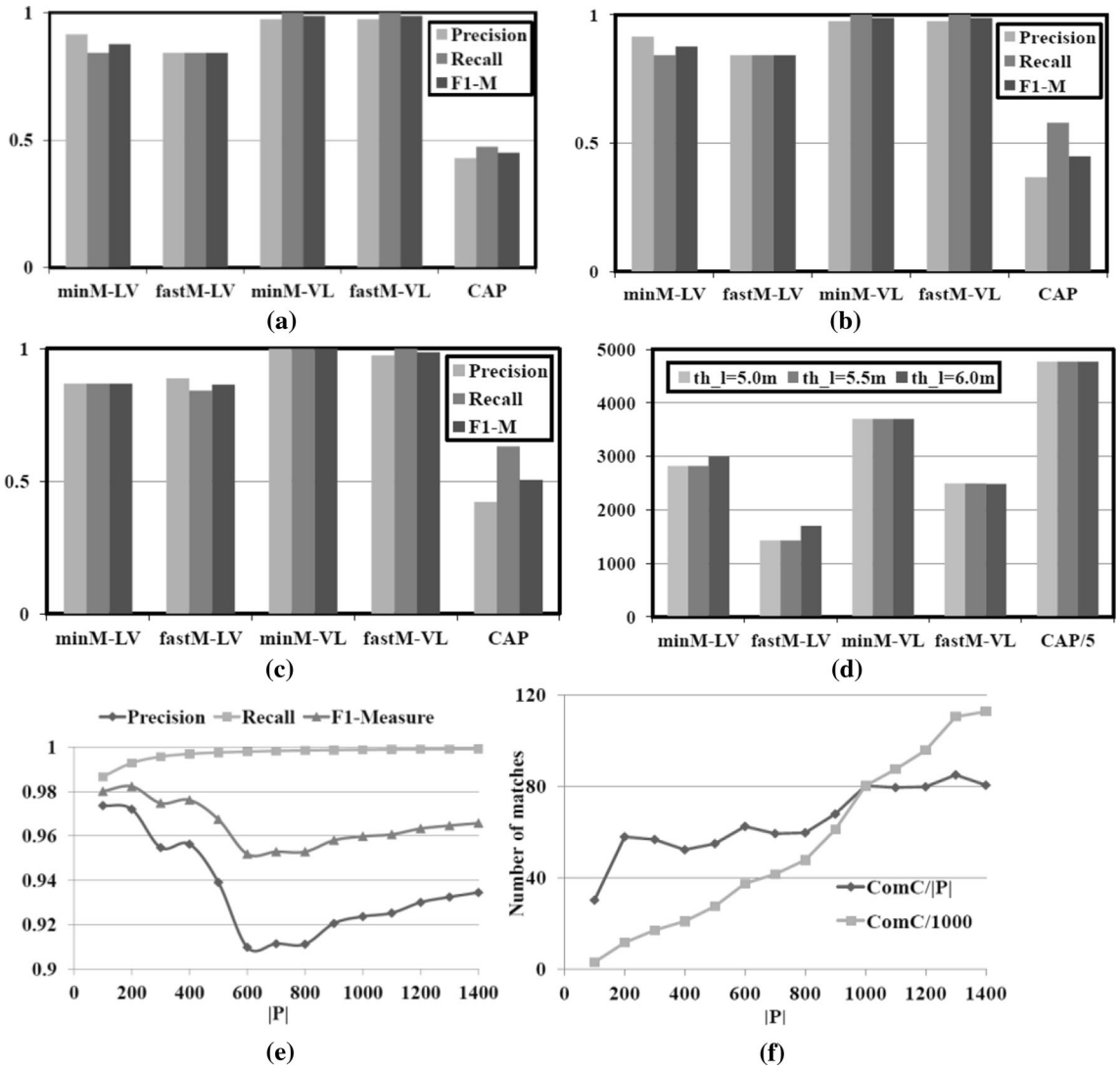

Fig. 4 Experiment results for clustering effectiveness, efficiency, and computation cost. **a** *th_l* = 5.0 m, *dataset* = *SESense*, **b** *th_l* = 5.5 m, *dataset* = *SESense*, **c** *th_l* = 6.0 m, *dataset* = *SESense*, **d** *ComC*, *dataset* = *SESense*. **e**, **f** *th_l* = 400 m, *th_t* = 4 days, *fastM*, *dataset* = FlierMeet

First, CAP is mainly based on content diversity, while the various semantic constraints are not considered. Second, the fusion of location and visual features at the same level also impacts its performance. In contrast, PTree achieves a higher performance by using hierarchical decision making, and the affect of each feature is addressed.

We also tested the flexibility of PTree over the FlierMeet dataset, using the data streams sized from 100 to 1400. As shown in Fig. 4e, the effectiveness of PTree on redundancy elimination changes quite slowly as the dataset size rises. It is not surprising that almost the whole computation cost increases linearly with the dataset size. However, we found that the average computation cost (*ComC*/|*P*|) of each picture changed much more slowly (it increases two times with a ten times increase in the dataset size). The results indicate that our approach is flexible, considering that most tasks have tens to thousands of user contributions.

Though VL is more effective than LV, its computation cost is considerably higher (2578 vs. 1449). We further compared the trees generated to explain the reason. Two PTrees with the SESense dataset and the same constraints, but different layer mappings (VL and LV), are shown in Fig. 5. The left tree has a bigger fan-out degree in the top layers compared with the right one, thus resulting in a higher computation cost. The experiment proves that setting proper layers can reduce *ComC*, and clustering the picture stream with the *A*-shape is more efficient.

6.3 A case study of PicPick data selection

To illustrate the effectiveness of our data selection framework, we made a case study based on the SESense dataset. The raw pictures contributed by five attendees (A1–A5) are shown in upper of Fig. 6 (sequenced along a timeline), and the pictures after data selection with fastM-LV are shown in the bottom of Fig. 6. Around 68 % pictures (13 over 19) were finally selected. The result indicates that PicPick selected pictures from the attendees at different shooting angles (the five attendees were located at different places in the forum venue: *front-left*, *front-middle*, *back-left*, etc.). Therefore, we can have multifaceted visual information (at the semantic-diversity level) about the event. Further, visually similar pictures taken from the same direction/place were not selected to reduce redundancy. For example, the first two pictures from A2 were visually similar and only the first one was selected, while the four pictures contributed by A4 were not visually similar and all of them were selected.

6.4 Discussion

The above experiments demonstrate the effectiveness and flexibility of PTree. There are several limitations that need to be improved in the future.

1. *Task modeling*. Our task model allows people to pre-specify the task constraints for data selection. However, sometimes task providers cannot predict the distribution or the context of sensed objects, and they might set ambiguous or incorrect task constraints. Also, the characteristics of the picture stream may evolve over time, which may result in changing constraint settings. In order to maintain the sensing quality, a more complex and dynamic task model should be built.
2. *PTree performance*. As the *ComC* increases when a PTree grows and some micro-clusters might not get new items for a certain period of time, it is possible to remove static branches to save computing cost. When and how to make branch-cuttings needs to be further studied.

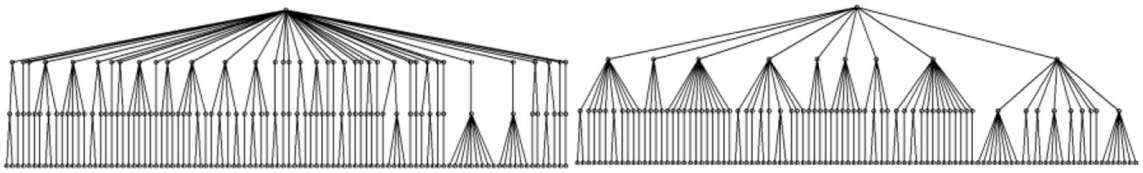

Fig. 5 PTree generation with two layer mappings: VL and LV ($th\_l = 5.5$ m, $th\_v = 0.1$, fastM)

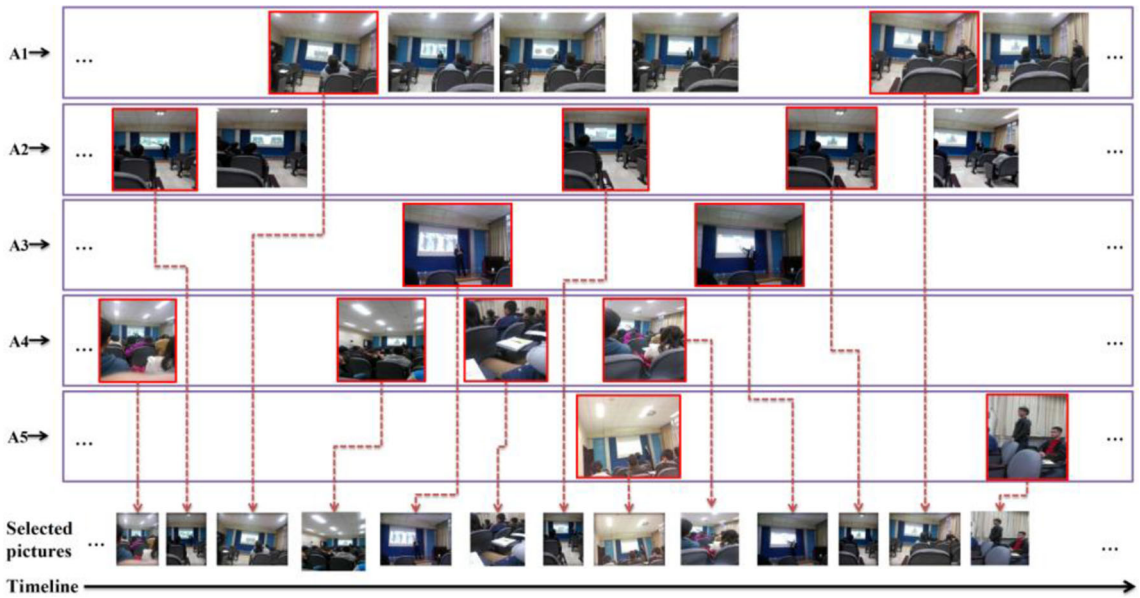

Fig. 6 Selection result to the SESense dataset (partial) using the LV layering strategy ($th\_l = 5.5$ m, $th\_v = 0.1$, fastM), A1–A5 represent the five attendees

3. *Quality of sensing*. In addition to data redundancy, data quality is another significant issue in crowd sensing. For example, participants may contribute low-quality pictures (e.g., pictures are blurred or taken under poor lighting environments) in MCP tasks. In PicPick, *time-priority* is used for picture selection. However, as we can cluster similar pictures into groups, later-coming pictures which have better quality than the previously selected ones (in the same group) can also be selected for full-size picture submission. In other words, an integrated *time-quality-priority* strategy can be used for picture selection. The quality of pictures can be measured based on its sensing contexts, as discussed in our previous work [6].
4. *Centralized versus cooperative*. In the current study, we used a centralized approach for picture selection, which may become a bottleneck when there are large-sale tasks in processing. Nevertheless, cooperation among peers was not considered. It is therefore important to leverage the local and opportunistic cooperation and data selection among peers to reduce the server burden. We leave this as future work.

7 Conclusion

We have presented our approach for optimized data selection and redundancy elimination in mobile crowd photography. A generic, task-driven MCP framework supports coverage-based data collection for varied MCP tasks. A PTree-based method is proposed to meet the multi-dimensional constraints by hierarchically and online clustering the picture stream. Experimental results indicate that PTree can successfully reduce data redundancy while maintaining coverage needs. Experiments on different-sized picture streams also prove the flexibility of our approach. We also use a case study about event sensing to demonstrate the usefulness of our framework.

In the future, we will first extend our task model to address evolvable and dynamic constraint settings. Second, PTree will be improved by using branch pruning techniques and quality-oriented data selection strategies. Third, we will investigate opportunistic collaboration of participants to achieve local data selection and optimized transmission.

Acknowledgments This work is partially supported by the National Basic Research Program of China (No. 2015CB352400), the National Natural Science Foundation of China (Nos. 61332005, 61373119), the Fundamental Research Funds for the Central Universities (3102015ZY095).

References

1. Guo B, Wang Z, Yu Z et al (2015) Mobile crowd sensing and computing: the review of an emerging human-powered sensing paradigm. ACM Comput Surv 48(1):1–31
2. Kim S, Robson C, Zimmerman T, Pierce J, Haber EM (2011) Creekwatch: pairing usefulness and usability for successful citizen science. In: Proceedings of ACM CHI'11, pp 2125–2134
3. Uddin MYS, Wang H, Saremi F et al (2011) PhotoNet: a similarity-aware picture delivery service for situation awareness. In: Proceedings of IEEE RTSS'11, pp 317–326
4. Wang Y, Hu W, Wu Y et al (2014) SmartPhoto: a resource-aware crowdsourcing approach for image sensing with smartphones. In: Proceedings of the ACM MobiHoc'14, pp 113–122
5. Goldman J, Shilton K, Burke J et al (2009) Participatory sensing: a citizen-powered approach to illuminating the patterns that shape our world. In: Foresight & Governance Project, White Paper, 2009, pp 1–15
6. Guo B, Chen H, Yu Z, Xie X, Huangfu S, Zhang D (2015) FlierMeet: a mobile crowdsensing system for cross-space public information reposting, tagging, and sharing. IEEE Trans Mobile Comput 14(10):2020–2033
7. Koukoumidis E, Peh L S, Martonosi MR (2011) SignalGuru: leveraging mobile phones for collaborative traffic signal schedule advisory. In: Proceeding of ACM MobiSys'11, pp 127–140
8. Rana RK et al. (2010) Ear-phone: an end-to-end participatory urban noise mapping system. In: Proceedings of the 9th ACM/IEEE international conference on information processing in sensor networks (IPSN'10), New York, USA, pp 105–116
9. Aly H, Basalamah A, Youssef M (2014) Map++: a crowd-sensing system for automatic map semantics identification. In: Proceedings of IEEE sensing, communication, and networking (SECON'14), pp 546–554
10. Gao R, Zhao M, Ye T et al (2014) Jigsaw: indoor floor plan reconstruction via mobile crowdsensing. In: Proceedings of ACM international conference on mobile computing and networking (MobiCom'14), pp 249–260
11. Ryong L, Shoko W, Kazutoshi S (2011) Discovery of unusual regional social activities using geo-tagged microblogs. World Wide Web 14(4):321–349
12. Ma H, Zhao D, Yuan P (2014) Opportunities in mobile crowd sensing. IEEE Commun Mag 52(8):29–35
13. Zhang D, Wang L, Xiong H et al (2014) 4W1H in mobile crowd sensing. IEEE Commun Mag 52(8):42–48
14. Guo B, Chen C, Yu Z et al (2015) Building human-machine intelligence in mobile crowd sensing. IEEE IT Prof 17(3):46–52
15. Guo B, Yu Z, Zhang D et al (2014) Cross-community sensing and mining. IEEE Commun Mag 52(8):144–152
16. Chen H, Karger DR (2006) Less is more: probabilistic models for retrieving fewer relevant documents. In: Proceeding of ACM SIGIR'06, pp 429–436
17. Yan T, Kumar V, Ganesan D (2010) Crowdsearch: exploiting crowds for accurate real-time image search on mobile phones. In: Proceeding of ACM MobiSys'10, San Francisco, USA, pp 77–90
18. Weinsberg U, Li Q, Taft N et al (2012) CARE: content aware redundancy elimination for challenged networks. In: Proceedings of the ACM HotNets'12, Redmond, USA, pp 127–132
19. Tuite K et al (2011) PhotoCity: training experts at large-scale image acquisition through a competitive game. In: Proceedings of ACM CHI'11, Vancouver, Canada, pp 1383–1392
20. Guha S, Meyerson A, Mishra N et al (2003) Clustering data streams: theory and practice. IEEE Trans Knowl Data Eng 15(3):515–528
21. Aggarwal CC, Han J, Wang J et al (2003) A framework for clustering evolving data streams. In: Proceedings of the 29th

international conference on Very large data bases. VLDB Endowment, Berlin, Germany, pp 81–92
22. Chen Y, Tu L (2007) Density-based clustering for real-time stream data. In: Proceeding of the ACM SIGKDD international conference on Knowledge discovery and data mining, San Jose, USA, pp 133–142
23. Gaber MM, Zaslavsky A, Krishnaswamy S (2005) Mining data streams: a review. ACM Sigmod Rec 34(2):18–26
24. Mizell D (2003) Using gravity to estimate accelerometer orientation. In: Proceeding of the international symposium on wearable computers (ISWC'03), Osaka, Japan, p 252
25. Muller E et al (2009) Relevant subspace clustering: mining the most interesting non-redundant concepts in high dimensional data. In: Proceeding of ICDM'09, Miami, USA, pp 377–386
26. Aggarwal CC et al (2003) A framework for clustering evolving data streams. In: Proceeding of the 29th international conference on Very large data bases. VLDB Endowment, Berlin, Germany, pp 81–92
27. Basak J, Krishnapuram R (2005) Interpretable hierarchical clustering by constructing an unsupervised decision tree. IEEE Trans Knowl Data Eng 17(1):121–132
28. Nan W, Guo B, Huangfu S, Yu Z, Chen H, Zhou X (2014) A cross-space, multi-interaction-based dynamic incentive mechanism for mobile crowd sensing. In: Proceeding IEEE international conference on ubiquitous intelligence and computing (UIC'14), Bali, Indonesia
29. Comer D, Sethi R (1976) Complexity of trie index construction. In: Proceeding of foundations of computer science, pp 197–207
30. Boppana R, Halldorsson MM (1992) Approximating maximum independent sets by excluding subgraphs. BIT Numer Math 32(2):180–196