CrowdPic: A Flexible Approach for Optimized Data Selection and Aggregation in Mobile Crowd Photographing Applications

Huihui Chen, Bin Guo, Senior Member, Zhiwen Yu, Senior Member, Liming Chen, Senior Member

Abstract—Mobile Crowd Photographing (MCP) is attracting an emerging area of interest for researchers as cameras of mobile devices are becoming an indispensable visual logging device in people’s everyday lives. In order to meet diverse MCP application constraints/requirements (e.g. where and when to sense, sampling frequency, single or multiple shooting angles) to sensing targets, a multi-facet task model should be defined as a generic MCP data collection framework. Furthermore, MCP collects pictures in a distributed way in which a large number of contributors upload pictures whenever and wherever it is suitable. This inevitably leads to evolving picture streams. This paper investigates the multi-constraint-driven data selection problem in MCP picture aggregation, and propose a pyramid-tree (PTree) model which can efficiently select an optimal subset from the evolving picture streams based on varied coverage needs of MCP tasks. By utilizing the PTree model in a generic MCP data collection framework called CrowdPic, we test and evaluate the effectiveness and efficiency of the proposed framework through both crowdsourcing-based and simulation-based experiments. Both the theoretical analysis and simulation results indicate that the PTree-based framework can effectively select a subset with high utility coverage and low redundancy ratio from the streaming data. The overall framework is also proved flexible and applicable to a wide range of MCP task scenarios.

Index Terms—Evolving data stream, maximum coverage, picture, mobile crowd photographing, pyramid tree.

1 INTRODUCTION

The increasing prevalence of smart devices, such as mobile phones, tablets, and wearable gadgets, and their inherent mobility, has led to the emergence and rapid adoption of a novel sensing paradigm, namely Mobile Crowd Sensing (MCS) [1]. MCS is to utilize the power of the public to accomplish specific sensing tasks without pre-deployed dedicated infrastructure. It can collect information of interest in remotely located physical environments from opportunistic and/or recruited participants who use their smart devices for sensing. In combination with the support of the cloud where data aggregation takes place MCS provides an effective and efficient approach to accomplishing some sensing tasks without requiring specifically tailored sensing infrastructures, in particular for applications like urban dynamics mining, public safety, traffic planning and environment monitoring.

MCS can be undertaken in different ways in terms of the modality of sensing, e.g. taking pictures, audio recording and GPS logging. Among them, Mobile Crowd Photographing (MCP) which utilizes the built-in cameras of smart phones to achieve large-scale visual sensing has become the dominant approach among various MCS systems. Previous research and applications, e.g. CreekWatch [2], GarbageWatch [3], PhotoNet [4], PhotoCity [5], WreckWatch [6], FlierMeet [7], and Mediascope [8], have indicated that MCP is useful and superior to traditional approaches to visual sensing, e.g. deployment of static cameras for monitoring.

Generic MCS platforms, such as McSense [9], Medusa [10], consider data collection for an MCS application/project as a sensing task, e.g. picture-taking tasks. But no generic MCS platform has paid much attention on the picture collection and no generic MCP platform has been proposed to the best of our knowledge. Our work, however, aims to build a generic framework to address the different requirements of MCP applications.

An MCP application is usually considered as a task in a generic MCP platform, and tasks differ in terms of their sensing targets and constraints, such as place, time, shooting angle, and so on. For example, WreckWatch [6] needs local pictures of a fatal car crash taken by standbys whenever the car crash is detected by the in-car cell phones. Creekwatch [2] collects global pictures combined with text tags of creeks all over the world to monitor pollution whenever the participant passes by the river. In addition, pictures with many different shooting angles will be helpful to reflect the situation of the accident for Wreckwatch, but one or few pictures of the garbage in the river with only one shooting angle is enough to learn the pollution status in Creekwatch.

Most of the existing MCP systems support only one or one type of specific tasks (e.g. river pollution monitoring [2], climate change sensing [11], congestion condition monitoring [12], disaster/event images collection [4], [8], [13], or posted flier reposting [7]). Our work, however, aims to build a generic framework to address the different requirements of MCP applications. The idea to build the generic framework for MCP is inspired by MTurk [10, 11], with the following merits. Firstly, it facilitates the rapid
specifications of MCP tasks with different constraints/needs without developing individual and proprietary systems to gather data. Secondly, it lowers the barrier for ordinary users to post MCP tasks and meets their diverse/personalized needs. Thirdly, it provides mobile users with a unique entrance to MCP tasks, which can simplify worker recruitment and task query/suggestion.

Despite the aforementioned benefits, there are several challenges to build a generic MCP data collection framework.

(1) Different MCP tasks have distinct needs and contextual constraints. For example, the project studying climate change [11] with pictures of trees in different seasons or months allow a changeable sampling interval (maybe two days when the tree sprouts or three months in winter) while a fire in a theater [14] needs to be reported with very short ones. Some tasks may need pictures about a sensing object from different shooting angles, e.g. Photocity [5], while others may need only one snapshot [7]. In order to build a generic MCP framework, we should make a thorough analysis of MCP concepts and constraints, and find a way to model different tasks.

(2) On-the-move pre-selection for evolving picture streams. In MCP tasks, the participants intentionally take the relevant pictures in a participatory manner according to predefined task requirements. The data are collected in a distributed manner and the data submitted by workers arrive at the backend server intermittently. These data constitute the evolving data stream. Since the later-coming duplicate data might be useless in the data sequence, data pre-selection should be conducted for the evolving picture streams to maintain the useful data and minimize the communication overhead [4], [8], [13]. For example, the data center should make a quick decision on whether the data is worth being uploaded according to the task constraints.

(3) Data selection with quality of sensing guarantee. Since some pictures can be discarded in the process of selection, the dynamic selection strategy (to the streaming data) still needs to ensure that the subset of pictures remained is with the maximum integrity compared to static data selection (i.e., selecting of data when all the data are already in the repository). Dataset integrity for MCP is domain-specific because the data center do not only leverage visual similarity to find visual-redundant pictures, but also put context of photographing in use to block utility-similar pictures arriving.

To address the above challenges, we studied the requirements, constraints, and quality demands of varied MCP applications, and proposed a generic data collection framework called CrowdPic for participatory picture collection. It can meet different MCP task requirements for high-quality and optimized data collection. Specifically, our contributions are as follows.

(1) Propose a generic picture collection framework for mobile crowd photographing, which is applicable to tasks of varying themes and constraints. Based on a multi-facet task model, the framework allows the task provider to specify the constraints on picture collection from multiple dimensions. In addition, it leverages a data selection method that can analyze and select an optimized subset of user-contributed data online from the original picture stream by means of the interaction between users and the backend server.

(2) Give a formal formulation of the optimal data selection problem for MCP. Since distributed participants might contribute redundant or irrelevant data, the selection problem for MCP is selecting diversified data to obtain the maximum coverage of the predefined constraints. It can be viewed as an extension of the vertex independent set covering problem.

(3) Develop a pyramid-tree model that can efficiently cluster the evolving picture stream and enable the near-optimized data selection. With the proposed pyramid tree structure and the associated tree generation rules (e.g., branching rules, layering rules), the framework can intelligently cluster the dynamic arriving pictures according to the task needs and constraints. The clustering result also facilitates the decision making process on picture acceptance/rejection and lowers the computation cost.

(4) Analyze and validate the performance of PTree-based selection method. A combination of theoretical analysis and crowdsourced dataset-based experiments are designed and conducted to evaluate the performance of the framework. The experimental results show that the PTree model achieves better tradeoff between efficiency and effectiveness in both theory and simulation analysis, and the framework can maintain the data integrity in the selection from the evolving picture stream with high efficiency and flexibility.

The rest of the paper is organized as follows. Section 2 outlines related work on MCP systems and picture collection/selection methods. Section 3 presents problems to be solved followed by the generic MCP framework focusing on the problems in Section 4. The solution based on a PTree-based clustering and data selection approaches is described in Section 5. We present the experimental results and the discussion of this work in Section 6, and conclude the paper and present the future work in Section 7.

2 Related work

2.1 MCS and MCP

Research on MCS has been undertaken in a wide range of application context. For example, Hu et al. [15] proposes a multidimensional context-aware social network architecture to provide a mobile ecosystem which enables context awareness in the development and utilization of MCS based applications. Zhang et al. [16] develops the 4W1H – a four-stage life cycle, to characterize the MCS process. Ma et al. [17] investigates the opportunistic characteristics of human mobility from the perspectives of both sensing and transmission, and presents approaches to collecting MCS data efficiently and effectively. Other related work includes sensing ability discovery [11], data aggregation [18], task management [19], and incentive mechanism [20]. In terms of implementation, MCS applications usually collect information from recruited participants, typically through application-specific Apps which are deployed on the participants’ smart phones, e.g. BikeNet [21], iMap [22], Ear-Phone [23], PEIR [24].

MCP has recently emerged as a dominant approach to MCS, which learns information and extracts knowledge
from a crowd-contributed picture set. Typical MCP achievements include monitoring the pollution of creeks [2], detecting traffic signals in urban environments [25], reposing and sharing the posters distributed in urban surfaces [7], extracting the prices of goods in the markets [26], gathering pictures of buildings for 3D city modeling [5], and reporting the scenes in emergency situations (e.g., disasters or fire) [4], [13], [14]. These MCP applications can be viewed as tasks with different sensing targets and sampling requests in a participatory collection manner.

In other words, participatory MCP tasks are application-specific and released by task providers or data requesters, who need pictures contributed by others. Different from existing studies, our work intends to build a generic data collection framework to meet diverse MCP application constraints/requirements.

### 2.2 Data Selection in MCP

Data contributed by a crowd of people can be redundant and of low quality in MCP data collection. To meet the systematic needs such as low transmission cost, low storage cost or high quality of sensing, it is an effective way to select a subset of high-value data from the evolving data stream.

In data pre-selection, a data selector is deployed between workers and the data center where the decision is made on whether a picture should be uploaded when its submission request is sent by a worker.

There have been works focused on the picture pre-selection, such as PhotoNet [4], SmartPhoto [27], MediaScope [8]. PhotoNet maximizes event coverage by maximizing the diversity of delivered photos through dropping pictures with little contributions to the diversity. Smartphoto uses a greedy algorithm to select photos with the most contribution to the total utility. MediaScope explore the selective, timely retrieval of media content from a collection of mobile devices. These applications use one or two features of the picture to measure the utility similarity. PhotoNet uses the location distance and the color histogram-based visual distance to evaluate the similarity degree. SmartPhoto focus on the field-of-view represented in angle of the camera lens and select limited photos that can cover $[0, 2\pi)$. MediaScope supports nearest-neighbor and other geometric queries on the feature space (e.g., clusters, spanners), and attempts to maximize the retrieval of relevant information. MediaScope incorporates a credit-assignment scheme to weight queries as well as differentiate query results by their importance.

Generally, the data selection is application-specific and sensing objectives are also personalized for most applications. Since targets sensed by MCP are different, varied requirements of MCP tasks should be considered. In Table 1, we illustrate some applications focused on different task constraints.

Aided by sensors in the smart phone, common picture features consist of visual, location, time, shooting angle. Current MCP applications consider one or more of them to eliminate redundant pictures. PhotoNet do not consider shooting angle but Smartphoto and PhotoCity focus on shooting angle to select the most useful pictures. Different from previous studies, this paper views each crowd photographing application as a task in a MCP platform and proposes the CrowdPic – a generic task-driven MCP framework which provides users with a multi-facet task model to address distinct task requirements/constraints. In particular, CrowdPic employs a novel pyramid-tree-based (PTree-based) algorithm to cluster the evolving picture stream and make efficient decisions on data selection.

### 3 System Modeling and Problem Formulation

#### 3.1 Four Stages of MCP tasks

We view each crowd photographing application as a task in a MCP platform. Figure 1 illustrates the generic four-stage process for a MCP task. This process involves three entities, namely task providers (i.e. data requesters), workers, and the backend server (e.g. the task manager and the data center).

![Fig. 1. Four Stages of MCP tasks.](image)

In the task initiation stage, task providers define their tasks with different requests or constraints and the task manager assigns these tasks to suitable workers. In the photographing and transmission stage, workers take and transmit pictures according to the task requirements to the backend server. As data is collected and uploaded by individual workers in a distributed manner, the backend server will receive pictures intermittently. Inevitably this will include low-quality or redundant pictures. As such, the data aggregation stage is responsible for grouping and selecting pictures over the evolving picture stream based on different task configurations. In the result handover stage, the task is completed, and the data repository will be made available for the task provider.

As low-quality or duplicate pictures can lead to unnecessary data traffic in MCP applications, one key research question in MCP is how to eliminate redundant pictures during picture aggregation. One approach to addressing this problem as described in [28] is that a thumbnail and related contextual information of a picture is first uploaded and analyzed in the server side based on the MCP application.
requirements. The analysis result will decide whether or not the full-size picture is needed. It is expected that such a decision should be made immediately once a thumbnail is uploaded, so that MCP based crowd sensing can be conducted efficiently. Based on the above analysis several requirements for building a general framework for participatory MCP data collection can be identified as follow.

- A multi-facet task model for varied MCP tasks specification (with different requests and constraints).
- Maintaining the quality of sensing when selecting data from the streaming pictures according to the task requirements.
- An efficient approach to deciding whether a full-size picture should be submitted or deleted.

### 3.2 MCP Task&Data Modeling

As discussed above, the development of a multi-facet task model is a fundamental requirement to build a generic MCP framework. To realize such a task model a formal data model for describing and representing pictures is also needed. Both are developed and described below.

#### 3.2.1 Multi-Facet Task Model

To adapt to various MCP tasks, a generic MCP framework needs to have a flexible and multi-facet task model which can define tasks with different types of demands and constraints. The model will cover the following two parts: (i) a task descriptor for workers to easily understand and execute a task, and (ii) the task specification module that allows task providers to define multi-dimensional constraints for picture collection and selection.

The **task descriptor** describes what should be captured, where the objects could be located, when and how the worker takes pictures and how many pictures should be taken. It is usually described in natural language and easily understood by any ordinary people. It tells workers what should be captured, where the objects will be or have been, when and how the worker takes pictures, and how many pictures should be taken. It is usually presented in natural language in such a way the descriptor can be easily understood by any ordinary people.

In order to collect highly relevant data the task specification module have quantified parameters to guide the picture collection process on the server. A 7-tuple, denoted as \( Tsk = \langle whn, whr, vlmn, cycl, grid, mview, imgSim \rangle \), is developed for the MCP task specification. \( whn \) is a time span defined by the start time \( TS \) and the end time \( TE \) of a task, and denoted as \( whn = (TS, TE) \). \( whr \) is a geographical area denoted by GPS points specified on the digital map (e.g. Google map) for performing the task. \( vlmn \) denotes the desired volume of the picture set. \( cycl \) is a time span, denoting the changing or refreshing cycle of the sensing target. \( grid \) is a geographical distance, within which the same target or similar targets might be seen. \( mview \) is a numerical value in \( [0, \pi) \), which denotes the multi-view photographing constraint with the angle of two shooting directions. Although the shooting angle of the camera lens is defined with a 3D vector, the angle difference of two shooting directions \( mview \) is denoted by an acute angle. \( imgSim \) is a method to detect similar images.

Overall these 7-tuple specifies multiple constraints in a picture collection task. The first three items characterizes the generic information about a task, and \( whn \) and \( whr \) will be sent to the mobile clients. The remaining items will be used by the backend server for picture selection.

In the following we use an example to illustrate the usage of the task model. Suppose that we want to know posters about discounting sales around a crowded area for two weeks before Christmas, an example task specification based on the above task model could be: \( [(20141210\sim20141225), (31.29, 121.47)\sim(31.09, 120.97), 2000, 5(day), 20(meter), \pi/2, (SIFT, high)] \). This can be interpreted as: this task needs to recruit workers to take about 2k pictures within the specified geographical area from Dec. 10, 2014 to Dec. 25, 2014. The geographical area is characterized by two GPS coordinates and can be defined on a digital map application. If two pictures are similar, only the early arriving picture will be selected and kept at the server side. The similarity is measured based on task constraints, i.e. 5 days, 20 meters, \( \pi/2 \), and high SIFT-based visual similarity degree, which imply that two pictures are similar if they are taken within 20 meters and 5 days, the difference of their shooting angles is less than \( \pi/2 \), and the visual similarity measured by SIFT features is at a high level.

#### 3.2.2 MCP Picture Record

Once a worker takes a picture, the mobile client saves the image file and records the associated context information. Each picture will be described by an MCP picture record (PR), which is modeled using a 9-tuple data structure \( PR = (tid, wid, pic, tStamp, tArr, loc, sAngle, light, acc) \).

The meaning of each items of the PR model is as follows. \( tid \) – the sensing task for which the picture is captured; \( wid \) – the worker who has taken this picture; \( pic \) – a full-size picture and its thumbnail saved by the mobile client App using different image resolution configurations; \( tStamp \) – the time point of a photographing. \( tArr \) – the time point that the picture arrives at the data center, which is null in the client and will be assigned in the server. \( sAngle \) – the shooting-angle of a picture defined by the three angle vectors, \( sAngle = (azimuth, pitch, roll) \) which can be obtained by the orientation sensor or calculated on the basis of observations of the accelerometer and the magnetometer [29]. \( loc \) – the location of the worker taking picture which is defined by \( loc = (latitude, longitude, errorRadius) \). \( light \) – the ambient light level observed by the light sensor. \( acc \) – 3D accelerometer readings at the moment of picture-taking.

Some associated context information of picture-taking will be utilized to match with task constraints for picture selection, such as \( tStamp \), \( sAngle \) and \( loc \). But both \( light \) and \( acc \) can be utilized for image quality evaluation, which is another topic discussed in [28].

#### 3.2.3 Evolving Picture Stream

The evolving picture stream consists of a series of picture records \( X = \{X_1, \ldots, X_k, \ldots\} \) for a certain task arriving at the data center at time points \( tArr_1, \ldots, tArr_k, \ldots \), and each \( X_i \in X \) is a multi-dimensional record denoted by \( X_i = (x_{i1}, \ldots, x_{id}) \). \( d \) denotes the dimension of \( X_i \). As pictures can be uploaded anytime (e.g. some nodes may upload data when Wi-Fi access points are available) before
the task deadline, there could be a delay between the time
the picture is taken and the time it arrives at the backend
server. As such, the arrival time $tArr_i$ of a picture record is
not always the same as its timestamp $tStamp_i$. This means
that $(tArr_i < tArr_j) \Rightarrow (tStamp_i < tStamp_j)$, and that
pictures can not be clustered only with static time windows.

3.3 Picture Selection with Max-coverage

As mentioned in Subsection 3.1, finding a high-quality
data subset for different tasks should be achieved by the
generic MCP data collection framework to improve user
experience. This high-quality data subset should meet the
multiple-coverage requirements of applications with little
data redundancy, which will be explained below.

3.3.1 Duplicate Picture Records

The utilities of redundant pictures have overlaps, so we use
a subset to represent the entire set. Y. Wang et al. [27] study
the online MaxUtility problem, and consider that the total
utility depends on how many aspects can be covered and
how they are covered. And we also consider that the quality
of sensing guarantee of a data subset can be measured with
its coverage $SubsCovr$.

**Definition 1 (Subset Coverage Ratio ($SubsCovr$)).** Given a
subset $X'$ of the entire picture set $X$, $X' \subseteq X$, $SubsCovr$
denotes the ratio of the utility of $X$ covered by that of $X'$
formulated by Eq. (1).

$$SubsCovr(X') = \frac{U(X')}{U(X)}$$  \hspace{1cm} (1)

Here, $U(X)$ refers to the utility of the picture set $X$.

Given two pictures $X_i.pic$ and $X_j.pic$, their similarity
degree $SimD$ is the overlap ratio of their utility, which can be calculated by

$$SimD(X_i, X_j) = \frac{U(\{X_i\}) + U(\{X_j\})}{U(\{X_i, X_j\})}$$

Then,

$$U(\{X_i, X_j\}) = \frac{U(\{X_i\}) + U(\{X_j\})}{1 + SimD(X_i, X_j)}$$

In general, items of a picture record are multi-
dimensional and heterogeneous, so it is application-specific
to find a numeral value of $SimD$. The traditional picture
similarity measurement is mainly based on visual features,
but it is not enough for the similarity measurement of the
generic MCP framework. Besides, for the MCP picture
collection, in order to make a timely decision for the picture
pre-selection on an evolving data stream, the similarity is
a qualitative value rather than a quantitative one. So we
use a logic express to denote the picture record similarity as follows.

**Definition 2 (Duplicate Picture Records).** Two records $X_i = (x_{i,1}, ..., x_{i,d})$ and $X_j = (x_{j,1}, ..., x_{j,d})$ is duplicate if and
only if $DP(X_i, X_j)$ in Eq. (2) is true.

$$DP(X_i, X_j) = \bigwedge_{m=1..d} sim(x_{i,m}, x_{j,m})$$  \hspace{1cm} (2)

Here, $sim(x_{i,m}, x_{j,m})$ is true if and only if $x_{i,m}$ is similar
with $x_{j,m}$. Each $sim(x_{i,m}, x_{j,m})$ is calculated with a distance
computation method ($d_{mthd_m}$) and a threshold ($d_{th_m}$) according to what $x_{i,m}$ actually is, e.g. image, text or location.

Once duplicate picture pairs are discovered, methods can be developed to find the high-quality subset by using
a graph and the maximum independent set.

3.3.2 Maximum Diversified Subset

Because selecting a high-quality subset equals to finding
a subset with max-coverage $SubsCovr$ in terms of subset
utility, we analyze the max-coverage optimization prob-
lem by means of an undirected and unweighted graph
$G = < X, DP >$. Next, we utilize the maximum vertex
independent set of a graph to analyze the max-coverage
optimization problem.

**Theorem 1.** The visual similarity relationship of pictures
have no transitivity.

**Proof 1.** $(p_i, \{J_m, ..., J_n\})$ denotes that objects $J_m, ..., J_n$
are captured in picture $p_i$. Given three pictures $p_1, p_2, p_3$ and four objects $\{J_1, J_2, J_3, J_4\}$, if the relationship between objects and pictures are $\{(p_1, \{J_1, J_2\}), (p_2, \{J_2, J_3\}), (p_3, \{J_3, J_4\})\}$, then
the similarity relationship might be $DP(p_1, p_2) = True$, $DP(p_2, p_3) = True$ and $DP(p_1, p_3) = False$. Therefore,

$$DP(p_1, p_2) = True \land DP(p_2, p_3) = True \Rightarrow DP(p_1, p_3) = True$$

If $DP(X_i, X_j) = Treee \Leftrightarrow SimD(X_i, X_j) = 1$, in order
to collect data with more coverage to the sensing objects, we
will choose $\{p_1, p_3\}$ rather than $\{p_2\}$ as the selected subset,
which is the motivation to find the maximum diversified
subset as the optimal selection result.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig2.png}
\caption{(a) Graph with a maximum vertex independent set \{a, c, e, g, i\} and a maximal vertex independent set \{b, d, f, h\}. (b) Graph with maximum vertex independent set \{a, c\} or \{a, f\} or \{b, d\} or \{b, f\} or \{c, d\}.}
\end{figure}

**Definition 3 (Maximum Independent Set (MIS)).** A subset
$S$ of the vertex set $V$ of a graph $G$ is called independent
if no two vertexes of $S$ are adjacent in $G$. $S \subseteq V$ is a
maximum independent set of $G$ if $G$ has no independent
set $S'$ with $|S'| > |S|$. A maximal independent set of $G$ is
an independent set that is not a proper subset of another
independent set of $G$ [30].

In order to select diverse photos that have the largest
$SubsCovr$ to maintain the quality of sensing, we define
the maximum diversified subset (MDS) $M$ of $X$ as follows
($M \subseteq X$).

$$M = \text{argmax}\{|M| : DP(X_i, X_j) = false, X_i, X_j \in M\}$$

As presented here, MDS and MIS have the same character-
istics, for example, for graphs in Figure 2, a maximum
independent set is also an MDS. Therefore, finding the MDS
is equal to find the maximum independent set of
$G = < X, DP >$. 

3.4 Efficiency of the Naive MDS-getting Method

Since finding MIS is NP-hard [31] and the length of the stream is scalable, we utilize the greedy-based algorithm to find the near-optimal resolution. Assume the MDS of an n-element data stream \( X(n) = \{X_1, ..., X_n\} \) is \( V_n \), \( V_n \subseteq X(n) \), then when \( X_{n+1} \) arrives, \( X(n+1) = X(n) \cup \{X_{n+1}\} \) and the MDS of \( X(n+1) \) is calculated with Eq. (3).

\[
V_{n+1} = \begin{cases} V_n \cup \{X_{n+1}\}, & v_{n+1} = 1 \\ V_n, & v_{n+1} = 0 \end{cases}
\]

\[
v_{n+1} = \begin{cases} 1, & \forall X_j \in V_n (DP(X_{n+1}, X_j) = False) \\ 0, & \exists X_j \in V_n (DP(X_{n+1}, X_j) = True) \end{cases}
\]

If the cost of computing one \( sim \) in Eq. (2) is 1, then the computation cost of getting \( v_n \) is \( d \times |V_{n-1}| \). The cost \( (ComC^N) \) of computing the MDS for an n-element picture stream \( X \) is

\[
ComC^N = d \times (|V_1| + |V_2| + ... + |V_{n-1}|)
\]

(4)

If once \( sim(x_i,m, x_j,m) = False \), no matter what \( sim(x_i,m+1, x_j,m+1) \) will be False, then

\[
\frac{n \times (n-1)}{2} \leq ComC^N \leq \frac{d \times n \times (n-1)}{2}
\]

Here, \( Mds(X) = X \) and \( |V_n| = n \).

In order to find an efficient way to determine whether \( X_{n+1} \) belongs to MDS or not when \( n \) increases, we introduce PTree-based clustering method and a PTree-based picture collection framework named CrowdPic in the following.

4 The CrowdPic Framework

Based on the requirements identified in Section 3 and the problems analyzed, we propose the CrowdPic framework. The workflow of picture collection under the framework will also be presented below.

4.1 The Architecture of CrowdPic

Based on the system requirements and challenges depicted in the introduction, we develop the CrowdPic framework, as shown in Figure 3. CrowdPic mainly addresses the optimal picture selection during two MCP stages, i.e., photographing and transmission, and data aggregation.

As mentions in Subsection 3.2.1, the sensing task contains a readable task descriptor and a set of task constraints. The task controller is responsible for assigning a task to a group of qualified workers according to the task needs [32].

The picture aggregator module collects and selects pictures from the picture stream in view of predefined task constraints. Supposing that a picture stream is composed of picture records for the same task, so there might be many different picture streams. Therefore, we need a picture selection method to satisfy a lot of different selection conditions extracted from task constraints of varied tasks. As shown in Figure 3, we use a Pyramid Tree (PTree)-based streaming dataset clustering method in CrowdPic. After being clustered, the data stream is divided into many micro-clusters and the MDS is composed of elements from each of these micro-clusters. During the clustering or the selection procedure, task constraints always work. As explained in Eq. (3), the old MDS is used to get the new MDS when a new picture record arrives. Overall, the picture aggregator is the crucial module of CrowdPic and the workflow of it is presented below.

4.2 The Workflow of the Picture Aggregator

The core component of the picture aggregator is the PTree-based clustering. As shown in Figure 4, PTree-based clustering contains two parameters, i.e., the layering mapping (LM) and the branching parameter (BP), and a procedure, i.e. new record matching with PTree. Detailed definitions of PTree and its parameters will be described in the next subsection. Here, we give an overview of the workflow.

A pyramid tree (PTree) is initially empty. When records arrive one by one, the PTree grows along with arrivals of these records who will also find positions in the PTree one by one. For the purpose of positioning in the PTree, each record will match with the PTree based on two parameters, i.e. the layering mapping (LM) and the branching parameter (BP). According to the matching result, the picture record will have a position in the PTree, and then the PTree refreshes before next picture record arrives. Which picture record will be selected as elements of the MDS depends on where these picture records are in the PTree. Therefore, finding the position for a picture record is the most important function of the PTree-based clustering. In the following, we describe the general concepts and features of the pyramid tree model and use examples to explain its working mechanism.
4.3 Definition of a Pyramid Tree

**Definition 4 (Pyramid tree (PTree)).** PTree is a \((d + 2)\)-layer tree structure that is generated according to a data stream \(X\) if the element \(X_i\) has \(d\) items. The leaf node only exists in the bottom layer, so the node count of the \(n\)-th layer is no less than that of the \((n - 1)\)-th layer.

In a PTree, the root node (RN) is in the 0-th layer and the other layer is denoted with \(p\)-th layer \((1 \leq p \leq d + 1)\). A non-leaf node (NLN) and a leaf node (LN) can be defined as a 3-tuple structure, i.e. \((no, id, cntr)\), and a 4-tuple structure, i.e. \((no, id, pr, accPro)\), respectively. \(no\) is the serial number of a node in its siblings \((no \geq 1)\). \(id\) is the identifier of a node, denoted with the path from the root node to itself, which is composed of a sequence of \(no\) s of nodes in the path. \(cntr\) is the center of an NLN and is calculated based on its micro-cluster. \(pr\) refers to a picture record. \(accPro\) is the probability of \(pr\) being accepted and implies whether \(pr\) will be selected or not.

**Definition 5 (NLN's Micro-cluster).** A micro-cluster of NLN \(N_j\), written \(MC(N_j)\), is composed of several picture records and can be found with

\[
MC(N_j) = \{N_m, pr : N_m \in CLN(N_j)\}
\]

Here \(CLN(N_j)\) denotes all offspring LNs of \(N_j\), and \(|MC(N_j)| = |CLN(N_j)|\).

If \(N_j\) is in the \(l\)-th layer, then \(MC(N_j)\) is considered as a \(l\)-th micro-cluster, and all \(l\)-th micro-clusters have no common elements with others.

Given a picture stream \(X = \{X_1, \ldots, X_{12}\}\), and \(X_i = (x_{i,1}, x_{i,2}, x_{i,3})\), the PTree in Figure 5 might be generated.

For the convenience of explanation, \(id\) is used as a node’s subscript in this paper, such as \(N_i.id = "1"\) and \(N_{11}.id = "1, 1, 1"\). \(N_{11}\) is the simplified expression of \(N_{1,1,1}\), so \(1 \leq no < 10\) is a limitation to this paper’s examples. As a result, the subset \(\{X_1, X_3, X_5\}\) is a 3-th micro-cluster of \(N_{11}\) and the subset \(\{X_1, X_2, X_3, X_4, X_5, X_6\}\) is a 2-th micro-cluster of node \(N_{11}\). Through this example, we show that the picture stream can be divided into micro-clusters based on the PTree’s structure, then we will introduce the PTree generation method next.

**Definition 6 (Feature name set (\(F\))).** The feature name set \(F\) is composed of names of all features that a picture record has, for example, \(F = \{\text{timestamp}, \text{location}\}\). Each item of a picture record is considered as a feature. If given \(X_i = \{x_{i,j} : 1 \leq j \leq d\}\), then \(F = \{f_j : 1 \leq j \leq d\}\).

**Definition 7 (Branching Parameter (\(BP\))).** A branching parameter is a \(2\)-tuple \((d_{\text{mthd}}, d_{\text{th}})\), composed of a method to calculate the distance \((d_{\text{mthd}})\) and a distance threshold \((d_{\text{th}})\) for clustering. Each feature of the picture record has a corresponding branching parameter, then the branching parameter of a PTree is \(BP = \{BP_i : 1 \leq i \leq d\}\), here \(BP_i = \{d_{\text{mthd}}, d_{\text{th},i}\}\) denotes the BP of a feature \(f_i\). \(f_j \in F\) and \(|BP| = |F|\).

\(BP\) is used to judge the similarity and is set according to the corresponding feature. \(d_{\text{ths}}\) are set based on task constraints. For example, if \(f_j\) refers to geography location expressed by a GPS coordinate, then \(d_{\text{mthd},i}\) can be the Euclidean distance and \(d_{\text{th},c}\) can be 30 meters if grid of Tsk is 30 meters.

**Definition 8 (Layer set (\(L\))).** The layer set \(L\) contains PTree layers except the 0-th layer and the bottom layer, written \(L = \{l_j : 1 \leq j \leq d\}\), and \(|L| = |F|\).

**Definition 9 (Layering Mapping (\(LM\))).** Layering mapping \(LM\) refers to the one-to-one mapping between \(F\) and \(L\), written \(LM : L \rightarrow F\), then \((l_i, f_j) \in LM\) and \(1 \leq i, j \leq d, l_i \in L, f_j \in F\).

**Definition 10 (Distance between a picture record and an NLN (\(Dis\))).** The distance between a picture record \(X_i\) and an NLN \(N_j\) in the \(p\)-th layer is written \(Dis(X_i, N_j)\).

If \((l_p, f_s) \in LM\), then \(Dis(X_i, N_j)\) is actually the distance between \(x_{i,s}\) and \(cntr_j\) calculated with the method \(d_{\text{mthd},p}\).

**Definition 11 (Matched NLN (\(matNLN\))).** A matched NLN of a picture record \(X_i\) in the \(p\)-th layer refers to a special NLN \(N_j\) satisfying a condition \(Dis(X_i, N_j) \leq d_{\text{th},p}\).

Any NLN satisfies the condition will be a candidate matNLN, but only one NLN can be selected as the matNLN for a picture record in one layer.

Because there might be more than one candidate matNLN in one layer, a matNLN selection method is required to select the unique matNLN for any picture record. The fast-match method (FastM) and minimum-match method (MinM) are two matNLN selection methods (MSN) used in this paper, \(\text{MSN} = \{\text{FastM}, \text{MinM}\}\).

If FastM is used, the first candidate matNLN will be selected as the only matNLN, and if MinM is used, the nearest one, i.e. \(Dis\) is minimum, will be selected. FastM is the default MSM in this paper.

4.4 PTree Generation

4.4.1 The Definition of Generation Parameters

In order to generate a PTree with a picture stream \(X = \{X_1, X_2, \ldots, X_n, \ldots\}\), here \(X_i = \{x_{i,j} : 1 \leq j \leq d\}\), we define some parameters as follows.

- \(X_i\) searches its matNLNs in turn from the 1-th layer to the \(d\)-th layer of a \((d + 2)\)-layer PTree.
- If the matNLN \(N_j\) of the picture record \(X_i\) in the \(p\)-th layer \((1 \leq p \leq d)\) is found, then (i) If \(p = d\), then a
new branch from NLN \( N_j \) to the bottom layer will be created. (ii) If \( p < d \), then the matNLN \( N_k \) in the \((p+1)\)-th layer will be searched in \( N_j \)'s child nodes.

- If the matNLN of the picture record \( X_i \) in the \( p \)-th layer \((1 \leq p \leq d)\) is not found, then (i) If \( p = 1 \), then a new branch from the root node to the bottom layer will be created. (ii) If \( p > 1 \), then a new branch from \( X_j \)'s matNLN in the \((p-1)\)-layer to the bottom layer will be created.

The new created branch is \( B = \{nl_{n1}, nl_{n2}, ..., nl_{nt}, ln: t \geq 1\} \) or \( B = \{ln\} \), here \( nl_{ni} \) is an NLN and \( ln \) is an LN. \( no \) of each element is a serial number in its sibling, and \( ln.pr = X_i \). The center \( cntr \) of NLNs and the acceptance probability \( accPro \) of NLNs will be introduced in the following subsections.

4.4.3 LM-based Distance Calculation for PTree Generation

As introduced above, finding matNLN for a picture record is the key step for generating PTree, and \( Dis \) is used to find candidate matNLNs. Because \( cntr \) of an NLN is used to calculate \( Dis \), next we will explain how to get \( cntr \).

According to Definition 9 and Definition 5, each layer is mapped with a feature of the picture record and each NLN has a unique micro-cluster, so each NLN has a set to contain items of picture records in its micro-cluster, written \( IC\). IC of an NLN \( N_j \) in the \( k \)-th layer can be obtained as follows.

\[
IC(N_j) = \{x_{i,t}: \forall X_i \in MC(N_j) \land (l_k, f_i) \in LM\} \tag{5}
\]

Here \( MC(N_j) \) is the micro-cluster of \( N_j \).

The relationship among a PTree, layering mapping, IC, and picture records are shown in Figure 6, by definition, \( IC(N_{j1}) = \{x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}\} \), \( IC(N_{j11}) = \{x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}\} \), \( IC(N_{j111}) = \{x_{i,2}, x_{i,2}, x_{i,3}\} \), and \( IC(N_{j112}) = \{x_{i,3}\} \).

For the PTree in Figure 6(a), if FaC is used, \( cntr \) of each NLN is \( N_{j1}.cntr = x_{1,1}, N_{j11}.cntr = x_{1,3}, N_{j111}.cntr = x_{1,2} \), and \( N_{j112}.cntr = x_{3,3} \). Conversely, if LaC is used, \( N_{j1}.cntr = x_{3,3}, N_{j11}.cntr = x_{3,3}, N_{j111}.cntr = x_{2,2} \) and \( N_{j112}.cntr = x_{9,3} \).

In the following discussion, the feature set \( F = \{\text{visual, location, timestamp, shooting angle}\} \). We use first-as-center (FaC) method for three layers, including the visual layer, the location layer, and the shooting angle layer. Because we cannot estimate the arrival time span of picture records, we use FaC and LaC together for the timestamp layer. In this case a 2-tuple \((ts, te)\) is used to denote \( cntr \) of an NLN \( N_j \) in the timestamp layer, \( ts = \inf(IC(N_j)) \) and \( te = \sup(IC(N_j)) \).

4.5 PTree-based Data Selection

Since the selection is based on the micro-cluster and the micro-cluster is dynamically created, in order to select all fresh pictures the first element of a micro-cluster will be selected. In a PTree, \( accPro \) of each picture record is saved in an LN. \( accPro \) of \( N_j (N_j.pr = X_i) \) can be calculated with Eq. (7).

\[
N_j.accPro = \begin{cases} 
1, & N_j.no = 1 \\
0, & N_j.no > 1 
\end{cases} \tag{7}
\]

The MDS of \( X \) selected based on a PTree, written as \( Mds(X) \), can be obtained by

\[
Mds(X) = \{N_j.pr : N_j.accPro = 1 \land N_j.pr \in X\}
\]

Then \( Mds(X) \) of the dataset in Figure 5 is \( \{X_1, X_2, X_4, X_7, X_8, X_9\} \).

5 THE PERFORMANCE OF PTree-BASED CLUSTERING

This section introduces factors which impact the performance of the PTree generation, and further analyzes the potential approaches to improve the efficiency of generating a PTree.

5.1 Efficiency Affected by PTree’s Shapes

Given different \( LM \) and \( BP \), the shape of the PTrees generated from the same picture stream might be different under different branching/layering processes. As shown in Fig. 7, there are three basic shapes of a PTree. An I-shape PTree has only one node (the root node or an NLN) that has a large number of child nodes and almost all descendant nodes of these child nodes have only one child node. The NLN of an inverted-T shape (iT-shape) PTree has barely any brother node and all LNs have the same parent NLN. Most NLNs in an A-shape PTree have more than one child node and their numbers of child nodes are slightly different.

During the PTree generation process, the computation cost of clustering each record into its micro-cluster is mainly on searching matNLNs. Assessing whether one NLN is a matNLN is equivalent to calculating \( sim \) once in Eq. (2). Therefore, we use the count of assessing NLNs for searching matNLNs as the computation cost of generating the PTree. Because the generation process is dynamic and the NLN count is increasing, it is difficult to estimate the computation.
cost \((\text{ComC})\) of generating a PTree. If \(\text{ComC}(n)\) denotes the computation cost of generating an \(n\)-LN PTree, then

\[
\text{AccNLN}(X_1) = 0 \quad (8) \\
\text{ComC}(1) = \text{AccNLN}(X_1) \quad (9) \\
\text{ComC}(n + 1) = \text{ComC}(n) + \text{AccNLN}(X_{n+1}) \quad (10)
\]

Here, \(\text{AccNLN}(X_{n+1})\) refers to the count of NLN accessed by \(X_{n+1}\) for searching matNLNs.

Assume that accessing NLNs for searching the matNLN is in an ascending order of the nos of NLNs and FastM (fast matching) is used. For the I-shape PTree shown in Figure 7(a), assume that the stream has \(n\) \(d\)-dimensional records and only the node in the \(t\)-th layer has multiple child nodes, then \(\text{AccNLN}(X_n) = t + n - 1\), and \(\text{ComC}(n) = t \ast (n - 1) + \frac{n \ast (n - 1)}{2}\).

Unlike an I-shape PTree, an iT-shape tree in Figure 7(b) has no branches except in the bottom layer, so \(\text{AccNLN}(X_i) = d \ast (1 < i \leq n)\) and \(\text{ComC}(n) = (n - 1) \ast d\).

The ComC of A-shape PTree is more complex than the other two, so we select several complete \(m\)-branch tree generated with \(n\) picture records, here \(m = \lceil \log_{d+1} n \rceil \). The ComC of PTrees in Fig. 7 is shown in Table 2, here \(d = 2\) and \(t = 0\). Given \(n = 8\), \(n = 27\) and \(n = 64\), the A-shape PTree will be a complete binary tree, ternary tree and quadtree respectively. As shown in Figure 7(c), the A-shape PTree is generated with 16 picture records and \(d = 3\).

\[
\begin{align*}
|n| & |I\text{-shape PTree}| & |iT\text{-shape PTree}| & |A\text{-shape PTree}| \\
8 & 28 & 14 & 22 \\
27 & 351 & 52 & 126 \\
64 & 2016 & 126 & 273
\end{align*}
\]

Although the ComC of generating an A-shape PTree or generating an iT-shape PTree is smaller than generating an I-shape PTree, but the number of selected NLNs of an iT-shape PTree are too little to guarantee the quality of sensing, so the A-shape is an ideal shape for a PTree.

Obviously, \(\text{ComC}\) of an I-shape PTree is larger than the other two. However, assume \(X = \text{Mds}(X)\), i.e. each micro-cluster has one element, then \(\text{ComC}^N\) of the naive method in Eq. (4) is \(\frac{n \ast (n - 1)}{2} \leq \text{ComC}^N \leq \frac{d \ast n \ast (n - 1)}{2}\) and \(\text{ComC}'\) of an I-shape PTree is \(\frac{n \ast (n - 1)}{2} \leq \text{ComC}' \leq (d - 1) \ast (n - 1) + \frac{n \ast (n - 1)}{2}\), then

\[
\text{sup}(\text{ComC}^N) - \text{sup}(\text{ComC}') = \left(\frac{d \ast n \ast (n - 1)}{2} - (d - 1) \ast (n - 1) + \frac{n \ast (n - 1)}{2}\right)
\]

We can conclude that if \(d > 1\) and \(n > 2\), even an I-shape PTree might be faster than the naive method. Actually, the naive method is at most equivalent to the PTree-based method with a static \(LM\), so it can not control the computing efficiency through building an A-shape PTree. Therefore, PTree-based clustering is much more efficient than the naive method.

### 5.2 MatNLN Selection Methods

As proposed in Subsection 4.4.1, matNLN selection methods (MSM) are FastM and MinM. The position of a picture record in the PTree might be different if using different MSM. The difference between these two methods can be depicted with the example in Fig. 8, where the PTree grows along with the arrival of records \(X = \{X_1, \ldots, X_3\}\) and FaC is used as the setting center method. The distance between two records are denoted with a line and the large circle represents the distance threshold, then records \(X_1\) and \(X_4\) at the center denote \(\text{ctr}\) of the NLN \(N_1\) and \(N_2\) respectively. The MSM of the PTree\(_A\) is FastM, and that of the PTree\(_B\) is MinM.

Using different MSMs, the PTree will be different because some branchings are different. The two examples in Figure 8 explain the difference of the two selection methods. First, because \(X_3\) arrives earlier than the micro-cluster of \(N_2\) is created, although \(\text{Dis}(X_3, N_2) < \text{Dis}(X_3, N_1)\), no matter which MSM is used, \(X_3\) is a child node of \(N_1\). Second, \(\text{Dis}(X_6, N_1)\) and \(\text{Dis}(X_6, N_2)\) are all less than the distance threshold and \(\text{Dis}(X_6, N_1) > \text{Dis}(X_6, N_2)\), then if FastM is used, \(N_1\) will be the selected matNLN of \(X_6\) in Figure 8(a) because \(N_1\) is found as the candidate matNLN earlier than \(N_2\), and if MinM is used, \(N_2\) will be the selected matNLN of \(X_6\) in Figure 8(b) because \(\text{Dis}(X_6, N_2)\) is the less.

As FastM will stop the search of matNLN whenever one matNLN is found, and MinM will assess all sibling NLNs to find all candidate matNLNs, the \(\text{AccNLN}\) of a picture record using MinM method is larger than using FastM method. The same happens to \(\text{ComC}\). This paper has focused on the use of FastM for experiments.

### 5.3 Equivalency of Multiple Solutions

Sometimes, we might got more than one MDSes. As shown in Figure 2(b), there are five MDSes and each of them can
be the optimal selection. In order to analyze the similarity of selected subsets, given two different selection results $X_i^{sel}$ and $X_j^{sel}$, $SelSim$ in Eq. (14) denotes the similarity degree of their elements and $UtiSim$ in Eq. (15) denotes the similarity degree of their utilities.

$$SelSim(X_i^{sel}, X_j^{sel}) = \frac{|X_i^{sel} \cap X_j^{sel}|}{|X_i^{sel} \cup X_j^{sel}|}$$ (14)

$$V(X_i, X_j^{sel}) = \begin{cases} 1, & \exists X_k \in X_i^{sel}(DP(X_i, X_k) = True) \\ 0, & \forall X_k \in X_j^{sel}(DP(X_i, X_k) = False) \end{cases}$$

$$W(X_i^{sel}, X_j^{sel}) = \sum_{X_m \in X_i^{sel}} V(X_m, X_j^{sel})$$

$$UtiSim(X_i^{sel}, X_j^{sel}) = \frac{W(X_i^{sel}, X_j^{sel}) + W(X_j^{sel}, X_i^{sel})}{|X_i^{sel}| + |X_j^{sel}|}$$ (15)

In Figure 9 $X_1^{sel}, X_2^{sel}, X_3^{sel}$ and $X_4^{sel}$ are selected subsets of $X = \{X_1, ... , X_8\}$ and they are all independent sets. Edges of graph (a) and graph (b) are slightly different. By definition, $SelSim(X_1^{sel}, X_2^{sel}) = 0$, $UtiSim(X_1^{sel}, X_2^{sel}) = 1$, $SelSim(X_1^{sel}, X_3^{sel}) = 1/8$ and $UtiSim(X_1^{sel}, X_4^{sel}) = 8/9$. Though two selected subsets look like different, their utility might be close.

Fig. 9. Two entire picture sets and their two selected subsets.

In order to evaluate which selected subset is the best, $Covr$ calculated with Eq. (16) leverages the size of the MDS to measure the coverage of the subset.

$$Covr(X^{sel}, X) = \frac{|Mds(X^{sel})|}{|Mds(X)|}$$ (16)

By definition, the MDS of $X$ in Figure 9 (a) is $Mds(X) = \{X_1, X_2, X_3, X_4, X_5\}$, then $Covr(X^{sel}) = 1$ and $Covr(X^{X_2^{sel}}) = 3/5$. The MDS of $X$ in Figure 9 (b) is $Mds(X) = \{X_1, X_2, X_3, X_4, X_5, X_7\}$, then $Covr(X^{sel}) = 5/6$ and $Covr(X^{X_5^{sel}}) = 4/6$. $Covr$ is a indicator to measure the proximity of the selected subset to the optimal solution and reflects $SubsCovr$ of $X^{sel}$. Therefore, subset $X_1^{sel}$ and $X_3^{sel}$ are the optimal selection.

6 EXPERIMENT AND EVALUATION

6.1 Metrics

To evaluate the algorithm we use five basic metrics, i.e. $Covr$, $ComC$, $Redn$, $SelRt$, and $AComC$. The prior two metrics $Covr$ and $ComC$ have been presented earlier in Eq. (16) and Eq. (10), respectively. $Redn$ denotes the redundancy ratio of the selected dataset, as formulated in Eq. (17), $SelRt$ refers to the ratio of the selected picture subset in Eq. (18), and $AComC$ depicts the average computation cost for each picture record in Eq. (19).

$$Redn = \frac{|X^{sel}| - |Mds(X^{sel})|}{|X^{sel}|}$$ (17)

$$SelRt = \frac{|X^{sel}|}{|X|}$$ (18)

$$AComC = \frac{ComC}{|X|}$$ (19)

Here $X^{sel}$ denotes the subset of selected pictures, and the MDS (maximum diversified subset) of $X$ is defined by $Mds(X)$, $Mds(X) \subseteq X$, and the MDS of $X^{sel}$ is denoted as $Mds(X^{sel})$.

$Covr$, $Redn$, and $SelRt$ are used to evaluate the effeciveness of the algorithm. $Covr$ reflects the sensing coverage by the selected subset. $SelRt$ reflects the degree of saving the traffic and the storage. $Redn$ is a negative indicator and reflects the problem of the selection result. $ComC$ and $AComC$ are utilized to measure the efficiency.

6.2 Experiment Settings

6.2.1 Distance Calculation Method

In the process of generating a PTree, the methods $d_{mthd}$ of $BP$ for calculating distances are different in different layers. The methods used in our experiments and their corresponding features are (Visual, $Dis_{SIFT}$), (Visual, $Dis_{CH}$), (Location, $Dis_{geo}$), (Shooting Angle, $Dis_{angle}$), and (Timestamp, $Dis_{time}$). $Dis_{geo}$ denotes a geography distance calculation method, and Euclidean distance is used in this paper. $Dis_{SIFT}$ denotes the SIFT-based (Scale-Invariant Feature Transform based) [33] near-duplicate image matching method. $Dis_{CH}$ denotes a visual similarity method based on the color histogram and KL-divergence (Kullback-Leibler divergence) distance [4]. $Dis_{angle}$ denotes the shooting angle distance and it can be calculated in Eq. (20), where both $C_i$ and $C_j$ are vectors denoting camera lens directions calculated with $sAngle_i$ and $sAngle_j$, respectively. $Dis_{time}$ is the temporal distance between a timestamp $tStamp_i$ and an NLP’s center $cntr_i$ in the temporal layer, i.e. $cntr_i = (ts_i, te_i)$, and it can be calculated in Eq. (21), where $H$ is a constant to limit the time span from $ts_i$ to $te_i$ and adjust the branching volume in the timestamp layer.

$$Dis_{angle}(sAngle_i, sAngle_j) = acos\left(\frac{C_i \cdot C_j}{|C_i| \cdot |C_j|}\right)$$ (20)

$$Dis_{time}(tStamp_i, cntr_i) = max\left\{|te_i - tStamp_i|, |ts_i - tStamp_i|\right\}$$ (21)

6.2.2 Generating Data for Simulation

We have developed an mobile social network application FlierMeet [7] and 38 students were recruited to use the application. After 8 weeks, we collected over 2000 geo-tagged pictures from these students. A student photographing at a specific place considered as check-ins implies that he is capable of accomplishing sensing tasks at that place. Five places having plenty of check-ins are chosen for the simulation and the distribution of these check-ins is shown in Figure 10.

To evaluate the performance of the PTree-based selection in picture streams of different sizes, we simulate more data streams based on the original FlierMeet dataset. Each of them has a dataset $X$ and the visual
distance matrix $Dv$ of $X$. The distribution of the timestamp and location of $X$ meets the CDF in Figure 10. The 3D shooting angle of $x_i$ is randomly created with the range of $[0, 2\pi)$ for each dimension. The parameters ($SimuPara$) used for generating $X$ include $Tsim$, $Asim$, $Dsmin$, $|X|$, $TS$ and $TE$. $X_i=\{tStamp_i, sAngle_i, loc_i\}$ denotes the simulated picture record, and $Dv(i,j)$ is the visual distance between $x_i$ and $x_j$. If $|tStamp_i - tStamp_j| \leq Tsim$, $Dis_{angle}(sAngle_i, sAngle_j) \leq Asim$ and $Dis_{geo}(loc_i, loc_j) \leq Dsmin$, then $Dv(i,j) = -1$, if not, $Dv(i,j) = 0$. Here, the range of the visual similarity threshold is $(-1, 0)$.

Four features of the picture record, i.e. location $L$, image I, shooting angle $A$, and timestamp $T$ are adopted in the evaluation. We use the permutations of these features to generate $LM$s, such as $LM=\{TLVA\}$. Here, “TLVA” is a simplified form of $\{(l_1, timestamp), (l_2, location), (l_3, visual), (l_4, shooting angle)\}$.

### 6.3 Simulation-Based Evaluation

#### 6.3.1 Performance Evaluation on PTree Shape

For the same picture stream, if PTrees are generated with different $LM$s or different $BP$s, their shape might be different. In order to observe the impact brought by the PTree shape to the generation performance, we select representative $LM$s. The data stream simulation parameter $SimuPara_1 = \{TS = 7am, TE = 11pm, Tsim = 20(minute), Asim = \pi/6, Dsim = 20(meter)\}$ and PTree branching parameter $BP_1 = \{(Dis_{geo}, 40(meter)), (Dis_{time}, (H = 2, 10(minute))), (Dis_{angle}, \pi/4)\}$ are used.

The experimental result shows that $ComC$ is related to the PTree shape which is in turn determined by $LM$ and $BP$. As shown in Figure 11, eight PTrees are generated with the same dataset, the same $BP$ while different $LM$s. We have hypothesized that selection results will be different when PTrees are different. The evaluations of selection results in Figure 11 are shown in Table 3. The $SelSim$s of selection results are shown in Table 4. Although $SelSim$s are different, we find that $UtiSim$ of each pair of results are 100%. The experimental result proves that the selection subset will be different if $LM$s are different. In order to decrease $ComC$, it is obvious that nodes in the upper layer should not have high fan-out degree. Therefore, setting proper $LM$ can reduce $ComC$ when $BP$ is static, and clustering the picture stream with an A-shape PTree is highly efficient.

In this subsection, we evaluate subsets selected from the same dataset and these subsets are different because they are extracted from different PTrees generated with different $LM$s. Because we select a large timespan to create simulation dataset, the data distribution is sparse and the dataset has few redundant data. Therefore, most $SelRt$s are very high in Table 3. In the following, we will use some simulated datasets with different sizes and densities to evaluate the performance.

#### 6.3.2 Performance Evaluation on Stability of the PTree

In order to evaluate the flexibility of the PTree-based selection algorithm when $|X|$ increases we choose two $LM$s, $LM_d = \{LTAV\}$ and $LM_f = \{TAVL\}$. The experiment parameters are still $SimuPara_1$ and $BP_1$ configured in Subsection 6.3.1.

The effectiveness of the PTree-based selection is shown in Figure 12 and Figure 14. The experimental result shows that the effectiveness of using $LM_d$ or $LM_f$ are similar. When $|X|$ increases, the values of $SelRt$ and $Mds(X)/|X|$ decrease, while the value of $Redn$ and $Covr$ remain nearly steady. $Covr$ reaches 100%, and $Redn$ remains about 20%.
The efficiency evaluation in Figure 13 proves that the computation cost is related to the PTree shape because ComC of $L_Md$ is much lower than that of $L_Mf$. Because ComC is linear with $|X|$, and the value of $|AComC|$ is always small, the selection efficiency could be very high. In other word, regardless of how long the stream is, each picture record will receive the feedback from the data center efficiently.

6.3.3 The Impacts of the Data Density on Picture Selection

The experimental result in Subsection 6.3.1 shows that choosing proper LM will significantly decrease ComC. Another finding in Subsection 6.3.2 is that SelRt decreases when $|X|$ increases. The experience tells us that when $|X|$ increases, the density will also increase, then more redundancy data emerge. The dataset in Figure 11 is very sparse which can be shown through the sizes of micro-clusters, so we simulate a dense dataset with SimuPara2, the same $BP$ and different LMs. Here $|X|=93$. From left to right and top to bottom, LMs of them are $L_Ma, L_Mb, ..., L_Mf$.

Fig. 15. Shapes of the PTrees generated with the same dataset simulated with SimuPara2, the same $BP$ and different LMs. Here $|X|=93$. From left to right and top to bottom, LMs of them are $L_Ma, L_Mb, ..., L_Mf$.

<table>
<thead>
<tr>
<th>LM</th>
<th>SelRt</th>
<th>Covr</th>
<th>ComC</th>
<th>AComC</th>
<th>Redn</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_Ma$ = &quot;VLTA&quot;</td>
<td>71.4%</td>
<td>100.0%</td>
<td>2,374</td>
<td>26.0</td>
<td>18.4%</td>
</tr>
<tr>
<td>$L_Mb$ = &quot;VTAL&quot;</td>
<td>71.4%</td>
<td>100.0%</td>
<td>2,362</td>
<td>25.9</td>
<td>18.4%</td>
</tr>
<tr>
<td>$L_MC$ = &quot;LVTAM&quot;</td>
<td>78.0%</td>
<td>100.0%</td>
<td>1,112</td>
<td>12.2</td>
<td>25.3%</td>
</tr>
<tr>
<td>$L_Mc$ = &quot;LTAV&quot;</td>
<td>76.9%</td>
<td>98.1%</td>
<td>623</td>
<td>6.8</td>
<td>25.7%</td>
</tr>
<tr>
<td>$L_Me$ = &quot;TVLA&quot;</td>
<td>78.0%</td>
<td>100.0%</td>
<td>1,181</td>
<td>12.9</td>
<td>25.3%</td>
</tr>
<tr>
<td>$L_Mf$ = &quot;TAVL&quot;</td>
<td>76.9%</td>
<td>100.0%</td>
<td>809</td>
<td>8.8</td>
<td>24.2%</td>
</tr>
<tr>
<td>$L_Mh$ = &quot;ALVT&quot;</td>
<td>70.3%</td>
<td>100.0%</td>
<td>659</td>
<td>7.2</td>
<td>17.1%</td>
</tr>
<tr>
<td>$L_Mb$ = &quot;ATLV&quot;</td>
<td>72.5%</td>
<td>100.0%</td>
<td>612</td>
<td>6.7</td>
<td>19.7%</td>
</tr>
</tbody>
</table>

Table 5. Evaluation of PTree generation when using different LMs and the dataset is simulated with SimuPara2.

Table 6. Selection similarity degree (SelSim) of Different LMs on the dataset generated with SimuPara2.

As shown in Figure 10, because the check-in distribution is different in different time spans at the same place and is also different in the same time span at different places, the spatial and temporal distribution of the entire picture set is uneven. To compare the impacts of the sensing data distributions on the selection performance we select three two-hour time spans $TS1 = (7am \sim 9am)$, $TS2 = (3pm \sim 5pm)$, and $TS3 = (5pm \sim 7pm)$ for simulation parameters. We compare the experimental results in Figure 16. The results show that $|Mds(X^{sel})|$ and SelRt is tightly related to the temporal and spatial density distribution of picture records, Redn is loosely related to it, and $AComC$ and Covr are almost not related to it at all.

6.3.4 Findings

From the simulation-based evaluation, findings with regard to the efficiency and effectiveness of the PTree-based data selection can be drawn as follows.

1. Although the computation cost of selection increases with the length of the data stream (i.e. $|X|$), the average computation cost is nearly steady or increases slowly.

2. The size of the selected dataset and the size of the maximal diversified subset have a positive relationship, which means that plenty of data need be selected to obtain maximal coverage of the selected subset. The coverage and redundancy of the selected dataset are also nearly steady.
when the data distribution reaches saturation (means plenty of data are in all time slots, geography grids, and shooting angle blocks).

(3) The efficiency can be assured if branching parameters and layering mapping are properly set to obtain an A-shape PTree.

These prove that our method has good flexibility to cluster the data stream when the length of the stream is generally unknown.

6.4 PTree-based Selection for Real Applications
CrowdPic is a generic framework for MCP which can be applied to many application scenarios. In the following, we discuss how to use the PTree-based selection method based on some published applications shown in the Table, such as PhotoNet, SmartPhoto, PhotoCity.

PhotoNet uses three factors, including time, location and visual feature. Pictures taken in two geographically dispersed locations (say, 1km apart) or at significantly different time points (say, 6 hours apart) are “dissimilar” (high spatio-temporal distance). When pictures are taken in closer spatio-temporal spaces, their similarity is further decided by the distance in the image-feature space [4]. We can use $BP = \{\{\text{Dis}_{\text{geo}},1(\text{km})\},\{\text{Dis}_{\text{time}},(H = 1,6(\text{hour}))\},\{\text{Dis}_{\text{CH}},0.5\}\}$, $LM = "\text{TLV}"$ or $LM = "\text{LTV}"$ to meet the selection requirement of PhotoNet. Since the threshold of the visual similarity is not given, 0.5 is inferred from experiments in [4].

Smartphoto selects a certain number of picture from a entire picture set, and it uses a greedy selection method based on the difference of the shooting angle. PTree can not fully simulate the method of Smartphoto, but if it selects $c$ pictures, then $BP = \{\text{Dis}_{\text{angle}},2 * \pi/c\}$ and $LM = "A"$ can meet the requirement.

PhotoCity collects pictures of buildings in the city for 3D reconstruction. To meet the 3D modeling requirement, the system acquires dense distribution pictures of the same building, then the parameter for this task could be defined as $BP = \{\{\text{Dis}_{\text{geo}},20(\text{meter})\},\{\text{Dis}_{\text{SIFT}},-0.5\},\{\text{Dis}_{\text{angle}},\pi/10\}\}$, $LM = "\text{LVA}"$.

6.5 Discussion
Initial results show that CrowdPic is practical and promising. Though, there are still some limitations to be improved in the future work, as discussed below.

Dealing with Ambiguous Task Constraints. Task providers sometimes cannot predict the distribution and the context of sensed targets, so they might set ambiguous or wrong task constraints. In order to maintain the quality of sensing, more complex and refined task model is needed and heuristic task-creating with guidelines will be also helpful for task providers to create tasks.

Pruning. Since the computation cost increases when a PTree grows and some micro-clusters might not get new items for a certain period and are out of date, it is possible to remove those static branches to save computation cost. When and how to make branch-cuttings are also issues to be focused.

PTree-based Image Searching. The image searching is widely used in many applications. PTree-based image clustering can also be utilized to search images. The most similar work like us is B-tree index for database. Many context information of picture-taking are utilized to construct layers of the PTree in this paper, so a lot of relative information of the images for image searching application need be selected to generate the PTree, which is the problem to be resolved for the PTree-based image searching application.

7 Conclusion and the Future Work
Selecting highly-relevant data from an evolving picture stream is a fundamental problem for mobile crowd photography (MCP). In this paper, we have introduced a generic task-driven MCP framework that supports optimal data selection for varied MCP tasks. A pyramid tree-based model is developed to efficiently cluster steaming pictures and its adjustable generation parameters meet the multiple constraints derived from different MCP tasks. Evaluation results have validated the effectiveness (in terms of sensing coverage and redundancy), efficiency and flexibility of our method. Based on the algorithm and the findings in this paper, our future works are as follows. First, we will pay attention to heuristic task-creating and task-accomplishing with guidelines to promote the data quality in the data sampling stage. Second, we will use dynamic layer mappings to keep a PTree in A-shape to obtain the high efficiency.
ACKNOWLEDGMENTS
This work is partially supported by the National Basic Research Program of China 973 (No. 2015CB352400), the National Natural Science Foundation of China (No. 61332005, 61373119, 61222209), the Scientific and Technology New Star of Shaanxi Province (2014KJXX-39).

REFERENCES