F-SED: Feature-Centric Social Event Detection

Elio Mansour, Gilbert Tekli, Philippe Arnould, Richard Chbeir, Yudith Cardinale

F-SED: Feature-centric Social Event Detection

Elio MANSOUR[1], Gilbert TEKLI[2], Philippe ARNOULD[1], Richard CHBEIR[1], and Yudith CARDINALE[3]

[1] UNIV PAU & PAYS ADOUR, LIUPPA, EA3000, FRANCE
{elio.mansour, philippe.arnould, richard.chbeir} @univ-pau.fr
[2] University of Balamand, Lebanon
gilberttekli@hotmail.com
[3] Dept. de Computación, Universidad Simón Bolívar, Venezuela
ycardinale@usb.ve

Abstract. In the context of social media, existent works offer social-event-based organization of multimedia objects (e.g., photos, videos) by mainly considering spatio-temporal data, while neglecting other user-related information (e.g., people, user interests). In this paper we propose an automated, extensible, and incremental Feature-centric Social Event Detection (F-SED) approach, based on Formal Concept Analysis (FCA), to organize shared multimedia objects on social media platforms and sharing applications. F-SED simultaneously considers various event features (e.g., temporal, geographical, social (user related)), and uses the latter to detect different feature-centric events (e.g., user-centric, location-centric). Our experimental results show that detection accuracy is improved when, besides spatio-temporal information, other features, such as social, are considered. We also show that the performance of our prototype is quasi-linear in most cases.

Keywords: Social Event Detection, Social Networks, Semantic Clustering, Multimedia Sharing, Formal Concept Analysis.

1 Introduction

With the rapid evolution of social media, more users are now connected to collaboration and information sharing platforms (e.g., Facebook, Google+) and other sharing applications (e.g., Iphotos[1]). These platforms offer users the opportunity to share and manage various multimedia objects (e.g., photos, videos) taken by different users, during life events [12]. Due to excessive sharing on these platforms, the organization of the shared objects has become challenging. As a result, many works, such as the ones on Social Event Detection (SED), have evolved around the organization of shared data on social media platforms and sharing applications. SED works propose an organization of the shared objects by grouping them based on their related social events.

Currently, some online platforms, stand-alone applications, and other works [3, 4, 16, 20, 21] provide an organization of shared objects through event detection, using clustering techniques. Some are based on metadata (e.g., Facebook organizes multimedia content based on publishing timestamps, Iphotos combines photo creation timestamps and locations to organize a user's photo library).

[1] http://www.apple.com/ios/photos

Others use the visual attributes of shared objects (e.g., textures, colors) coupled with metadata [11, 13, 15] in order to detect several events.
Even-though these works offer automatic organization, they present limitations related to: (i) no user-centric processing, these methods do not include user-related features (e.g., social, interesting event topics) in the processing; (ii) lack of extensibility, these methods heavily rely on spatio-temporal features and do not allow the integration of various event features; (iii) the processing is based on one data source, neglecting the consideration of multiple data sources (i.e., multiple publishers, users); (iv) incremental processing is only implemented in recent applications, allowing a continuous integration of new data (e.g., photos shared/published on different dates/times) in the set of already processed data. Then, there is a need of providing a better event detection approach, considering user-related features, in a more meaningful way for users.
To answer this need, we propose a Feature-centric Social Event Detection (F-SED) approach, for automatic detection of "feature-centric" social events. Our method considers various features of an event (e.g., time, geo-location, social (e.g., users), topics, etc.), and allows the user to choose at least one central feature. Based on the chosen feature (or set of features) our approach detects the corresponding feature-centric events (e.g., topic-centered events if the topic feature is selected, user-centric events if the social feature is selected). F-SED's clustering technique simultaneously considers the various features, objects shared by different sources (several data publishers) on different dates/times, as well as data from one user (i.e., to organize a user data library). In addition, by integrating the user and his interests (*social and topics* features) in the clustering, the user automatically receives the events that interest him the most. F-SED is defined based on an adaptation of FCA (Formal Concept Analysis) [8, 22], a backbone that provides an extensible and incremental clustering technique, handles high dimensional data, and requires low human intervention. The comparative study on clustering techniques cannot be shown in this paper due to space limitations. We implement F-SED as a desktop-based prototype in order to evaluate the approach in real case scenarios with the ReSEED Dataset [18]. Our experimental results show that the event detection accuracy is improved when the social feature is taken into consideration. In addition, our performance results show quasi-linear behavior in most cases.
The rest of the paper is organized as follows. Section 2 reviews Social Event Detection works. Section 3 introduces FCA. Section 4 details and formally defines the F-SED approach. The implementation and evaluation are discussed in Section 5. Finally, Section 6 concludes and highlights future perspectives.

2 Related Work

In the literature, several Social Event Detection (SED) approaches have emerged for detecting events. Since most shared objects (e.g., photos, videos) on Social Networks and sharing applications are uploaded randomly without prior knowledge on the occurring events, these approaches mainly use unsupervised clustering techniques [3, 13, 15, 16, 21]. Since there are no commonly adopted criteria, we propose the following set of criteria to compare the referenced works:

1. *User-centric processing*: This criterion measures if user related data (e.g., names, interesting event topics) is integrated in the detection process to provide more meaningful and personalized results.
2. *Extensibility*: This criterion states if multiple event features are considered (e.g., *visual*, *social*, *topics*) in addition to time and locations for improved event detection.
3. *Multi-source*: This criterion indicates if multiple data sources (other publishers, participants) are considered, since various participants publish/share event related data.
4. *Incremental (continuous) processing*: This criterion considers the possibility of processing newly published data without having to repeat the entire processing, because participants could share event related data on different dates/times.
5. *Level of human intervention*: This criterion measures how frequently users participate in the event detection process; since huge amounts of data are shared, it is important that user interventions become less frequent; we consider low intervention if users provide data input and initial configuration; moderate if users intervene in result correction/optimization; and high intervention when users participate in the whole process.

SED approaches can be grouped into two categories: approaches that rely on the metadata of shared objects [3, 16, 19, 21], denoted metadata-based, and approaches that rely on visual attributes (e.g., colors, shapes) and metadata [6, 7, 11, 13, 15], denoted hybrid. We could not find approaches that only rely on visual attributes since grouping visually similar objects does not necessarily mean that the latter belong to the same event.

Metadata-based approaches: In [16], the authors aim to detect social events based on image metadata, using temporal, geo-location, and photo-creator information. They perform a multi-level clustering for these features. A first clustering separates photos into groups by distinct time values. A second one is executed on the first level clusters based on geo-location information. Finally, a third clustering based on the second level clusters is executed using the creator-names. In [3], the authors use time and GPS data to cluster photos into events using the mean-shift algorithm. First, the authors find baseline clusters based on time, then GPS location attributes are integrated. In [19], the authors use time and location information from twitter feeds to detect various events (e.g., earthquakes). In [21], the authors rely on textual tags such as time, geo-location, image title, descriptions, and user supplied tags to cluster photos into events, thus detecting soccer matches that happened in Madrid. These approaches need moderate human intervention. However, they are not incremental, user-centric, nor extensible. Metadata is also used by stand-alone applications for photo management to detect social events in a user's multimedia library. For example, Iphotos is an iOS mobile application that clusters photos and videos based on time and geo-location. These applications require no human intervention, they automatically cluster objects found in a user's library. However, they do not consider other event features (e.g., social), nor other photo sources (photos taken by other participants/collaborators). They mainly focus on time and location, photos taken

at the same day and place of an event are merged with the event.

Hybrid approaches: Many hybrid approaches rely on both visual and metadata attributes. In [13] and [15], the authors combine visual object attributes with temporal information, geo-locations, and user-supplied tags for their clustering procedures. Visual and tag similarity graphs are combined in [13] for the clustering. The tags used are crucial for event detection because they enable the distinction between events and landmarks. The authors consider photos of landmarks to have variant and distant photo capture timestamps because landmarks are photographed at any given date of the year and by many users. They also consider events to have a smaller distance separating timestamps while the number of users (participants) is also lower. They define an event based on time and location. While in [15], although metadata information is used in a different way, it is also crucial for distinguishing objects from events. The authors divide the geographical map of the world into square tiles and then extract the photos of each tile using geo-location metadata. They later use other metadata combined with visual features to detect objects and events. In [11] and [7], the authors combine temporal metadata with different sets of visual attributes for annotation and event clustering purposes. In [7], they combine time with color, texture, and facial (face detection) information. While in [11], the authors add to the previously mentioned attributes, aperture, exposure time, and focal length. In [6], the author relies more on temporal metadata than visual attributes for correct event detection, since he considers that photos/videos associated with one event are often not visually similar. Hybrid approaches consider different types of object attributes (visual, temporal information, geo-locations, tags, etc.). However, regrouping visually similar objects does not imply that they belong to the same event. Therefore, metadata is required to boost the accuracy of such approaches. Since these methods process visual attributes (e.g., through photo/video processing techniques), they end up having a higher processing cost than the approaches that only process metadata. Some approaches require more human intervention, because they prompt the user to correct/optimize the results.

Table 1: SED approaches comparison

Criteria	**Metadata-Based**		**Hybrid**
	[3][16][21]	**Stand-Alone Applications**	**[6][7][11][13][15]**
The level of human intervention	Low - Moderate	Low	Moderate - High
The incremental processing	No	Partially[2]	No
Extensibility	No	No	No
Multi-source	Partially[2]	Partially[2]	Partially[2]
User-centric processing	No	No	No

Table 1 summarizes the evaluations of SED approaches based on the aforementioned criteria. Metadata-based approaches [3, 16, 21] need low to moderate human intervention and provide good event detection accuracy, since metadata describes data related to the events (e.g., dates, locations, tags). However, these works lack the incremental processing needed to match the flow of publishing/sharing. Recently, incremental processing was integrated in some works (e.g., stand-alone applications). Hybrid methods are costly computation-wise

[2] Partially states that not all approaches of a category are multi-source or incremental

and require human intervention thus making continuous processing hard to implement [14]. In contrast, hybrid methods [6, 7, 11, 13, 15] offer more event features by combining visual attributes with metadata to improve accuracy. Finally, the two categories of works do not fully consider the social feature in event detection and lack the extensibility and user-centric processing needed to provide more personalized, and therefore interesting, results to the user.

3 FCA Preliminaries & Definitions

After studying various clustering techniques [1, 2, 10, 17], we chose Formal Concept Analysis (FCA) [8, 22] as the backbone for our F-SED approach. FCA is incremental and extensible (criteria 4 and 2). It examines data through object/attribute relationships, extracts formal concepts and orders the latter hierarchically in a Concept Lattice which is generated through a four step process [5]:

Step 1: Defining a **Formal Context** (Def. 1) from the input data, based on object/attribute relations represented in a cross-table.

Definition 1 *A **Formal Context**: is a triplet $\langle X, Y, I \rangle$ where:*
- *X is a non-empty set of objects*
- *Y is a non-empty set of attributes*
- *I is a binary relation between X and Y mapping objects from X to attributes from Y, i.e., $I \subseteq X \times Y$.*

Table 2 shows an example, where photos are objects and photo attributes (locations, photo creator names, and dates) are attributes. The cross-joins represent the mapping of photos to their respective photo attributes, e.g., photo 1 *was taken in Biarritz by John on 17/08/2016.* ■

Table 2: Formal Context example

		Names				**Locations**			**Dates**		
		John	Patrick	Dana	Ellen	Biarritz	Munich	Paris	17/08/2016	12/12/2012	02/02/2016
Photos	1	x				x			x		
	2	x				x			x		
	3		x				x			x	
	4			x			x			x	
	5				x			x			x

Step 2: Adopting **Concept Forming Operators** to extract **Formal Concepts** (Def. 2). FCA has two concept forming operators:

- $\uparrow: 2^X \rightarrow 2^Y$ (Operator mapping objects to attributes)
- $\downarrow: 2^Y \rightarrow 2^X$ (Operator mapping attributes to objects).

For example, from the cross-table shown in Table 2, we have $\{3\}^{\uparrow} = \{$Patrick, Munich, 12/12/2012$\}$ and $\{02/02/2016\}^{\downarrow} = \{5\}$.

Definition 2 *A **Formal Concept** in $\langle X, Y, I \rangle$ is a pair $\langle A_i, B_i \rangle$ of $A_i \subseteq X$ and $B_i \subseteq Y$ such that: $A_i^{\uparrow} = B_i \wedge B_i^{\downarrow} = A_i$.*

Consider the set of photos A_1=$\{1, 2\}$ and the set of attributes $B_1 = \{$John, Biarritz, 17/08/2016$\}$. $A_1^{\uparrow} = \{$John, Biarritz, 17/08/2016$\}$ and $B_1^{\downarrow} = \{1, 2\}$. Thus, since $A_1^{\uparrow} = B_1$ and $B_1^{\downarrow} = A_1$, the pair $\langle A_1, B_1 \rangle$ is a Formal Concept. ■

Step 3: Extracting a **Subconcept/Superconcept Ordering** relation for **Formal Concept** (cf. Def. 2) ordering by defining the most general concept and the most specific concept for each pair. The ordering relation is denoted $\leq$.

For example, from Table 2, let $A_1 = \{3\}$, $B_1 = \{$Patrick, Munich, 12/12/2012$\}$, $A_2 = \{3, 4\}$, and $B_2 = \{$Munich, 12/12/2012$\}$. According to Def. 2, $\langle A_1, B_1 \rangle$ and $\langle A_2, B_2 \rangle$ are formal concepts. In addition, $A_1 \subseteq A_2$ therefore, $\langle A_1, B_1 \rangle \leqslant \langle A_2, B_2 \rangle$. This means that formal concept $\langle A_1, B_1 \rangle$ is a subconcept of formal concept $\langle A_2, B_2 \rangle$ (which is the superconcept).

Step 4: Generating the **Concept Lattice**, which represents the concepts from the most general one (top) to the most specific (bottom). The lattice is defined as the ordered set of all formal concepts extracted from the data (based on $\leq$).

For the example shown in Table 2, Fig. 1 illustrates the Concept Lattice. The next section formally describes our F-SED approach and how these FCA four steps are integrated and adapted for the clustering of shared objects (content).

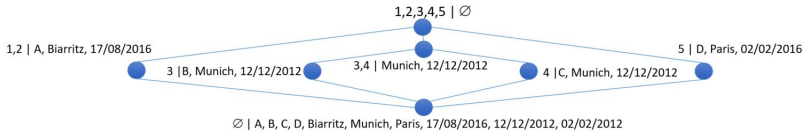

Fig. 1: The Concept/ Galois Lattice

4 An approach for Feature-centric Social Event Detection

In this section, we propose F-SED, an approach to detect feature-centric social events from a set of multimedia objects (e.g., photos, videos) shared by different users. F-SED mainly relies on a modular framework (Fig. 2) that integrates FCA as the backbone clustering technique, to provide an extensible and incremental approach. F-SED aims to detect feature-centric social events which are defined by at least three event features: (i) *temporal* feature (represented by a *time interval*), (ii) *geographical* feature (represented by a *location* value), and (iii) *social* feature (e.g., *photo creator name*). In order to focus more on the user's interests, F-SED also supports additional features such as *topics* (based on *tags* or *annotations*) and various levels of granularities for each feature. In order to organize a set of shared objects according to feature-centric events, F-SED processing is split into three main steps: (i) data pre-processing and extraction (executed by the Preprocessor and Attribute Extractor modules); (ii) lattice construction (executed by the Event Candidates Lattice Builder module); in this step we integrate and adapt the FCA clustering technique (described in Section 3) for shared objects clustering; (iii) event detection (carried out by the Feature-Centric Event Detector and Rule Selector modules). The user interacts with the system through the Front End. In the following, we detail each processing step and module.

4.1 Data pre-processing and extraction

Through the Front End, one or multiple user(s), when considering a multi-source scenario (criterion 3), send(s) a set of shared objects (Def. 3). The purpose of this

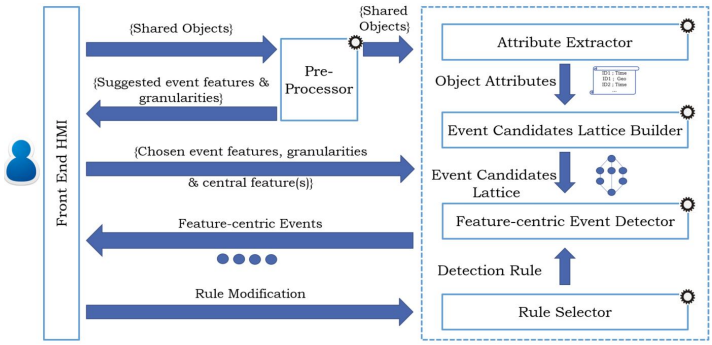

Fig. 2: F-SED Architecture

step is to extract the attributes of each shared object. An attribute is defined as a value associated with an attribute data type (Def. 4). We define a data type function denoted dt, that returns the attribute data type of a value based on the shared object attributes.

Definition 3 *A **Shared Object** is defined as a 2-tuple, so* : $\langle id, V \rangle$*, where:*

- id *is the unique identifier of a shared object*
- V *is a set of attribute values according to a given ADT, such that* $\forall a_i \in ADT \quad \exists v_i \in V \mid dt(v_i) = a_i$. ■

Definition 4 ***ADT*** *is a set of attribute data types defined in a metric space, such that* $\forall a \in ADT$, $a : \langle l, t, \Omega, d, f \rangle$*, where:*

- l *is a label*
- t *denotes the primitive data type of the attribute where* $t \in \{Integer, Float, Boolean, Date, Time, Character, String\}$
- Ω *is the domain (range) of the attribute values*
- d *is the function that returns the distance between any two values* v_i *and* v_j *of attribute data type* a
- f *is the event feature mapped to the attribute data type.* ■

The *Pre-processor* checks the attributes of each object and based on the data distribution, suggests event features (cf. Def. 5) and granularities to be used in the clustering. For example, the *Pre-processor* suggests the following features: *temporal*, *geographical*, *social*, and *topics* based on the availability of *photo creation timestamps*, *photo locations*, *photo creator names*, and *tags* or *annotations* respectively. The social feature can identify a single user (e.g., a user name) or a group of users (e.g., family, colleagues, friends), thus allowing the detection of user-centric events, or group-centric events (e.g., family events, work-related events). In addition, considering photo tags and annotations (either added manually by the user or detected by data inference or image processing techniques [9]), it can integrate a fourth event feature related to topics. Regarding granularities, the *Pre-processor* proposes for example to cluster *time* based on *day*, if photo creation values include day values. Finally, the *Pre-processor* detects the user's sharing space (cf. Def. 6), thus regrouping the collaborators, friends, other participants, and users that created/shared objects.

Definition 5 *Features **F** is a set of event features where* $\forall f \in F$, $f : \langle label, G, interval, gran \rangle$*, where:*

- *label is the feature label*
- *G is a set of granularities associated to the feature*
- *interval is a boolean value indicating if the feature is generated as an interval (true), or not (false)*
- *gran is a function that converts any granularity g_i to another g_j where $g_i, g_j \in G$.*

For example, the *temporal* event feature can be represented by:
$f_0 : (\text{"}Time\text{"}, \{year, month, week, day, hour, minute, second\}, True, gran_{time})$.
$gran_{time}(1year) = 12$ months. ■

Definition 6 *A Sharing Space for a user u_0, denoted as $\boldsymbol{SS_{u_0}}$, is defined as a 3-tuple: $SS_{u_0} : \langle U, USO, SO \rangle$, where:*

- *U is the set of user names in the sharing space of the user $u_0 \mid u_0 \in U$ (U is used to extract the social feature)*
- *$USO : U \longrightarrow SO$ is an association function mapping users to their respective shared objects such that, $USO(u_i) = \bigcup_{i=1}^{n} so_i$ where so_i is a shared object and n is the number of objects shared by u_i*
- *SO is a set of all shared objects in $SS_{u_0} \mid SO = \bigcup_{i=1}^{|U|} USO(u_i)$* ■

Once the user(s) validate(s) the features and granularities proposed by the *Pre-processor*, the *Attribute Extractor* extracts the object attributes. It also stores the shared objects, attributes, and values related to the chosen features for the lattice construction. In addition, the user specifies the feature (or set of features) that he would like to consider as central for the feature-centric event detection. No more human intervention is needed (low intervention-criterion 5).

4.2 Lattice Construction

In this step, we process the previously extracted object attributes, and shared objects into lattice attributes and objects, in order to generate one output: the lattice. The *Feature-centric Event Lattice Builder* is the FCA backbone. It integrates the four step process of FCA clustering described in Section 3. To do so, we formally define lattice attribute types in Def.7. These types will be used when defining the lattice attributes (cf. Def.8). Finally, for object/lattice attribute mapping, we define a binary cross rule denoted BXR (cf. Def.9).

Definition 7 ***lat*** *is a lattice attribute type representing an interval $[a, b[$ where $lat : \langle a, b, T \rangle$, where:*

- *a is the lower boundary value*
- *b is the upper boundary value*
- *T is a value representing the period having a primitive data type of either integer or float, such that:*
  - *$dt(a) = dt(b) \in ADT$ and*
  - *$b = a + T$.* ■

Definition 8 *A lattice attribute, denoted **la**, is defined as a 4-tuple $la : \langle f, SS_{u0}, lat, y \rangle$ where:*

- *$f \in F$ is the event feature mapped to lattice attribute la*
- *SS_{u_0} is the sharing space in which the detection will take place (cf. Def. 6)*
- *lat (cf. Def. 7) is the lattice attribute type*
- *y is a granularity $\mid y \in f.G$ and*

$$lat.T = \begin{cases} y & \text{if } f.interval = True \\ 0 & Otherwise \end{cases}$$

$lat.a = so_i.v_j$, where:

- $so_i \in SS_{u_0}.SO$ *and*
- $(v_j \in so_i.V) \wedge (dt(v_j).f = f)$. ■

Definition 9 *A binary cross rule, denoted as* ***BXR****, is defined as a function that maps a shared object x to its respective lattice attribute y where $x.v_i \in x.V$:*

$$BXR = \begin{cases} 1 & if\ (y.lat.T = 0 \wedge y.lat.a = x.v_i) \vee \\ & (y.lat.T \neq 0 \wedge x.v_i \in [y.lat.a, y.lat.b[) \\ 0 & Otherwise \end{cases}$$ ■

Then the *Feature-centric Event Lattice Builder* constructs the F-SED formal context, denoted ***ffc*** (cf. Def. 10). Once the ***ffc*** is created, formal concepts are extracted and a lattice is generated. This process is described in steps 2-4 of Section 3. This lattice is called an Event Candidate Lattice, where each node is a potential feature-centric event.

Definition 10 *A F-SED Formal Context, denoted* ***ffc****, is defined as a 6-tuple ffc : $\langle SS_{u0}, F, f_{LAG}, X, Y, I \rangle$, where*

- *SS_{u0} is the sharing space in which the detection takes place*
- *F is the set of event features*
- *f_{LAG} is the function that generates the lattice attributes, described in Algorithm 1*
- *$X = SS_{u_0}.SO$ is the set of shared objects*
- *$Y = \bigcup_{i=0}^{|X.V|-1}\{la_i\}$ is the set of lattice attributes | $X.V = \bigcup_{\forall so \in X}\{so.V\}$ is the union of all attribute values from the shared objects in SS_{u0}*
- *I is a BXR(x,y) where $x \in X \wedge y \in Y$.* ■

Algorithm 1: Lattice Attribute Generation (cf. Def. 10 - f_{LAG})

```
1  Input: SS_u0
2  Output: RES                                          // List of all lattice attributes
3  VAL = new List()                                     // Shared Objects attribute values list
4  PD = new List()                                      // Processed event features list
5  foreach so ∈ SS_u0.SO do
6  |  foreach v ∈ so.V                                 // This loop extracts all object attribute values
7  |  do                                                from all objects in SS_u0 and stores them in the
8  |  |  if (v ∉ VAL) then                             VAL list
9  |  |  |  VAL←v
10 |  end
11 end
12 foreach v ∈ VAL do
13 |  if (not dt(v).f.Interval)                        // If the value is not generated as an
14 |  then                                              interval
15 |  |  lat ← LAT(v, lat.a + lat.T, 0)
16 |  |  la ← LA(dt(v).f, SS_u0, lat, dt(v).f.g)       // Create la with lat.T=0
17 |  |
18 |  |  RES ← la
19 |  else
20 |  |  if (dt(v).f ∉ PD) then
21 |  |  |  RES ← (Create-Intervals(VAL,v,PD, SS_u0))  // Call
22 |  |  |                                               Create-Intervals
23 |  end                                               function
24 end
25 return RES
```

In Algorithm 1, we detail the lattice attribute generation process. This starts by extracting all object attribute values (lines 5-11). If the value is mapped to a feature that is generated as an interval (e.g., *time*), the algorithm calls the Create-Intervals function (lines 19-23). If not (e.g., *social*), the algorithm generates a lattice attribute type having a null period and creates the corresponding

lattice attribute (lines 13-18). This step allows the creation of generic lattice attributes from various features, thus providing extensibility (criterion 2). Algorithm 2 details the Create-Intervals function. This process extracts all values related to the same feature (lines 4-9), orders them (line 10), selects a minimum and a maximum value (lines 11-12), and creates periodic intervals starting from the minimum to the maximum value (lines 14-22). The period is calculated based on the chosen feature granularity (line 15). This makes the detection more user-centric (criterion 1). Finally, the result is added the the output of Algorithm 1.

Algorithm 2: Create-Intervals

```
1  Input: VAL, v, PD, SS_{u_0}   // Input provided by Algorithm 1, line 21
2  Output: LAI                   // Generated lattice attributes intervals
3  int i = 0
4  TEMP = new List()             // Temporary object attribute list
5  foreach val ∈ VAL do
6  |  if (dt(val).f == dt(v).f)  // Extract all object attribute
7  |  then                          values having the same
8  |  |  TEMP ← val               feature as v and store them
9  end                             in TEMP
10 Order_{ascending}(TEMP)                  // Order TEMP ascending
11 min ← TEMP.get(0)              // min is the first element of TEMP
12 max ← TEMP.get(|TEMP| − 1)     // max is the last element of TEMP
13 lat ← LAT()
14 while (lat.b < max) do
15 |  lat ← LAT(min, lat.a + (i+1) × lat.T, dt(v).f.g)
16 |  if (lat.b > max)                        // This loop creates
17 |  then                                      intervals of period
18 |  |  lat.b ← max                           lat.T = f.g
19 |  la ← LA(dt(v).f, SS_{u0}, lat, dt(v).f.g)   (feature
20 |  LAI ← la                                  granularity)
21 |  i++
22 end
23 PD ← dt(v).f      // Add feature to the list of processed features
24 return LAI
```

Feature	Condition
Time	Time Range
Geo	Geo Range
Social (central)	Specific value
Topic	Topic Range

(a)

Feature	Condition
Time	Time Range
Geo (central)	Specific Value
Social	Social Range
Topic	Topic Range

(b)

Feature	Condition
Time	Time Range
Geo	Geo Range
Social	Social Range
Topic (central)	Specific Value

(c)

Feature	Condition
Time (central)	Specific Value
Geo	Geo Range
Social	Social Range
Topic	Topic Range

(d)

Fig. 3: Default detection rule

4.3 Event Detection

The *Feature-centric Event Detector* module uses the previously generated lattice, an event detection rule, and the central features chosen by the user in order to detect feature-centric events (cf. Def. 11). We define a default detection rule, as a set of lattice attributes that comply with the two conditions mentioned in Def. 11. The rule is extensible, thus allowing the integration of multiple event features (e.g., *Time*, *Geo-location*, *Social*, *Topic*), each represented by the corresponding lattice attribute. This rule uses the selected central features in order to target the related feature-centric events. For example, the rules illustrated in Fig.3.(a), 3.(b), 3.(c), and 3.(d) detect user, geo, topic, and time-centric events respectively. Finally, for testing purposes, users can change/add detection rules using the *Rule Selector* module. Since the lattice is not affected by the rule change, only the event detection step is repeated based on the new detection rule.

Definition 11 *A feature-centric Event, denoted* ***fce****, is a Formal Concept defined as a 4-tuple* $fce : \langle ffc, central_F, A, B \rangle$*, where:*

- *ffc is a F-SED Formal Context (Def. 10)*
- $central_F$ *is the set of central features selected by the user* $|central_F \subseteq ffc.F$
- A *is a set of shared objects* $| A \subseteq ffc.X$
- B *is a set of lattice attributes* $| B \subseteq ffc.Y$ *where* $\forall b_i, b_j \in B \wedge i \neq j$*:*
  - ***Condition 1:*** $b_i.f \neq b_j.f$

- ***Condition 2:*** *if* $b_i.f.label = c_f.label | \forall c_f \in central_F$, *then* $d(b_i.lat.a, so_j.v_k)=0$ $| \forall so_j \in A \wedge \forall v_k \in so_j.V, \quad dt(b_i.lat.a) = dt(so_j.v_k)$. ■

5 Implementation & Evaluation

In order to validate our approach, we developed a Java desktop prototype and used the Colibri-java library for lattice generation. We evaluated the algorithm's performance based on execution time and memory consumption, and the quality of our detection process by measuring its accuracy. The objective of the experimentation is to show that the approach is generic and accurate when given optimal features/granularities. We do not aim at comparing accuracy results with other works.

ReSEED Dataset: To evaluate the detection results, we used the ReSEED Dataset, generated during the Social Event Detection of MediaEval 2013 [18]. It contains real photos crawled from Flickr, that were captured during real social events which are heterogeneous in size (cf. Fig. 4) and in topics (e.g., birthdays, weddings). The dataset contains 437370 photos assigned to 21169 events. In our evaluation, we used three event features: *time, location, and social*, since ReSEED photos have *time, geo, and social* attributes. In ReSEED, 98.3% of photos contain *capture time*, while only 45.9% of the photos have a *location*. We had to select photos having these attributes from the dataset. This left us with 60434 photos from the entire dataset. In ReSEED, the ground truth used

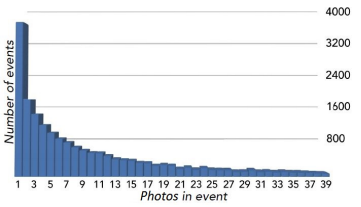

Fig. 4: ReSEED Photo distribution

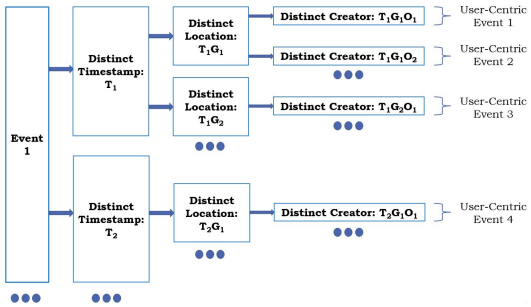

Fig. 5: Refactoring ReSEED ground truth

for result verification assigns photos to social events. Since, our approach is focused on feature-centric events (in this experimentation, user-centric events), we modified the ground truth to split the social events into their corresponding user-centric events. Since the splitting is based on the event features, we need to specify the feature granularities during the process. The latter are not specified in ReSEED, therefore we chose the lowest granularity values: *day* for *time*, *street* for *geo*, and *photo creator name* for *social*. The ground truth refactoring process is described in Fig. 5. First, we extracted the photos of each event in the ground truth. Second, we used the *timestamps of photo capture* to group photos by *day*. Third, we split the resulting clusters into distinct groups based on *street* values. Finally, the result was further split based on distinct *photo creators*.

Performance Evaluation: The performance tests were conducted on a machine equipped with an Intel i7 2.60 GHZ processor and 16 GB of RAM. The aim was to test the performance of our F-SED algorithm. We considered two criteria for this task: (i) total execution time and (ii) memory overhead.

Use Cases The performance is highly affected by the number of photos, generated attributes, and clusters. We noticed that granularities *day* for *time* and *street* for *geo* generate more clusters and attributes than any other granularity combination. Therefore, we used *day* and *street* to test the prototype's performance in three worst case scenarios:

- Case 1: We selected the biggest event (1400 photos) as input. We varied the number of photos progressively from 1 to 1400. Since all photos are related to one event, the number of detected clusters should be one.
- Case 2: We extracted 400 events each having exactly one photo. We varied the number of photos from 100, 200, 300 to 400. The number of generated clusters for each iteration should be 100, 200, 300, and 400 respectively.
- Case 3: The goal is to test with as many photos as possible related to different events. We varied the number of photos from 15000, 30000, 45000 to 60434. Since thousands of events contain only one or two photos per event (worst case scenario), this case will generate the most clusters.

Results & Discussion In Cases 1 and 2 (Figures 6.a and 6.b), where the number of photos does not exceed 1400 and 400 respectively, the total execution time is quasi-linear. However, in Case 3 (Figure 6.c), we clustered the entire dataset (60434 photos). The total execution time tends to be exponential, in accordance with the time complexity of FCA. When considering RAM usage, we noticed a linear evolution for the three cases (Figures 6.d, 6.e, and 6.f). RAM consumption is significantly higher in Case 2, where we generated 400 clusters, than in Case 1, where we generated one cluster. In Case 3, RAM consumption is the highest because both the number of photos at the input, and the number of generated clusters (detected events) were the highest. Other tests were conducted, Fig.7 (left) shows that low granularities (e.g., day) consume more execution time than high ones (e.g., year). This is due to the generation of more lattice attributes and clusters. In addition, Fig.7 (right), shows that considering more features in the processing is also more time consuming. Nonetheless, the evolution from one to three features remains quasi-linear, making the process extensible.

Accuracy Evaluation: We chose to consider the criteria proposed by MediaEval for clustering quality evaluation. We calculated the F-score, based on the Recall (R) and Precision (PR), and the Normalized Mutual Information (NMI) using ReSEED's evaluation tool. These criteria are commonly adopted in information retrieval and social event detection. A high F-score indicates a high quality of photo to user-centric event assignment while NMI will be used to measure the information overlap between our clustering result and the ground truth data. Therefore, a high NMI indicates accurate clustering result.

Use Cases: Since we considered the *time*, *geo*, and *social* features, we identified all possible combinations of the detection rule (see Table 3). In order to test granularity impacts, Table 4 sums up the different granularity combinations.

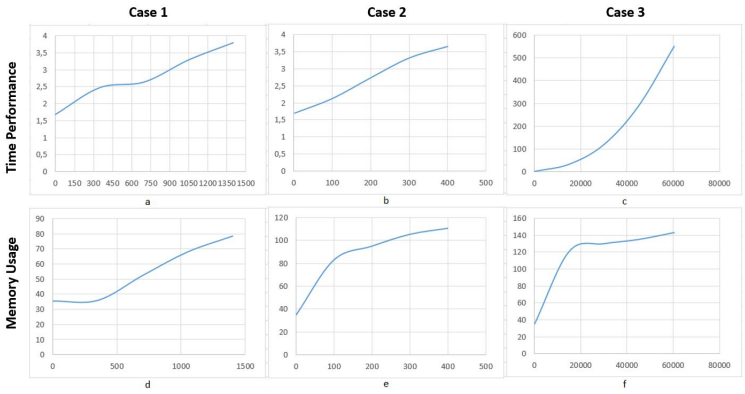

Fig. 6: Performance Results

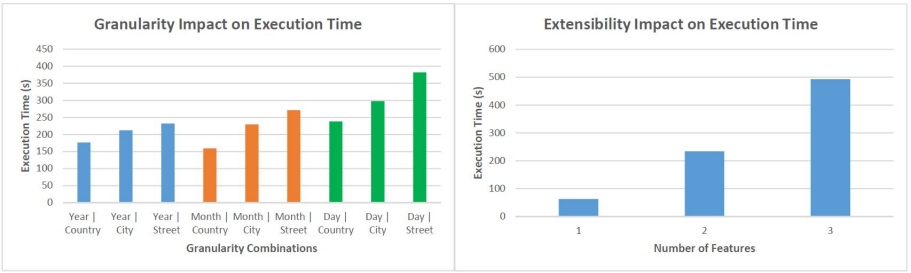

Fig. 7: Granularity and Extensibility Impact

When applying detection rules to granularity combinations, we get 63 use cases. We measured for each one the NMI and F-Score.

Table 3: Detection Rule

Combination	Number of Features	Features Considered in the Detection Rule
1	3	Time, Geo, Social
2	2	Time, Geo
3		Time, Social
4		Geo, Social
5	1	Time
6	1	Geo
7	1	Social

Table 4: Granularity Combinations

Combination	Granularities: Time / Geo
1	Year / Country
2	Year / City
3	Year / Street
4	Month / Country
5	Month / City
6	Month / Street
7	Day / Country
8	Day / City
9	Day / Street

Results & Discussion: Results shown in Table 5, highlight the following:
(i) *Detection rule/features impact:* Our detection rule (based on *time*, *geo*, and *social* features) generates the highest NMI and F-score (NMI: 0.9999 and F-Score: 0.9995). It also exceeds all other detection rules (e.g., the one including solely *time* and *geo* features) in every granularity combination. This underlines the importance of the *social* feature in the detection task. Moreover, it highlights F-SED's extensibility, which allows the integration of additional features and the accurate detection of user-centric events. Nonetheless, accuracy can still be improved, few photos were assigned to the wrong clusters, due to the closeness in time and space of the latter.
(ii) *Granularity impact:* The results improve, when the clustering is based on granularities closer to the ones used in the ground truth. For example, in the

case of granularities *year, country*, the F-Score achieved based on *time* and *geo* features is 0.1911, but for the detection rule that considers only the *social* feature the F-Score is higher: 0.5376. This is because the granularities for *time* and *geo* are the most general (*year and country*). Therefore, the impact factor of granularities is more important than that of the number of features considered in the detection rule. Some rules can exceed others for specific granularity combinations (e.g., *Time Geo* exceeds *Time Social* and *Geo Social* for granularities Year/Month/Day-Street while Time Social exceeds the other two rules for Year/Month/Day-Country). The best result can be achieved by considering the maximal number of features having correct granularities. This indicates that the granularities should not be fixed for all scenarios. When given the best granularities, our approach detects the user-centric events very accurately. With these results, we find it interesting to study the user data distribution to deduce the granularities that boost his detection results the most or the possibility of disregarding certain features with non optimal (result-wise) granularities. This can become an advantage for reducing processing costs. In addition, we would also like to investigate inter user-centric event distances for *time* and *geo* essentially.

6 Conclusion & Future Work

Social Event Detection (SED) has become essential towards automatic, semantic organization of photo collections and other multimedia objects on the web. In this paper, we propose a generic framework for Feature-centric Social Event Detection (F-SED) based on Formal Concept Analysis (FCA). Our approach aims at integrating various features (e.g., *social*, *topics*), making the detection task more user-centric, extensible, incremental, and as automated as possible, thus reducing human intervention. We developed a prototype for testing purposes. The results show that our approach achieved high accuracy in most cases, especially when the social feature is considered. As future work, we are investigating the detection of optimal granularities based on the user's data distribution and sharing frequency and the impact of other features. We would also like to improve our accuracy by considering spatio-temporal distances between clusters and noise handling techniques. In addition, we aim at testing the incremental processing of our algorithm. Finally, we want to extend our work in order to detect social events based on existent ties between feature-centric events.

Table 5: Clustering Results

Detection Rule	**Measure**	**Granularities**								
		Year			Month			Day		
		Country	City	Street	Country	City	Street	Country	City	Street
Time Geo Social	F-Score	0.6399	0.8180	0.8662	0.7964	0.8619	0.8948	0.9535	0.9742	**0.9995**
	NMI	0.9181	0.9602	0.9729	0.9549	0.9703	0.9789	0.9880	0.9938	**0.9999**
Time Geo	F-Score	0.1911	0.7678	0.8473	0.4943	0.8367	0.8821	0.8854	0.9542	0.9892
	NMI	0.7113	0.9475	0.9684	0.8707	0.9637	0.9759	0.9729	0.9894	0.9977
Time Social	F-Score	0.6245	0.6245	0.6245	0.7939	0.7939	0.7939	0.9534	0.9534	0.9534
	NMI	0.9143	0.9143	0.9143	0.9544	0.9544	0.9544	0.9879	0.9879	0.9879
Geo Social	F-Score	0.5085	0.7718	0.8357	0.5085	0.7718	0.8357	0.5085	0.7718	0.8357
	NMI	0.8742	0.9470	0.9653	0.8742	0.9470	0.9653	0.8742	0.9470	0.9653
Time	F-Score	0.0220	0.0220	0.0220	0.1399	0.1399	0.1399	0.7278	0.7278	0.7278
	NMI	0.3971	0.3971	0.3971	0.7069	0.7069	0.7069	0.9392	0.9392	0.9392
Geo	F-Score	0.0559	0.6958	0.8343	0.0559	0.6958	0.8343	0.0559	0.6958	0.8343
	NMI	0.5084	0.9241	0.9646	0.5084	0.9241	0.9646	0.5084	0.9241	0.9646
Social	F-Score	0.5376	0.5376	0.5376	0.5376	0.5376	0.5376	0.5376	0.5376	0.5376
	NMI	0.8755	0.8755	0.8755	0.8755	0.8755	0.8755	0.8755	0.8755	0.8755

References

1. Berkhin, P.: A survey of clustering data mining techniques. In: Grouping multidimensional data, pp. 25–71. Springer (2006)
2. Burmeister, P.: Formal concept analysis with ConImp: Introduction to the basic features. Fachbereich Mathematik, Technische Universität Darmstadt (2003)
3. Cao, L., et al.: Image annotation within the context of personal photo collections using hierarchical event and scene models. IEEE Transactions on Multimedia 11(2), 208–219 (2009)
4. Chen, L., Roy, A.: Event detection from flickr data through wavelet-based spatial analysis. In: Conf. on Information and Knowl. Manag. pp. 523–532 (2009)
5. Choi, V.: Faster algorithms for constructing a concept (galois) lattice. Clustering Challenges in Biological Networks p. 169 (2006)
6. Cooper, M., et al.: Temporal event clustering for digital photo collections. ACM Transac. on Multimedia Computing, Communic., and App. 1(3), 269–288 (2005)
7. Cui, J., et al.: Easyalbum: an interactive photo annotation system based on face clustering and re-ranking. In: Conf. on Human factors in computing systems. pp. 367–376. ACM (2007)
8. Ganter, B., Wille, R.: Formal concept analysis: mathematical foundations. Springer Science & Business Media (2012)
9. Hanbury, A.: A survey of methods for image annotation. Journal of Visual Languages & Computing 19(5), 617–627 (2008)
10. Jain, A.K.: Data clustering: 50 years beyond k-means. Pattern recognition letters 31(8), 651–666 (2010)
11. Mei, T., et al.: Probabilistic multimodality fusion for event based home photo clustering. In: Internat. Conf. on Multimedia and Expo. pp. 1757–1760 (2006)
12. Oeldorf-Hirsch, A., Sundar, S.S.: Social and technological motivations for online photo sharing. Journal of Broadcasting & Electronic Media 60(4), 624–642 (2016)
13. Papadopoulos, S., et al.: Cluster-based landmark and event detection for tagged photo collections. IEEE MultiMedia 18(1), 52–63 (2011)
14. Park, S.C., Park, M.K., Kang, M.G.: Super-resolution image reconstruction: a technical overview. IEEE signal processing magazine 20(3), 21–36 (2003)
15. Quack, T., Leibe, B., Van Gool, L.: World-scale mining of objects and events from community photo collections. In: Internat. Conf. on Content-based image and video retrieval. pp. 47–56 (2008)
16. Raad, E.J., Chbeir, R.: Foto2events: From photos to event discovery and linking in online social networks. In: Internat. Conf. on Big Data and Cloud Computing. pp. 508–515. IEEE (2014)
17. Rehman, S.U., et al.: Dbscan: Past, present and future. In: Internat. Conf. on Applications of Digital Information and Web Technologies,. pp. 232–238 (2014)
18. Reuter, T., et al.: Reseed: social event detection dataset. In: Conf. on Multimedia Systems. pp. 35–40. ACM (2014)
19. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes twitter users: real-time event detection by social sensors. In: Proc. of Internat. Conf. on WWW. pp. 851–860. ACM (2010)
20. Sayyadi, H., et al.: Event detection and tracking in social streams. In: Icwsm (2009)
21. Sheba, S., Ramadoss, B., Balasundaram, S.: Event detection refinement using external tags for flickr collections. In: Intelligent Computing, Networking, and Informatics, pp. 369–375. Springer (2014)
22. Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Ordered sets, pp. 445–470. Springer (1982)