Extracting Topics with Focused Communities in Multi-dimensional Social Data

Theodore Georgiou
Department of Computer Science
University of California, Santa Barbara
teogeorgiou@cs.ucsb.edu

Amr El Abbadi
Department of Computer Science
University of California, Santa Barbara
amr@cs.ucsb.edu

Xifeng Yan
Department of Computer Science
University of California, Santa Barbara
xyan@cs.ucsb.edu

ABSTRACT

Understanding what social media users discuss and what is happening in the real world can be enabled through the automatic analysis and summarization of Online Social Media. Trend Discovery, through the extraction of trending topics, is utilized in marketing campaigns and by companies to identify customer interests and potential new markets. While there is a plethora of techniques to identify trending topics, there is a lack of focus on the characteristics of the underlying population that participate in a trend. Users that mention a topic define a multivariate vector of demographics and user characteristics with the potential to offer insights into the communities that are focused on a topic, both latent and obvious. We propose a novel algorithmic framework for the efficient and scalable extraction of the combination of user characteristics that define a community interested in a topic. Potential results include cases like "female residents of Boston, MA, that support the Democratic party, are focused on the activistic topic #FreeJustina" or "young adults, living in the US, are focused on topic #NavyYardShooting with a Negative sentiment". Such topics might be significantly popular or not, but community extraction emphasizes the importance and focused interest in a topic even if it is not as popular as other topics with a more defused audience. The proposed framework can support any number of attributes and scales linearly. We assessed our algorithm's accuracy and efficiency on synthetic and real Twitter data and results show high accuracy and efficiency.

1. INTRODUCTION

The study of social patterns in Online Social Media like Twitter or Facebook can be very helpful in identifying collective user behavior among specific segments of society. Trending Topics have been popularly used in the detection of breaking news, as well as in marketing and advertising mechanisms. In its general definition, a Trending Topic is a collection of words or phrases that refer to a temporarily popular topic. Usually, the origin of a trending topic is a popular real life event that is being discussed on social media or a meme that is spreading. Trending topics are used to understand and explain how information and memes diffuse through vast social networks with hundreds of millions of nodes. Many algorithms have been proposed for discovering interesting trending topics utilizing techniques from the areas of Anomaly Detection, Data Streams, and Clustering. In existing studies, trending topics are mined for specific interest areas like Sports [18], Earthquakes [21], News reporting [16, 22], or general event detection [20, 1]. In this paper we research the novel idea of identifying the underlying user communities that are interested in social media topics and then utilize this knowledge to rank topics by importance. Such a task can be challenging in terms of complexity when dealing with a non trivial number of community characteristics. The official Twitter Trending Topics are personalized to the user by displaying the top topics from categories the user is interested in. This is a simple approach to serve relevant trends but focuses only on topic categories (e.g. Technology, Politics, etc.) or location, and does not identify topics where the underlying population has specific properties, thus, can miss less popular topics with highly interesting community characteristics.

We observed that the user population involved in a trend offers high potential in understanding a trend and how other users might react to it. In a previous study on Twitter trending topics even simple social relations between the participants of a topic trend could greatly enhance the understanding of trending topics [9] or spammer detection [7]. Then, a space efficient framework was proposed, named GeoScope [10], that extracts trending topics which are highly focused in specific geographical locations. Human evaluations showed that topics with a high geographical correlation tend to be more interesting than topics with a dispersed population.

In this paper, we propose a novel community detection algorithm utilizing a spectrum of social characteristics rather than just geographic locations. The detection of community characteristics that are interested in a topic, like gender, age, location, race, ethnicity, political affiliation, etc., can yield powerful results which are useful in a variety of domains. Marketers can understand their customers better by identifying the communities interested in their or competitive products. Advertisements, which usually are linked to a trending topic or event, can become more personalized. Content recommendation can be improved through the extraction of target groups interested in specific topics. Last but not least, the understanding of the underlying population of a topic can enable the study and discovery of latent communities that sociologists might not be aware of.

Communities focused on topics, can sometimes be expected and sometimes unexpected. It is easy to anticipate that young boys will be interested in the PlayStation 4 gaming console even without monitoring the widely popular trending topic #PS4. But we might not expect that women in the area of Boston, MA, that also support the Democratic party, showed their solidarity to an arrested female teen named Justina with the not so popular topic #FreeJustina. It is even more unexpected to observe the hijacking of the hashtag campaign #ReasonsToVisitEgypt that was originally created to promote tourism in Egypt, but local citizens used it negatively to raise awareness for the country's political situation. The important take away is that using only the popularity of a trending topic is not enough; a better understanding of the underlying community can yield a better ranking for interesting topics that might not be globally popular.

We define a specific type of community that we call *focused community* to enable the efficient extraction of communities interested in topics on social media. We exploit specific properties of this definition to propose a novel framework that receives a social stream as its input and reports topics and the corresponding focused communities as its output. The framework scales linearly with the number of attributes, and reports communities that might lie within in any attribute combination or sub-dimension.

The current research project makes the following contributions:

- The definition of a maximally focused community.
- Information Retrieval metrics, like TF/IDF, are adapted to rank topics based on their communities' relative popularity and exclusivity.
- A probabilistic and scalable algorithm for the discovery of maximally focused communities with amortized linear time complexity. The algorithm also deals with missing values in records, in a stochastic way that maintains high confidence on the reported results.
- The effectiveness and efficiency of the proposed approach is tested through extensive experimentation and analysis on a massive social dataset from Twitter and synthetic data.

Related work is presented in Section 2. The definition of a Focused Community is given in Section 3 and the description of the proposed algorithm is offered in Section 4. The methodology for dealing with missing values in real social data is described in Section 5 and, finally, experimental results are presented in Section 6.

2. RELATED WORK

While there are multiple approaches to extract trending topics from social content we are the first to explore the underlying focused communities. Our work builds on many techniques in areas that share common properties with this problem, most notably from Subspace Clustering and Frequent Itemset Extraction/Association Rule Mining.

Association Rule Mining using Frequent Itemset Extraction [6] is a well studied area and poses similarities to the attribute-based community extraction. Techniques that sample the data to perform fast itemset extraction are the closest to our proposed approach since probabilistic algorithms are used to reduce complexity. Such techniques include Toivonen [25] and Chakaravarthy et al. [11].

Clustering algorithms for data in multiple dimensions, known as subspace clustering algorithms, are usually divided in two categories: density-based methods and k-means-based methods. A detailed survey on both categories can be found in [15]. Density-based methods assume that a cluster is a data space region in which the element distribution is dense and is separated from other clusters by regions of low density. Examples of such algorithms include CLIQUE [5], P3C [17], and 4C [8]. K-means based algorithms come closer to the nature of our framework for community extraction of a trending topic. In the k-means or the k-medoids [14] methods k cluster centroids are initially picked and through an iterative process, where each datapoint is assigned to its closest cluster centroid, convergence is eventually achieved – or a maximum number of iterations is reached. Subspace clustering methods that follow this approach include PROCLUS [3], and ORCLUS [4]. The same algorithmic principles are also used to solve the frequent itemset and association rule mining problems (e.g. a-priori pruning in used in both [6] and CLIQUE [5]).

Our approach is similar to the k-means based methods in the sense that k datapoints are also randomly picked at the beginning, although this number, k, does not dictate the final number of the extracted communities which, in fact, might be less than k. In contrast, our method probabilistically locates the sub-space of a community interested in a Trending Topic starting with these k random datapoints and then climbs through the attribute lattice to identify the interested community. Thus, we can detect focused communities quickly, a task that is possible but inefficient through existing sub-space clustering and association rule mining techniques.

Missing value imputation (handling) is important given the nature of social data, something that is consistently overseen in traditional subspace clustering and frequent itemset methods [13]. Typically, records with missing values are completely removed and algorithms are then applied only on the remaining or a subset of the records. Our algorithmic framework includes a stochastic methodology to deal with missing values in a less severe way and yields results with high confidence that otherwise could be discarded if records were completely removed.

Similar to our approach, probabilistic or Monte Carlo based methods for community extraction have also been explored in Perozzi et al. [19]. They study extracting community attributes that form highly connected subgraphs within the social network. To detect the correct values for each attribute they utilize Monte Carlo sampling to randomly select values until a connected subgraph is formed. Our approach seeds the process by sampling k datapoints from a trending topic's population. This leads to a much more agile and efficient attribute value selection process.

3. FOCUSED COMMUNITIES

The process of identifying focused communities has to be applied on each topic individually. To extract and understand the underlying communities interested in a particular topic, T, two pieces of information are necessary. 1) The topic population P which includes every social posting that mentions topic T. We will refer to these social postings using the general term *datapoints* but in the specific case of Twitter they are called *tweets*. 2) The corresponding social characteristics (attribute values) for every datapoint. These attributes can include user demographics like Location, Age, Gender, Race, or characteristics like political affiliation, supporting soccer team, hobbies, etc. Each user that mentions a topic can be represented by an attribute value vector. For example, a hypothetical 5-dimensional attribute vector could be: [Location: Los Angeles, Age: 18, Gender: Male, Citizenship: USA, Political Affiliation: Republican]. Certain attributes can be hierarchical, like Location or Age. If a user lives in Los Angeles, then she also lives in California, or USA, or the World. If a user is 15 years old then she also belongs in the "teenager" age group. Ultimately, given the population of a topic T, we want to extract a value for each attribute in order to discover a "maximally focused community" interested in topic T. In the currrent section we will formally define what a focused and maximally focused community is.

Let D be a domain with N attributes. Each attribute $a_i \in D$ has a finite set of values V_{a_i}. Let P be a set of datapoints where each datapoint is represented by a vector of N attribute values $v_i \in V_{a_i}$. We refer to any vector of attribute values as a *tuple*; therefore, any datapoint is considered a tuple. The *support* of a single attribute value is equal to the number of datapoints in P that contain the value. The *support* of a tuple instance is equal to the number of datapoints with values that match the values of the tuple.

Categorical attribute values may follow a tree-like hierarchical pattern. The Location attribute can be described using a tree hierarchy of 4 levels: city, region/state/province, country, and "World-wide". The Ethnicity attribute can be described using a tree hierarchy of 3 levels: ethnicity, minority/non minority, and "Any ethnicity". Values in each level of the hierarchy are connected to a single ancestor from the previous level and to an arbitrary number of successors in the next level (which can be zero for the values of the bottom level). The root of the hierarchy is symbolized with the value "*". Note that every attribute can be described at least by the trivial hierarchy of 2 levels where the bottom level contains all the values and the top level contains just "*". Numerical attribute values can be viewed as hierarchical attributes as well. Using a radius r the hierarchical ancestor of a numerical value v can be dynamically estimated as the range $[v-r, v+r]$. Alternatively, the values of a numerical attribute can be discretized so it becomes categorical. In the current work we focused on categorical attributes but the proposed algorithm works also with numerical attributes.

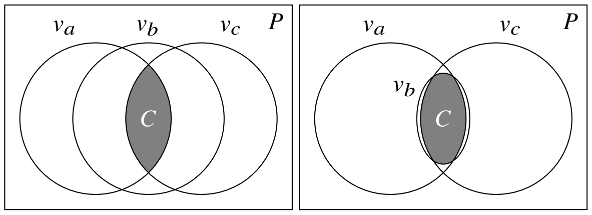
We define a *homophilic* community C to be a partition of the population P for a specific topic T which share common values for a subset of the N attributes. For example, a homophilic community can include all the users who live in Los Angeles and are male. Homophilic communities are important because *homophily* dictates that every member of the community interested in T share some common characteristics. However, the community members are not necessarily connected in the social graph. From this point on we will refer to homophilic communities simply as *communities*.

For a given topic with a group of datapoints P, a community $C \subseteq P$ is defined by N attribute values $[v_1 \in a_1, v_2 \in a_2, ..., v_N \in a_N]$ and can also be represented as a tuple. We will use the symbol C to refer to the members of the community and C_t to refer to the tuple with the attribute values that define the community. We define a community C to be *focused* if there is at least one attribute value $v \in C_t$ which no datapoint in $P - C$ shares. This attribute value v is an exclusive feature of the community: every member of the community has this value because of the homophily property but no other person in P shares it. Let P_v be the set of all datapoints in P that contain the attribute value v. The following must hold for C to be *focused*: $\exists v \in C_t$ so that $P_v \equiv C$.

Figure 1 illustrates the difference between an arbitrary community (non focused) and a *focused* community with three attributes. As an example we can assume that attribute a is Location with value v_a equal to Los Angeles, attribute b is Age with value v_b equal to 18 years old, and attribute c is Gender with value v_c equal to Male. In the first case, the population corresponding to the intersection of the three attributes defines a non focused community, ie., 18 year old males who live in Los Angeles. In the second case, the population corresponding to the attribute $v_c \equiv$ 18 years old is almost identical to the intersection of all three attributes. Therefore, the support of the Los Angeles male community is almost equal to the number of users in P that are 18 years old, since almost none else in $P - C$ has this age.

We can relax the definition of a focused community by introducing a relaxation threshold ϵ so that we can measure how close a community is to being perfectly focused. The support of a community C must be smaller or equal to the size of P_v: $|C| \leq |P_v|$. If $P_v \neq \emptyset$ this inequality can be re-written as:

$$\frac{|C|}{|P_v|} \leq 1 \implies \frac{|C|}{|P_v|} - 1 \leq 0 \qquad (1)$$



(a) Non focused community (b) Focused community ($\epsilon > 0$)

Figure 1: Illustration of a non focused community (a), which is the simple intersection of three attribute values v_a, v_b, and v_c. A focused community (b) has at least one attribute (v_b) that is as close to the intersection of v_a, v_b, and v_c, as ϵ permits.

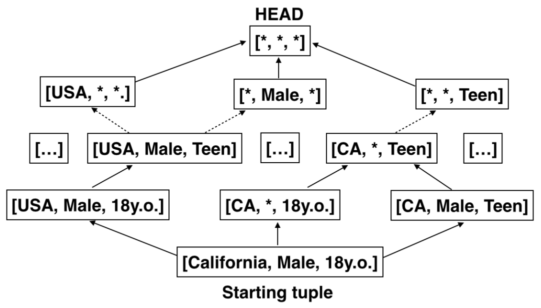


Figure 2: Partial view of the attribute lattice. Two connected nodes (solid arrow) in the lattice indicate that a tuple can be reached from the other through a single attribute generalization. A dashed arrow indicates that two nodes have other nodes between which are omitted due to space restrictions.

For a focused community where $P_v \equiv C$, Inequality (1) must be exactly equal to 0. With the introduction of the relaxation threshold ϵ inequality (1) becomes:

$$\left| 1 - \frac{|C|}{|P_v|} \right| \leq \epsilon \qquad (2)$$

We call the absolute value of the fraction at the left side of the inequality the *focus metric* of the community. A value of 0 indicates that the community is perfectly focused. A value above ϵ indicates that the community is not focused.

The relaxation of the focused community definition is necessary for the discovery of focused communities in the presence of data noise or missing values. The threshold ϵ practically defines how many datapoints in P might share the "exclusive" feature of the focused community without being part of it.

The generalization of an attribute value v is the process of replacing the value v with a direct ancestor of v in the attribute's hierarchy. The generalization of any value except "*" is possible. The root value "*" cannot be generalized since it has no ancestors. We denote the case of a missing attribute value using the "$\perp$" operator (bottom). A "$\perp$" value can be directly generalized to "*" through a single generalization step no matter how high the attribute hierarchy is. In the general case, an attribute a can be generalized from value v_a to value v_b if v_b *precedes* or is equal to v_a in attribute a's hierarchy. We denote this relation between v_a and v_b using the operators $\succeq$ (succeeds) and $\preceq$ (precedes): $v_b \preceq v_a$ or $v_a \succeq v_b$.

As an example, for the Location attribute the following relations are true: Los Angeles $\succeq$ Los Angeles, Los Angeles $\succeq$ California, Los Angeles $\succeq$ USA, California $\succeq$ USA, California $\succeq$ *.

The support of a generalized attribute value in P is equal to the number of datapoints that contain any successor of the value. For example, in a two-dimensional space, the tuple [Location:California, Gender:*] matches datapoints like [Los Angeles, Male] or [San Francisco, Female]. The tuple that contains all the hierarchy roots is called *HEAD*: $HEAD \equiv [*,*,...,*,...,*]$. The *HEAD* tuple matches every datapoint in P: $|HEAD| = |P|$. Figure 2 shows an example of the formed lattice given a specific starting tuple with three attributes: Location, Gender, and Age. Connected nodes are reachable through a series of attribute value generalizations (*climbing*).

Given a focused community C, it is possible to reach another focused community C' by generalizing one or more attributes ($C \succ C'$). A *maximally focused community* is a focused community that has no attribute value which can be generalized to obtain another focused community. Any two maximally focused communities in P are guaranteed to share a number of members between 0 and $\epsilon|P|$ (upper bound). When $\epsilon \to 0$, any two maximally focused communities should share no common members. This is an important characteristic since it leads to the extraction of non overlapping (partitioned) groups, a desirable property in trend analysis [3].

Since every single tuple with unique attribute values is a potentially self-contained focused community, we further require a focused community to meet a minimum support requirement, relative to the population P. More specifically, we introduce a support threshold $\xi \leq 1$ so that every *maximally focused community* has support of at least $\xi|P|$.

4. ALGORITHM DESCRIPTION

The proposed algorithm aims to extract a new topic ranking that utilizes the identified focused communities. Given a set of topics, all the maximally focused communities of each topic with a focus metric less or equal to ϵ and support greater or equal to ξ are extracted. The community characteristics are then used to calculate a score which leads to a novel topic ranking. The output of the algorithm is the ranked list of topics and the extracted maximally focused communities interested in each topic.

4.1 Algorithm Overview

Figure 3 illustrates the overview of the algorithm. Given a stream of datapoints being traversed through a *sliding window*, topics are identified (keywords, entities, phrases, hashtags) and the datapoints of the window are grouped by topic. All the topics with at least a minimum number of datapoints form the pool of candidate topics. For each topic in the candidate topics, the focused communities are extracted through the Sample&Climb module (dashed rectangle in Figure 3). The extraction of a maximally focused community is an optimization problem: find a combination of attribute values (tuple C_t) that maximizes the size of the community defined by C_t, while minimizing the focus metric (Equation (2) from Section 3). The Sample & Climb algorithm selects a random sample of datapoints from P (seeding phase) and uses each datapoint as a starting point to reach the attribute values of a focused community through a series of value generalizations (climbing phase). The output of the Sample&Climb module is used to calculate a topic score for each topic. These scores are then used to rank the pool of candidate topics, and result in a top-k list of trending topics. Topic scoring is performed by normalizing the raw frequency of a topic with the *relative popularity* and *exclusivity* of the corresponding focused community. The intuition behind such a scoring is to identify surprising communities, i.e. communities that are relatively more popular in a topic's population than the global population, or communities with exclusive focus on a topic.

As an example, the Twitter hashtag "#ObamaInThreeWords" has a focused community that includes supporters of the Republican Party (Political affiliation), that are Male (Gender), between the

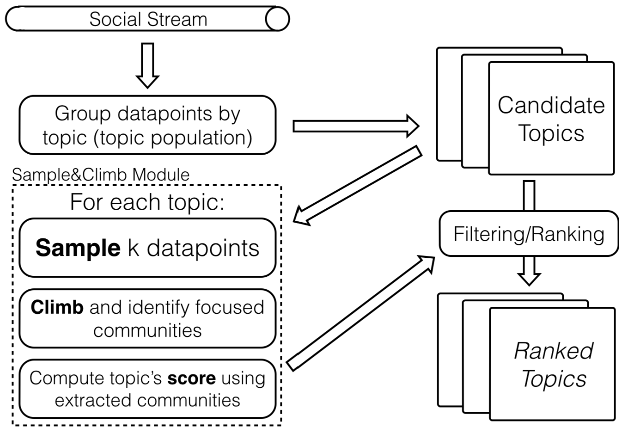

Social Stream
Group datapoints by topic (topic population)
Candidate Topics
Sample&Climb Module
For each topic:
Sample k datapoints
Climb and identify focused communities
Compute topic's score using extracted communities
Filtering/Ranking
Ranked Topics

Figure 3: Algorithm overview.

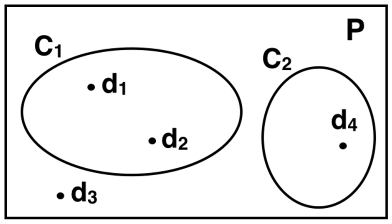


Figure 4: Seeding phase example.

ages 19-22 (Age), and that live in the United States (Location). These characteristics can boost the topic to the top 10 of the list of ranked topics, even with just 246 mentions; a population size that would not be significant for general trend detection algorithms.

In Subsections 4.2 and 4.3 we describe how the focused community extraction (Sample&Climb) is performed *per topic* and in Subsection 4.4 we describe the scoring and ranking process which is applied on all candidate topics of the sliding window.

4.2 Sample&Climb Seeding Phase

The Seeding phase must efficiently bootstrap the optimization problem of extracting a tuple that defines a topic's maximally focused community. To that end, we *uniformly* sample k tuples from P (datapoints) and create a new set S; every tuple $t \in S$ is then fed to the climbing phase which will reach a potential maximally focused community. If the sampled tuple is actually a member of a maximally focused community (checked by Equation 2), the climbing phase should extract the community. If the sampled tuple is not a member of any focused community the climbing phase will not extract a community. The intuition behind this approach is to probabilistically select datapoints that might belong to a maximally focused community. This intuition is visualized in Figure 4 where we assume that in a population P two *focused* communities C_1 and C_2 exist. The sampling of datapoints d_1 or d_2 can enable the extraction of community C_1. The sampling of datapoint d_4 can enable the extraction of community C_2. The sampling of datapoint d_3 does not enable the extraction of any community and a different datapoint needs to be sampled. If the datapoint is indeed a member of a community, then a series of attribute generalizations and focus metric computations can lead us to the actual attribute values of the community. For example, if the following focused community exists: [Location: USA, Gender: *, Age: 13-18] and the datapoint: [Santa Barbara, Male, 18] is randomly selected then the location value can be generalized twice (Santa Barbara $\to$ California $\to$ USA), the gender value once (Male → *), and the age value once (18 → 18-23) to reach the tuple that describes the community.

When a sampled tuple successfully leads to the extraction of a maximally focused community, the result is saved. If the next sampled tuple succeeds an already extracted community, by a previous iteration in the seeding phase, then the tuple is skipped since it can only lead to a known community and would be a waste of resources to process it. Pseudocode for the *seeding* phase is provided in Algorithm 1. Line 5 tests if the new sampled tuple is succeeds an already extracted community c. If the tuple is already a successor of an extracted community, the climbing phase is skipped since it will yield the same result given that the climbing process is deterministic. The returned result of the climbing phase is a maximally focused community if climbing was successful, or equal to *NULL* if a focused community could not be extracted (line 8).

Algorithm 1: Seeding phase

```
  Data: Tuples $P$, attribute hierarchies $H[N]$
  Result: Set of maximally focused communities $C$
1 begin
2     $C \leftarrow \{\}$;
3     $S \leftarrow sample(P)$;
4     for $t \in S$ do
          // If $t$ succeeds any community $c$ in $C$,
          // skip it (optimization).
5         if $\exists c \in C \;\; t \geq c$ then
6             continue;
7         $c \leftarrow climb(t,P,H)$;
          // $c$ is a community tuple
8         if $c \neq$ NULL then
9             $C \leftarrow C \cup \{c\}$;
```

Based on the desired success probability of the seeding phase p_b, the appropriate minimum size of the sample S can be determined. Let k be the number of datapoints sampled during the Seeding phase and C a unique maximally focused community in P.

The sampling of datapoints can be simulated through a series of Bernoulli trials where success is defined as the selection of a datapoint tuple t so that $t \geq C_t$. The number of trials is equal to the size of the sample: $|S| = k$. The probability of success in a single trial is equal to $p = |C|/|P|$. The probability of *at least one success out of k trials* (we can assume that P is large enough for the trials to be independent even without replacement) is equal to 1 minus the probability of getting 0 successes. This probability is defined by the geometric equation that describes the CDF of k Bernoulli trials: $1-(1-p)^k$. Therefore, we have:

$$p_b = 1-(1-|C|/|P|)^k = 1-(1-p)^k \tag{3}$$

We want to find the *minimum* value of k so that the right hand of Equation (3) is greater or equal to p_b. Let $q = 1-p$ be the probability of failure in a single trial.

$$p_b \leq 1-(1-p)^k \implies q^k \leq 1-p_b \underset{q<1}{\implies}$$

$$k \geq \log_q(1-p_b) \implies k \geq \frac{\log(1-p_b)}{\log(q)} \underset{argmin}{\implies} k = \left\lceil \frac{\log(1-p_b)}{\log(q)} \right\rceil$$

Note that k is not directly dependent to the size of the population P, only on the probability of success p_b. As an example, to find focused communities with at least 30% the size of population P and with success probability $p_b = .99$ we need at least 13 samples. For communities with size 70% or more and the same probability p_b we need only 4 uniformly selected samples.

4.3 Sample&Climb Climbing Phase

The Climbing phase follows the Seeding phase by consuming the sampled datapoint and producing a maximally focused community. More specifically, a tuple t is received from the Seeding phase and the focus metric from Equation (2) is utilized to *climb* the *lattice* (see Figure 2) from t to a new tuple $t' \leq t$, so that the support of t' in P is maximized and is at least ξ, and t's focus metric remains below the relaxation threshold ϵ. Similar to hill-climbing techniques, in every new iteration a new neighbor of the current solution is generated until an acceptable solution is reached. A tuple t has N possible neighbors: each one can be reached by generalizing a different attribute value of t.

4.3.1 Basic Climbing Approach

The pseudocode in Algorithm 2 describes this process. Starting from a tuple t, a new neighbor is produced in every iteration till a maximally focused community or a HEAD tuple is reached. *HEAD* represents the unique tuple that has all of its attribute values fully generalized: $HEAD \equiv [*,*,\ldots,*,\ldots,*]$.

An accepted solution (focused community) is reached when both conditions in line 5 in Algorithm 2 are satisfied (focus metric and support). These two conditions alone do not guarantee maximality therefore the algorithm will not return at this point but will continue until the *HEAD* is reached and at this point will return the most recent accepted value for t'. In line 7 the next attribute for generalization is selected: a_g. Different selection policies will yield different results and offer different guarantees. Using the selected attribute, a new tuple t_{temp} is generated, identical to the previous t_{temp} on all attributes except a_g, which gets generalized.

Since the climbing process always follows an upward path – a neighbor is created only by *generalizing* a single attribute – there is a well defined maximum number of iterations, equal to:

$$\sum_{i=1}^{N}(H[i].numLevels-1)$$

where $H[i].numLevels$ is the number of hierarchical levels for the i^{th} attribute. This sum can be approximated by $O(N)$. However, the selection policy for the next attribute to generalize has a significant impact on the performance of extracting a tuple t' that eventually corresponds to a maximally focused community. We will first discuss the *exact selection policy* that guarantees the discovery of a maximally focused community and then propose a *greedy policy* for a more efficient selection.

Algorithm 2: Climbing phase

```
  Data: Attribute tuple $t$, all tuples $P$, attribute hierarchies $H[N]$
  Result: Maximally generalized tuple $t'$
1 begin
2     $t_{temp} \leftarrow t$;
3     $t' \leftarrow NULL$;
4     while $t_{temp} \neq$ HEAD do
5         if $focus(t_{temp},P) \leq \epsilon$ and $support(t_{temp},P) \geq \xi|P|$
          then
6             $t' \leftarrow t_{temp}$;
7         $a_g \leftarrow getNextAttributeToGeneralize(t_{temp},P,H)$;
          // Replace $a_g$ with parent of $a_g$ in $t_{temp}$
8         $t_{temp} \leftarrow \{a \in t_{temp} | a_g \leftarrow H.parentValue(a_g)\}$;
```

We start with a policy for selecting the next attribute of a tuple t to generalize (a_g) which guarantees reaching the correct attribute values of a maximally focused community C, if one exists *and* $t \geq C_t$. This policy involves choosing the attribute with a value that when generalized to the next hierarchical level results in the largest gain on the new tuple's support.

$$\underset{a_g \in t}{\operatorname{argmax}}\, support(\{a \in t | a_g.value \leftarrow H.parent(a_g.value)\}, P)$$

where $a_g.value$ is the current value of the attribute a_g (e.g. if the attribute is Location, it could be Los Angeles or California). The argmax function returns the attribute value for which the tuple support attains its maximum value. The main drawback of this approach is the need to calculate the support of N different tuples in each iteration. Since a total of $O(N)$ iterations is required to reach a maximally focused community, the total time complexity becomes quadratic ($O(N^2)$).

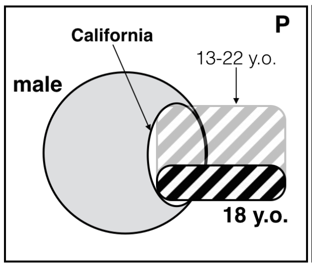
Theorem 1. *The attribute generalization policy will lead to a maximally focused community C if the starting tuple $t \geq C_t$.*

Proof. Let C be a *maximally focused* community with size $|C| \geq \xi P$ and with a focus metric less than ϵ. Let t be a starting tuple with n attribute values so that $t \geq C_t$ (C_t can be reached by generalizing attribute values in t). C_t can be correctly reached from t if after $O(n)$ iterations t' becomes C_t. The only way that a selection policy can fail to reach C_t, during the climb from t to *HEAD*, is if one attribute value of t gets generalized beyond the corresponding attribute value of C_t. To prove the theorem we need to show that the selection policy will never select to generalize an attribute of t that has reached the same value with the corresponding attribute of C_t.

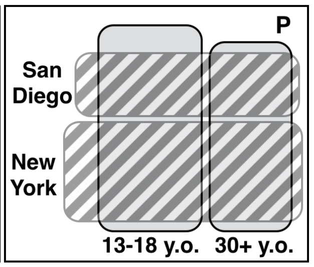
Let t_i and t_j be the i^{th} and j^{th} attribute values of t, and c_i and c_j the i^{th} and j^{th} attribute values of C_t. Assume that t_i has reached the same value with c_i, and that t_j has not: $t_j \succ c_j$. C is a maximally focused community so given the maximality property any further generalization of an attribute in C_t cannot lead to a new focused community. Therefore, the generalization of t_i will not increase the support of t while the selection of attribute t_j (or any other attribute not generalized to the same level with C_t) will result in a new tuple t' with an increased support. Thus, as long as there are attribute values in t that are not generalized to the same level of C_t, their selection will always be prioritized over attribute values that have reached the correct level of generalization, till all of them are correctly generalized. □

4.3.2 Greedy Attribute Selection Approach

To improve the efficiency of the focused community extraction algorithm and render it scalable, we propose a *greedy* policy to select the attribute a_g: choose the attribute value of the tuple that has the smallest support in P (argmin). The intuition behind this approach is that in a homophilic community defined by N characteristics, the characteristic with the smallest support is the one that most likely constrains the size of the community the most. More specifically, the support of a tuple t is equal to the size of the intersection of the N attribute values in t and the size of this intersection is bounded by the support of the attribute value with the smallest support. The only way to increase this bound is by generalizing the smallest attribute in order to match more datapoints. This observation is illustrated in Figure 1b: if either of the two values v_a or v_c is generalized, the intersection of the three attributes will still be limited by value v_b and remain almost the same size. Instead, the generalization of v_b has the greatest potential to increase the intersection of the three attributes. The mathematical form of this policy is the following:



(a)



(b)

Figure 5: (a) An example of the specific case for which the greedy attribute selection policy can fail to select the correct attribute.
(b) An example of a community with split attribute values.

$$\underset{a_g \in t}{\operatorname{argmin}}\, support(a_g.value, P) \tag{4}$$

The main benefit of the greedy policy over the exact approach, is the improvement of time complexity. While we need to compute the support of N attribute values in each iteration, we do not need to actually perform the operation for every attribute value in every iteration, since only one of the support values changes: the support of attribute a_g which gets generalized. All other attribute values of the tuple remain the same therefore their support does not change in the next iteration. Storing in memory the support of the $N-1$ attribute values only a single support calculation needs to be performed per iteration. With an $O(1)$ time complexity per iteration the total climbing time complexity becomes $O(N)$.

The downside of the greedy policy is that it does not offer specific guarantees for reaching a maximally focused community. In fact, there is a specific case where the greedy approach might choose to generalize an attribute value that is not the correct one. Figure 5a visualizes this scenario where all of the necessary requirements to fail are met: Assuming that a correct community exists and is [male, California, 13-22], if the climbing process seeded by the tuple [male,San Francisco,18] has currently reached tuple [male, California, 18] then the greedy policy will select attribute value *California* for generalization since it has the smallest support. However, the correct choice would be to generalize the value 18 to 13-22 in order to reach the correct focused community. If California is generalized, the focused community will not be reached.

4.3.3 Climbing Phase Extension

The Climbing algorithm, as described so far, requires that a community characteristic is precisely described through the hierarchy. However, there can be cases where a community expands in a level that is not present as a node in the hierarchy. For example, a focused community might be exclusively split in two remote locations: San Diego and New York. The only way to describe both cities with one attribute value would be to generalize them to the country level: USA. Ideally, we want to avoid this generalization since USA also indicates a community that expands in the whole country and not only San Diego and New York.

If this scenario occurs for a single attribute only, then two individual focused communities can be extracted. When this pattern is observed in more than one attributes then the extraction of individual communities might not be possible. Figure 5b illustrates such an example where a focused community resides in two separate cities *and* two disjoint age groups. Note that the intersection between a single location value and single age value does not form a focused community by itself (based on the definition). A focused community can only be reached if the location attribute has value $SanDiego|NewYork$ and the age attribute has value $13-18|30+$, where the "|" operator stands for "or". We will be referring to such values as "split attribute values" and to the individual values of a split attribute value as *siblings*.

The generalization of an attribute value from v to v', given a population of datapoints P, should be *acceptable* if the majority of attribute values directly succeeding v' are also represented in P. For example, we can generalize the value Male to "*" only if there is a significant number of females in the population as well (at least $\xi\%$). Or generalize the value Los Angeles to California only if the majority of cities in California also have a presence is P. Calculating the presence of a majority can be computationally expensive so instead we utilize a simpler *heuristic*: If at least m different attribute values directly succeeding v' are also represented in P then the generalization is acceptable. In the experiments, we used a value of $m = 3$ for all attributes except Location which had a value of $m = 7$. This results in an accuracy of *93%*, with the exact approach to test majority being the ground truth.

To extract communities with split attribute values the climbing algorithm has to be altered to handle such cases. The necessary changes are shown in Algorithm 3 which *replace* lines 7 and 8 in Algorithm 2. The intuition is that when an attribute value generalization is not *acceptable* there might still be other sibling values which are part of a homophilic community. By sampling a few datapoints from P we can discover those sibling values (lines 5 and 6). It is still necessary to generalize these sibling values (line 8) with regards to t_{temp} to eventually reach a maximally community. For example, let a focused community expand in two regions *California* and *Florida* and only there. Starting from the attribute value a_g = *Los Angeles*, we can generalize to *California* ($a_g \leftarrow$ *California*) but generalizing *California* to *USA* is not acceptable since the whole country of 50 states can not be represented by just two states. If a new sample from P is seeded, other cities from *California* and *Florida* will be discovered as sibling values. Cities in *California* are already part of the current a_g and can be ignored. Cities in *Florida*, like *Miami* can be accepted but should be tested for further generalization, i.e. to *Florida*. Since this generalization is acceptable, *Florida* will become a sibling value to *California*.

This algorithmic extension has no significant impact on the complexity. Though the cases of split attribute values were quite rare in our experiments, even when met, the additional complexity only includes the additional steps for checking and generalizing the new attribute siblings (line 8 in Algorithm 3). Given that there can be a maximum of $1/\xi$ siblings for each attribute the total complexity becomes $O(n/\xi)$, with $\xi \leq 1$.

Algorithm 3: Extension for handling split attribute values

```
Result: Maximally generalized tuple $t'$ with split attributes
1  if $t' \neq$ NULL and $focus(t_{temp}, P) > \epsilon$ then
2      return;
3  $a_g \leftarrow getNextAttributeToGeneralize(t_{temp}, P, H)$;
4  if generalization of $t_{temp}[a_g]$ not acceptable then
5      $S' \leftarrow sample(P)$;
6      $splitValues \leftarrow \{t[a_g] | t \in S'$ and $t \geq t_{temp}\}$;
7      for $splitValue \in splitValues$ do
8          $splitValue \leftarrow generalize(splitValue)$;
9  else
10     $splitValues = \{H.parentValue(a_g)\}$;
11 $t_{temp} \leftarrow \{a \in t_{temp} | a_g \leftarrow splitValues\}$;
```

4.4 Topic Ranking Phase

During the traversal of datapoints using a sliding window of size $|W|$, the Sample&Climb module identifies a set of topics with their corresponding maximally focused communities. This set can be quite large and may include uninteresting cases. We therefore developed a score metric to order this set in a way that ensures more interesting or surprising topics are ranked higher. Initially, a filtering pre-step is necessary to remove topics with unsurprising focused communities. For example, if a topic has a focused community of males with support 50%, and half of the total Twitter population in W is male anyway, then the focused community is actually expected. We perform a Pearson's chi-squared test to compute the anomaly of a topic's community (expected Vs. observed) and further filter-in candidate topics with significance of $p \leq 0.05$.

To obtain an interesting ranking of topics, after filtering out the topics with expected focused communities, we use a combination of two measures: Inverse Community Frequency and Relative Community Popularity. Both measures aim to normalize the raw frequency of a topic in order to boost those topics with interesting focused communities. Inverse Community Frequency (icf) is inspired by Inverse Document Frequency from text document ranking in Information Retrieval. Here we use it in a similar context: to tune down community characteristics that get associated with many topics. A community characteristic that appears in few topics only should be more interesting. Inverse Community Frequency, measures how many topics in the whole window W of datapoints also share a community characteristic. For example, the icf of location Santa Barbara will depend on how many topics in W have a focused community that contains Santa Barbara. The icf score of a community C is the product of icf scores for each attribute value in C. The **icf score** for a single attribute value a is equal to:

$$icf(a) = log \frac{N_t}{|\{T \in W | a \in C\}|}$$

where N_t is the total number of topics in W and the fraction denominator is equal to the number of topics T in W with a community C that contains the attribute value a. Relative Popularity takes values between 0 and 1 and practically compares the size of a topic's focused community with the size of the community with the same characteristics in the window W of datapoints. The **relative popularity score** is calculated as the fraction of the support of a community in P over the support of the community in W:

$$rp(C) = \frac{support(C,P)}{support(C,W)}$$

For example, if a topic is being discussed by 100 women and the number of women in W is also 100, then this community has a relative popularity of 1. The overall scoring function is based on each topic's extracted focused community C and uses both notions of relative popularity and exclusive focus:

$$score(T) = |P| \times rp(C) \times icf(C) \tag{5}$$

where C is a focused community of the topic T, P is the population of the topic. The overall score of a topic is proportional to the topic's raw popularity (size of P), the relative popularity score of the topic's community, and the icf score of its community. Using this score metric we rank the candidate topics and obtain a list of top-k Trending Topics as the final result of the algorithm.

5. HANDLING MISSING VALUES

One of the main challenges when dealing with real social datasets is the sparsity of attribute values. This observed sparsity (missing values) is due to the low recall of specific inference tasks which usually originates in the general lack of sufficient information to infer attributes with high confidence (e.g., not enough textual information to infer the age of a user). For example, it is possible that a topic is mentioned by 1000 users, and within this population 100 have complete location information and 250 have their age successfully inferred. The intersection of the users that have both location and age values extracted is even smaller.

Let P be a population of datapoints and C a maximally focused community in P if there were no missing values. Let N be the number of attributes and $m_i \in [0,1]$ the portion of missing values for each attribute a_i. The discovery of a focused community requires at least one attribute value (exclusive feature) with size equal to the size of the community, assuming 0 relaxation ($\epsilon = 0$). Note that the size of the community is equal to the number of datapoints in P that match the tuple that defines the community. In the presence of missing values, an attribute tuple will not match every datapoint that it should. For example, the tuple [California, Male, *] does not match the datapoint [Los Angeles, $\perp$, 18] because $\perp$ does not succeed Male. Therefore, if there are randomly missing values in each attribute, the observed size of the community C and the size of the exclusive feature will differ and the focus metric will not be low enough to identify the focused community.

To overcome this problem, we allow a tuple to match missing values during counting. Referring back to the previous example, we allow the tuple [California, Male, *] to match the datapoint [Los Angeles, $\perp$, 18]. This alteration fixes the issue of under-counting a tuple, but introduces over-counting: additional datapoints are now counted as part of a community. However, the community size over-estimation is statistically bounded. Let v_f be the attribute value that plays the role of the exclusive feature in the focused community C and let m_f be the ratio of missing values for the attribute a_f. The focused community can be divided in two parts: the datapoints that belong in the community and have a value v_f for the attribute a_f and the datapoints that belong in the community and have a value $\perp$ for the attribute a_f (missing value). Similarly, the datapoints outside the focused community can be divided in two parts: the datapoints that have a value $v'_f \neq v_f$ for the attribute a_f and the datapoints that have a value $\perp$ for the attribute a_f (missing value). Note that there are no datapoints outside the community with value v_f for the attribute a_f based on the definition of the focused community. The datapoints that could be mistakenly counted are the ones outside the community, with a missing value. Among these datapoints, the other $N-1$ attribute values must also match the corresponding attribute values of the community or be missing. The expected size of this subset of datapoints is bounded by:

$$m_f|P-C| \leq m_f|P| - \xi|P| = m_f(1-\xi)|P|$$

This is an upper bound of the expected count of wrongly counted datapoints since some of these datapoints will likely have a different non-missing value for some attribute $a_i \neq a_f$ and will not match to the community's tuple. In any case, in the presence of many missing values it is recommended to use a higher support threshold ξ for the correct detection of focused communities since the above value gets closer to 0 when $\xi \to 1$.

6. EXPERIMENTS AND ANALYSIS

To understand the performance and accuracy of the proposed algorithm we performed experiments on two different datasets, one real and one synthetic. The real dataset comprises of a 10% sample of the Twitter stream from 2013 and 2014 and tested the performance of the algorithm and quality of the results. The synthetic dataset was specifically constructed to examine the accuracy and recall of the proposed approach and includes a complete spectrum of scenarios — some that might be rare in a real dataset.

We introduce the details of the Twitter dataset and attribute extraction process in Subsection 6.1, then offer a qualitative analysis of the produced results in Subsection 6.2. In Subsection 6.3 we offer a quantitative evaluation of the Twitter results to measure their usefulness and attractiveness. Finally, we discuss the performance of the algorithm on synthetic data in Subsection 6.4.

6.1 The Twitter Dataset

The used Twitter dataset contains a uniform 10% sample of all the tweets and Twitter users from the following two periods: September 12 to October 26 of 2013 (45 days) and April 16 to May 24 of 2014 (39 days). The pool of topics contains every mentioned hashtag or capitalized entity from the tweets' raw text. The extracted tweet features include location, the list of external user mentions (@-replies), the device the tweet was posted from (e.g. iPhone, Android, web browser), and the general sentiment. Location extraction was done on (1) the tweet level using Twitter's geo-tagging mechanism, and to further improve the recall, on (2) the user level using a user-provided raw text field (similarly to [27, 2]). To infer location based on the user's field we applied a simple but precise pattern matching process that could identify location patterns like: "City, Region, Country", or "Region, Country", or just "Country". To validate the patterns we used a Location hierarchy provided by the MaxMind database [12]. The user device was extracted from the available information provided by the Twitter API [26]. To infer the sentiment of a tweet we used the SentiStrength tool [24]. Note that not all features were available in every tweet; for example, less than 2% of the tweets had an explicit location tag or non-neutral sentiment (missing values).

Meaningful and interesting community extraction requires a diverse set of user characteristics/demographics. To expand the number of extracted attributes from the Twitter dataset we additionally infer the users' age, gender, political affiliation, and sports team preference. To extract gender and age we applied existing language models extracted from Schwartz et al. [23] on social media data. To apply the models we gathered all the tweets of every user for each of the two analyzed periods of data. While this is an expensive process, especially space-wise, it can be done offline and does not affect the complexity of our Sample&Climb algorithm. For political affiliation we gathered the Twitter accounts associated with the three most popular US political parties: Democratics, Republicans, and Libertarians. Then, a user's political affiliation was determined based on their interactions (@-replies) with these accounts. Similarly for sports, we collected the Twitter accounts of teams, players, and coaches for the following four US professional sports: Baseball, Basketball, Football, and Hockey. For every sport, a user's team preference was inferred based on their interactions with each team's accounts. Table 1 shows the **precision** of each inference task. Given that the language models are in English, age and gender inference only works for english speaking users. Similarly, political affiliation and sports teams are focused on users within the United States and Canada. Precision was calculated manually on a sample of 100 users from each category. However, for the age and gender inference we list the calculated precision from Schwartz et al. Note that their models were tested on Facebook data and not on Twitter, so precision might differ slightly.

In total, the experimental setup contained **10 attributes**: 1) Location (either from the tweet or the user), 2) Age, 3) Gender, 4) Political affiliation, 5) Baseball team, 6) Basketball team, 7) Football team, 8) Hockey team, 9) Tweeting device (e.g. iPhone), and 10) Sentiment. While the sentiment is not strictly a user characteristic, it helps with the interpretation of the results by hinting at the attitude of the community towards the topic. Apart from Loca-

Inferred Attribute	Source	Precision
Location	Geo-tagged tweets	100%
Location	User specified location	96.1%
Device	Twitter API	100%
Gender	Schwartz et al. [23]	91.9%
Age	Schwartz et al. [23]	.84 (R value)
Political Affiliation	Interaction with parties	83.4%
Baseball Team	Interaction with teams	91.5%
Basketball Team	Interaction with teams	93.7%
Football Team	Interaction with teams	87.8%
Hockey Team	Interaction with teams	95.0%
Sentiment	SentiStrength [24]	68.7%

Table 1: Inference accuracy of Twitter attributes

tion, and Device all attribute hierarchies have only 2 levels (trivial hierarchy). Specifically, the Location hierarchy has 4 levels: city, region, country, and "*". The Device hierarchy has 3 levels: specific device, "mobile"/"desktop", and "*".

Settings. The execution of the community extraction algorithm was applied on the stream of tweets using a sliding window of size 500,000. On a typical day this amount of tweets can be produced within two minutes of real time. For every new window new topics get introduced, existing topics receive additional mentions, and old topics get evicted. To reduce noise, candidate topics are required to have at least 50 mentions during the window. The rest of the algorithm settings are: selection policy: greedy, seeding sample size (k): 20 datapoints, $\epsilon : 0.15$, $\xi : 0.3$. The choice of ϵ is based on the fact that Twitter data is noisy and the community extraction should be relaxed enough to accommodate this noise. The value of the support threshold ξ is based on the average population of a Trending Topic on Twitter, which is usually between 1K and 200K tweets, therefore we can expect communities of size between 300 and 60K users (smaller communities would not be interesting). Note that our dataset only contains 10% of the tweets in Twitter Firehose, therefore all reported size numbers in the results section should be $1/10$ of what we would get on the whole Twitter dataset.

6.2 Qualitative Evaluation of Twitter Results

The final product of this experiment, is an ordered list of Trending Topics and the corresponding maximally focused communities that were extracted, in each window. The extracted community of a topic might differ between different windows as different or additional users mention the topic and the population changes.

We cherry-picked some topics to showcase interesting behaviors and qualitatively argue that the results actually make sense. These topics are listed in Tables 2 (general interest trends) and 3 (trends with a sports related focus). A "*" value indicates that the attribute got generalized to its top level of the hierarchy. A "$\perp$" value indicates that there was not enough information to extract a specific attribute value (due to missing values). Attribute values for Device and Basketball team are omitted due to lack of space. Topics that appear twice are taken from different windows, or even days, and are listed to show the dynamic nature of focused communities as the topic population grows or just changes.

One interesting result includes hashtag *#DisneySide* which was a social media campaign by US Disney Parks. Disney asked from fans to tweet photos of their 'Disney Side' from their visit to a Disney theme park. During the first day, most of the tweets occurred in the two cities where a Disney park is located: Anaheim, California and Orlando, Florida (split values). The next day, the campaign audience expanded to the include whole states of California and Florida.

Other topics that are not self explanatory are briefly discussed here: Topic *#PS4* stands for 'Play Station 4', the gaming console. *#Bring1DtoGreece* stands for 'Bring 1Direction to Greece' where One Direction is a globally popular boy band. *#NavyYardShooting* is about a mass shooting that occurred on September 16, 2013 on a US military base at Washington, D.C. Topic *#ReasonsToVisitEgypt* started as a touristic campaign for Egypt but got highjacked with citizens' complains, hence the negative sentiment. Topic *Penn State* is related to a college football match where college Penn State played in Bloomington, Indiana. Indianapolis is also in the results since it is the capital of the Indiana state and it is very likely that fans/students might have specified it as their location. *#auspol* is a hashtag about police brutality in Australia. In the early stages of the trend it was mostly mentioned in the two largest cities of Australia but as it became popular, the whole country became the focused community. The topic #FreeJustina is about an arrested female teen named Justina from Boston. We observe that women in the area of Boston, MA, that also support the Democratic party, showed their solidarity to Justina through this hashtag. *#cdnpoli* stands for 'generic canadian political issues' and this is why the topic's location is in Canada. *#AZvsNO* stands for 'Arizona vs New Orleans' and it is an example were the extracted community has a split value in the Football attribute. Finally, *#Boston* is an interesting case with a focused community that consisted of users which were fans of local teams in all three sports attributes.

An interesting general observation in the Twitter results is that for topics related to activism or politics, usually the male demographic was prevalent (with exceptions like #FreeJustina). For topics related to memes or pop culture, usually the female demographic was prevalent. A similar pattern based on age was not observed.

To compare the community-based topics with a baseline, we list the top 10 topics based on each ranking: raw popularity ranking and community-based ranking (relative community popularity and inverse community frequency). Both topic lists are extracted from the same randomly selected window. The results are shown in Table 4. Raw popularity results in popular but not necessarily informative or diverse trending topics (e.g. #ipad). On the other hand, topics ranked based on their focused community characteristics appear to be more interesting and are further enhanced with the information of *who* is interested in each topic. The average similarity between the two rankings was measured with the *Set Based Measure* described in [28] (*average ranked correlation* cannot be applied since our method filters out some topics). In general, the idea is to determine the fraction of content overlapping (set intersection) at different depths. The average set based measure with a depth of 20 is equal to 0.089 while with a depth of 10 is 0. These values indicate that the two rankings produce heterogeneous top-k lists with minimal overlap and signifies that surprising communities do not have a focus on extremely popular topics.

6.3 Quantitative Evaluation of Twitter Results

As with many unsupervised learning tasks, evaluating the produced results is a challenging task. In content recommendation systems used by real users, one can run A/B tests to compare the success of the algorithm with a baseline. To evaluate the community-based topics in terms of potential usefulness and interestingness we (a) measure the entropy of the results as an objective quantitative measure, and (b) asked human evaluators to choose their favorite topics from a mixed pool of trending topics.

Using the notions of Self-information and Entropy from Information Theory we provide a measure of the information content for community-based trending topics. Self-information can capture how surprising an event is based on the probability of the event.

Table 2: Examples of general Trending Topics.

Topic	Size	Sentiment	Location	Age	Gender	Politics	Size
#PS4	114	*	*	13-18	Male	⊥	111
#Bring1DtoGreece	117	*	Athens:AT:GR	13-18	Female	⊥	110
#NavyYardShooting	5427	Negative	US	19-22	*	*	5218
#ReasonsToVisitEgypt	50	Negative	AL:EG, CA:EG	*	*	⊥	49
#DisneySide (day 1)	54	Positive	Anaheim:CA:US, Orlando:FL:US	*	Female	⊥	50
#DisneySide (day 2)	53	*	CA:US, FL:US	*	Female	⊥	51
Penn State	64	Negative	Bloomington:IN:US, Indianapolis:IN:US	19-22	Male	*	56
#auspol	55	*	Melbourne:VIC:AU, Sydney:NSW:AU	*	Male	⊥	51
#auspol	461	Negative	AU	*	*	⊥	457
#FreeJustina	54	Negative	Boston:MA:US	*	Female	Democrats	51
#cdnpoli	151	Negative	ON:CA	23-29	Male	Republicans	139
White House	2989	*	US	*	Male	Republicans	2868
#ObamaCare	5090	Negative	US	*	Male	Republicans	4818
#ObamaInThreeWords	246	Negative	US	19-22	Male	Republicans	224

Table 3: Examples of Trending Topics in sports.

Topic	Size	Location	Age	Gender	Baseball	Football	Hockey	Size
#TMLtalk	3437	Toronto:ON:CA	19-22	*	⊥	⊥	Toronto Maple Leafs	3096
#AZvsNO	50	⊥	19-22	*	⊥	Arizona Cardinals, New Orleans Saints	⊥	50
#RedSox	528	Boston:MA:US	19-22	Male	Boston Red Sox	⊥	⊥	411
#Boston	51	⊥	⊥	⊥	Boston Red Sox	New England Patriots	Boston Bruins	51

Table 4: Comparison between raw popularity trending topics (left) and trending topics ranked by the proposed ranking score (right).

#	Topic	Size	Topic	Score	Size	Community attribute values
1	#gameinsight	8592	#auspol	62570.109	59	Sydney,AU, Negative sentiment, Male
2	# нет	5555	#miami	42330.137	50	Miami,US, mobile, 19-22, Male
3	#android	5117	#iraq	35321.723	61	Mobile, Negative sentiment, Male, Republicans
4	#androidgames	4999	#news	29135.463	958	Mobile web, negative, Male
5	#ipadgames	4057	#uk	19354.53	105	Sevenoaks,GB, Web, Male
6	#ipad	3958	#nature	17530.416	78	Browser, Positive sentiment, 23-29, Female
7	#mpn	3012	#benghazi	13585.276	59	Mobile, Negative sentiment, Male, Republicans
8	#iphonegames	2229	#tcot	10481.482	229	Negative sentiment, Male, Republicans
9	#tbt	2211	#hair	5858.8447	93	iPhone, Positive sentiment, 13-18, Female
10	#nowplaying	1972	#redsox	5346.331	60	Male, Boston Red Sox

Then, we can calculate the entropy of the experiment (extracting community-based trending topics) as the expected value of every trending topic's self-information. The self-information of the community C_T for a single topic T is $I(C_T) = -log_2(Prob(C_T))$. Intuitively, the less likely a community is to observed the higher its self-information. The prior probability of C_T can be measured in the sliding window as the percentage of datapoints that contain C_T. The entropy of the results is equal to the expected value of all topic communities: $E[I(C_T)]$. Since we are using logarithm with base 2 in the self-information calculation, the final entropy will be measured in bits. We also measured in the same way the entropy of communities associated with trends ranked by raw frequency, i.e. the more standard popularity-based trending topics. In this case, topics do not have a focused community that can be defined as a single attribute tuple. However, we can still calculate the probability of the observed population characteristics for each topic based on the prior probabilities from the sliding window.

The average entropy for the community-based topics was found to be **1.87** bits. The average entropy for frequency-based topics was found to be much lower: **0.27** bits. This indicates that the extracted topics using our method contain surprising and potentially useful communities that cannot be trivially anticipated or that are not observed in topics ranked by raw frequency.

To measure exactly how much more interesting real users would find the extracted topics we performed the following experiment with human evaluators (colleagues and Amazon Turkers): We randomly picked a timeslot from our dataset and created a pool of 10 topics, which included a random selection of top ranked topics by our algorithm and top ranked topics by raw popularity (similarly to Table 4). These 10 topics were then presented, *unlabeled and randomly ordered*, to an evaluator. We asked each evaluator to pick 5 topics (in no particular order) that they find the most interesting, where a topic is defined as interesting to a user if they would like to know more about it; get tweets that contain it, read news articles, see related images, etc.

The histogram in Figure 6 shows the results of this experiment performed on 12 Computer Science graduate students from the University of California, Santa Barbara. Topics denoted by (C) are community-based and topics denoted by (F) are frequency-based. All of the top selections ended up being community-based topics even though some evaluators were interested in simply popular topics like #ThrowbackThursday or #NowPlaying.

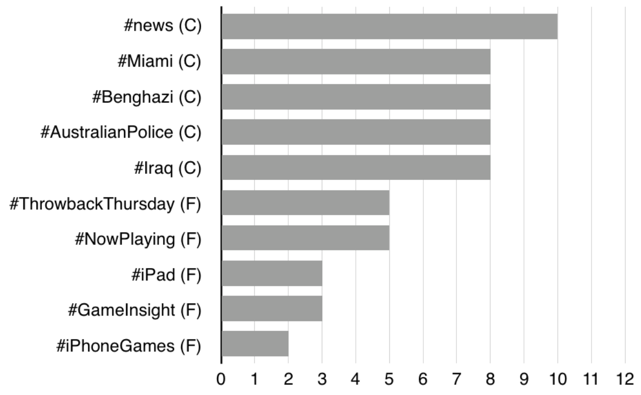


Figure 6: People's topic preference from a pool of 10 topics. All of the top ranked topics are community-based.

To reduce any bias on the reported evaluations results, we performed the same experiment with 5 topic pools on Amazon Turk with an average of 72 participants per pool and 10 topics per pool. All participating turkers were based in the US. The results are shown in Table 5. The first two rows display the percentage of community-based topics (c-topics) in the top-3 and top-5 of the evaluators' selections. On average, community-based topics were ranked in the top-3 on 73.3% of the times, and in the top-5 on 60% of the times. These two values indicate that in both the top-3 and top-5 cases, the *majority* of interesting topics was community-based. The final two rows of Table 5 show the percentages of c-topic and frequency-based topic (f-topic) selections — how many times an evaluator clicked a topic of each category as interesting. This value can also be viewed as the probability of each category/method to produce an interesting topic. On average, community-based topics have 26.86% better chance to be more interesting than raw frequency ranked topics, which shows that in most cases users will find our algorithm's results more interesting. Overall, a few topics ranked by raw frequency will still be interesting to users due to their popularity, but overall our method delivers more appealing results to the average person as represented by Amazon Turkers.

The right-most column of Table 5 displays the above statistics for the experiment we run on the CS grad students which was applied on the topics of Pool 1. It is interesting to note the difference between these values and the values of the first column of the table which shows the results of Pool 1 with Amazon Turkers. This difference indicates that a group of people with a biased interest in news (like graduate students) might find topics with a community focus more interesting than a random diverse population (like US based Amazon turkers). In future work we plan to explore personalized topic scoring that produces different rankings for different users to further increase relevance.

6.4 Experiments on Synthetic Data

The synthetic data was mainly used to test the effectiveness and efficiency of our proposed algorithm on specific scenarios. Using a pseudo-random data generation process we were able to inject communities and then test the algorithm for the expected result. The first part of the generation process includes the random building of the dimensional space: number of attributes n (between 5 and 20), possible values for each attribute a_i (between 2 and 50000), and the number of levels in the attribute's hierarchy h_i (between 2 and 5). The second part includes the creation of a community C by randomly selecting a value c_i for each attribute a_i, given equal selection probability to each level of the hierarchy h_i. The result is an array of attribute values $[c_1,c_2,...,c_n]$ that defines the expected focused community of the population. This community is also assigned a randomly selected size ratio p_C between 30% and 90% of the total size of the population. The third and final part is the creation of the population with a random size between 10,000 and 1,000,000 to simulate the numbers of Twitter Trending Topics. Every created datapoint has a probability of being part of the community C equal to p_C. If the datapoint is detemined to be part of the community, each attribute value v_i is selected randomly from the leave attributes of hierarchy h_i which are also descendants of (or equal to) c_i. If the datapoint is not part of the community, then each attribute value v_i can be uniformly selected from every possible value of attribute a_i. A total of 10000 population groups were created, each with a single maximally focused community.

The above construction process creates simple communities. To test the more rare extended version of a community where attribute values are split (like in Figure 5b), we altered 1000 cases so that the selected values in each attribute dimension were more than one; half of the times under the same direct ancestor in the hierarchy and the other half with separate direct ancestors. We limited the number of split attributes of a community to a maximum of 5. The algorithm settings that we used on the synthetic dataset were the same as with the Twitter experiments: selection policy: greedy, seeding sample size: 20 datapoints, $\epsilon : 0.15$, $\xi : 0.3$

Results: The algorithm was able to find the correct communities in each synthetic population with an accuracy of 93.1%. A community extraction was labeled as successful when the exact correct community could be identified. In the rest of the cases that failed, most of the time there would be a community attribute value or two that were further generalized than expected. By measuring the ratio of correctly extracted attribute values for each community the average accuracy is 97.2%. The running time for all 10000 cases was a little less than 10 minutes on a 2.6GHz Inter Core i5 workstation.

7. CONCLUSIONS

We study the problem of extracting multi-dimensional communities focused on individual topics by introducing the notion of a maximally focused community with properties that enable the efficient discovery of interested communities defined by a subset of social attributes. These properties led to the implementation of an algorithmic framework for the extraction of maximally focused communities of any topic with proved linear time complexity. Finally, we provide a robust ranking that boosts topics with *relatively popular* or *exclusively focused* communities through metrics adapted from Information Retrieval.

Extensive experimentation was conducted on two different datasets: one real from Twitter with data from large periods in 2013/14 and one synthetic. The results highlight the efficiency, correctness, and stability of our proposed algorithm. We are able to identify interesting communities for Trending Topics, sometimes expected and sometimes unexpected. It is interesting to observe that females in Boston, which also support the Democratic party, show their solidarity to an arrested teen (#FreeJustina). It is unexpected to discover the hijacking of a touristic hashtag in Egypt from local citizens that try to raise awareness for the country's political situation (#ReasonsToVisitEgypt). Such data can be used by advertisers, analysts, the police, or any other entity, to better understand a topic's participating population or who contributes to a trend.

8. REFERENCES

[1] H. Abdelhaq, C. Sengstock, and M. Gertz. Eventweet: Online localized event detection from twitter. *Proc. VLDB Endow.*, 6(12):1326–1329, Aug. 2013.

Table 5: Evaluation results from Amazon Turk on 5 different pools of topics.

	Pool 1	**Pool 2**	**Pool 3**	**Pool 4**	**Pool 5**	**All Pools Average**	Local (Pool 1)
% of c-topics in top-3	33.3%	100%	66.6%	100%	66.6%	73.3%	100%
% of c-topics in top-5	60%	60%	40%	80%	60%	60%	100%
% of clicks on c-topics	49.75%	54.86%	52.28%	64%	58.75%	55.92%	70%
% of clicks on f-topics	50.25%	45.14%	47.71%	36%	41.25%	44.08%	30%

[2] H. Achrekar, A. Gandhe, R. Lazarus, S.-H. Yu, and B. Liu. Predicting flu trends using twitter data. In *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*, pages 702–707. IEEE, 2011.

[3] C. C. Aggarwal, J. L. Wolf, P. S. Yu, C. Procopiuc, and J. S. Park. Fast algorithms for projected clustering. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, SIGMOD '99, pages 61–72, New York, NY, USA, 1999. ACM.

[4] C. C. Aggarwal and P. S. Yu. Finding generalized projected clusters in high dimensional spaces. *SIGMOD Rec.*, 29(2):70–81, May 2000.

[5] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, SIGMOD '98, pages 94–105, New York, NY, USA, 1998.

[6] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. *SIGMOD Rec.*, 22(2):207–216, June 1993.

[7] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida. Detecting spammers on twitter. In *In Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS)*, 2010.

[8] C. Böhm, K. Kailing, P. Kröger, and A. Zimek. Computing clusters of correlation connected objects. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, SIGMOD '04, pages 455–466, New York, NY, USA, 2004. ACM.

[9] C. Budak, D. Agrawal, and A. E. Abbadi. Structural trend analysis for online social networks. In *PVLDB 4(10)*, pages 646–656, 2011.

[10] C. Budak, T. Georgiou, D. Agrawal, and A. E. Abbadi. Geoscope: Online detection of geo-correlated information trends in social networks. In *PVLDB 7(4)*, pages 229–240, 2013.

[11] V. T. Chakaravarthy, V. Pandit, and Y. Sabharwal. Analysis of sampling techniques for association rule mining. In *Proceedings of the 12th International Conference on Database Theory*, ICDT '09, pages 276–283, New York, NY, USA, 2009. ACM.

[12] Maxmind world cities with population. `http://www.maxmind.com/app/worldcities`.

[13] J. Kaiser. Dealing with missing values in data. *Journal of Systems Integration*, 5(1):42–51, 2014.

[14] L. Kaufman and P. Rousseeuw. *Clustering by Means of Medoids*. Reports of the Faculty of Mathematics and Informatics. Faculty of Mathematics and Informatics, 1987.

[15] H.-P. Kriegel, P. Kröger, and A. Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Trans. Knowl. Discov. Data*, 3(1):1:1–1:58, Mar. 2009.

[16] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 591–600, New York, NY, USA, 2010. ACM.

[17] G. Moise, J. Sander, and M. Ester. Robust projected clustering. *Knowledge and Information Systems*, 14(3):273–298, 2008.

[18] J. Nichols, J. Mahmud, and C. Drews. Summarizing sporting events using twitter. In *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces*, IUI '12, pages 189–198, New York, NY, USA, 2012. ACM.

[19] B. Perozzi, L. Akoglu, P. Iglesias Sánchez, and E. Müller. Focused clustering and outlier detection in large attributed graphs. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 1346–1355, New York, NY, USA, 2014. ACM.

[20] S. Petrović, M. Osborne, and V. Lavrenko. Streaming first story detection with application to twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 181–189, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[21] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: Real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 851–860, New York, NY, USA, 2010. ACM.

[22] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman, and J. Sperling. Twitterstand: News in tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '09, pages 42–51, New York, NY, USA, 2009. ACM.

[23] H. Schwartz, J. Eichstaedt, M. Kern, L. Dziurzynsk, and S. Ramones. Personality, gender, and age in the language of social media: The open-vocabulary approach. In *PLoS ONE 8(9)*, 2013.

[24] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas. Sentiment in short strength detection informal text. *J. Am. Soc. Inf. Sci. Technol.*, 61(12):2544–2558, Dec. 2010.

[25] H. Toivonen. Sampling large databases for association rules. In *Proceedings of the 22th International Conference on Very Large Data Bases*, VLDB '96, pages 134–145, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers.

[26] Twitter api tweet object. `https://dev.twitter.com/overview/api/tweets`.

[27] S. Vieweg, A. L. Hughes, K. Starbird, and L. Palen. Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1079–1088. ACM, 2010.

[28] W. Webber, A. Moffat, and J. Zobel. A similarity measure for indefinite rankings. *ACM Trans. Inf. Syst.*, 28(4):20:1–20:38, Nov. 2010.