What are they up to? Distilling the Twitter Stream of Subpopulations

Ido Dangur
University of Haifa
Haifa, Israel
idodangur@gmail.com

Ron Bekkerman
University of Haifa
Haifa, Israel
ron.bekkerman@gmail.com

Einat Minkov
University of Haifa
Haifa, Israel
einatm@gmail.com

ABSTRACT

Social network researchers have been tackling community detection / community search for over a decade. Detecting communities – small groups of people who know each other and interact with each other – have numerous applications, starting from marketing and computational advertisement, all the way to the homeland security domain. By now, the problem can be considered mostly solved, in either its unsupervised form (community detection) or semi-supervised form (community search). In our quest to answer general – and very exciting – questions *What are people up to? What do they care about? What are they discussing?*, we move beyond detecting communities to circumscribing subpopulations – large groups of people who share some common characteristics, for example activists, students, engineers, New Yorkers, football fans etc. We want to know what are $<\cdots>$ talking about on Twitter, where $<\cdots>$ is any subpopulation. Initially, the subpopulation is characterized by a few representative members, who are treated as seeds in the iterative Personalized PageRank (PPR) framework that enlarges the subpopulation at each iteration. We immediately hit the scalability limitation, which we overcome by proposing the Splash PPR algorithm, inspired by Splash Belief Propagation. We implement Splash PPR on Apache Spark and show its efficiency and effectiveness on extracting the Twitter stream of a subpopulation of machine learning practitioners, by which we pave the road to distilling valuable signal out of the sea of Twitter noise.

KEYWORDS

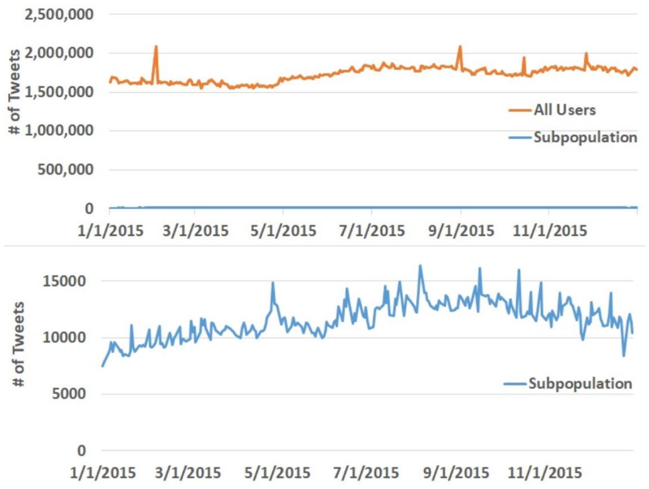
Twitter, Subpopulations, Personalized PageRank, Big Data

ACM Reference format:
Ido Dangur, Ron Bekkerman, and Einat Minkov. 2018. What are they up to? Distilling the Twitter Stream of Subpopulations. In *Proceedings of KDD 2018 14th International Workshop on Mining and Learning with Graphs (MLG'18), London, UK, August 20th, 2018 (MLG'18),* 8 pages.
https://doi.org/

1 INTRODUCTION

Social media platforms like Twitter let users enjoy a great visibility of their content, targeting millions of other users. This is a strong

Figure 1: An illustration of a strong noise / weak signal problem on Twitter. The top panel shows the stream volume of the entire Twitter population, which is mostly flat besides a few peaks (the left one corresponds to the Superbowl). A subpopulation stream is so weak that is looks like a straight line on the bottom. However, if we zoom in that stream (the bottom panel), we could see that it is full of bursts, corresponding to events of the subpopulation's interest.



value proposition for those who seek attention and ability to influence, which is why Twitter has become a mandatory tool for politicians, celebrities, and other outspoken personalities. All this made Twitter a honeypot for computational social scientists and hence it has been studied extensively [2, 5, 28, 30].

Despite the close attention Twitter has been paid to over the past decade, we still cannot view the Twitter stream holistically, to answer questions such as *What is currently happening? What are people up to? What are they talking about?*. Those questions are notoriously difficult because there are just too many people talking at the same time. If a major event, such as the Superbowl, is taking place, the entire Twitter is reacting. However, there are many sources of information on the Superbowl, out of which Twitter is arguably not among the strongest. We would love to analyze the Twitter stream for smaller, under-the-radar events, as well as trends and public opinions, but those signals drown in the ocean of Twitter noise (see Figure 1). Therefore, instead of trying to tackle the toughest questions as they are, we chose to adjust them towards specific groups, which we call *subpopulations*. Our goal is to find out what a specific subpopulation is talking about, by which we avoid the noise produced by the general Twitter population.

A subpopulation is a large subset of the Twitter population who share some common characteristics. A subpopulation can be characterized by a common profession (e.g. *physicians*), interest (*U2 fans*), nationality (*Philippians*), activity (*Louvre visitors*), and many others, or a combination of them (*Maryland residents of African origin*). While we wish to identify the largest such subpopulation, we admit that it is not always possible from the practical perspective. For example, if we look for engineers on Twitter, would engineering students count? How about retired engineers? How about highly skilled technicians? Fortunately, to find out what engineers are talking about, we do not need to identify all the engineers on Twitter. Since the information distribution on Twitter often takes the form of a cascade [20] that starts from an influencer [2], we are interested in identifying prominent members of the subpopulation, who would most probably define the subpopulation's update stream.

This intuition is prescriptive for choosing the methodological paradigm behind an adequate solution: PageRank would be our best choice. Another important design choice to make is about how the system should find out *which* subpopulation to identify. Here we have two options: (a) a textual query, such as *"engineers"*, which has the drawbacks discussed above, and (b) subpopulation member examples, which would be used as seeds to bootstrap the identification process. The latter choice is more plausible as it allows easy customization of the subpopulation boundaries, without getting into word semantics of textual queries. Also, it would be safe to say that the larger the initial set of seeds is, the better the subpopulation would be identified. Following Kloumann & Kleinberg [18] and others, we chose the seed option, which led us to *personalized PageRank (PPR)* as the most adequate underlying technology [10, 22].

Given a set of k seed nodes in the Twitter graph, we apply k PPR processes (one process per seed) that rank Twitter users while prioritizing users in the seed vicinity. Here we face our main scalability challenge: the Twitter graph consists of hundreds of millions of nodes. Since we seek subpopulations that are large groups of users, and therefore our seed set can be large as well (e.g. dozens of thousands of users), we will need to run an order of 10^4 PPR processes, each over an order of 10^8 nodes. Even for the most advanced cloud computing infrastructure, such a computation would be too heavy and too expensive.

We note, however, that an application of PPR to a large graph would only affect the close vicinity of the seed node. Therefore, inspired by *Splash Belief Propagation* [13], we propose an approach to parallel PPR execution that we call *Splash Personalized PageRank*, implemented in a MapReduce. The Map phase applies PPR to small-radius subgraphs around each seed node. The Reduce phase then takes care of the subgraph overlaps. For extremely large graphs, such as the Twitter graph, that are stored in a distributed file system, the operation of bringing all the neighbors of a node's neighbors is expensive. To address this problem, we propose a two-MapReduce algorithm for constructing subgraphs around all the seed nodes simultaneously. Once subpopulation members are identified in a process described above, we add them to the pool of the seeds, and start the identification process over again, in an incremental manner. At each such iteration, the subpopulation grows into areas of the Twitter graph which are gradually more distant from the original seeds.

We implement our methodology in Spark Python over AWS, and test it on a dataset of more than 600 million tweets that came from the US during 2015. In this work, our case study is a subpopulation of machine learning practitioners: after identifying the ML subpopulation on Twitter, we detect bursts in the subpopulation activity, associated with major events in the ML world.

2 RELATED WORK

There exists ample research on the detecting and analyzing online communities (see [15, 21] and many others). Communities are most often defined topologically, as groups of nodes in the social graph that densely interconnected among themselves and loosely connected to other nodes. In contrast, we define subpopulations *topically*, as groups of people who share a semantic property, and we base on the *topical locality* property [9] to identify subpopulations.

Among the community detection works, the most relevant to ours is the work of Whang et al. [29] who expand sets of seed community members using personalized PageRank and a clustering procedure, while taking into account community overlaps. Whang et al. detect small, heavily connected communities, while we identify large, loosely connected subpopulations. This crucial difference prescribes the scalability treatment: Whang et al. implement their method on a single machine in a multi-threaded environment which is limited in scalability while fitting their needs – we, however, have to go for a fully distributed environment in which the data is stored in a distributed file system.

Community search aims at retrieving the smallest and densest community to which query nodes belong (see, e.g. [7, 26], and [3] for the survey). Community search is related to our task of subpopulation identification in the respect that we also start with a set of query nodes (i.e. seeds). Our objective, however, differs from the objective of community search, as we identify a large group of nodes that has something in common, while the emphasis is on identifying prominent members of the subpopulation who are primarily responsible for the subpopulation's update stream.

The most relevant community search work is by Kloumann and Kleinberg [18] who proposed seed expansion methods and investigated their effect on finding densely connected communities. In particular, they showed that PageRank and its variants outperformed other methods, with the highest recall over different datasets. Our focus is on identifying as many subpopulation members as possible, which leads us to resolving scalability problems by parallelization. We achieve a broader coverage of a subpopulation by applying the iterative process of expanding the set of top-ranked users discovered in the previous iteration.

Applying PPR to a graph of all Twitter users is a challenging task for the exisiting methods. Computing PPR scores using power iteration [17, 19] requires massive computational resources for large networks. Jeh and Widom [16] suggested a scalable PPR framework for vertices linked to heavily connected hubs. Their idea was based on the notion that relevance scores for a given distribution can be approximated as a linear combination of those originated from a single vertex in network hubs. Hence, they compute approximate relevance scores for a given distribution using precomputed relevance scores. This idea was later improved by Fogaras et al. [11] who introduced a method for calculating PPR scores using a randomized approximation based on random walk simulations, which can be scaled to the entire web. We apply (non-personalized) PageRank to the entire graph only once, while keeping small-radius subgraphs around seed nodes, on which PPR is executed in parallel.

Parallelization of PageRank methods have been covered in the literature as well. Bahmani et al. [1] enhanced MapReduce paradigm to develop a scalable fully personalized PageRank. Their method performs a Monte Carlo based PPR approximation and enables to compute single random walks starting from all nodes in a graph. Fujiwara et al. [12] introduced Castanet, a PPR based method aimed to identify the top-k nodes with the exact node ranking, as well as to support interactive similarity search. They estimated the similarity bounds in iterative style performed by pruning irrelevant nodes to effectively detect the top-k nodes in each iteration. Their method was fast due to the construction subgraphs for similarity computations – which is exact as it returned the top-k nodes with the exact ranking, and robust since no pre-computation or inner-parameters configurations were required.

Zhang et al. [31] introduced two parallel algorithms, Forward Push and Reverse Push for updating and maintaining approximate Personalized PageRank vectors, enabling to restore accuracy on large dynamic graphs. Guo et al. [14] suggested an efficiently distributed framework for calculating exact Personalized PageRank Vectors (PPV) for all nodes in a given graph. They proposed a graph partition algorithm, *GPA*, that separates graphs into disjoint subgraphs of similar sizes to distribute the PPV computation, and even improved it by introducing *HGAP*, which is based on a hierarchy of subgraphs. The latter method was shown to be more efficient and outperformed the power iteration method in terms of space communication and scalability. Our method takes a similar approach to Guo et al. [14], but instead of performing PPR calculation on subgraphs with similar sizes, our distributed mechanism allocates each subgraph to a different mapper, enabling an efficient parallel computation. In addition, our method distributes PPR by running it locally on subgraphs around a pre-selected set of seeds, in a semi-supervised approach.

3 SPLASH PPR FOR SUBPOPULATION IDENTIFICATION

Our goal is to reveal subpopulations from Twitter traffic. We assume that an initial set of example members of the subpopulations of interest S_q can be specified by a user with minimal effort, and refer to this initial set of seed examples as a *query*. The proposed framework aims to *expand* the initial set S_q into a larger sample of the underlying subpopulation. To identify a subpopulation given a set of seed users, we apply Personalized PageRank on small-radius subgraphs around the seed nodes in the Twitter graph, aggregate scores of nodes that belong to more than one subgraph, and subtract the global (non-personalized) PageRank scores to get rid of very popular users who are not a part of the subpopulation. We then add detected subpopulation members to the pool of seeds and repeat the process iteratively, to include more subpopulation members.

We consider a large-scale graph, which summarizes a large 'bulk' of Twitter traffic over a period of a year (or more). There are two main configurations of the social graph that we consider in this work.

Algorithm 1 Subgraph Generation step 1: MapReduce

```
1:  $flatmapper_1$ : $Find\ Seed\ Neighbors$ (graph record);
2:  process $graphrecord$;
3:  find center $c$;
4:  find all neighbors $NE$;
5:  output center $c$, neighbors $NE$
6:  for neighbor $ne$ in neighbors $NE$ do
7:      if neighbor $ne$ in seeds $S$ then
8:          add neighbor $n$ to Seed Found List $SFL$;
9:      if Seed Found List $SFL$ is not empty then
10:         for neighbor $ne$ in neighbors $NE$ do
11:             for seed $s$ in $SFL$ do
12:                 if neighbor $ne$ != seed $s$ then
13:                     output neighbor $ne$, (center $c$,seed $s$);
14: $reducer_1$ : $Aggregate\ Output\ Per\ Neighbor$ (neighbor, value);
15:     return neighbor + value;
```

- *Social structure.* This variant of the graph represent Twitter users as vertices, and observed interactions between users as directed edges. Following previous works [23, 27], we consider retweets and user mentions as a signal of inter-user affinity, having a mention of user y by user x represented as a directed edge in the graph from x to y.
- *Content-augmented graph.* Assuming that a subpopulation is characterized by common properties or interests, it may be beneficial to model the language, content or topics used by the subpopulation members. In this work, we consider hashtags as topical cues, modeling distinct hashtags as graph nodes, and linking users to the hashtags mentioned by them. This design choice is inspired by Cunha et al. [8] who mentioned that hashtags are used to categorize messages in Twitter, as well as to help other users to find tweets that have a common topic, and also by Ramage et al. [24] who used hashtags to label topics in the Twitter stream.

Applying PPR on a large scale graph starting with a single query node is known to affect only a small-radius neighborhood of the query node [6]. That is why we propose the *Splash* PPR approach. We use the term "splash" to describe a process that starts from a certain position in a large graph and affects its closer environment. This process can be compared to as a raindrop falling in the middle of a lake, generating a very small splash that affects the surrounding of the position it met the water. Several drops falling close to each other create splashes that interfere. This way, multiple local actions can cause a wide effect. We employ the Splash approach to perform multiple local PPR executions in parallel, each originating from a subpopulation seed and affecting only a small-radius subgraph around it. The splash interferences are then taken care of.

Given the enormous scale of our task, the need to run parallel processes on the distributed environment led us to seeking solutions for constructing subgraphs around each seed user – and hopefully constructing those subgraphs in parallel as well. After investigating various off-the-shelf solutions such as subgraph construction using the built-in "subgraph" method in Apache Spark "GraphFrames" package, or using the Neo4j graph database, we concluded that all of them had issues that prevented us from scaling up the process to the required level. The GraphFrames package, for example, repeatedly

Algorithm 2 Subgraph Generation step 2: MapReduce

```
1: $flatmapper_2$ : $Determine\ Node\ Depth\ Per\ Seed$ (tuple $t$);
2: split to node $n$ and value $v$;
3: for value $v$ in values $V$ do
4:     if value $v$ is tuple then
5:         add $v$ to tuples $T$;
6:     else
           add $v$ to neighbors $NE$;
7: if neighbor $ne$ in seeds $S$ then
8:     for neighbor $ne$ in neighbors $NE$ do
9:         output node $n$ , (node $n$ : '0', neighbor $ne$ : '1');
10: for neighbor $ne$, seed $s$ in tuples $T$ do
11:     output seed $s$ , (node $n$ : '2', neighbor $ne$ : '1');
12:     flag neighbor $ne$ and seed $s$ as seen;
13: if tuples $T$ then
14:     for neighbor $ne$ in neighbors $NE$ do
15:         if neighbor $ne$ not seen then
16:             output seed $s$ , (node $n$ : '2', neighbor $ne$ : '?');
17: $reducer_2$ : $Union\ Results\ By\ Seed$ (seed, value);
18:     return seed.union(value);
```

calls a routine that brings neighbors of a node – the process that might be efficient enough for constructing one subgraph, but we needed to accommodate the construction of up to 10^4 subgraphs. We decided on proposing our own subgraph construction algorithm, based on the MapReduce paradigm.

A subgraph is a set of nodes and edges in a close vicinity around a seed node, which is the center of the subgraph. We call its immediate neighbors the *first hop* from the seed, and their neighbors the *second hop* from the seed. We propose a MapReduce algorithm that given a set of seed nodes and the graph represented by a list of nodes and their first-hop neighbors, outputs sets of edges connecting nodes at the distance 1 and 2 from the seed. The choice to deal with only two hops from the seed is motivated by the strong interconnectedness of the Twitter graph, and the iterative nature of our subpopulation identification method: on Twitter, chances are low that an influential member of a subpopulation would be on the third hop from any seed, and even if this happens, that member will be eventually discovered over the process iterations which direct the identification towards influential members.

We eliminated users with too many first-degree neighbors as these users are most probably celebrities or media accounts. Assuming having accounts that operate as hubs, like @twitter or @cnn, based on the graph structure, the "retweet" functionality might link it to a huge number of users. The second hop can reach millions of users, and the idea of building a focused subgraph around the seed would not hold then. Due to that, our decision to only deal with nodes of the degree less than 10,000 is crucial in the process. Overall, each constructed subgraph was between few kilobytes in case of a fairly isolated seed, up to 200 megabytes for the largest subgraphs, composed of millions of edges.

The proposed iterative pipeline includes several steps. First, an initial iteration of PageRank will be executed to rank all "user" nodes in the full graph. This rank will be used as the background model. In addition, some basic graph statistic will be measured, such as degree of each node, and connectivity. The first steps will

Algorithm 3 Subgraph Generation step 3: Postproccesing

```
1: $postprocessing$ : $Generate\ Final\ Subgraphs$ (tuple $t$);
2: for edge $e$ in edges $E$ do
3:     if edge $e[1]$ is not '?' then
4:         add edge $e[0]$ to nodes $N$;
5:     if edge $e[3]$ is not '?' then
6:         add edge $e[2]$ to nodes $N$;
7: for edge $e$ in edges $E$ do
8:     if edge $e[0]$ in nodes $n$ and edge $e[2]$ in nodes $n$ then
9:         add ($e[0]$,$e[2]$) to edges set $ES$
10: return seed $s$ + edges set $ES$
```

be done once, as a part of the network preprocessing. Then, we build subgraphs around each seed (see Algorithms 1, 2 and 3), and run a PPR in each subgraph. This approach will generate a ranked list of nodes in the subgraph, and a summation of their PPR scores will give us the final rank of all affected nodes. Next, we will use the PR background model to eliminate nodes that are very popular, hence received high rank in the global PR. The PR background model will eventually filter out less "interesting" nodes, which were ranked high mainly because of a large number of followers. In other words, we will compare the PPR ranks to the PR ranks and will look for nodes that popped up in the PPR but were ranked low in the overall PR. We will then add the top nodes as new seeds to the iterative algorithm, and repeat this process till there will be no significant change in ranks. The idea is to look for new users related to the seed subpopulation, as their local PPR rank will be higher compared to their PR rank which might be significantly lower. The entire process is summarized in Algorithm 4.

The main challenge of this work is simultaneous construction of a large number of small-radius subgraphs in a graph of an enormous scale, stored in a distributed file system – for which the procedure of accessing neighbors of a node's neighbors is computationally heavy. For subgraphs of radius 2, the intuition behind the three-MapReduce process is that the first hop nodes are neighbors of the seed and of second hop nodes, so that only the access to a node's neighbors is enough (Algorithm 1). Algorithm 2 deals primarily with taking care of edges that go out of second hop nodes: if an edge like that ends at a second hop node, it will be accepted, otherwise rejected. Algorithm 3 is the result aggregation.

4 EXPERIMENTAL SETUP

4.1 Twitter data and graph

We use a large corpus of Tweets, including more than over 600M tweets posted by U.S. accounts in English during 2015. Overall, this data measured 2TB as raw text, and was stored in an Amazon S3 account. Table 1 shows details about this experimental data.

User representation. We represent user interactions as a graph, having distinct users represented as nodes, and directed edges denote direct user interactions, considering observed retweets or user mentions. Having represented this data as a graph, we process it using the GraphFrames package[1], which implements a variety of graph analytics algorithms in the Spark distributed system.

[1] https://graphframes.github.io/; GraphFrames is based on Spark DataFrames, which is an advanced version of RDDs.

Algorithm 4 Subpopulation Identification: Detailed

```
1:  Perform only once:
2:  Generate the Graph $G_u$ composed of Users $RTs$ and $UMs$;
3:  Generate the Graph $G_h$ composed of Hashtags mentioned by
    users;
4:  Generate A joint Graph $G_{uh}$ Composed of both User and Hash-
    tag Graphs;
5:  Generate the list of User Edges $E_u$ and User Nodes $V_u$ as an
    input to the $PR$ Algorithm;
6:  Execute $PageRank$ over the entire graph to generate a Back-
    ground Model ($BM$);
7:  for each Subpopulation $SP$ do
8:      Collect key users to perform as seeds ($S$);
9:      while repeat till convergence do
10:         Generate subgraphs for $S$ while using each graph of $G_u$,
            $G_h$, $G_{uh}$;
11:         Run $PPR$ originated from $S$ in $SP$;
12:         Subtract $BM$ from ranks obtained by $PPR$;
13:         Aggregate results of $S$ based on Aggregated Scores ($SA$)
            and Mutual Appearances ($CA$);
14:         Take the Top $N$ Users subject to maximize $SA$ or $CA$;
15:         Add $N$ users to the $S$;
```

Using GraphFrames, we computed *graph degree* of nodes in the user graph, defined as the total number of incoming and outgoing edges per node. As expected, the distribution of node degrees is long-tailed, where a small number of users (several thousands) are densely connected to more than 1,000 other nodes, while the majority of users are weakly connected. Some of the highest degree accounts belong to popular websites ('YouTube'), celebrities (Taylor Swift), and possibly bots. In order to avoid bias in our analysis towards such high-degree nodes [4], we remove them from the graph as a preprocessing step. Overall, we eliminated roughly 12.7K users with in-degree or out-degree greater than 1,000.

Hashtag extraction. In addition to users and their interactions, we represent topical evidence in the graph in the form of *hashtags*. Again, we wish to avoid the representation of widespread hashtags (e.g., 'tbt') and focus on hashtags that are adopted selectively by subpopulations. To identify meaningful hashtags, we evaluate them using a measure similar in spirit to inverse document frequency (IDF) in information retrieval. Concretely, we first extract all of the hashtags mentioned more than k times in our experimental data. For each hashtag we then compute the number of different users who mentioned it at least once, reflecting the uniqueness of that hashtag across users. Only those hashtags that are unique enough (10<IDF<10K) are incorporated into the graph. A dedicated node is used to represent each of the hashtags included, where users who mentioned that hashtag at least once are linked to it over an outgoing edge.

4.2 Graphs variants

We generated several graph variants, described in this section, in order to evaluate the efficiency and accuracy of the task of identifying new users related to a certain subpopulation.

User Graph. For our proposes, we wanted the graph to be undirected, to make the process smoother. The idea is to generate 2 edges instead of one direct edge. For instance, when user A mentions user B, the User Graph will generate an edge between User A to B and another edge in the opposite direction. This decision makes sense for mentions, as user mentioning each other might be familiar, but also for retweets since we like to capture the opposite direction as well. Another attribute of the user graph is the number of mentions. This attribute indicated how many times each user was mentioned by the other. For the final graph, we excluded all users having out-degree or in-degree of more than 1000, as we wanted to capture the associations between individuals who are not famous or very popular. This approach reduced PageRank's tendency toward popular accounts [4].

Table 1: Basic Statistics on the dataset

Data stats:	Tweets	616,486,595
	Users who tweeted at least once	10,679,145
	Users who retweeted at least once	6,796,592
Graph stats:	User Nodes	24,145,648
	User-to-User Edges	143,415,392

Hashtag Graph. In addition to the User Graph, which represents the linkages between user to user, we wanted to enrich our features and add linguistic attributes. Our main challenge was to keep the unsupervised approach of generating both user and hashtag graph only once. This objective will improve the performance of the system since it will prevent the change of the global large graph after each iteration of the algorithm. After considering the option of taking words as well, we concluded with hashtags only. There are several reasons for using only hashtags: (a) hashtags are used to mark topics; (b) hashtags are generally less frequent than words; (c) hashtags's purpose is to make the tweet more searchable. For each hashtag tweeted by a certain User, we generated an edge from this hashtag to the same user. This enables us to reach from one user to another user, in case those are sharing the same hashtag. To filter out too common and too rare hashtags, we decided to take hashtags that were retweeted by between 10 and 10,000 users. We ended up with a large graph, composed of more than 540,000 unique hashtags, and having over 52M edges.

Joint Graph. The joint graph is a combination of the User Graph and the Hashtag Graph. It contains user to user and user to hashtag edges, as well as user and hashtag nodes. This graph was generated by a direct concatenation of both graphs into one while keeping each graph properties the same as before. The joint graph enables us to experiment whether a combination of features will yield better results that each graph independently. This hypothesis derives from an intuition that different subpopulations have different means of communication. For example, more socially involved subpopulation like the population of U.S. activists will tend to express their ideas by using hashtags, while a subpopulation of academic researchers might use mentions and retweets instead of hashtags, in order to communicate with each other.

Background Model. We apply non-personalized PageRank to identify the user nodes which eventually have the greatest chance for a random walker to end up in. Generating such a *background model* for all users in the system is a crucial as it allows avoiding "celebrity" users who are unlikely to belong to any specific subpopulation. The implementation of PageRank on the entire user graph was coded using GraphFrames package, implemented in Spark.

Table 2: Example of Machine Learning Researchers Seeds

User	Description
andrewyng	Andrew Ng is one of the founders of the Google Brain Team, chairman and founder of Coursera
ylecun	Yann Le Cunn is the head of Facebook's Artificial Intelligence laboratory
awmcmu	Andrew Moore is the Dean of Carnegie Mellon University's School of Computer Science
drfeifei	Fei-Fei Li is an associate professor of computer science at Stanford university
demishassabis	Demis Hassabis is the founder of DeepMind
hmason	Hilary Mason, founder of Fast Forward Labs
karpathy	Andrej Karpathy is a Director of AI at Tesla

Evaluation Metrics. We use *precision-at-rank-k* and *mean average precision* to evaluate the quality of the results. We split our training set into 5 folds each, composed of randomly selected subpopulation users, in order to prevent bias. Training set users were labeled in advance by manual judgments, to increase the accuracy and minimizing the noise caused by mislabeled data. After applying the proposed algorithmic process, we generated the final ranks for all users, and measured the accuracy based on the MAP measure, while focusing on the ranks received from the test group users. Eventually, the ranking quality obtained for the test group will determine the overall method accuracy.

List consolidation. Having run PPR for each of the k seed users using PPR-Splash, we obtain k ranked lists, which then have to be consolidated into a single list. There are two approaches for ranked list consolidation which we experiment with in our work:

(1) *Score-based aggregation (SA).* For each user i, sum the PPR scores assigned to this user in all lists: $SA(i) = \sum_k PPR_k(i)$. The motivation for aggregating PPR scores is supported by the Linearity Theorem.
(2) *Count-based aggregation (CA).* Instead of summing scores, we summed the number of appearances of each user in all lists [25]: $CA(i) = \sum_k G_k(i)$, where G_k is an indicator function.

5 CASE STUDY: MACHINE LEARNING PRACTITIONERS

We experimented with a few subpopulations but due to the space limitations we can only discuss one – a relatively small one but related to a hot topic nowadays: Machine Learning (ML). The machine learning subpopulation attracts a lot of attention these days, and it might be interesting to observe whether our method is able to find out what machine learning practitioners are up to.

Seed Data and Labeling. In order to find seeds, which are ML researchers and practitioners in this usecase, we simply crawled several websites recommending top ML researchers and influencers to follow on social networks[2]. Eventually, we labeled 100 users as ML subpopulation seeds – we want to find out whether a relatively small number of seeds will lead to confident results, while taking into account that the manual effort of obtaining an initial seed list for different subpopulations might be expensive. We carefully assume that 100 seeds can be manually labeled for a certain subpopulation one will be interested in, and this kind of benchmark number is likely to be easily achieved.

[2] https://goo.gl/CjhLmq; https://goo.gl/Njy6X7; https://goo.gl/UAux6R; https://goo.gl/ofVgtq; https://goo.gl/Q1XdTu; https://goo.gl/6mjtEq

Table 3: Mean Average Precision for 5-fold CV for the 3 Graph Settings. Average Precision is measured by Aggregated Scores (SA) and Mutual Appearances (CA). Std Error and Avg. Users (U) retrieved are also reported.

Setting	User Graph			Hashtag Graph			Joint Graph		
Measure	SA	CA	U	SA	CA	U	SA	CA	U
MAP	0.03	0.05	61	0.01	0.01	21	0.03	0.04	63
Std. Err.	0.004	0.012	-	0.003	0.005	-	0.003	0.013	-

5.1 Results

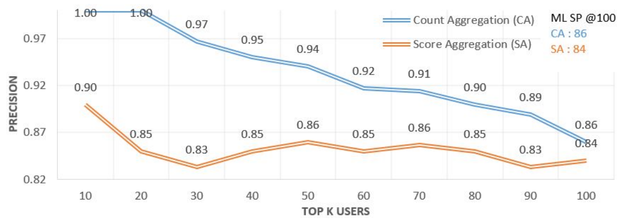
In order to evaluate our method, we generated subgraphs for each one of the 100 initial seed users retrieved by the manual labeling process. We randomly divided our 100 labeled ML subpopulation members into 5 groups of 20 users each. The idea in this experiment was to evaluate each graph setting based on its Average Precision, on the task of retrieving the other 80 users in a cross-validation setting. We aimed to experiment with the three graph configurations on the same seed group, in order to eventually select the best configuration that represents the ML subpopulation and manages to retrieve a large amount of subpopulation members with the highest precision possible.

Background Model. Assuming that the ML research domain has low chances to be associated with celebrity-related content, we set a low threshold on the number of users to be excluded based on the background model: we excluded the top 10,000 users only. This choice eliminated users like @nytimes (New York Times) and other popular accounts that we are not interested in, but still preserved users such as @hmason (Hilary Mason) and @andrewyng (Andrew Ng) who have relatively high global PR scores.

Different Subgraphs. For each fold in 5-fold cross validation, we generated different subgraphs based on the three configurations; User Graph, Hashtag Graph, and Joint Graph. The purpose of this experiment was to examine which graph setting will present the highest precision, hence will be selected as the preferred setting for this subpopulation. Precision was measured after performing the two manipulations discussed earlier, the first is aggregating PPR scores (SA) for each user retrieved from each seed within a group, and the second is summing up appearances (CA) of each user retrieved within a group.

Table 3 shows that the User Graph setting receives the highest Average precision, for both aggregation methods. Both the User Graph and the Joint Graph show similar results, with a slight advantage towards the User Graph setting. The Hashtag Graph, however, shows poor results for both aggregation methods. This behavior can be explained by the strong assumption that ML Researchers, as opposed to social subpopulations like Activists or celebrities, do not tend to use hashtags quite often. Unlike social subpopulations, which aim to be searched by other Twitter users, Researchers keep a more general and professional language. Another important finding was that method of Aggregated Counts (CA) outperformed the Aggregated Scores (SA) one. An explanation for that might be that each fold aggregated 20 subgraphs only, causing the Aggregated

Figure 2: Precision @100 for the ML Researchers Subpopulation while applying CA and SA Aggregation methods



Scores (SA) method not to converge, while counting the Mutual Appearances (CA) seems to be immune to small data inputs. It is worth mentioning that SA outperformed CA in other experiments, which did not fit the scope of this paper.

A combination of small folds, together with relatively low number of tweets per seed, cause the low coverage between folds, expressed by relatively low values of Precision on the CV task of retrieving 80 seeds from 20 input seeds (still, they were chosen out of 24 million users). Moreover, the step of choosing the best graph configuration guides as to the next experiments, as we are able to select both graph setting and aggregation method that maximize precision for seeds we are confident with, so we do not need to manually explore all other different configurations.

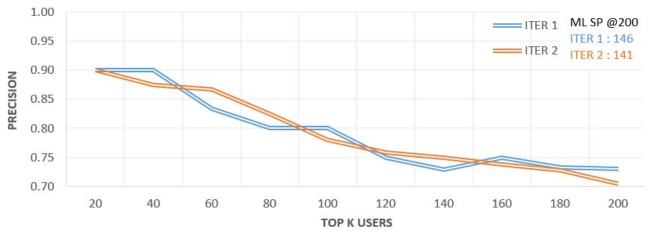
The results received after running the experiment on all 100 seeds in one bulk support the previous five-fold CV experiment, having 6.7% for CA and 3.4% for SA methods. In addition, 89 out of the 100 seeds were ranked in the top 100, while the other seeds were not mentioned or have very few amounts of data. A precision of 6.7% is not high, which corresponds to the fact that the ML subpopulation is not well connected, and ML users rarely mention or retweet each other.

However, the fact that this subpopulation is weakly connected, does not impact the identification of new subpopulation users. Hence, we manually evaluated the second 100 results after one iteration of our algorithm, in order to find out whether it managed to identify new Twitter users associated with the ML research area. After having sorted lists of users having the highest scores for User Graph setting, which achieved the highest precision score, we explored its content. The manual evaluation effort included an investigation of the first 100 non-seed users; scamming Twitter current profiles, reviewing LinkedIn page, blogs, academic and company websites. As expected, some accounts belong to research, analytics and big data companies, which a user might find worth to follow, in order to be up to date with research, new products, and events happening in the industry. In general, the majority of the top accounts is relevant to ML, AI or related topics, and managed by individuals.

Figure 2 shows that the performance of both CA and SA is similar, and the absolute results are high. This is very encouraging, as it increases the confidence of the proposed method to reveal new users that share the same features and interests within the initial seeds. Some results from the second 100 include KDnuggets[3] (@kdnuggets) and Kaggle[4] (@kaggle) which are very known websites for

[3]https://www.kdnuggets.com/
[4]https://www.kaggle.com/

Figure 3: Precision values Percentile of the first 200 users (excluding seeds). Comparison was made between results after the First and Second iterations based on CA



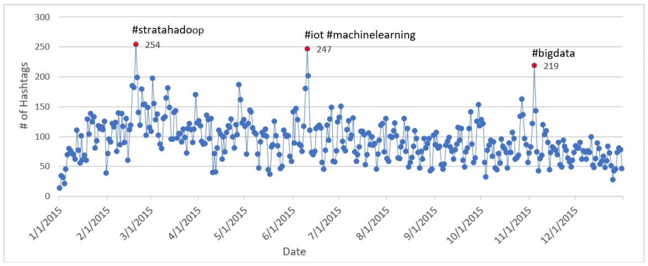
Data Science and Machine Learning topics, as well as researchers such as Jeremy Howard (@jeremyphoward) who is the founder of Fast.ai company, and DJ Patil (@dpatil),the former U.S. Chief Data Scientist. In addition, among the second 100 are ML-related accounts owned by industry consultants and technology writers, who publish and tweet the latest news in the ML area. To conclude, the suggested method proved to be effective for revealing additional accounts relevant to the ML domain. These users show a direct association to Machine Learning and related topics, hence the above results strengthen the validity of our method.

Second Iteration. In order to enrich our analysis, and to examine whether the second iteration will contribute to the identification task, we performed another step of the PPR process. We decided to focus on the User Graph using the CA aggregation method, as this combination had given the highest precision in the automated evaluation task. We have expanded the initial 100 seeds list with additional 100 users – the second 100 of the first iteration ranking. We generated subgraphs for the new users and aggregated the CA scores for the combined list of users, which is now consists of 200 seeds. The resulting ranked lists of Iteration 1 and Iteration 2 were compared to each other. For evaluation, we performed the same task as we did for the first iteration: we manually evaluated the second and third 100 of users in the two ranked lists.

Figure 3 displays precision values of the first 200 users after the exclusion of the original 100 seeds, in terms of their relevance to the ML domain, which was concluded after the manual evaluation process. It can be seen that the two curves are quite similar, having a similar trend, which means that the second iteration does not show a direct improvement in precision. Based on the manual evaluation, we can conclude that the second iteration did not improve ranks for users at the top, but this might not be the case for users which are still relevant at lower ranks, say between 300-500 or 500-1000.

Due to the lack of labeling resources, we could not compare results for top 2,000 users ranked after both iterations. We performed a manual analysis to understand whether there were accounts related to Machine Learning and Data Science that improved their position between the two iterations, and non-related accounts got downgraded. This effort showed a similarity of 81.25% between the ranked lists after both iterations, meaning that 375 users gained or lost their positions among the top 2,000, after performing the second iteration. Using the same manual evaluation technique, we randomly sampled 40 users who improved their rank compares to the first iteration, resulting in 22 of them labeled as related accounts. We performed the same on randomly chosen 40 users whose rank

Figure 4: Count of hashtags tweeted per day for the top 2000 ML Researchers subpopulation, after the first iteration



got decreased, having 12 labeled as ML related accounts. The above analysis shows that the second iteration helps to fine-tune the final output.

In order to find out what ML users are up to, we constructed Figure 4 that shows the daily distribution of the most popular hashtags among our seeds. It can be seen that there are several peaks, for instance on February 19, when the "Strata Hadoop conference"[5] presented by O'Reilly and Cloudera took place, and on June 10, when the CVPR conference[6] took place in Boston. These events are correlated with a high frequency of common ML related hashtags, which suggests that peaks of ML people activity on Twitter are related to major conferences in the field.

6 CONCLUSION

The proposed research aims to enhance the task of identifying subpopulations by generating a mechanism that may operate as a basis for a novel methodology, which competes with today's state-of-the-art techniques. Subpopulation identification may contribute to analyzing relationships between many seemingly unrelated users and capturing the update stream of these subpopulations in the digital space. This approach will be one step towards a better understanding of what people discuss, how they interact and behave in the era of social media.

REFERENCES

[1] Bahman Bahmani, Kaushik Chakrabarti, and Dong Xin. 2011. Fast personalized pagerank on mapreduce. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. ACM, 973–984.

[2] Eytan Bakshy, Jake M Hofman, Winter A Mason, and Duncan J Watts. 2011. Everyone's an influencer: quantifying influence on twitter. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 65–74.

[3] Nicola Barbieri, Francesco Bonchi, Edoardo Galimberti, and Francesco Gullo. 2015. Efficient and effective community search. *Data Mining and Knowledge Discovery* 29, 5 (2015), 1406–1433.

[4] Suratna Budalakoti and Ron Bekkerman. 2012. Bimodal invitation-navigation fair bets model for authority identification in a social network. In *Proceedings of the 21st international conference on World Wide Web*. ACM, 709–718.

[5] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, and P Krishna Gummadi. 2010. Measuring user influence in twitter: The million follower fallacy. *Icwsm* 10, 10-17 (2010), 30.

[6] Fabrizio Costa and Kurt De Grave. 2010. Fast neighborhood subgraph pairwise distance kernel. In *Proceedings of the 26th International Conference on Machine Learning*. Omnipress, 255–262.

[7] Wanyun Cui, Yanghua Xiao, Haixun Wang, and Wei Wang. 2014. Local search of communities in large graphs. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 991–1002.

[8] Evandro Cunha, Gabriel Magno, Giovanni Comarela, Virgilio Almeida, Marcos André Gonçalves, and Fabrício Benevenuto. 2011. Analyzing the dynamic evolution of hashtags on twitter: a language-based approach. In *Proceedings of the Workshop on Languages in Social Media*. Association for Computational Linguistics, 58–65.

[9] Brian D Davison. 2000. Topical locality in the Web. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*. 272–279.

[10] Sela Ferdman, Einat Minkov, Ron Bekkerman, and David Gefen. 2017. Quantifying the web browser ecosystem. *PloS one* 12, 6 (2017).

[11] Dániel Fogaras, Balázs Rácz, Károly Csalogány, and Tamás Sarlós. 2005. Towards scaling fully personalized pagerank: Algorithms, lower bounds, and experiments. *Internet Mathematics* 2, 3 (2005), 333–358.

[12] Yasuhiro Fujiwara, Makoto Nakatsuji, Hiroaki Shiokawa, Takeshi Mishima, and Makoto Onizuka. 2013. Efficient ad-hoc search for personalized pagerank. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. ACM, 445–456.

[13] Joseph Gonzalez, Yucheng Low, and Carlos Guestrin. 2011. Parallel belief propagation in factor graphs. In *Scaling Up Machine Learning: Parallel and Distributed Approaches*, Ron Bekkerman, Mikhail Bilenko, and John Langford (Eds.). Cambridge University Press, Chapter 10, 190–216.

[14] Tao Guo, Xin Cao, Gao Cong, Jiaheng Lu, and Xuemin Lin. 2017. Distributed algorithms on exact personalized pagerank. In *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 479–494.

[15] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. 2007. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*. ACM, 56–65.

[16] Glen Jeh and Jennifer Widom. 2003. Scaling personalized web search. In *Proceedings of the 12th international conference on World Wide Web*. ACM, 271–279.

[17] Sepandar D Kamvar, Taher H Haveliwala, Christopher D Manning, and Gene H Golub. 2003. Extrapolation methods for accelerating PageRank computations. In *Proceedings of the 12th international conference on World Wide Web*. ACM, 261–270.

[18] Isabel M Kloumann and Jon M Kleinberg. 2014. Community membership identification from small seed sets. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1366–1375.

[19] Amy N Langville and Carl D Meyer. 2004. Deeper inside pagerank. *Internet Mathematics* 1, 3 (2004), 335–380.

[20] Kristina Lerman and Rumi Ghosh. 2010. Information contagion: An empirical study of the spread of news on Digg and Twitter social networks. *Icwsm* 10 (2010), 90–97.

[21] Daifeng Li, Bing He, Ying Ding, Jie Tang, Cassidy Sugimoto, Zheng Qin, Erjia Yan, Juanzi Li, and Tianxi Dong. 2010. Community-based topic modeling for social tagging. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, 1565–1568.

[22] Frank Lin and William W Cohen. 2010. Semi-supervised classification of network data using very few labels. In *International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*.

[23] Zongyang Ma, Aixin Sun, and Gao Cong. 2013. On predicting the popularity of newly emerging hashtags in twitter. *Journal of the Association for Information Science and Technology* 64, 7 (2013), 1399–1410.

[24] Daniel Ramage, Susan T Dumais, and Daniel J Liebling. 2010. Characterizing microblogs with topic models. *ICWSM* 10, 1 (2010), 16.

[25] Lior Rokach. 2010. Ensemble-based classifiers. *Artificial Intelligence Review* 33, 1-2 (2010), 1–39.

[26] Mauro Sozio and Aristides Gionis. 2010. The community-search problem and how to plan a successful cocktail party. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 939–948.

[27] Bongwon Suh, Lichan Hong, Peter Pirolli, and Ed H Chi. 2010. Want to be retweeted? large scale analytics on factors impacting retweet in twitter network. In *Social computing (socialcom), 2010 ieee second international conference on*. IEEE, 177–184.

[28] Katrin Weller, Axel Bruns, Jean Burgess, Merja Mahrt, and Cornelius Puschmann. 2014. *Twitter and society*. Vol. 89. P. Lang.

[29] Joyce Jiyoung Whang, David F Gleich, and Inderjit S Dhillon. 2013. Overlapping community detection using seed set expansion. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2099–2108.

[30] Shaozhi Ye and S Felix Wu. 2010. Measuring message propagation and social influence on Twitter. com. In *International Conference on Social Informatics*. Springer, 216–231.

[31] Hongyang Zhang, Peter Lofgren, and Ashish Goel. 2016. Approximate Personalized PageRank on Dynamic Graphs. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1315–1324.

[5] https://conferences.oreilly.com/strata/big-data-conference-ca-2015
[6] http://www.pamitc.org/cvpr15/