Proceedings of the Ninth International AAAI Conference on Web and Social Media

WhichStreams: A Dynamic Approach for Focused Data Capture from Large Social Media

Thibault Gisselbrecht[+ *], **Ludovic Denoyer**[*], **Patrick Gallinari**[*], **Sylvain Lamprier**[*]
Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6, F-75005, Paris, France
CNRS, UMR 7606, LIP6, F-75005, Paris, France
[+]thibault.gisselbrecht@irt-systemx.fr [*]firstname.lastname@lip6.fr

Abstract

Due to the huge amount of data produced on large social media, capturing useful content usually implies to focus on subsets of data that fit with a pre-specified need. Considering the usual API restrictions of these media, we formulate this task of focused capture as a dynamic data sources selection problem. We then propose a machine learning methodology, named *WhichStreams*, which is based on an extension of a recently proposed combinatorial bandit algorithm. The evaluation of our approach on various Twitter datasets, with both offline and online settings, demonstrates the relevance of the proposal for leveraging the real-time data streaming APIs offered by most of the main social media.

1 Introduction

In the last decade, social networks have become an invaluable source of data for many applications and services. Collecting data from such social media thus stands as a key issue for several academic and industrial organizations. Alternatively to giving (costly) access to huge repositories of historical data, most of social media also offer streaming services that allow real-time tracking of their users' activity. However, leveraging such online streaming services to collect useful data for a specified task usually faces some restrictive constraints, both technical - related to the required computational resources - and business - limitation policies set by the queried social media. Real-time capture of the whole activity from a social media is therefore usually impossible. A classical strategy is then to define filters that allow the collection process to focus on useful data that fit with a predefined need. This corresponds to selecting data sources to follow (users, topics, keywords, etc...) in order to restrict the capture to a focused subset of the whole activity. However, such data sources sampling is very difficult to handle manually, as it is proposed by various specialized companies. In this paper, we propose an intelligent sampling strategy able to adapt itself to both the specified data need and the operational constraints of the capture. Besides its benefits for real-time applications, our proposal is also useful for collecting social datasets with expected properties.

Let us consider, for instance, the case of the very famous social media Twitter, on which more than an average of 7000 messages (*tweets*) are posted every second. Being able to consume such a huge amount of data requires to own very important computational and storage capacities, especially if one wishes to extract usable information for a given particular task. Moreover, as most of the main social media, Twitter has quickly understood the wealth of its data and now avoids the capture of its whole activity. Only data related to a limited set of indicators (authors or keywords for examples) can be collected simultaneously, which limits the knowledge about the media to a restricted subset of its global activity. In this context, defining a given data need can turn out to be a very difficult problem: How to define a static set of relevant indicators, while one does not know the distribution of the data over the network ? Moreover in dynamic contexts like social media ? While a data collection about a topic can be performed by defining a specific list of keywords, data obtained with such a method are likely to be very noisy or out of scope, due to the poor expressiveness of such a need formulation and the usually high number of matching messages. The companies which offer data access services well know this effect and many of them use human operators for tracking and modifying the target sources, which is expensive and cannot be done on a large scale.

In this work, we consider streaming systems which, given source users to follow, delivers the content produced by these sources during a specific time interval. Being given a reward function defined for the task in concern according to the utility of the data delivered by each source, we propose a solution to this sampling problem from unknown environments, which is based on a machine learning method, namely an extension of combinatorial bandit algorithms. Starting from an initial set of seed users, our *WhichStreams?* method explores, evaluates and continuously redefines the set of source users to follow. This allows one to progressively learn to focus on the most relevant sources of the media, under the specified operational constraints (limitations on the simultaneous capture ability, bounded resources, no prior information about the media, restricted knowledge about its activity). Note that this method works for any information goal that can be expressed as a reward function as this will be defined later on. It can be used for example for collecting topical messages, identifying thematic influencers or capturing data that tend to satisfy a panel of given end-users.

The contributions of the paper are the following:

- We present a task of focused data capture from large social media with unknown distributions under restrictive constraints;
- We formalize this task as a combinatorial bandit problem and propose an algorithm for solving the problem;
- We provide experimental evaluations on both offline datasets and live streaming experiments, that show the ability our method to automatically orient a realtime data capture toward relevant data sources in complex environments such as large social media.

The paper is organized as follows: Section 2 introduces our task of dynamic data collection from social media. Section 3 describes our algorithm for optimizing data collection from sources according to some reward criterion. Section 4 reports experiments allowing us to assess the performance of our approach. Section 5 presents related work and discusses possible extensions.

2 Dynamic Data Capture

The task of dynamic data capture from social media can then be seen as a problem of sources selection: the capture of the whole activity of the media being not possible, the aim is to efficiently define a subset of users that are likely to produce relevant content for the specified data need. We propose to consider this problem in the context of very large social media, where the choice of data sources cannot be performed manually due to the huge amount of data produced and to the high number of sources to investigate. For instance in our experiments (see section 4), although our approach could be applied on several other social media (all of which that offer real-time data streaming APIs), we focus on data capture from Twitter, where the total number of users is greater than 240 millions, with 5000 as the limit of simultaneously followed ones. In this context, selecting which users to follow is a difficult question, which requires the use of machine learning techniques that enable an efficient exploration of the possible sources of the queried media.

Consider a process which, at each time step, is allowed to collect the content produced by a subset of users in a social network. Given a reward function that renders the usefulness of the collected content from any followed user for a specified task, the aim is to define a decision strategy that allows the process to progressively move towards the most relevant users of the media. This decision strategy is defined so that it maximizes the cumulative score - evaluated by the reward function - provided by the followed users over time. In the following, this decision strategy is called *selection policy*. On Twitter for instance, the data captured from a user during a given time interval correspond to the set of tweets she publishes during this period, and the reward function can correspond to some thematic or influence score functions (see section 4).

In our context, a major difficulty is that one do not know anything *a priori* about the users of the media nor the relations that can exist between them: getting the list of all user accounts or the list of the friends of a given user for examples is usually not possible in our case, as it would require to query an often costly or restricted API (query frequency being usually very limited). We thus consider in this paper that, except from what is obtained via the streaming API considered, not any additional information is available. Among others, this implies that we cannot ground in the social graph to explore the set of users (which therefore avoids the use of techniques employed for related tasks of *Crawling*, see section 5). Traditional techniques for sampling, such as those that would be applied in a stored database, are not possible either.

Then, starting from a set of seed users, the aim is to define a process that, beyond exploiting relevant known users, is able to explore new unknown ones, by incrementally feeding the pool of possible sources as long as new users are met in the collected content. For instance, unknown users can be mentioned in messages posted by followed ones. Typically on Twitter, users can reply to others or can re-post (*retweet*) messages from other users (see section 4), which offers the opportunity to discover new data sources without requiring any external resources. The general process of *WhichStreams?*, depicted by figure 1, can be sketched as follows. At each period of the data capture, the process:

1. Selects users to follow among known ones according to the current selection policy;
2. Collects their published content during the time interval;
3. Feeds the set of known users with unknown users referenced in the collected messages;
4. Evaluates the collected data according to the reward function rendering their relevance for the task to solve;
5. Update the selection policy according to the obtained rewards.

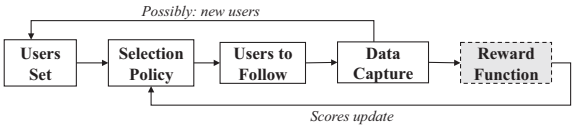

Figure 1: Overall process

3 Multi-armed bandit approach

In the this section, we formalize our problem as a bandit problem, which corresponds to a family of algorithms for the exploration of unknown environments with immediate reward feedback. After introducing general background about multi-armed bandits, we show that our problem can be expressed using the recently proposed combinatorial bandit framework. We also show that because of the extreme variability of user behaviors, existing bandit algorithms developed for this setting are not well adapted to our problem. We then propose an extension of the combinatorial UCB algorithm which better suits our needs and analyze its theoretical convergence.

3.1 Background: Multi-Armed Bandit

The multi-armed bandit problem is aimed at tackling the well-known trade off between exploration and exploitation in decision processes where, at each round (or time step), an agent has to choose an action among a finite set. According to the playing outcome of this action, the agent then receives a reward which quantifies the quality of the action. The goal for the agent is to maximize its cumulative reward trough time. The name "bandit" comes from the image of a gambler in front of several slot machines who can pull one arm every minute and then receives as a reward an amount of money. Her goal is obviously to maximize the total amount of earned money, or equivalently her cumulated reward. A good introduction to multi-armed bandit is given in (Auer, Cesa-Bianchi, and Fischer 2002). A general overview of the multi-armed bandit problem can be found in (Bubeck and Cesa-Bianchi 2012).

In the classical bandit setting, only one arm at a time is pulled and then evaluated. Here, one will consider the case where a unique gambler has to pull **several arms simultaneously** at each time step. This problem has been recently formalized and studied (Chen, Wang, and Yuan 2013), and is now known as the *Combinatorial bandit* problem. We provide a brief introduction to this problem below.

Notations Let us denote $\mathcal{K}$ the set of all K available actions and $\omega_{i,t} \in \Omega$ the playing outcome of action i at time step t. We suppose available a reward function g, which provides a score $g(\omega_{i,t})$ for every outcome $\omega_{i,t}$. Then, at each timestep $t = 1, 2..., n$ the agent has to:

- Choose a subset $\mathcal{K}_t \subseteq \mathcal{K}$ of k actions according to a selection policy π;
- Observe the outcome $\omega_{i,t}$ provided by each action i in $\mathcal{K}_t$ and receive the corresponding reward $g(\omega_{i,t})$;
- Try to improve the action-selection strategy based on these new observations in order to obtain a higher reward in future action selection.

The objective is thus to find the optimal selection policy π^* that provides the highest possible cumulative reward value:

$$\pi^* = \arg\max_{\pi} \sum_{t=1}^{n} \sum_{i \in \mathcal{K}_t} g(\omega_{i,t}) \tag{1}$$

where n is the total number of steps. Moreover, we suppose that we are in the so-called stochastic case where each reward distribution follows an unknown law ν_i with values in $[0, 1]$: $g(\omega_{i,.}) \sim \nu_i$ with mean μ_i to be estimated from the observations. Every time the action i is chosen, the knowledge about its empirical mean increases. In order to increase the knowledge about the whole environment, a tradeoff between exploitation of good actions and exploration of new or badly known actions has to be defined, which is the central point the different bandit policies differ on.

Combinatorial UCB The Combinatorial UCB algorithm recently proposed in (Chen, Wang, and Yuan 2013) is an extension of the original *UCB* (Auer, Cesa-Bianchi, and Fischer 2002) for cases where several actions can be performed simultaneously by a unique player. We consider the case where the reward obtained for a set of actions corresponds to the sum of its individual rewards. Outcomes of actions can then be considered individually.

As the classical *UCB* does, the combinatorial *UCB* algorithm uses a ranking score $v_{i,t}$ computed for each action i. At each time-step t of the process, it selects the k best actions according to this score rather than the best one for the classical *UCB*. Theoretical convergence guarantees have been given in (Chen, Wang, and Yuan 2013) for this algorithm.

Let us denote, $\tau_{i(t)}$ the number of times a given action i has been chosen during the first t time steps of the process and $g_{i,s}$ the s-th reward provided by playing i. The ranking score $v_{i,t}$ given to action i at timestep t is based on the empirical mean $\widehat{\mu}_{i,\tau_{i(t-1)}}$ of the rewards received for each action i in the $t-1$ first time steps, where $\widehat{\mu}_{i,x}$ is defined for any i in $\{1..K\}$ and $x \geq 1$ as:

$$\widehat{\mu}_{i,x} = \frac{1}{x} \sum_{s=1}^{x} g_{i,s} \tag{2}$$

The score $v_{i,t}$ used in (Chen, Wang, and Yuan 2013) to rank the arms at each round t is defined as:

$$v_{i,t} = \widehat{\mu}_{i,\tau_{i(t-1)}} + B_{i,t} \tag{3}$$

where $B_{i,t} = \sqrt{\frac{3 \ln(t)}{2\tau_{i(t-1)}}}$. This value defines a tradeoff between exploitation (the $\widehat{\mu}_{i,\tau_{i(t-1)}}$ term) and exploration (the $B_{i,t}$ term), as it sums a first term estimating the utility of arm i with a second term that decreases with the number of times this arm has been played before time t.

3.2 Collecting Data with Bandits

In this section, we formalize the task - dynamically collecting data on media sites - as a combinatorial bandit problem and propose an extension of the Combinatorial UCB algorithm better suited to this specific problem. Assuming that we are able to assess the collected data w.r.t. a given reward, our goal is to orient the users selection process towards users that produce the most useful content during the capture period. The bandit framework fits well our data capture problem: an action (or arm) corresponds to the selection of a user from the social network and playing outcomes correspond to the textual content produced by these selected users during a given period.

Data Capture as a Combinatorial Bandit Problem Let us denote $\mathcal{U}$ the set of users of the social network we are interested in. Let us also consider that the data capture period is divided into n time steps. The content produced by user i during the t-th time interval of capture is denoted $\omega_{i,t} \in \Omega$. On Twitter for instance, $\omega_{i,t}$ corresponds to the set of tweets published by user i during time interval t. We consider a reward function g which provides a score indicating the utility

of the collected data:

$$g : \begin{cases} \Omega \rightarrow [0;1] \\ g(\omega) = \text{ quality of } \omega \text{ for the given task} \end{cases} \quad (4)$$

This reward function depends on the task to solve and can for example correspond to a thematic quality or a popularity of the collected content. Different reward functions are defined in Section 4.

As mentioned before, we consider throughout the paper the frequent case of streaming APIs that give the opportunity to collect content produced by a given number k of users simultaneously. Our problem comes down to a combinatorial bandit problem as stated above, where multiple actions have to be selected at each time step. Given a time period of n time steps, a set of available users $\mathcal{K} \subseteq \mathcal{U}$ and an operational constraint k corresponding to the number of users that can be simultaneously captured, a selection policy is a function $\pi : \{1, ..., n\} \rightarrow \mathcal{K}^k$, where $\pi(t)$ defines, for a given time step t a subset of k users to follow. Our goal is then to find the best policy π^*, as defined in equation 1, that allows us to collect the best cumulative reward over captured content from followed users.

With all users from $\mathcal{U}$ known at the beginning of the process, and assuming hidden utility distributions for these users, the *Combinatorial UCB* (*CUCB*) algorithm presented above could be used to orient our data capture process towards the most useful users w.r.t. the task in concern. However, in our case, the **full set of actions is unknown**: assuming the frequent case where no exhaustive list is provided by the streaming API, only some seed users are known at the beginning of the process. Users are then added to the pool $\mathcal{K}$ iteratively, according to new users discovered in the data collected at each step.

Our dynamic data capture algorithm, *WhichStreams?*, is described in algorithm 1. At each time step t of the process, the algorithm computes a ranking score for every known user, then selects the k top-ranked users according to these scores, collects the content produced by these k users during the t-th streaming period, records their rewards to update the selection policy and finally feeds new users potentially met in the collected data to the pool of available users $\mathcal{K}$. The ranking score $v_{i,t}$ considered for each available user i at each time step t is defined as:

$$v_{i,t} = \begin{cases} \widehat{\mu}_{i,\tau_{i(t-1)}} + B_{i,t} & if\ \tau_{i(t-1)} > 0 \\ +\infty & if\ \tau_{i(t-1)} = 0 \end{cases} \quad (5)$$

where $B_{i,t}$ stands for the exploration term of our policy. Since the full set of users $\mathcal{U}$ is unknown, it is not possible to initialize the empirical reward mean of each user by choosing it once at the beginning of the process, as it is done with the classical *CUCB* policy. At each time step t, we therefore define $v_{i,t} = \infty$ for every user i for which $\tau_{i(t-1)} = 0$. This forces the algorithm to consider it in the set of followed users during the current period, in order to compute its mean reward value.

The Combinatorial UCBV policy For our sources selection problem, the reward for each user is computed based on the relevance of his content production during finite periods of streaming. Since users do not produce content at a fixed, continuous rate, high reward variations are likely to be observed during the capture process. Typically, most of the time, the users do not produce any content. Under the *CUCB* policy (i.e., with $B_{i,t} = \sqrt{\frac{3\ln(t)}{2\tau_{i(t-1)}}}$), the ranking score $v_{i,t}$ in equation 3 would penalize users that produced few content during the first periods they have been followed. Moreover, no difference would be done between a user who produces a large amount of poorly relevant content and a user who produces fewer but highly valuable data. In order to take into consideration this high variability in the user behaviors, we propose in this section a new combinatorial bandit algorithm called **Combinatorial UCBV (CUCBV)**, where "V" holds for variance, which considers the variance of the collected rewards.

Algorithm 1: The *WhichStreams?* algorithm

```
Input: K, k, n
for t ← 1 to n do
    for i ← 1 to K do
        Compute v_{i,t} with formula 5;
    end
    Order all users in decreasing order w.r.t. v_{i,t};
    Select the k first to set K_t;
    for i ∈ K_t do
        Follow i and observe ω_{i,t};
        Receive the reward g(ω_{i,t});
        Feed K with newly encountered users j, j ∉ K;
    end
end
```

The CUCBV algorithm extends the *UCBV* algorithm, proposed in (Audibert, Munos, and Szepesvári 2007), for the combinatorial case where several actions are performed simultaneously. UCBV, which considers the empirical reward variance of actions in its exploration term, indeed appears better fitted for our data capture task since it allows us to favor users with high reward variations. To the best of our knowledge, variance dependent bandit algorithms have never been proposed in a combinatorial setting.

Following *UCBV*, we therefore consider an exploration term $B_{i,t}$ based on the reward variance of users:

$$B_{i,t} = \sqrt{\frac{2\ln(t)\widehat{\sigma}^2_{i,\tau_{i(t-1)}}}{\tau_{i(t-1)}}} + \frac{3\ln(t)}{\tau_{i(t-1)}} \quad (6)$$

where $\widehat{\sigma}^2_{i,x}$ stands for the empirical reward variance of user i after its x-th selection:

$$\widehat{\sigma}^2_{i,x} = \frac{1}{x}\sum_{s=1}^{x}(g_{i,s} - \widehat{\mu}_{i,x})^2 \quad (7)$$

With such an exploration term, the policy tends to explore more frequently users that have a high empirical variance, since more information is required for these users to get a good estimation of their utility. In the following, we analyze the theoretical convergence for this extension of *UCBV* in a combinatorial context.

CUCBV Theoretical Convergence In bandit problem, algorithm performance is usually measured through the notion of regret, which corresponds to the reward loss an agent is expected to regret by choosing a given action i rather than the optimal one i^*. In our context, if we denote by $\mathcal{K}^*$ the subset of k actions that provides the highest value of cumulative reward over a time period of n time steps, the cumulative regret R_n computed at time step n is defined by:

$$R_n = \sum_{t=1}^{n} \sum_{j \in \mathcal{K}^*} g(\omega_{j,t}) - \sum_{i \in \mathcal{K}_t} g(\omega_{i,t}) \tag{8}$$

Hence we are interested in finding strategies that lead to a small value of the expected cumulative regret.

Proposition 1. *The expected cumulative regret $\mathbb{E}_\pi[R_n]$ for a given policy π and after a time period of n timesteps, can be defined as:*

$$\mathbb{E}_\pi[R_n] = \sum_{i=1}^{K} \mathbb{E}_\pi[\tau_{i(n)}]\Delta_i \tag{9}$$

where Δ_i is defined as the difference between $\overline{\mu}^*$, the average of the means of the reward distributions of the actions in $\mathcal{K}^*$, and μ_i, the mean of the reward distribution of action i: $\Delta_i = \overline{\mu}^* - \mu_i$ [1].

Proposition 2. *If we consider that the full set of actions $\mathcal{K}$ is known, we get, with our Combinatorial UCBV policy with $n \in \mathbb{N}^+$, the following bound:*

$$\boxed{\mathbb{E}[R_n] \leq \ln(n) \sum_{i \notin \mathcal{K}^*} \left(C + 8(\frac{\sigma_i^2}{\delta_i^2} + \frac{2}{\delta_i}) \right) \Delta_i + D} \tag{10}$$

where C and D are constants, δ_i corresponds to the difference between $\underline{\mu}^*$, the mean of the reward distribution of the worst action in $\mathcal{K}^*$ and μ_i, σ_i^2 is the variance of action i.

Proof. The general scheme of the proof can be sketched in three main steps[1]: 1) We separate the contribution of optimal and non optimal actions; 2) We cancel the optimal actions part using $\sum_{i \in \mathcal{K}^*} \Delta_i = 0$ and $\tau_{i(n)} \leq n \; \forall i$; 3) Following a method similar to (Audibert, Munos, and Szepesvári 2007), we upper-bound the non optimal actions contribution using Bernstein's inequality. □

Up to some constants, this result allows us to guarantee a logarithmic convergence for our Combinatorial UCBV algorithm. Note that, while this guarantee is defined for the case where the full set of users is known, it ensures that, from the moment when an optimal user (one with an expected reward in the k best ones) enters the pool, the process logarithmically converges towards a policy that recognizes it as so. Note also that the convergence analysis consider the *stationary* case where the reward distributions do not change over time. This usual assumption does not always hold in our context. However, our convergence results indicate that, if some users are useful sources during a sufficient period of time, our algorithm tends to select them.

In the next part, we describe a set of experiments allowing to evaluate our algorithm in different contexts.

[1]Details of the proof are available as supplementary content at *http://webia.lip6.fr/~lampriers/CUCBVproof.pdf*

4 Experiments

This section presents the results obtained on two sets of experiments. A preliminary set has been performed *offline*, by using already collected data with known properties, in order to compare our algorithm to some baselines. A second set of experiments has been performed *online*, in real-world capture settings, by using the Twitter streaming API during three weeks, and aims at demonstrating the ability of the proposed method to efficiently capture a specific relevant information under operational constraints.

4.1 Rewards definition

The rewards considered in our experiments correspond to three different examples of realistic problems of data capture. First, we propose a topical model which focuses on the content of published messages. This reward corresponds to a problem in which one wants to collect information on a predefined topic and orient the capture toward users that are the most likely to produce data on this topic. Then, we consider a task where one wants to capture content from influential users without any reference to a specific topical information need. Finally we propose an hybrid model, taking into account both the content of the posts and the influence of their authors to capture data from the best "thematic influencers" in the network. Note that many other rewards could be considered depending on the information need.

Topical model This reward model corresponds to a task where one wants to gather information concerning a topic, specified by a query defined as a weighted vector of terms[2]. Let us suppose that we have a dictionary of d terms, a query is defined as a vector $\mathcal{Q}$ of size d denoted: $\mathcal{Q} = (w_j^{\mathcal{Q}}, j = 1..d)$, where $w_j^{\mathcal{Q}}$ is the weight of term j in the query $\mathcal{Q}$.

We also represent the content in $\omega_{i,t}$ obtained by capturing data from a given user i at time step t using a vector $\mathcal{D}_{i,t} = (w_j^{i,t}, j = 1..d)$, where $w_j^{i,t}$ is the weight of term j in the messages posted during the corresponding time period[3]. A document vector is built by applying a Porter Stemmer on the collected messages, filtering terms not present in the dictionary and then assigning tf weights to the remaining ones ($w_j^{i,t}$ equals the number of occurrences of the term j in the messages posted by user i at time step t).

The reward function $g_1(\omega_{i,t})$ is then defined for data $\omega_{i,t}$ captured from a given user i at time step t as the cosine between the content $\mathcal{D}_{i,t}$ of $\omega_{i,t}$ and the query $\mathcal{Q}$:

[2]Note that, we consider here a simple bag of words model, but more complex text models could be used as well.

[3]A user can post several messages at each time step. For this study, we consider that a set of messages arriving at the same time step are grouped into a single one by simply adding the weights of all the corresponding vectors.

$$g_1(\omega_{i,t}) = \frac{\mathcal{D}_{i,t}.\mathcal{Q}}{\|\mathcal{D}_{i,t}\|\,\|\mathcal{Q}\|} = \frac{\sum_{j=1}^{d} w_j^{i,t} w_j^{\mathcal{Q}}}{\sqrt{\sum_{j=1}^{d}(w_j^{i,t})^2}\sqrt{\sum_{j=1}^{d}(w_j^{\mathcal{Q}})^2}}$$

Influence Model Via the Twitter streaming API, messages from non-followed users that correspond to *Retweet* (i.e., a re-post) or *Reply* (i.e., an answer) to messages posted by a given user i are obtained when i is followed. In (Cha et al. 2010), the authors show that the influence of a user on Twitter can be measured through its number of *Retweet* and *Mention*. This allows us to consider an example of reward function using those indicators of the influence of a particular user, regardless of the published content.

More formally, by denoting $n_{i,t}^{Rt}$ and $n_{i,t}^{Rp}$ respectively the number of *Retweet* and *Reply* of messages from user i between t and $t+1$, our second reward function $g_2(\omega_{i,t})$ is defined as:

$$g_2(\omega_{i,t}) = \tanh(n_{i,t}^{Rt} + n_{i,t}^{Rp})$$

where $\tanh$ is the hyperbolic tangent function.

Thematic Influence Model This hybrid model combines the two previous ones. The reward function characterizes both the content of the collected messages and the influence of the considered users. What we want here is to orient the process toward thematic influencers on a particular topic. We propose the following reward function:

$$g_3(\omega_{i,t}) = \tanh(g_1(\omega_{i,t}) \times n_{i,t}^{Rt})$$

4.2 Preliminary experiments

Offline preliminary experiments were performed on two datasets. They allow us to easily compare different algorithms. On these datasets, the algorithms can be compared to a ground truth providing which sources are the most useful for the task. In particular, we consider in the following a policy that selects the subset of users that finally provide the best cumulative reward. Of course, such policy cannot be used for online data capture since, by nature, best users are unknown in real-world settings.

The first dataset is user-centered, while the second is focused on a particular keyword.

- The first dataset, called *usElections*, corresponds to the set of all messages obtained via the Twitter streaming API plugged on 5000 user accounts during the 10 days preceding the 2012 US presidential elections. The chosen accounts were the 5000 first ones to use one of the keywords "Obama", "Romney" or "#USElections" . It finally contains 2148651 messages from the 5000 streamed users, plus 1686797 messages that correspond to *Retweet* or *Reply* to messages of these users. On this dataset, we consider time steps of 100 seconds.
- The second dataset, called *Libya*, is the result of 3 months of static capture using the keyword "Libya" from the Twitter streaming API. It contains 1211475 messages from 17341 users. By its nature, this dataset does not allow to experiment influence based reward models, since data collected via a given keyword do not provide *Retweet* or *Reply* information about a specific set of users. Only the topical model is then considered on this dataset. For this dataset we considered time intervals of 500 seconds.

Note that, with both datasets, in order to simulate realistic experimental settings, we started the capture processes with a set of available users $\mathcal{K}$ that only contained the k first users of the corpus (those that posted the earliest messages). At each time step t, new users are then added to $\mathcal{K}$, according to those that are either referenced in messages collected during time interval t or have posted replies (or retweets) to messages from followed users in $\mathcal{K}_t$.

For the topical model described in section 4.1, we use a specific dictionary for each dataset, which has been built by using a Porter stemmer algorithm on every message and then choosing the 2000 most frequent terms in the dataset in concern. Each message is then transformed into a frequency vector using this dictionary. For the queries, we generated 500 queries for each dataset by randomly sampling weight vectors from a uniform distribution in $[0,1]^{2000}$ and then selecting 50 terms at random. This strategy generates very different queries. Note that considering a large set (500 here) of queries allows to get more reliable results than when considering a unique topical search. All offline reported results concerning the topical reward model thus correspond to an average over these 500 randomly generated queries.

Results Besides our CUCBV policy which makes use of ranking scores defined by equation 5, we also consider a classical combinatorial *CUCB* version that uses scores given by equation 3 to analyze the benefits resulting from using the empirical variance in our selection strategy. We compare these two algorithms with two baseline policies:

- A *BestSubset* policy, that captures rewards from the optimal subset of users $\mathcal{K}^*$ (as defined above);
- A *Random* policy, which uniformly selects k users to follow at each time step.

Every tested policy consider subsets of $k = 100$ simultaneously followed users.

Figures 2, 3 and 4 present the evolution of the cumulative reward w.r.t time for these four policies tested on the *usElections* dataset and for the three reward models (there are thus 4×3 curves in total). Figure 5 presents cumulative reward results for the topical reward model for the *Libya* dataset.

First, it may be noticed that, as expected, the *BestSubset* curve is the highest, on the whole process, in every experimental setting. The fact that this *BestSubset* policy exhibits a greater slope than the *Random* one in every situation and at every time step shows that some users have a better reward distribution than others during the whole streaming period.

Although not at the same level as the *BestSubset* policy, both dynamic capture policies $CUCBV$ and $CUCB$ get significantly greater performance results than the *Random* strategy, which confirms that using bandit algorithms

for data capture from flows of social activity is relevant, at least with the ratios k/K used in the experiments (0.02 for the *usElections* dataset and 0.005 for the *Libya* one). The influence of the ratio over the performance will be explored later. Our general dynamic data capture framework *WichStreams?* thus appears valid for collecting useful data from targeted users.

For every reward type and on the *usElections* dataset, we observe that our *CUCBV* policy outperforms the *CUCB* policy. Considering the empirical variance in the exploration term leads to more reliable estimates for users with high reward variability, by allowing more frequent explorations of these users. In our data capture setting, this is particularly relevant since obtained rewards greatly depend on the message posting frequencies of the users. During several time steps for instance, null rewards may be obtained for some users uniquely because of bad luck, for example the process did not follow them when they posted useful material. Increasing the exploration probability of users with high reward variance allows the process to reconsider these users when some good rewards are obtained. Moreover, some users may be slightly less active than others while owning a greater utility for the reward in concern: because more focused on a given topic for example, they post less messages but with greater impact on the cumulative reward. Considering the variance in the exploration term of the policy allows one to give more chances to such users.

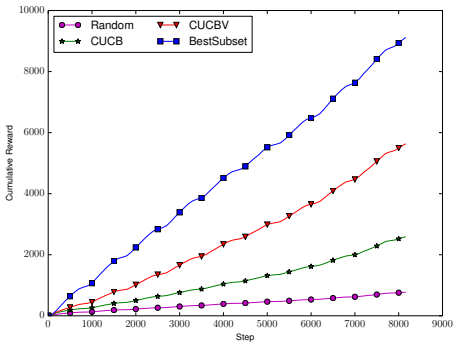

Figure 2: Cumulative reward versus timestep for the *usElections* dataset using the topical model.

On the *Libya* dataset, we also observe that our algorithms collect more valuable data than the random baseline. In this case, we also notice a particular shape of the curves around the 8000^{th} time step. This period corresponds to a large peak of activity regarding political events in Libya. Even if our data capture algorithms can follow this trend, both *CUCB* and *CUCBV* policies are quite far from the *BestSubset* one. Such sudden activity peaks are difficult to handle for these algorithms which have not enough time to stabilize themselves on useful users. However, in real-world settings or with data collected by following user accounts rather than a specific keyword as it has been done for this dataset, such activity peaks are less likely observed.

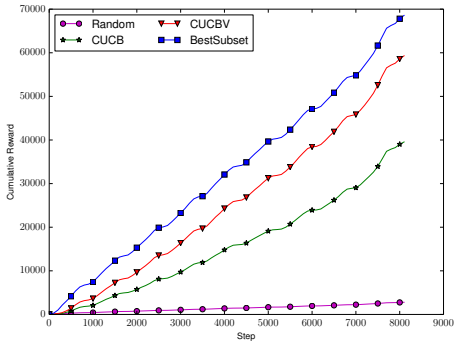

Figure 3: Cumulative reward versus timestep for the *usElections* dataset using the influence model.

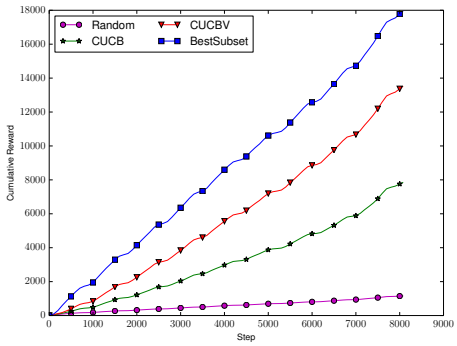

Figure 4: Cumulative reward versus timestep for the *usElections* dataset using the thematic influence model.

Figure 6 plots the evolution of the *capture rate* on the *usElections* dataset with thematic influence reward. This capture rate is defined for a timestep t and a policy π as the average of rewards obtained by the policy π during a time window $[t-500, t+500]$, normalized by the average of rewards that would have been obtained by following the best subset of users $\mathcal{K}^*$ on the same period:

$$C(\pi, t) = \frac{\sum_{s=t-500}^{t+500} \sum_{i \in \mathcal{K}_s^\pi} g(\omega_{i,s})}{\sum_{s=t-500}^{t+500} \sum_{i \in \mathcal{K}^*} g(\omega_{i,s})} \tag{11}$$

This quantity represents the relative velocity of the capture w.r.t. the *BestSubset* strategy. These results allow to highlight the fact that when this relative capture rate remains constant for the *Random* strategy, it increases for all of our bandit policies during the data capture process. At the end of the process, *CUCBV* even reaches a capture rate very close to the *BestSubset* strategy ($C(CUCBV, 8000) > 0.9$).

Finally, figure 7 plots the cumulative reward for *Random*, *CUCB* and *CUCBV* algorithms w.r.t to both time and the number of users k used for the capture, on the *usElections* dataset with the topical reward model. We notice that the

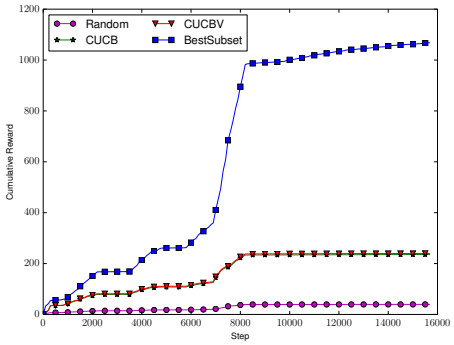

Figure 5: Cumulative reward versus timestep for the *Libya* dataset using the topical model.

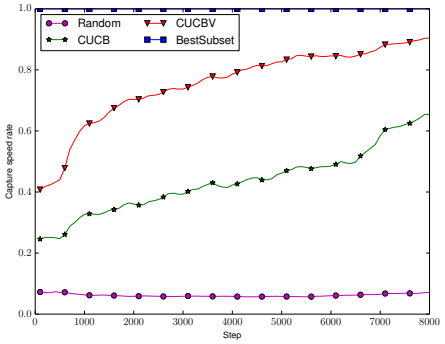

Figure 6: Capture rate evolution for the thematic influence model on the *usElections* dataset.

surfaces never cross each other and that *CUCBV* is always above *CUCB*, which itself is above random. This means that no matter the number of selected users, *CUCBV* appears to be the best option for this dataset. We also observe that, as expected, performances of our method increase fast with the number of users k simultaneously followed.

4.3 Real-World Experiments

Offline experiments allowed us to analyze the behavior and performances of our model in a controlled setting. Now, we propose to consider their application on a real-world capture task, in order to assess their effectiveness when dealing with the entire social activity of a social media, under the operational constraints that are set by the streaming API.

Experimental Settings The social network considered for our online experiments is again Twitter. It limits the number of simultaneously followed users and the frequency of streaming settings modifications. We have set $k = 4000$ and consider a time step corresponding to intervals of 300 seconds, which fits with these constraints while providing a good visibility of the current social activity and a sufficient re-orientation frequency for our capture process. Because

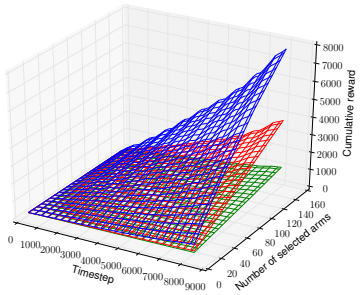

Figure 7: Reward as a function of time and number of streamed users for *usElections* dataset and topical model: above surface corresponds to CUCBV, middle to CUCB and below to Random.

the number of connections with the same account is limited by the Twitter API, we only considered *CUCBV*, which appeared as the best policy in our offline experiments, and a random baseline policy, which uniformly selects the users to follow at each time step. Note that for this real-world online capture setting the best set of users $\mathcal{K}^*$ is unknown, so that the *BestSubset* policy cannot be considered.

With two Twitter accounts, one for *CUCBV* and one for the random policy, we simultaneously captured data from the streaming API of Twitter during three weeks, starting from a set of three initial users: *BBC*, *CNN* and *FoxNews*, which correspond to very active accounts allowing one to quickly obtain useful data. The set of potentially followed accounts $\mathcal{K}$ is fed at each iteration t with users that either replied to (or retweeted) already followed users in $\mathcal{K}_t$ or are referenced in the collected messages during this time period. Obviously, it may take some time to reach a value of 4000 users. So, at the very beginning of the process, every user is followed at each time step whatever the considered strategy (before reaching k users, no selection decision can be taken as every user from the pool will be followed). For these live experiments, we consider the thematic influence model, which uses a thematic query created by considering the 300 most employed stems on the Wikipedia Libya page (we used binary weights). Every post is stemmed on the fly as we collected in real time. Note lastly that, as a large amount of new users can be encountered during some capture periods, we limited the number of new users added to the pool to 1000 at each iteration, in order to prevent cases where the number of new encountered users reaches k, which would avoid any exploitation by *CUCBV*.

Results Figure 8 plots the cumulative reward obtained by the random and the *CUCBV* capture policies as a function of time. It highlights the good performances of our proposal in this real-world settings, as the curve of *CUCBV* increases significantly faster than the one of the random strategy. At the end of the process, the cumulative reward obtained with *CUCBV* is 7.5 times bigger than the one obtained with the

random policy. It can be reasonably assumed that this difference would even increase when considering a streaming period longer than three weeks.

Note that the number of 500 000 users was reached much faster with the *CUCBV* than with the random policy (three days against one week). This is coherent with the fact that *CUCBV* orients better toward influential users. Indeed, the former being frequently re-tweeted, this allows the process to discover significantly more data sources than a random capture process.

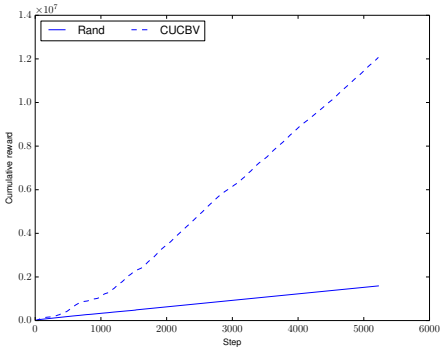

Figure 8: Cumulative reward versus timestep for the live stream experiment.

Finally, figure 9 represents the empirical mean of 50 000 randomly chosen users with respect to the number of times they have been followed with the *CUCBV* at the end of the process. Considering an arbitrary threshold mean of 0.6 to distinguish "good" and "bad" users, we note that the good ones represent 1.73% of the whole set of profiles. Besides, the bad ones, which represent the majority of the users, were followed just a few time, which again shows the ability of the policy to focus on good users. Moreover our algorithm succeeded in following 75% of the good users at least 500 times. The 25% of (uncertainly) good users which have been streamed less than 500 times are users will be likely considered during the next iterations of the process.

5 Related Work

5.1 Data Capture

Works on emerging topics detection naturally follow on from those on Topic Detection and Tracking (TDT). Given a flow of text messages, TDT aims at identifying trending topics in a streamed source. For instance, (Cataldi, Di Caro, and Schifanella 2010) proposed an approach that continuously deals with new messages posted on Twitter for discovering keyword vectors that correspond to current hot topics on the network. However, the question of data collection, and of capture orientation, is left aside by this work, that simply uses the random stream proposed by Twitter. More recently, (Colbaugh and Glass 2011) proposed a model that considers the blogosphere as a network, where each node corresponds to a blog that can be followed. With the aim of identifying emerging topics, they search for blogs that tend to report

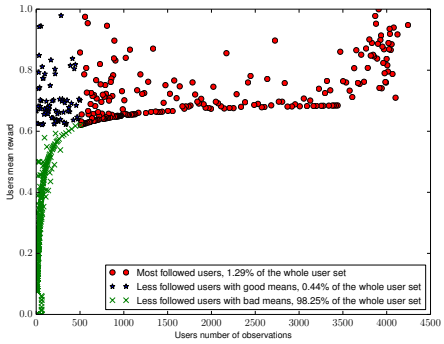

Figure 9: Users mean reward versus number of observations for the live experiments and CUCBV policy. Every symbol represents a user.

tracked contents early in their life-cycles. In that sense, they perform an orientation toward useful data sources, but this is done statically, on training data collected from every node, which is only possible thanks to the small size of the network. The work reported in (Li, Wang, and Chang 2013) is probably one of the closest to ours, since it includes an automatic re-orientation of the data capture process via the streaming API of Twitter. However, it is only fitted for the task of emerging topics detection and for the specific Twitter APIs. It also differs from our work by the fact that, rather than incrementally moving toward useful areas of the network, it makes use of a random sampling strategy to determine the keywords to follow at each time step.

Web page crawling is another research area that can be considered as related to our work, since it defines strategies to select which nodes of the web to visit at each step of its process. Focused crawling, first introduced in (Chakrabarti, van den Berg, and Dom 1999), defines whether a web page should be crawled or not according to the relevance of its context and of its link structure for a predefined topic. Variants of this concept have been proposed, to focus data collection on useful web pages, such as those presented in (Micarelli and Gasparetti 2007) which define strategies that attempt to adapt themselves according to the neighborhood of the current crawling environment. However, all these approaches require a prior knowledge of the link structure of the data sources. This is usually not the case in our context and the extraction of the underlying structure may reveal as a very difficult task. (Boanjak et al. 2012) proposed a distributed architecture to apply focused crawling on Twitter, by intensively querying the Twitter API from many clients to get the social graph (i.e., the followers graph) of the network. However, while it was already limited by the Twitter API (in term of crawling frequency) at the time of this work, this approach would not be viable today given the current access restriction of the API.

5.2 Multi-armed Bandit

There is a wide number of publications concerning bandit algorithms and their applications to a variety of prob-

lems. Bandits are particularly fitted in online contexts, such as dynamic data capture problems, for continuous learning from data flows when no prior information is available. In the context of social networks, bandit algorithms have for example been successfully applied for recommendation (Kohli, Salek, and Stoddard 2013), network crawling (Bnaya et al. 2013), online advertising (Buccapatnam, Eryilmaz, and Shroff 2014) or audience maximization (Lage et al. 2013). Nevertheless, none of these works had to deal with simultaneous decisions, as it is the case in the multi-sources data capture problem we address in this paper.

The combinatorial bandit problem has been studied in (Chen, Wang, and Yuan 2013) and the authors propose the *CUCB* algorithm. Recently in (Gopalan, Mannor, and Mansour 2014), the authors generalized Thompson sampling, another bandit algorithm, to select a subset of size k among a larger set of K arms at each time step. This approach could appear well fitted for our task of dynamic data capture. It however cannot be efficiently deployed in contexts with a large number of arms, since each iterations in this method requires a complex sampling process on every available arm.

6 Conclusion

In this paper, we have considered the problem of data capture from large social media under restrictive operational constraints based on multi-arm bandit approaches, which define a framework for the learning of exploration strategies in unknown environments with immediate rewards. We have extended the recent Combinatorial UCB approach, which allows us to solve problems where simultaneous decisions have to be taken, for our problem of data capture from simultaneous sources. Faced to the high variability of the rewards obtained in our context of data capture, we have proposed the Combinatorial UCBV approach, which aims at better handling such problems by including the variance of the rewards in its exploration strategy. After giving convergence guarantees for this general algorithm, we have experimented it in our context of data capture from social media. Experiments on both offline data sets and real-world settings have shown the ability of our algorithm to automatically discover relevant information sources given a goal specified as a reward function.

The work proposed in this paper constitutes a new way of managing online social data. Various perspectives can thus be considered. First, while the developed approaches are based on the assumption that the reward distribution is stationary over time (i.e, users keep a same behavior during the whole process), we are now experiencing new methods for non-stationary distributions, that are expected to better fit for long term capture in complex environments. Another perspective is also to extend the present work to combinatorial rewards, where the usefulness of a subset of users does not simply correspond to a sum of individual rewards, but can for example consider the diversity of the selected users. It can also serve as a basis for many concrete online tasks that have to deal with big social data, such as content tracking through social networks or online modeling of information diffusion.

Acknowledgments

This research work has been carried out in the framework of the Technological Research Institute SystemX, and therefore granted with public funds within the scope of the French Program "Investissements d'Avenir". Part of the work was supported by project Luxid'x financed by DGA on the Rapid program.

References

Audibert, J.-Y.; Munos, R.; and Szepesvári, C. 2007. Tuning bandit algorithms in stochastic environments. In *ALT '07*, ALT '07, 150–165.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine Learning* 47(2-3):235–256.

Bnaya, Z.; Puzis, R.; Stern, R.; and Felner, A. 2013. Bandit algorithms for social network queries. In *SocialCom '13*, 148–153.

Boanjak, M.; Oliveira, E.; Martins, J.; Mendes Rodrigues, E.; and Sarmento, L. 2012. Twitterecho: A distributed focused crawler to support open research with twitter data. In *WWW '12 Companion*, 1233–1240.

Bubeck, S., and Cesa-Bianchi, N. 2012. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Foundations and Trends in Machine Learning* 5(1):1–122.

Buccapatnam, S.; Eryilmaz, A.; and Shroff, N. B. 2014. Stochastic bandits with side observations on networks. In *SIGMETRICS '14*, 289–300.

Cataldi, M.; Di Caro, L.; and Schifanella, C. 2010. Emerging topic detection on twitter based on temporal and social terms evaluation. In *MDMKDD '10*, 4:1–4:10.

Cha, M.; Haddadi, H.; Benevenuto, F.; and Gummadi, K. P. 2010. Measuring user influence in twitter: The million follower fallacy. In *ICWSM '10*.

Chakrabarti, S.; van den Berg, M.; and Dom, B. 1999. Focused crawling: A new approach to topic-specific web resource discovery. In *WWW '99*, 1623–1640.

Chen, W.; Wang, Y.; and Yuan, Y. 2013. Combinatorial multi-armed bandit: General framework and applications. In *ICML '13*, volume 28, 151–159.

Colbaugh, R., and Glass, K. 2011. Emerging topic detection for business intelligence via predictive analysis of 'meme' dynamics. In *AAAI Spring Symposium: AI for Business Agility*.

Gopalan, A.; Mannor, S.; and Mansour, Y. 2014. Thompson sampling for complex online problems. In *ICML '14*, volume 32 of *JMLR Proceedings*, 100–108. JMLR.org.

Kohli, P.; Salek, M.; and Stoddard, G. 2013. A fast bandit algorithm for recommendation to users with heterogenous tastes. In desJardins, M., and Littman, M. L., eds., *AAAI*. AAAI Press.

Lage, R.; Denoyer, L.; Gallinari, P.; and Dolog, P. 2013. Choosing which message to publish on social networks: A contextual bandit approach. In *ASONAM '13*, 620–627.

Li, R.; Wang, S.; and Chang, K. C.-C. 2013. Towards social data platform: Automatic topic-focused monitor for twitter stream. *Proc. VLDB Endow.* 6(14):1966–1977.

Micarelli, A., and Gasparetti, F. 2007. Adaptive focused crawling. In *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*. 231–262.