Proceedings of the Tenth International AAAI Conference on
Web and Social Media (ICWSM 2016)

# Dynamic Data Capture from Social Media Streams: A Contextual Bandit Approach

**Thibault Gisselbrecht**[+*] and **Sylvain Lamprier**[*] and **Patrick Gallinari**[*]
[+]Technological Research Institute SystemX, 8 Avenue de la Vauve, 91120 Palaiseau, France
*thibault.gisselbrecht@irt-systemx.fr*
[*]Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606, 4 place Jussieu, 75005 Paris
*firstname.lastname@lip6.fr*

**Abstract**

Social Media usually provide streaming data access that enable dynamic capture of the social activity of their users. Leveraging such APIs for collecting social data that satisfy a given pre-defined need may constitute a complex task, that implies careful stream selections. With user-centered streams, it indeed comes down to the problem of choosing which users to follow in order to maximize the utility of the collected data w.r.t. the need. On large social media, this represents a very challenging task due to the huge number of potential targets and restricted access to the data. Because of the intrinsic non-stationarity of user's behavior, a relevant target today might be irrelevant tomorrow, which represents a major difficulty to apprehend. In this paper, we propose a new approach that anticipates which profiles are likely to publish relevant contents - given a predefined need - in the future, and dynamically selects a subset of accounts to follow at each iteration. Our method has the advantage to take into account both API restrictions and the dynamics of users' behaviors. We formalize the task as a contextual bandit problem with multiple actions selection. We finally conduct experiments on Twitter, which demonstrate the empirical effectiveness of our approach in real-world settings.

## 1 Introduction

In recent years, many social websites that allow users to publish and share content online have appeared. For example Twitter, with 302 million active users and more than 500 millions posts every day, is one of the main actors of the market. These social media have become a very important source of data for many applications. Given a pre-defined need, two solutions are usually available for collecting useful data from such media: 1) getting access to huge repositories of historical data or 2) leveraging streaming services that most media propose to enable real-time tracking of their users' activity. As the former solution usually implies very important costs for retrieving relevant data from big data warehouses, the latter may constitute a very relevant alternative that enables real-time access to focused data. However, it implies to be able to efficiently target relevant streams of data.

In this paper, we consider the case of user-centered streams of social data, i.e. where each individual stream furnishes access to the data published by a particular user, under restrictive constraints related to the number of streams that can be simultaneously considered. On Twitter for instance, data capture is limited to 5000 simultaneous streams[1]. Regarding the huge number of available users on this media, it requires an incredibly important effort for targeting relevant sources. Selecting relevant users to follow among the whole set of accounts is very challenging. Imagine someone interested in politics wishing to keep track of people in relation to this topic in order to capture the data they produce. One solution is to manually select a bucket of accounts and follow their activity along time. However, Twitter is known to be extremely dynamic and for any reason some users might start posting on this topic while others might stop at any time. So if the interested user wants to be up to date, he might change the subset of followed accounts dynamically, his goal being to anticipate which users are the most likely to produce relevant content in a close future. This task seems hard to handle manually for two major reasons. First, the criteria chosen to predict whether an account will potentially be interesting might be hard to define. Secondly, even if one would be able to manually define those criteria, the amount of data to analyze would be too large.

Regarding these important issues, we propose a solution that, at each iteration of the process, automatically selects a subset of accounts that are likely to be relevant in the next time window, depending on their current activity. Those accounts are then followed during a certain time and the corresponding published contents are evaluated to quantify their relevance. The algorithm behind the system then learns a policy to improve the selection at the next time step. We tackle this task as a contextual bandit problem, in which at each round, a learner chooses an action among a bigger set of available ones, based on the observation of action features - also called context - and then receives a reward that quantifies the quality of the chosen action. In our case, considering that following someone corresponds to an action, several actions have to be chosen at each time step, since we wish to leverage the whole capture capacity allowed by the streaming API and therefore collect data from several simultaneous streams.



[1]In this paper we focus on Twitter, but the approach is valid for many other social platforms.

With the current activity of a certain user corresponding to its context features, our task suits well the contextual bandit framework. However, in its traditional instance, features of every action are required to be observed at each iteration to perform successive choices. For our specific task, we are not able to observe the current activity of every potential user. Consequently, we are not able to get everyone's context and directly use classical contextual bandit algorithms such as *LinUCB* (Chu et al. 2011). On the other hand, it cannot be treated as a traditional (non-contextual) bandit problem since we would lose the useful information provided by observed context features. To the best of our knowledge, such an hybrid instance of bandit problem has not been studied yet. To solve our problem, we propose a data capture algorithm which, along with learning a selection policy of the $k$ best streams at each round, learns on the feature distributions themselves, so as to be able to make approximations when features are hidden from the agent.

The paper is organized as follows: Section 2 presents some related work. Section 3 formalizes the task and presents our data capture system. Section 4 then describes the model and the algorithm proposed to solve the task. Finally, section 5 reports various sets of experimental results.

## 2 Related Work

In this section we first present some literature related to bandit learning problems, then we discuss about some data capture related tasks and finally present some applications of bandits for social media tasks.

The multi-armed bandit learning problem, originally studied in (Lai and Robbins 1985) in its stationary form has been widely investigated in the literature. In this first instance, the agent has no access to side information on actions and assume stationary reward distributions. A huge variety of methods have been proposed to design efficient selection policies, with corresponding theoretical guarantees. The famous Upper Confidence Bound (*UCB*) algorithm proposed in (Auer, Cesa-Bianchi, and Fischer 2002) and other UCB-based algorithms ((Audibert, Munos, and Szepesvari 2007; Audibert and Bubeck 2009)) have already proven to solve the so-called stochastic bandit problem. This type of strategies keeps an estimate of the confidence interval related to each reward distribution and plays the action with highest upper confidence bound at each time step. In (Auer, Cesa-Bianchi, and Fischer 2002), the authors introduce the contextual bandit problem, where the learner observes some features for **every** action before choosing the one to play, and propose the *LinRel* algorithm. Those features are used to better predict the expected rewards related to each action. More recently, the *LinUCB* algorithm, which improves the performance of *LinRel* has been formalized (Chu et al. 2011). Those two algorithms assume the expected reward of an action to be linear with respect to some unknown parameters of the problem to be estimated. In (Kaufmann, Cappe, and Garivier 2012), the authors propose the *Bayes-UCB* algorithm which unifies several variants of *UCB* algorithms. For both the stochastic and contextual bandit case, Thompson sampling algorithms, which introduce randomness on the exploration by sampling action parameters from their posterior distributions, have been designed (Kaufmann, Korda, and Munos 2012; Agrawal and Goyal 2012a; Chapelle and Li 2011; Agrawal and Goyal 2012b). More recently, the case where the learner can play several actions simultaneously has been formalized respectively in (Chen, Wang, and Yuan 2013) (*CUCB* algorithm) and (Qin, Chen, and Zhu 2014) ($C^2UCB$ algorithm) for the non contextual and the contextual case. Finally in (Gisselbrecht et al. 2015) the authors propose the *CUCBV* algorithm which extends the original *UCBV* of (Audibert, Munos, and Szepesvari 2007) to the multiple plays case. To the best of our knowledge, no algorithm exists for our case, i.e. when feature vectors are only observable with some given probability.

Regarding social media, several existing tasks present similarities with ours. In (Li, Wang, and Chang 2013), the authors build a platform called ATM that is aimed at automatically monitoring target tweets from the Twitter *Sample* stream for any given topic. In their work, they develop a keyword selection algorithm that efficiently selects keywords to cover target tweets. In our case, we do not focus on keywords but on user profiles by modeling their past activity on the social media. In (Colbaugh and Glass 2011) the authors model the blogosphere as a network in which every node is considered as a stream that can be followed. Their goal is to identify blogs that contain relevant contents in order to track emerging topics. However, their approach is static and their model is trained on previously collected data. Consequently, this approach could not be applied to our task: first we cannot have access to the whole network due to APIs restrictions and secondly, the static aspect of their methods is not suitable to model the dynamics of users' behaviors that can change very quickly. In (Hannon, Bennett, and Smyth 2010) and (Gupta et al. 2013), the authors build recommendation systems, respectively called *Twittomender* and *Who to Follow*. In the former collaborative filtering methods are used to find twitter accounts that are likely to interest a target user. In the latter, a *circle of trust* (which is the result of an egocentric random walk similar to personalized *PageRank* (Fogaras et al. 2005)) approach is adopted. In both cases, authors assume the knowledge of the whole followers/followees Twitter graph, which is not feasible for large scale applications due to Twitter restriction policies. Moreover, the task is quite different from ours. While these two models are concerned about the precision of the messages collected from the selected followees (in order to avoid relevant information to be drown in a too large amount of data), we rather seek at maximizing the amount of relevant data collected in a perspective of automatic data capture.

Bandit algorithms have already been applied to various tasks related to social networks. For example in (Kohli, Salek, and Stoddard 2013), the authors handle abandonment minimization tasks in recommendation systems with bandits. In (Buccapatnam, Eryilmaz, and Shroff 2014), bandits are used for online advertising while in (Lage et al. 2013), the authors use contextual bandit for an audience maximization task. In (Gisselbrecht et al. 2015), a data capture task is tackled via non-contextual bandit algorithms. In that case, each user is assumed to own a stationary distribution and the process is supposed to find users with best means by trading

off between the exploitation of already known good users to follow and the exploration of unknown ones. The proposed approach mainly differs from this work for the non-stationnarity assumptions it relies on. This work is considered in the experiments to highlight the performances of the proposed dynamic approach.

# 3 Data Streams Selection

## 3.1 Context

As we described in the introduction, our goal is to propose a system aimed at capturing some relevant data from streaming APIs proposed by most of social media. Face to such APIs, two different types of data sources, users or keywords (or a mixture of both kinds), can be investigated. In this paper, we focus on users-centered streams. Users being the producers of the data, following them enables more targeted data capture than keywords. Following keywords to obtain data about a given topic for example would be likely to lead to the collection of a greatly too huge amount of data[2], which would imply very important post-filtering costs. Moreover, considering user-centered streams allows a larger variety of applications, with author-related aspects, and more complex topical models, than focusing on keywords-centered streams.

Given that APIs usually limit the ability to simultaneously follow users' activity to a restricted number of accounts, the aim is to dynamically select users that are likely to publish relevant content with regard to a predefined need. Note that the proposed approach would also be useful in the absence of such restriction, due to the tremendous amount of data that users publish on main social media. The major difficulty is then to efficiently select user accounts to follow given the huge amount of users that post content and that at the beginning of the process, no prior information is known about potential sources of data. Moreover, even if some specific accounts to follow could be found manually, adapting the capture to the dynamics of the media appears intractable by hand. For all these reasons, building an automatic solution to orient the user's choices appears more than useful.

To make things concrete, in the rest of the paper we set up in the case of Twitter. However, the proposed generic approach remains available for any social media providing real-time access to its users' activity.

## 3.2 A Constrained Decision Process

As described above, our problem comes down to select, at each iteration $t$ of the process, a subset $\mathcal{K}_t$ of $k$ user accounts to follow, among the whole set of possible users $\mathcal{K}$ ($\mathcal{K}_t \subseteq \mathcal{K}$), according to their likelihood of posting relevant tweets for the formulated information need. Given a relevance score $r_{a,t}$ assigned to the content posted by user $a \in \mathcal{K}_t$ during iteration $t$ of the process (the set of tweets he posted during iteration $t$), the aim is then to select at each iteration $t$ over $T$ the set of user accounts that maximizes the sum of collected relevance scores:

$$\max_{(\mathcal{K}_t)t=1..T} \quad \sum_{t=1}^{T} \sum_{a \in \mathcal{K}_t} r_{a,t} \tag{1}$$

Note that, in our task, we are thus only focused on getting the maximal amount of relevant data, the precision of the retrieved data is not of our matter here. This greatly differs from usual tasks in information retrieval and followees recommendation for instances.

Relevance scores considered in our data capture process depend on the information need of the user of the system. The data need in question can take various forms. For example, one might want to follow the activity of users that are active on a predefined topic or influent in the sense that their messages are reposted a lot. They can depend on a predefined function that automatically assigns a score according to the matching of the content to some requirements (see section 5 for details).

Whereas (Gisselbrecht et al. 2015) relies on stationarity assumptions on the relevance score distributions of users for their task of focused data capture (i.e., relevance scores of each user account are assumed to be distributed around a given stationary mean along the whole process), we claim that this is a few realistic setting in the case of large social networks and that it is possible to better predict future relevance scores of users according to their current activity. In other words, with the activity of a user $a \in \mathcal{K}$ at iteration $t-1$ represented by a d-dimensional real vector $z_{a,t}$, there exists a function $h : \mathbb{R}^d \rightarrow \mathbb{R}$ that explains from $z_{a,t}$ the relevance score $r_{a,t}$ that a user $a$ would obtain if it was followed during iteration $t$. This correlation function needs to be learned conjointly with the iterative selection process.

However, in our case, maximizing the relevance scores as defined in formula 1 is constrained in some different ways:

- Relevance scores $r_{a,t}$ are only defined for followed accounts during iteration $t$ (i.e. for every $a \in \mathcal{K}_t$), others are unknown;
- Context vectors $z_{a,t}$ are only observed for a subset of users $\mathcal{O}_t$ (i.e., it is not possible to observe the whole activity of the social network).

Constraints on context vectors are due to API restrictions. Two streaming APIs of Twitter are used to collect data:

- On the one hand, a *Sample* streaming API furnishes real-time access to 1% of all public tweets. We leverage this API to discover unknown users and to get an important amount of contexts vectors for some active users;
- On the other hand, a *Follow* streaming API provides real-time data published by a subset of the whole set of Twitter users. This API allows the system to specify a maximal number of 5000 users to follow. We leverage this API to capture focused data from selected users.

For users in $\mathcal{O}_t$, their relevance score at next iteration can be estimated via the correlation function $h$, which allows to capture variations of users' usefulness. For others however, we propose to consider the stationary case, by assuming some general tendency on users' usefulness. Therefore,

[2]Note that on some social media such as Twitter, a limitation of 1% of the total amount of produced messages on the network is set. Considering streams centered on popular keywords usually leads to exceed this limitation and then, to ignore some important messages.

we get an hybrid problem, where observed contexts can be used to explain variations from estimated utility means.

### 3.3 System Description

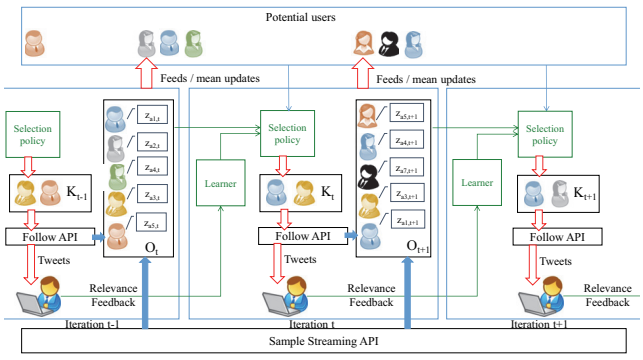


Figure 1: System illustration.

Figure 1 depicts our system of focused data capture from social media. The process follows the same steps at each iteration, three time steps are represented on the figure. At the beginning of each iteration, the selection policy selects a set of users to follow ($\mathcal{K}_t$) among the pool of known users $\mathcal{K}$, according to observations and some knowledge provided by a learner module. Then, the messages posted by the $k$ selected users are collected via the *Follow* streaming API. As we can see in the central part of the illustration, after having followed users in $\mathcal{K}_t$ during the current iteration $t$, the collected messages are analyzed (by a human or an automatic classifier) to give feedbacks (i.e., relevance scores) to the learner.

Meanwhile, as a parallel task during each iteration, current activity features are captured from the *Sample* streaming API. This allows us to feed the pool of potential users to follow $\mathcal{K}$ and to build the set $\mathcal{O}_t$ of users with observed contexts. Context vectors for iteration $t$ are built by considering all messages that the system collected from the *Follow* streaming and the *Sample* streaming APIs for each user during this iteration. Therefore, every user whose at least one message was included in these collected messages at iteration $t$ is included in $\mathcal{O}_{t+1}$ (messages from a same author are concatenated to form $z_{a,t+1}$, see section 5 for an example of context construction from messages). Contexts vectors collected during iteration $t$ serve as input for the selection strategy at iteration $t+1$.

## 4 Model and Algorithm

This section first introduces some settings and backgrounds for our contextual bandit approach for targeted data capture, details the policy proposed to efficiently select useful users to follow, and then describes the general algorithm of our system.

### 4.1 Problem setting

As defined above, we denote by $\mathcal{K}$ the set of $K$ known users that can be follow at each iteration of the data capture process. At each round $t \in \{1, .., T\}$ of the process, feature vectors $z_{a,t} \in \mathbb{R}^d$ can be associated with users $a \in \mathcal{K}$. The set of $k$ users selected to be followed during iteration $t$ is denoted by $\mathcal{K}_t$. The reward obtained by following a given user $a$ during iteration $t$ of the process is denoted by $r_{a,t}$. Note that only rewards $r_{a,t}$ from users $a \in \mathcal{K}_t$ are known, others cannot be observed.

In the field of contextual bandit problems, the usual linear hypothesis assumes the existence of an unknown vector $\beta \in \mathbb{R}^d$ such that $\forall t \in \{1, .., T\}, \forall a \in \mathcal{K} : \mathbb{E}[r_{a,t}|z_{a,t}] = z_{a,t}^T \beta$ (see (Agrawal and Goyal 2012b)). Here, in order to model the intrinsic quality of every available action (where an action corresponds in our case to the selection of a user to follow during the current iteration), we also suppose the existence of a bias term $\theta_a \in \mathbb{R}$ for all actions $a$, such that finally:

$$\forall t \in \{1, .., T\}, \forall a \in \mathcal{K} : \mathbb{E}[r_{a,t}|z_{a,t}] = z_{a,t}^T \beta + \theta_a \quad (2)$$

This formulation corresponds to a particular case of the *LinUCB* algorithm with the hybrid linear reward proposed in (Li et al. 2010) by taking a constant arm specific feature equal to 1 at every round. Note that individual parameters could have also been considered in our case but this is not well suited for the types of problems of our concern in this paper, where the number of available actions at each round is usually high (which would imply a difficult learning of individual parameters). For the sake of simplicity, we therefore restrict this individual modeling to this bias term.

In our case, a major difference with existing works on contextual bandit problems is that every context is not accessible to the selection policy before it chooses users to follow. Here, we rather assume that every user $a$ from the social media owns a probability $p_a$ ($0 < p_a < 1$) to reveal its context[3]. The set of users for which the process observes the features at time $t$ is denoted $\mathcal{O}_t$, whose complement is denoted $\bar{\mathcal{O}}_t$. Then, since $\mathcal{K}_t \subset \mathcal{K}$ and $\mathcal{K} = \mathcal{O}_t \cup \bar{\mathcal{O}}_t$, selected users can then belong to the set of users without observed context $\bar{\mathcal{O}}_t$.

Finally, as told above, while our approach would remain valid in a classical bandit setting where only one action is chosen at each step, our task requires to consider the case where multiple users can be selected to be followed at each iteration (since we wish to exploit the whole capture capacity allowed by the API). At each round, the agent has then to choose $k$ ($k < K$) users among the $K$ available ones, according to observed features and individual knowledge about them. The reward obtained after a period of capture then corresponds to the sum of individual rewards collected from the $k$ users followed during this period.

### 4.2 Distribution assumptions

To derive our algorithm, we first perform a *maximum a posteriori* estimate for the case where every context are available, based on the following assumptions:

- **Likelihood:** reward scores are identically and independently distributed w.r.t. observed contexts: $r_{a,t} \sim$

[3]The case $p_a = 1 \quad \forall a \in \mathcal{K}$ corresponds to the traditional contextual bandit.

$\mathcal{N}(z_{a,t}^T\beta + \theta_a, \lambda_a)$, with $\lambda_a$ the variance of the difference between the reward $r_{a,t}$ and the linear application $z_{a,t}^T\beta + \theta_a$;

- **Prior:** the unknown parameters are normally distributed: $\beta \sim \mathcal{N}(0, b\mathbb{I}_d)$ and $\theta_a \sim \mathcal{N}(0, \rho_a)$, where $b$ and $\rho_a$ are two values allowing to control the variance of the parameters and $\mathbb{I}_d$ is the identity matrix of size $d$ (the size of the features vectors).

For the sake of clarity, we set $b$, $\rho_a$ and $\lambda_a$ equal to 1 for all $a$ in the following. Note that every results can be extended to more complex cases.

**Proposition 1** *Denoting $T_a$ the set of steps when user $a$ has been chosen in the first $n$ time steps of the process ($T_a = \{t \leq n, a \in \mathcal{K}_t\}$, $|T_a| = \tau_a$), $c_a$ the vector containing rewards obtained by $a$ at iterations it has been followed ($c_a = (r_{a,t})_{t \in T_a}$) and $D_a$ the context matrix related to user $a$ ($D_a = (z_{a,t}^T)_{t \in T_a}$), the posterior distribution of the unknown parameters after $n$ time steps, when all contexts are available, follows:*

$$\beta \sim \mathcal{N}\left(\bar{\beta}, A_0^{-1}\right) \tag{3}$$

$$\theta_a + \bar{z}_a^T\beta \sim \mathcal{N}\left(\bar{\mu}_a, \frac{1}{\tau_a + 1}\right) \tag{4}$$

*With:*

$$A_0 = \mathbb{I}_d + \sum_{a=1}^{K}(\tau_a + 1)\bar{\Sigma}_a \qquad b_0^T = \sum_{a=1}^{K}(\tau_a + 1)\bar{\xi}_a$$

$$\bar{\Sigma}_a = \frac{D_a^T D_a}{\tau_a + 1} - \bar{z}_a\bar{z}_a^T \qquad \bar{\xi}_a = \frac{c_a^T D_a}{\tau_a + 1} - \bar{\mu}_a\bar{z}_a^T$$

$$\bar{\beta} = A_0^{-1}b_0^T \qquad \bar{\mu}_a = \frac{\sum_{t \in T_a} r_{a,t}}{\tau_a + 1} \qquad \bar{z}_a = \frac{\sum_{t \in T_a} z_{a,t}}{\tau_a + 1}$$

**Proof 1** *The full derivation is available at* [4]. *It proceeds as follows: denoting $D = ((r_{a_1,1}, z_{a_1,1}), ...(r_{a_n,n}, z_{a_n,n}))$ and using Bayes's rule we have:*

$p(\beta, \theta_1 ... \theta_K | D)$

$\propto p(D|\beta, \theta_1 ... \theta_K) p(\beta) \prod_{a=1}^{K} p(\theta_a)$

$\propto e^{-\frac{1}{2}\left(\sum_{t=1}^{n} \frac{(r_{a_t,t} - z_{a_t,t}^T(\beta - \theta_a)^2}{\lambda_a} + \frac{\beta^T\beta}{b} + \sum_{a=1}^{K}\frac{\theta_a^2}{\rho_a}\right)}$.

*Rearranging the terms and using matrix notations leads to:*

$p(\beta, \theta_1 ... \theta_K | D)$

$\propto e^{-\frac{1}{2}\left(\beta^T A_0 \beta - 2b_0^T\beta + \sum_{a=1}^{K}(\tau_a + 1)(\theta_a + \bar{z}_a^T\beta - \bar{\mu}_a)^2\right)}$.

*The two distributions are directly derived from this formula.*

All the parameters above do not have to be stored and can be updated efficiently as new learning example comes (see algorithm 1).

[4] http://www-connex.lip6.fr/~lampriers/ICWSM2016-supplementaryMaterial.pdf

**Theorem 1** *For any $0 < \delta < 1$ and $z_{a,t} \in \mathbb{R}^d$, denoting $\alpha = \sqrt{2}erf^{-1}(1-\delta)$* [5]*, for every action $a$ after $t$ iterations:*

$$P\Big(|\mathbb{E}[r_{a,t}|z_{a,t}] - \bar{\mu}_a - (z_{a,t} - \bar{z}_a)^T\bar{\beta}| \leq \alpha\,\sigma_a\Big) \geq (1-\delta)$$

$$\textit{with } \sigma_a = \sqrt{\frac{1}{\tau_a + 1} + (z_{a,t} - \bar{z}_a)^T A_0^{-1}(z_{a,t} - \bar{z}_a)}$$

(5)

**Proof 2** $\mathbb{E}[r_{a,t}|z_{a,t}] = (z_{a,t} - \bar{z}_a)^T\beta + \theta_a + \bar{z}_a^T\beta$. *By combining equations 2, 3 and 4:* $\mathbb{E}[r_{a,t}|z_{a,t}] \sim \mathcal{N}\left(\bar{\mu}_a + (z_{a,t} - \bar{z}_a)^T\bar{\beta}, \frac{1}{\tau_a+1} + (z_{a,t} - \bar{z}_a)^T A_0^{-1}(z_{a,t} - \bar{z}_a)\right)$. *The announced result comes from the confidence interval of a Gaussian.*

This formula is directly used to find the so-called *Upper Confidence Bound*, which leads to choose the $k$ users having the highest score values $s_{a,t}$ at round $t$, with:

$$s_{a,t} = \bar{\mu}_a + (z_{a,t} - \bar{z}_a)^T\bar{\beta} + \alpha\,\sigma_a \tag{6}$$

We recall that this formula can be used in the traditional contextual bandit problem, where every context is available. In the next section, we propose a method to adapt it to our specific setting of data capture, where most of contexts are usually hidden from the agent.

### 4.3 Dealing with Hidden Contexts

Selection scores derived in the previous section are defined for cases where all context vectors are available for the agent at each round. However, in our case, only users belonging to the subset $\mathcal{O}_t$ show their context. This particular setting requires to decide how judging users whose context is unknown. Moreover, it also implies questions about knowledge updates when some users are selected without having been (contextually meaning) observed.

Even though it is tempting to think that the common parameter $\beta$ and the user specific ones $\theta_a$ can be learned independently, in reality they are complexly correlated as formula 3 and 4 highlight it. It is important to emphasize that to keep probabilistic guarantees, parameters should only be updated when a chosen user was also observed, i.e. when it belongs to $\mathcal{K}_t \cap \mathcal{O}_t$. So, replacing the previous $T_a$ by $T_a^{both} = \{t \leq n, a \in \mathcal{K}_t \cap \mathcal{O}_t\}$ (we keep the notation $|T_a^{both}| = \tau_a$) allows us to re-use the updates formula of Proposition 1.

However, computing the selection score for a user whose context is hidden from the agent cannot be done via equation 6 since $z_{a,t}$ is unknown. To deal with such a case we propose to use an estimate of the mean distribution of feature vectors of each user. The assumption is that, while non-stationarity can be captured when contexts are available, different users own different mean reward distributions, and that it can be useful to identify globally useful users.

**New notations:**

- We denote the set of steps when user $a$ revealed its context vector after $n$ steps by $T_a^{obs} = \{t \leq n, a \in \mathcal{O}_t\}$ with $|T_a^{obs}| = n_a$.

[5] $erf^{-1}$ is the inverse error function, $erf(x) = 2/\pi \int_0^x e^{-t^2}\,\mathrm{d}x$.

- The empirical mean of the feature vector for user $a$ is denoted $\hat{z}_a$, with $\hat{z}_a = \frac{1}{n_a} \sum_{t \in T_a^{obs}} z_{a,t}$. Note that $\hat{z}_a$ is different from $\bar{z}_a$ since the first is updated every time the context $z_{a,t}$ is observed while the second is only updated when user $a$ is observed and played in the same iteration.

**Main assumptions:**

- Without loss of generality, assume that $\beta$ is bounded by $M \in \mathbb{R}^{+*}$, i.e. $||\beta|| \leq M$, where $||\ ||$ stands for the $L^2$ norm on $\mathbb{R}^d$.
- For any user $a$, at any time $t$ the context vectors $z_{a,t} \in \mathcal{R}^d$ are iid sampled from an unknown distribution with finite mean vector $\mathbb{E}[z_a]$ and covariance matrix $\Sigma_a$ (both unknown).

Note that the requirement $||\beta|| \leq M$ can be satisfied through proper rescaling on $\beta$.

**Proposition 2** *The excepted value of the mean reward of user $a$ with respect to its features distribution, denoted $\mathbb{E}[r_a]$ satisfies after $t$ iterations of the process:*

$$\begin{aligned}\mathbb{E}[r_a] &= \mathbb{E}[z_{a,t}^T \beta + \theta_a] \\ &= \mathbb{E}[z_a]^T \beta + \theta_a \\ &= (\hat{z_a} - \bar{z_a})^T \beta + \theta_a + \bar{z_a}^T \beta + (\mathbb{E}[z_a] - \hat{z_a})^T \beta\end{aligned} \tag{7}$$

**Theorem 2** *Given $0 < \delta < 1$ and $0 < \gamma < \frac{1}{2}$, denoting $\alpha = \sqrt{2} erf^{-1}(1-\delta)$, for every user $a$ after $t$ iterations of the process:*

$$P\left(|\mathbb{E}[r_a] - \bar{\mu}_a - (\hat{z_a} - \bar{z_a})^T \bar{\beta}| \leq \alpha\, \hat{\sigma}_a + \frac{1}{n_a^\gamma}\right) \geq (1-\delta)\left(1 - \frac{C_a}{n_a^{1-2\gamma}}\right)$$

$$\text{with } \hat{\sigma}_a = \sqrt{\frac{1}{\tau_a + 1} + (\hat{z_a} - \bar{z_a})^T A_0^{-1} (\hat{z_a} - \bar{z_a})} \tag{8}$$

*Where $C_a$ is a positive constant specific to each user.*

**Proof 3** *The full proof is available at URL[4], we only give the main steps here. Denoting $\hat{m}_a = (\hat{z_a} - \bar{z_a})^T \bar{\beta} + \bar{\mu}_a$, $\hat{\sigma}_a^2 = \frac{1}{\tau_a + 1} + (\hat{z_a} - \bar{z_a})^T A_0^{-1} (\hat{z_a} - \bar{z_a})$, $X_a$ the random variable such that $X_a = (\hat{z_a} - \bar{z_a})^T \beta + \theta_a + \bar{z_a}^T \beta$ and using Cauchy Schwarz inequality, we have: $\frac{|\mathbb{E}[r_a] - \hat{m}_a|}{\sigma_a} \leq \frac{|X_a - \hat{m}_a|}{\sigma_a} + \frac{||\mathbb{E}[z_a] - \bar{z_a}||M}{\sigma_a}$ Then, using the Gaussian property of $X_a$, we show that: $P\left(\frac{|X_a - \hat{m}_a|}{\sigma_a} \leq \alpha\right) = 1 - \delta$, with $\alpha = \sqrt{2} erf^{-1}(1-\delta)$. On the other hand, with Chebyshev inequality, we prove that:*

$P\left(\frac{||\mathbb{E}[z_a] - \bar{z_a}||M}{\hat{\sigma_a}} \leq \frac{1}{\hat{\sigma_a} n_a^\gamma}\right) \geq \left(1 - \frac{M^2 d}{n_a^{1-2\gamma}} Trace(\Sigma_a)\right)$

*Finally, combining the two previous results proves the theorem.*

For $\gamma < \frac{1}{2}$, the previous probability tends to $1 - \delta$ as the number of observations of user $a$ increases (as in equation 5). Then, the inequality above gives a reasonably tight UCB for the expected payoff of user a, from which a UCB-type user-selection strategy can be derived. At each trial $t$, if the context of user $a$ has not been observed (i.e. $a \notin \mathcal{O}_t$), set:

$$s_{a,t} = \bar{\mu}_a + (\hat{z_a} - \bar{z_a})^T \bar{\beta} + \alpha\, \hat{\sigma}_a + \frac{1}{n_a^\gamma} \tag{9}$$

Given that $\gamma > 0$, and that each user $a$ has a probability $0 < p_a < 1$ to reveal its context at each time step $t$, the extra-exploration term $1/n_a^\gamma$ tends to 0 as the number of observations of user $a$ gets larger. The above score then tends to a classical LinUCB score (as in equation 6) in which the context vector $z_{a,t}$ is replaced by its empirical mean $\hat{z_a}$, as $t$ increases.

## 4.4 Contextual Data Capture Algorithm

The pseudo-code of our contextual data capture algorithm from social streams is detailled in algorithm 1.

The algorithm starts by initializing the set of observed users $\mathcal{O}_t$ from the *Sample* API during a time period of $\mathcal{L}$. Then, for each iteration $t$ over $T$, it proceeds as follows:

1. Lines 10 to 15: every user from $\mathcal{O}_t$ that does not belong to the pool of potential users to follow is added to $\mathcal{K}$, in the limit of $newMax$ new arms at each iteration to cope with cases where too many new arms are discovered each iteration (in that case only the $newMax$ first new ones are considered, others are simply ignored). This allows to avoid over-exploration for such cases;
2. Lines 16 to 20: updates of empirical context means $\hat{z}_a$ (according to data collected from user $a$ at the previous iteration) and observation counts $n_a$ for every observed user $a$ in $\mathcal{O}_t$;
3. Line 22: parameters $\bar{\beta}$ of the estimation model are updated according to current counts;
4. Lines 23 to 37: selection scores are computed for every user in the pool $\mathcal{K}$. As detailed in the previous section, selection scores are computed differently depending on the availability of the current context of the corresponding user. Note also that the selection scores of new users added to the pool during the current iteration are set to $+\infty$ in order to enforce the system to choose them at least once and initialize their empirical mean reward;
5. Line 38: selection of the $k$ users with the best selection scores;
6. Lines 39 to 43: simultaneous capture of data from the *Sample* and *Follow* Apis during a time period of $\mathcal{L}$. The *Follow* API is focused on the subset of $k$ selected users $\mathcal{K}_t$;
7. Lines 44 to 55: computation of the rewards of users in $\mathcal{K}_t$ according to the data collected from the *Follow* API. Then, using users in $\mathcal{K}_t \cap \mathcal{O}_t$, both the global model parameters ($A_0$ and $b_0$) and arm specific ones ($\tau_a$, $\bar{\mu}_a$ and $\bar{z}_a$) are updated;
8. Line 57: Update of the set of observed users by gathering all users whose at least one post has been collected from both APIs during the current iteration.

**Algorithm 1**: Contextual Data Capture Algorithm

**Input**: $k, T, \alpha, \gamma, \mathcal{L}, maxNew$
$A_0 \leftarrow I_{d\times d}$ (identity matrix of dimension $d$);
$b_0 \leftarrow 0_d$ (zero vector of dimension $d$);
$\mathcal{K} \leftarrow \emptyset$;
$\Omega_1 \leftarrow$ Data collected from *Sample* API
    during a period of $\mathcal{L}$;
$\mathcal{O}_1 \leftarrow$ Authors of at least one post in $\Omega_1$;
**for** $t \leftarrow 1$ **to** $T$ **do**
    $nbNew \leftarrow 0$;
    **for** $a \in \mathcal{O}_t$ **do**
        **if** $a \notin \mathcal{K}$ ***and*** $nbNew < maxNew$ **then**
            $nbNew \leftarrow nbNew + 1$;
            $\mathcal{K} \leftarrow \mathcal{K} \cup \{a\}$;
            $\tau_a \leftarrow 0$; $n_a \leftarrow 0$; $s_a \leftarrow 0$; $b_a \leftarrow 0_d$;
            $\bar{\mu}_a \leftarrow 0$; $\bar{z}_a \leftarrow 0_d$; $\hat{z}_a \leftarrow 0_d$;
        **end**
        **if** $a \in \mathcal{K}$ **then**
            Observe context $z_{a,t}$ from $\Omega_t \cup \Psi_t$;
            Update $\hat{z}_a$ w.r.t. $z_{a,t}$;
            $n_a \leftarrow n_a + 1$;
        **end**
    **end**
    $\bar{\beta} \leftarrow A_0^{-1} b_0$;
    **for** $a \in \mathcal{K}$ **do**
        **if** $\tau_a > 0$ **then**
            **if** $a \in \mathcal{O}_t$ **then**
                $\sigma_a \leftarrow \sqrt{\frac{1}{\tau_a+1} + (z_{a,t} - \bar{z}_a)^T A_0^{-1} (z_{a,t} - \bar{z}_a)}$;
                $s_{a,t} \leftarrow \bar{\mu}_a + (z_{a,t} - \bar{z}_a)^T \bar{\beta} + \alpha\, \sigma_a$;
            **end**
            **else**
                $\hat{\sigma}_a \leftarrow \sqrt{\frac{1}{\tau_a+1} + (\hat{z_a} - \bar{z_a})^T A_0^{-1} (\hat{z_a} - \bar{z_a})}$;
                $s_{a,t} \leftarrow \bar{\mu}_a + (\hat{z_a} - \bar{z_a})^T \bar{\beta} + \alpha\, \hat{\sigma}_a + \frac{1}{n_a^\gamma}$;
            **end**
        **end**
        **else**
            $s_{a,t} \leftarrow \infty$;
        **end**
    **end**
    $\mathcal{K}_t \leftarrow \underset{\hat{\mathcal{K}} \subseteq \mathcal{K}, |\hat{\mathcal{K}}| = k}{\arg\max} \sum_{a \in \hat{\mathcal{K}}} s_{a,t}$ ;
    **during** a time interval of $\mathcal{L}$ **do**
        $\Omega_t \leftarrow$ Collect data from the *Sample* API;
        $\Psi_t \leftarrow$ Collect data from the *Follow* API
            focused on the users in $\mathcal{K}_t$ ;
    **end**
    **for** $a \in \mathcal{K}_t$ **do**
        Compute $r_{a,t}$ from $\Psi_t$;
        **if** $a \in \mathcal{O}_t$ **then**
            $A_0 \leftarrow A_0 + b_a b_a^T / (\tau_a + 1)$;
            $b_0 \leftarrow b_0 + b_a s_a / (\tau_a + 1)$;
            $\tau_a \leftarrow \tau_a + 1$;
            $s_a \leftarrow s_a + r_{a,t}$; $\bar{\mu}_a \leftarrow s_a / (\tau_a + 1)$;
            $b_a \leftarrow b_a + z_{a,t}$; $\bar{z}_a \leftarrow b_a / (\tau_a + 1)$;
            $A_0 \leftarrow A_0 + z_{a,t} z_{a,t}^T - b_a b_a^T / (\tau_a + 1)$;
            $b_0 \leftarrow b_0 + r_{a,t} z_{a,t} - b_a s_a / (\tau_a + 1)$
        **end**
    **end**
    $\Omega_{t+1} \leftarrow \Omega_t$; $\Psi_{t+1} \leftarrow \Psi_t$ ;
    $\mathcal{O}_{t+1} \leftarrow$ Authors of at least one post in $\Omega_{t+1} \cup \Psi_{t+1}$;
**end**

# 5 Experiments

Classical bandit algorithms usually come with an upper bound of the regret, which corresponds to the difference between the cumulative reward obtained with an optimal policy and the bandit policy in question. In our case, an optimal strategy corresponds to a policy for which the agent would have a perfect knowledge of the parameters $\beta$ and $\theta_a$ (for all $a$) plus an access to all the features at each step. However, in our case, in the absence of an asymptotically exact estimate of $z_{a,t}^T \beta + \theta_a$, due to the fact that some contexts $z_{a,t}$ are hidden from the process, it is unfortunately not possible to show a sub-linear upper-bound of the regret. Nevertheless, we claim that the algorithm we proposed to solve our problem of contextual data capture from social media streams well behaves on average. To demonstrate this good behavior, we present various offline and online experiments in real-world scenarios. Our algorithm is the first to tackle the case of bandits with partially hidden contexts, the aim is to demonstrate its feasibility for constrained tasks such as data capture from social media streams.

## 5.1 Experimental Setup

Beyond a random policy that randomly selects the set of users $\mathcal{K}_t$ at each iteration, we compare our algorithm to two bandit policies well fitted for being applied for the task of data capture: *CUCB* and *CUCBV* respectively proposed in (Qin, Chen, and Zhu 2014) and (Gisselbrecht et al. 2015). These algorithms do not take features into account and perform stationary assumptions on reward distributions of users. Compared to *CUCB*, *CUCBV* adds the variance of reward distributions of users in the definition of the confidence intervals. This has been shown to behave well in cases such as our task of data capture, where a great variability can be observed due to the possible inactivity of users when they are selected by the process. At last, we consider a naive version of our contextual algorithm that preferably selects users with an observed context (it only considers users in $\mathcal{O}_t$ when this set contains enough users), rather than getting the ability of choosing users in $\mathcal{O}_t$ as it is the case in our proposal.

For all of the reported experiments we set: 1) The exploration parameter to $\alpha = 1.96$, which corresponds to a 95 percent confidence interval on the estimate of the expected reward when contexts are observed; 2) The reduction parameter of confidence interval for unknown contexts to $\gamma = 0.25$, which allows a good trade-off between the confidence interval reduction rate and the probability of this interval; 3) The number of new users that can be added to the pool at each iteration to $newMax = 500$, to avoid over-exploration, especially for online experiments, when many new users can be discovered from the *Sample* API;

## 5.2 Offline experiments

**Datasets** In order to be able to test different policies and simulate a real time decision process several times, we first propose a set of experiments on offline datasets:

- *USElections*: dataset containing a total of 2148651 messages produced by 5000 users during the ten days preceding the US presidential elections in 2012. The 5000 cho-

sen accounts are the first ones who used either "Obama", "Romney" or "#USElections".

- *Libya*: dataset containing 1211475 messages from 17341 users. It is the result of a three-months capture from the $Follow$ API using the keyword "Libya".

In the next paragraph, we describe both how we transform messages in feature vectors and which reward function we use to evaluate the quality of a message.

**Model definition**

**Context model** The content obtained by capturing data from a given user $a$ at time step $t$ is denoted $\omega_{a,t}$ (if we get several messages for a given author, these messages are concatenated). Given a dictionary of size $m$, messages can be represented as $m$ dimensional bag of word vectors. However the size $m$ might cause the algorithm to be computationally inefficient since it requires the inversion of a matrix of size $m$ at each iteration. In order to reduce the dimension of those features, we used a *Latent Dirichlet Allocation* method (Blei, Ng, and Jordan 2003), which aims at modeling each message as a mixture of topics. However, due to the short size of messages, that standard LDA may not work well on Twitter (Weng et al. 2010). To overcome this difficulty, we choose the approach proposed in (Hong and Davison 2010), which aggregates tweets of a same user in one document. We choose a number of $d = 30$ topics and learn the model on the whole corpus. Then, if we denote by $F : \mathbb{R}^m \longrightarrow \mathbb{R}^d$ the function that, given a message returns its representation in the topic space, the features of user $a$ at time $t$ is $z_{a,t} = F(\omega_{a,t-1})$.

**Reward model** We trained a SVM topic classifier on the *20 Newsgroups* dataset in order to rate each content. For our experiments, we focus on 4 classes to test: *politics*, *religion*, *sport*, *science*. We propose to consider the reward associated to some content as the number of times it has been retweeted (re-posted on Tweeter) by other users if it belongs to the specified class according to our classifier, or 0 otherwise. Finally, if a user posted several messages during an iteration, his reward $r_{a,t}$ corresponds to the sum of the individual rewards obtained by the messages he posted during iteration $t$. This instance of reward function corresponds to a task of seeking to collect messages, related to some desired topic, that will have a strong impact on the network (note that we cannot directly get high-degree nodes in the user graph due to API restrictions).

**Experimental Settings** In order to obtain generalizable results, we test different values for parameters $p$, the probability for every context to be observed, $k$, the number of users that can be followed simultaneously, and $\mathcal{L}$, which stands for the duration of an iteration. More precisely, we experienced every possible combination with $p \in \{0.1, 0, 5, 1.0\}$, $k \in \{50, 100, 150\}$ and $\mathcal{L} \in \{2min, 3min, 6min, 10min\}$.

**Results** For space reasons and brevity, we only report results for $k = 100$, $\mathcal{L} = 2min$ (which means that every 2 minutes the algorithms select 100 users to follow) for the *USElections* dataset and the four rewards defined above. For the *Libya* dataset, we only show the results for the *politics* reward. Similar tendencies were observed for different values of $k$ and $\mathcal{L}$.

Figures 2 and 3 represent the evolution of cumulative reward for different policies and rewards, respectively for the *USElections* and the *Libya* dataset. For contextual algorithms, their naive versions that select observed users in priority are given with same symbols without solid line.

First, it should be noticed that every policy performs better than the *Random* one, which is a first element to assert the relevance of bandit algorithms for the task in concern. Second, we notice that *CUCBV* performs better than *CUCB*, which confirms the results obtained in (Gisselbrecht et al. 2015) on a similar task of focused data capture.

More interesting is the fact that when every context is observable, our contextual algorithm performs better than stationary approaches *CUCB* and *CUCBV*. This result shows that we are able to better anticipate which user is going to be the more relevant at the next time step, for a particular information need, given what he said right before. This also confirms the usual non-stationarity behavior of users. For instance, users can talk about science during the day at work while being more focused on sports when they come back home. Considering contexts also allows one to converge faster towards interesting users since every user account share the same $\beta$ parameter.

Results show that even for low probabilities of context observation $p$, our contextual policy behaves greatly better than non-contextual approaches, which empirically validates our approach: it is possible to leverage contextual information even if a large part of this information is hidden from the system. In particular, for the *Libya* dataset where no significant difference between the two non-contextual *CUCB* and *CUCBV* is noteworthy, our algorithm seems much more appropriate. Moreover, for a fixed probability $p$, the naive versions of the contextual algorithm offer lower performances than the original ones. By giving the ability of selecting users even if their context is unknown, we allow the algorithm to complete its selection by choosing users whose average context corresponds to a learned profile. If no user in $\mathcal{O}_t$ seems currently relevant, the algorithm can then rely on users with good averaged intrinsic quality. For $k = 100$, the average number of selected users for which the context was observed at each time step is 43 for $p = 0.1$ and 58 for $p = 0.5$, which confirms that the algorithm does not always select users in $\mathcal{O}_t$.

## 5.3 Online experiments

**Experimental Settings** We propose to consider an application of our approach on a real-world online Twitter scenario, which considers both the whole social network and the API constraints. For these experiments, we use the full potential of Twitter APIs, namely 1% of all public tweets for the *Sample* API that runs in background, and 5000 users simultaneously followed (i.e., $k = 5000$) for the *Follow* API.

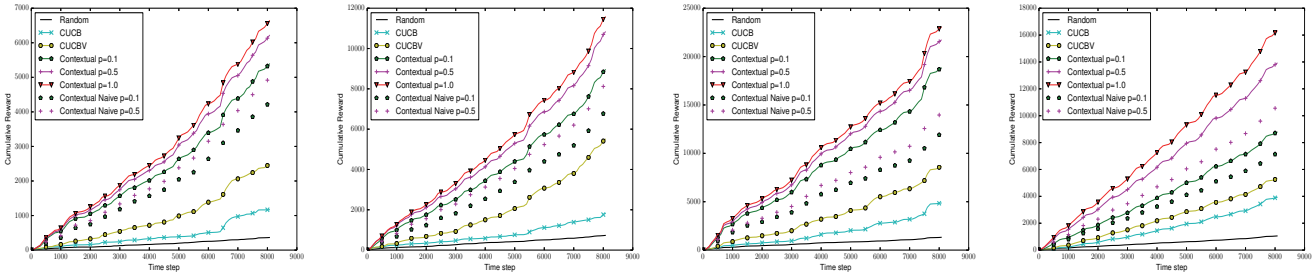


Figure 2: Cumulative Reward w.r.t time step for different policies on the *USElections* dataset. From left to right, the considered topics for reward computations are respectively: *politics*, *religion*, *science* and *sport*.

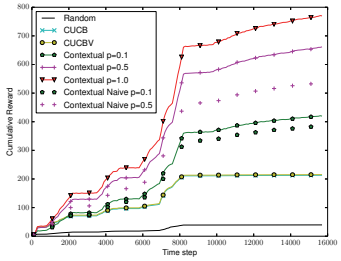


Figure 3: Cumulative Reward w.r.t time step for different policies on the *Libya* dataset and the *politics* reward.

At each iteration, the selected users are followed during $\mathcal{L} = 15minutes$. The messages posted by these users during this interval are evaluated with the same relevance function as in the offline experiments, using the *politics* topic. We also use an LDA transformation to compute the context vectors.

Given Twitter policy, every experiment requires one Twitter Developer account, which limits the number of strategies we can experiment. We chose to test the four following ones: our contextual approach, *CUCBV*, *Random* and a another one called *Static*. This *Static* policy follows the same 5000 accounts at each round of the process. Those 5000 accounts have been chosen by collecting all tweets provided during 24 hours by the *Sample* streaming API, evaluating them and finally taking the 5000 users with the greatest cumulative reward. For information, some famous accounts such that *@dailytelegraph*, *@Independent* or *@CNBC* were part of them.

**Results** Figure 4 on the left represents the evolution of the cumulative reward of a two-weeks-long run for the four tested policies. From these curves, we note the very good behavior of our algorithm on a real-world scenario, as the amount of rewards it accumulates grows greatly much more quickly than other policies, especially after the 500 first iterations of the process. After these first iterations, our algorithm appears to have acquired a good knowledge on the reward distributions, w.r.t. observed contexts and over the various users of the network.

In order to analyze the behavior of the experimented policies during the first iterations of the capture, we also plot on the right of the figure a zoomed version of the same curves on the first 150 time steps. At the beginning of the process, the *Static* policy performs better than every others, which can be explained by two reasons: 1) bandit policies need to select every user at least once in order to initialize their scores and 2) users who are part of the *Static* policy's pool are supposed to be relatively good targets considering the way we chose them. Around iteration 80, both *CUCBV* and our *Contextual* algorithm become better than *Static* policy, which corresponds to the moment they start to trade off between exploration and exploitation. Then, after a period of around 60 iterations (approximately between the $80^{th}$ and the $140^{th}$ time step) where *CUCBV* and our *Contextual* approach behave similarly, the latter then allows the capture process to collect greatly more valuable data for the specified need. This is explained by the fact that our *Contextual* algorithm requires a given amount of iterations to learn the correlation function between contexts and rewards and then taking some advantage of it. Note the significant changes in the slope of the cumulative reward curve of our *Contextual* algorithm, which highlight the ability of the algorithm to be reactive to the environment changes. Finally, the number of times every user has been selected by our *Contextual* algorithm is more spread than with a stationary algorithm such as *CUCBV*, which confirms the relevance of our dynamic approach. To conclude, in both offline and online settings, our approach proves very efficient at dynamically selecting user streams to follow for focused data capture tasks.

## 6 Conclusion

In this paper, we tackled the problem of capturing relevant data from social media streams delivered by specific users of the network. The goal is to dynamically select useful user-streams to follow at each iteration of the capture, with the aim of maximizing some reward function depending on the utility of the collected data w.r.t. the specified data need. We formalized this task as a specific instance of the contextual bandit problem, where instead of observing every feature vectors at each iteration of the process, each of them has a certain probability to be revealed to the learner. For solving this task, we proposed an adaptation of the popular con-

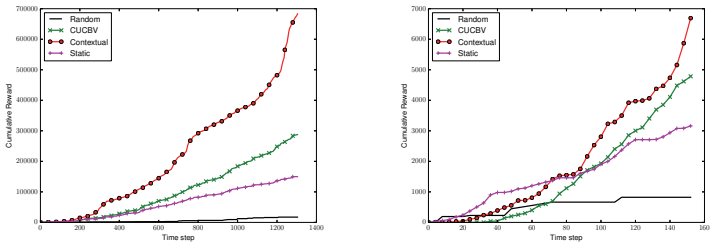


Figure 4: Cumulative Reward w.r.t time step for different policies on the live experiment. The figure represents the whole experiment on the left and a zoom on the 150 first steps on the right.

textual bandit algorithm $LinUCB$ to the case where some contexts are hidden at each iteration. Although not any sub-linear upper-bounds of the regret can be guaranteed, because of the great uncertainty induced by the hidden contexts, our algorithm behaves well even when the proportion of hidden contexts is high, thanks to some assumptions made on the hidden contexts and a well fitted exploration term. This corresponds to an hybrid stationary / non stationary bandit algorithm. Experiments show the very good performances of our proposal to collect relevant data under real-world capture scenarios. It opens the opportunity for defining new kinds of online intelligent strategies for collecting and tracking focused data from social media streams.

## 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

Agrawal, S., and Goyal, N. 2012a. Analysis of thompson sampling for the multi-armed bandit problem. In *COLT*.

Agrawal, S., and Goyal, N. 2012b. Thompson sampling for contextual bandits with linear payoffs. *CoRR*.

Audibert, J.-Y., and Bubeck, S. 2009. Minimax policies for adversarial and stochastic bandits. In *COLT*.

Audibert, J.-Y.; Munos, R.; and Szepesvari, C. 2007. Tuning bandit algorithms in stochastic environments. In *ALT*.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*

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

Chapelle, O., and Li, L. 2011. An empirical evaluation of thompson sampling. In *NIPS*. Curran Associates, Inc.

Chen, W.; Wang, Y.; and Yuan, Y. 2013. Combinatorial multi-armed bandit: General framework and applications. In *ICML*.

Chu, W.; Li, L.; Reyzin, L.; and Schapire, R. E. 2011. Contextual bandits with linear payoff functions. In *AISTATS*.

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

Fogaras, D.; Rácz, B.; Csalogány, K.; and Sarlós, T. 2005. Towards scaling fully personalized pageRank: algorithms, lower bounds, and experiments. *Internet Math.*

Gisselbrecht, T.; Denoyer, L.; Gallinari, P.; and Lamprier, S. 2015. Whichstreams: A dynamic approach for focused data capture from large social media. In *ICWSM*.

Gupta, P.; Goel, A.; Lin, J.; Sharma, A.; Wang, D.; and Zadeh, R. 2013. Wtf: The who to follow service at twitter. In *WWW*.

Hannon, J.; Bennett, M.; and Smyth, B. 2010. Recommending twitter users to follow using content and collaborative filtering approaches. In *RecSys*.

Hong, L., and Davison, B. D. 2010. Empirical study of topic modeling in twitter. In *ECIR*.

Kaufmann, E.; Cappe, O.; and Garivier, A. 2012. On bayesian upper confidence bounds for bandit problems. In *ICAIS*.

Kaufmann, E.; Korda, N.; and Munos, R. 2012. Thompson sampling: An asymptotically optimal finite-time analysis. In *ALT*.

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

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

Lai, T., and Robbins, H. 1985. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics* 6(1):4 – 22.

Li, L.; Chu, W.; Langford, J.; and Schapire, R. E. 2010. A contextual-bandit approach to personalized news article recommendation. In *WWW*.

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

Qin, L.; Chen, S.; and Zhu, X. 2014. Contextual combinatorial bandit and its application on diversified online recommendation. In *SIAM*.

Weng, J.; Lim, E.-P.; Jiang, J.; and He, Q. 2010. Twitterrank: Finding topic-sensitive influential twitterers. In *WSDM*.