Mining Hidden Constrained Streams in Practice: Informed Search in Dynamic Filter Spaces

Nikolaos Panagiotou*, Ioannis Katakis*, Dimitrios Gunopulos*,
Vana Kalogeraki†,Elizabeth Daly‡, Jia Yuan Yu§ and Brendan O Brien¶
National and Kapodistrian University of Athens*, Athens University of Economics and Business†,
IBM Research, Ireland‡, Concordia University§ and Dublin City Council, Ireland¶
Email: n.panagiotou@di.uoa.gr, katak@di.uoa.gr, dg@di.uoa.gr, vana@aueb.gr
elizabeth.daly@ibm.ie.com, jiayuan.yu@concordia.ca, brendan.obrien@dublincity.ie

***Abstract*—In this paper we tackle the recently proposed problem of *hidden streams*. In many situations, the data stream that we are interested in, is not directly accessible. Instead, part of the data can be accessed only through applying filters (e.g. keyword filtering). In fact this is the case of the most discussed social stream today, Twitter. The problem in this case is how to retrieve as many relevant documents as possible by applying the most appropriate set of filters to the original stream and, at the same time, respect a number of constrains (e.g. maximum number of filters that can be applied). In this work we introduce a search approach on a dynamic filter space. We utilize heterogeneous filters (not only keywords) making no assumptions about the attributes of the individual filters. We advance current research by considering realistically hard constraints based on real-world scenarios that require tracking of multiple dynamic topics. We demonstrate the effectiveness of our approaches on a set of topics of static and dynamic nature.**

The development of the approach was motivated by a real application. Our system is deployed in Dublin City's Traffic Management Center and allows the city officers to analyze large sources of heterogeneous data and identify events related to traffic as well as emergencies.

I. Introduction

Social media streams have been shown to provide a valuable source of real-time information and insights into the physical world in scenarios ranging from information dissemination and coordination during a flood [12], earthquake response [11], sport summarization [8] or traffic updates [1]. In most of these applications, the challenge of identifying valuable relevant content can be like finding the proverbial needle in a haystack. Research has shown that this problem is particularly difficult for short text messages like in Twitter, compared to longer documents [5]. An additional challenge is the highly dynamic nature of these streams. Adding to these many challenges, the complete information stream is not directly available for many social media architectures such as Twitter.

A motivational example. Let's assume our analysis task is the following:

> **Task 1.** *"Analyze the micro-blog stream of the city of Dublin and identify as many tweets as possible in real-time that talk about traffic issues in the road network".*

► @paraicgallagher There was a collision before J9, now cleared. Those are the delays.
► @LiveDrive the M50 collision has the N4 inbound backed up to hermitage golf club ...
► Stuck in traffic on the M50 and my plc interview is at half six ... I'm gonna cry
► My bus has taken 40 minutes to get from Harolds Cross Road to Aungier Street, I hate you ...

TABLE I: Traffic tweets coming from Dublin identified by our approach.

The output of our analysis should be a set of relevant tweets reported by the citizens or authorities of Dublin. An example of the desired output is displayed in Table I. In fact, these are tweets identified by our method as described in later Sections. This is a *real world* task inspired and funded by the European Research Project INSIGHT (www.insight-ict.eu) where managing traffic in emergency situations was a problem of major importance.

Our method is implemented in a system that is **deployed** (see Section VII) in Dublin City's Traffic Control room. Despite the fact that the motivation and the requirements were problem-specific, our solution is generic and can be utilized in similar cases.

This type of analysis necessitates the need for efficient algorithms to mine the data that is available through the Twitter API. However, the data stream that is accessible through the API is *constrained* in the following ways:

- The output is originating from the original Twitter stream *but* one has to apply a type of filtering on it (area based, keyword based, user based, etc) in order to access the data. By applying different filters, you get different sub-streams.
- One can apply up to 400 keyword filters, up to 5000 user filters and up to 25 location filters.
- Data volume generated by the application of filters is bounded by a limit (currently close to 1% of the original stream).

The above constraints are applied to the typical and most-widely Twitter API usage. Full access of the stream is only possible through a commercial license . Similar constraints

IEEE/ACM ASONAM 2016, August 18-21, 2016, San Francisco, CA, USA
978-1-5090-2846-7/16/$31.00 ©2016 IEEE

exist in other social media. Examples include Instagram, Flickr and Reddit. Therefore the issue is not Twitter-specific.

Task 1 can be broken down into addressing the following two problems: (i) Given a twitter stream and a maximum set of filters that can be applied, our goal is to determine an appropriate set of filters, such that, the number of tweets that we find from an area of interest is maximized, (ii) train a machine learning classifier, using a trustworthy stream, to identify relevant tweets. In practice, often such trustworthy streams exist for many areas and therefore we utilize them.

For the classifier part, there are a lot of knowledge discovery challenges. Many of them have been addressed in the literature [3]. However, in this paper we focus on the first part of the problem: *to identify as many relevant tweets as possible using an appropriate set of filters and utilize the classifier in order to select high quality content*. We argue that feeding the classifier with a richer stream will eventually have positive impact on the number of true positives, precision, and recall without making any adjustments to the classifier.

The **contribution** of this work can be summarized in the following points:

- We present a new dynamic approach that analyzes and extracts potentially useful filters (keywords, users, etc). The method has the following advantages over the state of the art: (a) it requires no apriori knowledge of precision, recall or the overlap of the filters, (b) it takes advantage of heterogeneous filters, (c) it updates the filters dynamically adapting to new information, hence making it effective for drifting topics, (d) it is suitable for problems with tight constraints (e.g. low number of allowed filters).
- We evaluate our approach against the state-of-the-art on datasets that we make available. This will enable the repeatability of our experiments.
- We present two case studies of real world retrieval problems. In contrast to the state of the art, our approach solves both parts of the problem by: (a) enriching the data stream by applying an appropriate set of filters, (b) automatically building a classifier to identify which of the items of the stream are relevant.

II. Related Work

Information retrieved from Twitter requires the identification of domain specific, topic relevant content. A combination of queries must be constructed as part of the data collection process. Typical solutions include keyword or hashtag queries, following specific users or constructing a bounding box query for the geographic location of interest. In order to retrieve and monitor a topic of interest a combination of the about queries may be used however a large amount of unrelated content may also be returned. As a result, further filtering may also be employed to discard unrelated content. Each step in this process tends to be a hand crafted and customized solution in order to satisfy the specific application area.

The Hotstream system consists of pre-defined search queries such as #breakingnews that were used to retrieve Twitter messages, similar messages are then clustered in order to identify emerging news stories [9].

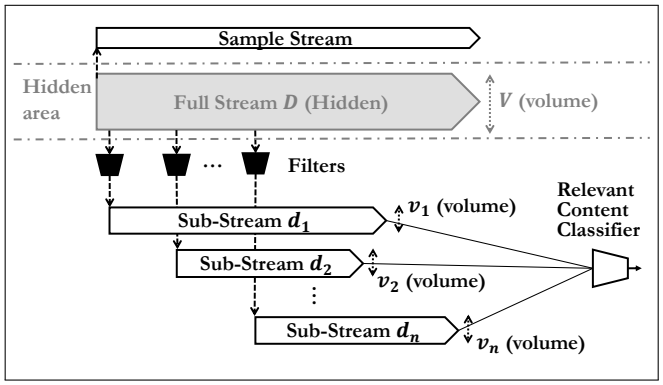

Fig. 1: Hidden Streams & Information Retrieval. The goal is to apply the proper set of filters to the Hidden Stream in order to build a set of high quality sub-streams that will contain relevant content.

Queries of specific focus may be used to first generate a dataset used for training a classifier which can then be used to filter the entire Twitter stream. Dan et. al. propose using keyword based queries in order to generate a small sample dataset which is then harnessed to create a reusable classifier in order to identify related social TV messages [2]. [6] uses hashtags as a label for specific topics. These labelled topic specific tweets are then used to create a language model which is then used to filter the entire Twitter stream.

To our knowledge Ruiz et. al. presented the first work tackling the problem of hidden streams [10]. They defined two variations, a static and a dynamic one. Having a set of keywords identified by a classifier, the goal is to select the best of them. These are keywords that lead to the maximum number of retrieved documents under a set of constraints. The problem is formulated as an optimization task and solved with Greedy and Dynamic programming algorithms.

The differentiation and contribution of our work is that we make no assumptions about the type of the filter (in [10], only keyword-based filters are considered). In [10], the precision and coverage (static case) of each keyword-filter are assumed to be known. Our approach doesn't require this information and dynamically estimates keywords' relevance. Finally, In [10], the keywords to be explored are considered given. In our approach, we *dynamically* extract relevant keywords as well as users or suitable bounding boxes from the stream.

III. Problem Definition

The information retrieval task in Hidden Constrained Document Streams is to collect as many relevant documents as possible by applying a set of filters to the original (hidden) stream and, at the same time, respect a set of constraints or limitations defined by the system architecture or by the data provider policy.

Definition 1: **Constrained Hidden Stream.** A constrained hidden stream D is a series of documents

$$D = \{\delta_1, \delta_2, \ldots, \delta_n\}$$

that has the following three properties.

P1. The originating stream is hidden.
P2. The number of filters that can be applied is limited.
P3. The aggregated volume of the resulting substreams is bounded.

In this section, we explain the above properties in detail. *Hidden Stream.* Documents δ_i in D can only be accessed through filters. The application of a filter f to stream D leads to a sub-stream d. In other words $f(D, P) = d_P \subset D$. Where P is the set of parameters that uniquely define the filter f. For example, a user-based filter can be defined as $f(D, U)$ where U is a set of users that the filter will track. The sub-stream d in this case will contain items generated by users in U. The $size$ of the filter is the size of its parameter set ($|P|$). Each substream d_i produces v_i data volume per time unit (see Figure 1).

Limited Number of Filters. The number of filters allowed is constrained by an integer Φ_{max}. In this work, we consider heterogeneous filters of different types $T_1, \ldots T_n$. In this context, the constraint can be defined as:

$$(C_{\text{fil}}): \sum_{i=1}^{n} |T_i| \leq \Phi_{max},$$

where $|T_i|$ is the size (number of parameters) of all filters of type i.

Bounded Data Volume. The sum of the returned documents from all filters can not be over Δ_{max} per window. Formally, given that n filters are applied and produce $d_1, \ldots, d_n$ sub-streams and each substream produces $v_1, \ldots, v_n$ data volume per time unit, then the constraint can be written as:

$$(C_{\text{vol}}): \sum_{i=1}^{n} v_i \leq \Delta_{max}$$

Figure 1 illustrates a hidden stream D, the application of n filters and the resulting substreams $d_1, ..., d_n$. The sample stream is a random sample of the original stream. Such functionally is available through the Twitter API.

Definition 2: **Information Retrieval in Hidden Constrained Document Streams.** Given a hidden constrained document stream D, and a topic of interest T, the task is to identify as many relevant documents to T in D by applying a set of filters $f_1, f_2, \ldots, f_n$ to D. The filters and their resulting substreams $s_1, s_2, \ldots, s_n$, should satisfy constraints C_{vol} and C_{fil}.

The *dynamic* version of the problem, where the filters in the set can change over time while their number remains fixed, is more important in a real setting since it requires incremental discovery of keywords related to topics that might drift over time. At the same time it is more difficult since it becomes essential to avoid overloading the selected substreams with popular keywords. To cope with these challenges we introduce a method that explores the filter space in an *on-line* fashion and, when available, utilizes trusted sources as stabilizers in order not to be carried away by the dynamics of the social network. Note that as a consequence we may have to monitor a given topic with only a small number of filters, given that the Twitter API imposes a hard constraint on the total number of filters.

IV. Dynamic Filters (DynHF)

The most common approach in retrieving relevant information from a hidden stream is to apply a single filter. In our task the most intuitive approach would be to apply a bounding box filter (BoxSinF) on the hidden stream and then feed the classifier with the resulted sub-stream. This approach, although straight forward, limits the amount of information fed into the classifier.

We build upon the common single filter approach by extending the information fed to the classifier by applying *multiple heterogeneous filters*. On top of the bounding box filter we utilized two additional streams. The first one, is a set of *user filters* collecting tweets of a specific user group. The second set of filters are keyword based.

We formulate our solution as *search in the dynamic filter space*. Our search criterion is the number of relevant documents collected in the latest time window. The novelty of the problem and the approach is that the space is *dynamic* since new filters (e.g. keywords, users) are extracted continuously from the stream while data are collected. Instead of solving a static optimisation problem in each step, our algorithm defines a heuristic local search in the space of possible filters. To do this we provide a formal framework for defining the filter space, and propose a search strategy in this space. A detailed description can be found in Algorithm 1.

States. A state is a particular set of filters with their parameters (Users, Keywords). Formally, $S_i = [f_{(1,i)}, \ldots, f_{(1,N)}]$, where $f_{(j,i)}$ is the filter of type j as it is set up at state S_i. What differentiates one state from another is the parameters of the filters. At Figure 2, we have two states in addition to the initial one. Each state is defined by the settings of its filters and an initial state S_0 is required. To move from S_0 to S_2 a set of different actions (operators) can be applied.

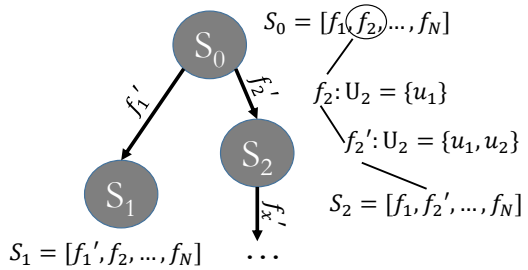

Fig. 2: A state is a set of filters along with their operators (users, keywords, locations). Moving through the dynamic filter space requires the addition or removal of an operator.

Trustworthy Stream. The trustworthy stream is a subset of the original stream that we know that it contains documents

Algorithm 1: DynHF - Search in a Filter Space

Input: Hidden Stream (D), initial state (S_0);
operator evaluation function $evop(item, data)$;
heuristic function $h(s)$, constraints Φ_{max}, Δ_{max}
Result: for each window a state S
Set $S = S_o = [f_1, f_2, \ldots, f_N]$;
Closed set of states $C = \emptyset$;
Open set of states (frontier) $A = [S_0]$;
Priority list of operators $O = \emptyset$
while *stream is active* **do**
  aggregate all d_i to D_w;
  if *reached* Φ_{max} *or* Δ_{max} **then**
    remove operators o_i using $evop(o_i, D_w)$
  end
  while *window* w **do**
    apply filter operators in $f_1, \ldots, f_N$ of S to D;
    collect substreams $d_1, \ldots, d_N$;
  end
  aggregate all d_i to D_w;
  evaluate $h(S, D_w)$;
  put tuple $< S, h(S, D_w) >$ to A (sorted);
  $O_w \leftarrow$ extract list of operators from D_w ;
  evaluate all o_i in O_w using $evop(o_i, D_w)$;
  Put tuples $< o_i, evop(o_i, D_w) >$ in O;
  Sort O according to $evop(o_i, D_w)$;
  $o_{next} \leftarrow$ top of O;
  `// move to` S_{next} `by applying` o_{next} `to` S
  $S \leftarrow o_{next}(S)$
end

related to the topic of interest. In practice, such a source always available; for example, in our experiments, we leverage such resources as traffic authorities' Twitter accounts that are used for public announcements. Taking advantage of such streams improves the quality and flexibility of our system. Note that the utilization of such a stream not only serves as a good starting point (see Initial State), but, more importantly, can be used to ameliorate the problem of focusing on only very popular topics. This is achieved by updating the relevant content classifier with data from the trustworthy stream. In any case, the approach is fully functional even without such a stream.

The Initial State. The initial state can be a randomly selected state or a carefully selected one depending on the domain knowledge available. In the case of Twitter for example, we can consider setting the filters to follow accounts or keywords that are relevant to the topic T or trustworthy accounts.

Extracting Action Operators. We consider as actions operators the potential change of filter settings from one state to another. This comes down to adding (or removing) users/keywords from the attributes of one of the filters. For example, in Figure 2, we observe that in order to move from state S_0 to S_2 a user (u_2) was added to the user-based filter U_2. In other words S_2 is based on S_0 but now f_2' has a changed parameter. The list of candidate operators must be dynamically discovered since keywords, users, etc, are not predefined. For each time step, we extract a new list of operators and we evaluate them using an evaluation function $evop$. Operators are maintained in an ordered list O. The evaluation function is problem-dependent. When a state S reaches the limits Δ_{max} or Φ_{max} the removal operation $rmop$ removes inappropriate operators according to $evop$ score.

Fitness Function. This is the necessary function h that will guide the algorithm through the search space evaluating the states. Again, this function is problem-dependent.

Dynamic Search Space. It is important to note that the filter search space is bounded by Φ_{max} and Δ_{max}. The algorithm cannot reach any state that contains more filters than the limit or create volume of data that exceeds the acceptable rate as shown in Figure 3. The search space is dynamic since new operators can be discovered (see the dotted lines in Figure 3).

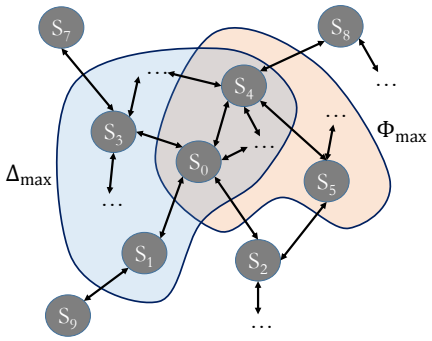

Fig. 3: The filter space. The coloured areas represent areas of the space that satisfy one of the two constraints (Δ_{max}, Φ_{max}). States in the intersection satisfy both constraints.

This consideration of the dynamic properties of the search space is not addressed by the state of the art. In [10], the authors considered an optimization approach to set up a *static* set of homogeneous (keyword) filters assuming that the following information is known a priori: a) the initial pool of keywords from which we have to select the optimal subset, b) the utility of each one of these words in terms of precision and overlap with other keywords, c) the volume that each word generates. The requirement of this information makes the approach not applicable to many cases where this type of knowledge is not available or very costly to obtain. In addition, we argue that these qualities are not static since the information value of users and keywords might change from time to time. In this context, the algorithm should constantly update and adapt to any changes. Our approach calculates the above filter qualities in a streaming fashion and discovers new filters from the stream.

V. Framework Instantiation

In this section we present the necessary implementation decisions required to put our framework into practice.

Relevant Content Classifier. We formulated the task of automatically identifying traffic related tweets as a data classification task. This required a set of *annotated* tweets (with labels 'traffic-related' or 'other'). Unfortunately, such a dataset is not publicly available. To overcome this issue and to avoid

manually annotating tweets, we trained the classifier using the tweets from the trustworthy stream assuming they are are related to traffic issues. This classifier is periodically updated using the trustworthy stream in order to capture *topic drifts*. The classifier uses as features the TF-IDF weighted term vectors of the tweets. The performance of the classifier utilizing an SVM in terms of F-Measure was 0.72 with precision / recall 0.63 and 0.84 respectively (based on a set of ground-truth relevant tweets - see Section VI).

Terminology Note: Throughout the text we use the term ***relevant*** tweets to indicate tweets that the classifier has identified as positive for the class 'traffic'. On the other hand we call ***ground-truth relevant*** tweets, the tweets that were manually annotated as 'traffic related' by human annotators. To evaluate the suggested approaches we use both relevant and ground-truth relevant tweets (see Section VI-A - Quality Assessment).

Initial State In our use case, the initial state was set to be the trustworthy stream. A random state could also be utilized with the downside of a slower learning rate. For our case, the trustworthy stream originates from authority Twitter accounts in Dublin that inform citizens on traffic issues. Such sources are LiveDrive, AARoadWatch and Garda Traffic. For the 'Greek Politics' use case (see Section VI-B) we utilized the keywords 'Greece', 'crisis', 'government'.

Operators. The operators are the tools that enable moving between states. For example, the operator "add user u_x" would add a user to a filter. One of the main advantages of the dynamic approach is that operators are extracted *incrementally* as the system receives new information from the stream.

Extracting Operators. Since we integrate heterogeneous filters to our framework, we need to define a different extraction approach for each type of filter. For *keyword* filters, the TextRank [7] algorithm is utilized. This algorithm creates a keyword co-occurrence graph over the stream's documents and extracts the most influential keywords. The keywords with the highest TextRank score are then evaluated according to the relevance of their substreams at the most recent window w. A keyword k applied to a hidden stream D as a filter will generate a substream d_k of tweets (containing this keyword). The relevance of these tweets will be evaluated based on the output of the traffic classifier (i.e. how many times the classifier considered tweets in this substream as relevant). More specifically, keyword relevance is defined as $\frac{|P_{w,k}|}{|T_{w,k}|}$ where $P_{w,k}$ is the set of tweets classified as traffic related by the classifier during window w and $T_{w,k}$ is the number tweets containing the keyword k. *User Extraction* can be rather simple. New candidate users are the ones that tweeted during the latest window.

Evaluating Operators. A *ranked* list of operators discovered so far is maintained according to their evaluation scores (see Algorithm 1). Evaluation of candidate operators (users, keywords, etc) is vital for two parts of the approach: a) prioritizing which operators will be tried next, b) removing operators when one of the two limits (Φ_{max}, Δ_{max}) has been reached. In the first case, the top-ranked operator is selected whereas in the second, the lowest ranked operator is removed. The *keyword* filter evaluator defined as

$$evop_k(k, D_w) = \alpha \frac{|P_{k,w}|}{|T_{k,w}|} + (1-\alpha)e^{-\gamma(t_c - t_k)}$$

It assigns a score to the sub-stream generated by a keyword filter and utilizes two parameters capturing *relevance* and *temporal value*.

The relevance factor is expressed as the ratio of the potentially relevant tweets $|P_{k,w}|$, over the total number of tweets $|T_{k,w}|$ containing the keyword k during the window w. The temporal activity factor considers the last appearance time t_k of keyword k and the current timestamp t_c. The above formula: a) penalizes keywords that appear only in the beginning of the window, and b) favors the ones that appear at the end of the window. Parameter γ was set empirically to 10^{-4} and α to 0.5. The *user* evaluator $evop_u(u, D_w) = |P_{u,w}|$ considers the number of potentially relevant tweets the user has posted during the window.

Fitness Function. An evaluation criterion (heuristic) is necessary to guide the dynamic search described in Algorithm 1. The evaluation function (h) of a state S during a window w is defined as $h(S, D_w) = \frac{|P_{S,D_w}|^\lambda}{|T_{S,w}|}$.. $P_{S,w}$ is the number of relevant tweets during window w using filters at state S. $T_{S,w}$ is the total number of tweets collected during window w. If the ratio $\frac{h(S_i, D_w)}{h(S_{i-1}, D_w)}$ is greater than 1 the system will extract operators from the current state S_i. If not, it will backtrack to the state with the highest fitness value h from the open set of states (see Algorithm 1). Similar to α, λ controls how much emphasis is given to the number of relevant tweets and the two can be considered as one parameter. Higher values of λ achieve higher Recall.

VI. Experimental Evaluation

The goal of the experimental evaluation is to investigate the utility of the dynamic exploration of our proposed approach (DynHF).

Competitor Methods. In the experiments, in addition to our proposed approach (DynHF), we include:

- DNOV-R / DNOV-K, are originally described in [10]. The assumption is that the keywords are given a priori and also that their precision is known. The algorithm utilizes up to Φ_{max} filters according to their precision trying not to surpass Δ_{max} rate limit. In our implementation, the required keywords were selected similarly to [10], using two days of data preceding the testing period. The difference between DNOV-R and DNOV-K is that the first targets precision whereas DNOV-K targets high recall.

A. Search in Dynamic Filter Spaces

DynHF aims to improve upon the two main disadvantages of DNOV-R and DNOV-K: a) the requirement for a priori knowledge of the quality of the filters, and b) the static filters and static precision assumptions in an extremely dynamic environment.

The keyword filter limit was set to $\Phi_{max} = 2, 5, 10$. These settings are intentionally selected in order to simulate real-world cases where organizations need to capture multiple topics. Hence, the allowed keywords per topic should be rather small. For our use cases the allowed Twitter data rate was more than enough hence, there was no point in studying the effect of Δ_{max}. Preliminary experimentation showed that a good value for DynHF's λ and α is 2 and 0.8 respectivelly. Location and user filter were not used in DynHF in order to fairly compare with DNOV-R and DNOV-K that only utilize keywords.

Dataset Construction. In order to collect a dataset for the experiments of this section we set-up crawlers that followed a set of Dublin users (obtained via the method suggested in [4]), keywords that appear in authority accounts related to traffic (e.g. LiveDrive) and a bounding box set up in Dublin. All these filters were applied for a period of approximately three months (01/09/15 to 22/11/15) and the collection resulted in a dataset of 1,105,748 tweets. We make this dataset available at: http://www.insight-ict.eu/hiddenstreams/

Quality Assessment. Since we are working on an un-labeled Twitter feed, we followed an unsupervised assessment strategy. The evaluation of each method is based on the quality of the sub-streams they generate that are forwarded to the content classifier (see Figure 1). If the approach provides a high-quality stream with lots of informative tweets, then the classifier will be able to identify them. Hence, we asses the quality of each approach by calculating the number of relevant tweets as identified by the classifier (see note in Section V). Eventually, some of our experiments are based on an annotated subsets, hence Precision, is calculated based on *ground truth* relevant tweets. For this section, in order to simplify the language, we use the term 'relevant tweets' to describe the tweets that were identified as positive by our classifier.

Results. Figure 4 presents the number of traffic relevant tweets retrieved, as identified by the classifier, under different Φ_{max} values for the three month period. DynHF performs better under all settings. When the allowed keywords are only a few, it is very important to select the best combination for that *particular time window*. Hence, DynHF is able to track in real time what topics are discussed and retrieve relevant content.

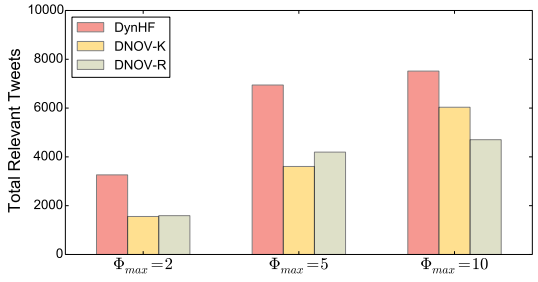

Fig. 4: Relevant Tweets Identified in total by the three methods for different values of Φ_{max}.

This is also demonstrated in Figure 5, where the number of identified tweets is presented with respect to time (with $\Phi_{max} = 10$ where DNOV-R and DNOV-K perform best). We observe that DynHF is consistently providing more relevant tweets. As shown in this figure all peaks of the DynHF line are related to an event that affected the traffic (floods, fire, etc). In some cases however (like the 'Truck Overturn' accident) generic keywords of DNOV-R and DNOV-K were sufficient to capture the event. We calculated that from all tweets identified by DynHF for $\Phi_{max} = 5$, 30% and 37% are novel tweets, i.e. tweets that were not identified by DNOV-R and DNOV-K respectively. The results are similar for other values of Φ_{max}.

The number of keywords discovered by DynHF with respect to time is presented in Figure 6. In a period of 20 days, our method explored more than 150 traffic related keywords which means that even in the context of this static topic (traffic) there were drifts in some sub-concepts that was important to be captured. In the next section, we explore a more dynamic topic, where the benefit of our approach is even more evident.

A Note on Precision. In order to evaluate the quality of the streams generated by the algorithms in terms of precision we utilized a smaller annotated dataset. We only include DNOV-R since DNOV-K targets recall. The annotation included 1600 tweets from 6am to 9pm during three days using $\Phi_{max} = 5$. The days were selected from Figure 5, where the following events occurred: i) Major Floods (11/09/15), ii) Port tunnel fire (18/09/15), iii) Truck overturn at port tunnel (20/11/15). Table II presents the results of this experiment. Stream Precision (StrPR) is the ratio of ground-truth relevant tweets to the total number of tweets returned by the filtering method. In the same Table we have included the number of ground truth relevant tweets (GTRel) found on the streams. What can be observed is that DynHF retrieves 2.3 times more tweets with a 10% drop in precision. Moreover, if higher precision is required, then the user can adjust parameters α and λ (see Section V). We also found that 63% of true positive tweets gathered from DynHF are novel to DNOV-R.

	DynHF			DNOV-R		
Date	StrPr	GTRel	Tot	StrPr	GTRel	Tot
11/09/15	0.68	280	412	0.82	111	135
18/09/15	0.63	268	425	0.7	112	160
20/11/15	0.62	208	335	0.78	104	133
Average	0.64			0.76		

TABLE II: Stream precision (StrPr) and ground truth relevant tweets (GTRel) during three days with major events.

B. Dynamic Topics

In order to demonstrate the potential of our approach we studied an additional retrieval task where the topic of interest is rather dynamic. The topic is 'Greek Politics' during a period that there were many developments (June-July, 2015, Referendum, Bail-out, etc).

We experimented using tweets written in English from June 20 to July 30, 2015. The dataset included 58 million tweets (221 GB). We observe that DynHF has an advantage even

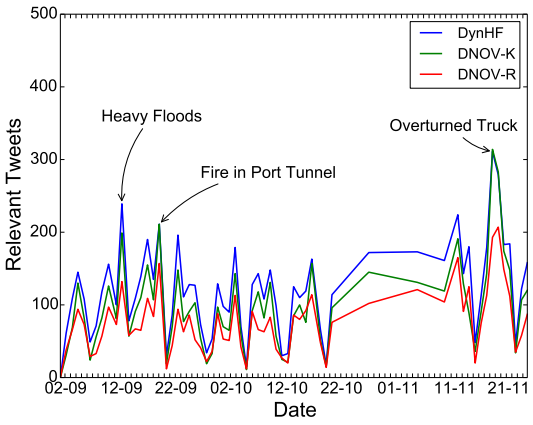

Fig. 5: Relevant tweets per day identified during the three months by all methods.

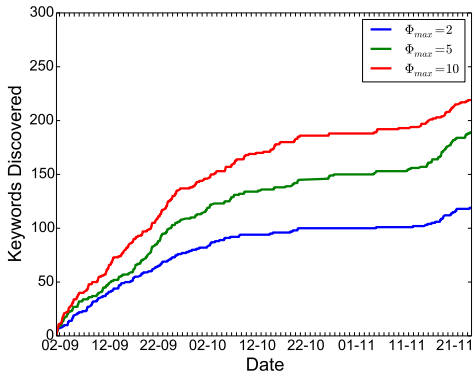

Fig. 6: Keywords discovered from DynHF during the period of the three months.

when only two keywords are allowed returning 154% more relevant tweets that DNOV-R. In cases of major events like the 'Referendum' or 'Payment Miss' DynHF outperforms the rest of the approaches by identifying and applying time-specific keyword-filters. DynHF in less than a month explored more than 100 keywords. This demonstrates again, the dynamic nature of the topic under study. The experimental run on this dataset required less than two hours on a four-core machine.

On Table III, we present some keywords discovered by DynHF on four non overlapping periods. Before the announcement of the referendum (Period 1 - until June 26) the keywords look rather generic while after the announcement (Period 2 - 27 June) the surname of the prime minister of Greece 'Tsipras' is added to the keywords list along with the keyword 'referendum'. Shortly after the Greece referendum (Period 3 - until July 5) keywords 'polls' and 'votes' also appear on the DynHF list due to the referendum outcome. Finally, in mid July (Period 4 - July 15) due to the crisis in Syria, we observe keywords like 'refugees' and 'migrant' capturing discussions about refugees trying to pass to Europe through Greece. We observe that these keywords are strongly related to unforeseen

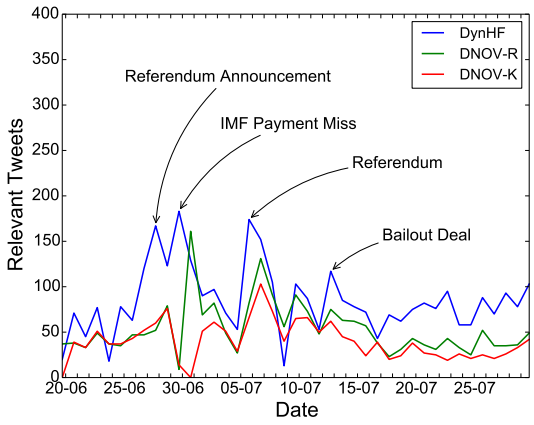

Fig. 7: Comparison of the three methods in terms of relevant tweets discovered with respect to time.

political developments that a static method could not track.

Period 1	Period 2	Period 3	Period 4
crisis	tsipras	tsipras	political
humanitarian	translifeline	referendum	eurozone
political	bailout	defiant	syriza
eurozone	referendum	polls	refugees
correctness	markets	votes	migrant

TABLE III: Examples of keywords discovered by DynHF on the Greece politics use case.

VII. SYSTEM DEPLOYMENT, END USER EVALUATION AND IMPACT

Deployment. The approach discussed in this paper (DynHF) is implemented and utilized in a system deployed in Dublin City's Council (DCC) traffic control room since June 2015. The system is called INSIGHT and it aids the personnel to monitor events and emergencies happening in Dublin (Figure 8). Although Dublin collects a lot of information the operators are not able to track all this data sources manually and identify issues. INSIGHT analyzes a set of heterogeneous sources like sensors measuring traffic volume in intersections, GPS information from Buses and social media data (Twitter). The system is deployed on an Intel i7, 3.4GHz, 8GB Ram.

The Twitter analysis component is responsible for analyzing the twitter stream and identifying tweets discussing traffic issues, emergencies, or floods (see Figure 9). On top of that, the user can investigate identified relevant tweets (historical and real time) and adjust the system's parameters.

Evaluation - Setup. We run a two-day full evaluation of the System at DCC. The process was set up in such way that we evaluate the system's features in terms of: a) functionality and b) usability. Simply put, functionality relates to how the system aids DCC employees achieving more than when using their conventional workflow. Usability relates to how easy to use and intuitive the system was. Note that the majority (95%) of participants were not familiar with the system. A Training Step

Fig. 8: Multiple Monitors at DCC's Traffic Control room. One of the monitors displaying the INSIGHT system automatically identifying events and emergencies.

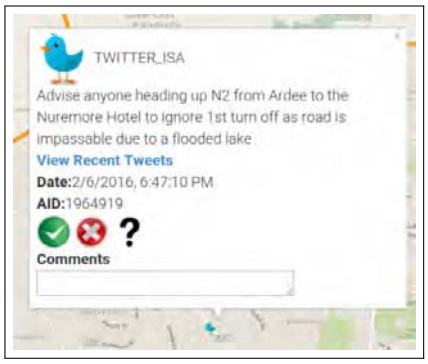

Fig. 9: Twitter Analysis in INSIGHT. The system analyzes tweets coming from the city, automatically inferring the location when necessary[1] and raises alerts on the map. DCC employees can investigate the event further or leave feedback.

preceded the evaluation by providing the participants a user guide and a set of video tutorials[1]. 27 employees participated in the evaluation and filled in a questionnaire.

Results. Based on their questionnaire answers DCC employees found the system and its Twitter analysis component very helpful. More specifically they found that social media analysis is extremely valuable in comparison to the analysis of the other sensors since: a) Citizens through their tweets publish information about the event in natural language and hence it is easier to comprehend, b) Citizens are actually 'moving social sensors' and are spread throughout the city covering areas where static sensors are not available.

On top of the above feedback we requested from the DCC team to examine independently Tweets list that the INSIGHT Twitter analysis identified as 'events' during the evaluation period (Traffic / Flood / Fire related). Results are shown in Table IV where we see the high accuracy of the system (91%).

VIII. Conclusions and Future Work

This paper deals with the problem of information retrieval in hidden constrained data streams. We formulate the problem as a search problem in a dynamic filter space. We compare our approach with a single filter baseline, a static alternative and a method recently proposed in the literature. Through experimental evaluation and two real-world use cases, we demonstrate that the dynamic nature of our approach (DynHF) is able to incrementally extract high quality filters and therefore generate sub-streams with relevant content. The method is implemented, deployed and evaluated in Dublin's Traffic Control Room. It currently aids operators to detect events and emergencies in the City by studying the identified tweets.

This is an exciting field and an unexplored area. We noted many items for our research agenda. For example, it would be interesting to conduct a more exhaustive study of various fitness functions as well as search strategies and apply the same principle in other domains.

[1]www.insight-ict.eu/hiddenstreams/

Tweets identified by DynHF	179
Confirmed Relevant Tweets	91%
Confirmed Relevant Tweets in Dublin	63%

TABLE IV: Accuracy of the Twitter analysis component

Acknowledgments

This research has been financed by the European Union through the FP7 ERC IDEAS 308019 NGHCS project, the Horizon2020 688380 VaVeL project and a Yahoo Faculty award.

References

[1] E. M. Daly, F. Lecue, and V. Bicer. Westland row why so slow?: fusing social media and linked data sources for understanding real-time traffic conditions. In *Proc of the 2013 int conf on Intelligent user interfaces*, pages 203–212. ACM, 2013.

[2] O. Dan, J. Feng, and B. D. Davison. A bootstrapping approach to identifying relevant tweets for social tv. In *ICWSM*, 2011.

[3] Y. Duan, F. Wei, M. Zhou, and H.-Y. Shum. Graph-based collective classification for tweets. In *Proc of the 21st ACM Int Conf on Information and Knowledge Management*, CIKM '12, pages 2323–2326, New York, NY, USA, 2012. ACM.

[4] D. Kotzias, T. Lappas, and D. Gunopulos. Home is where your friends are: Utilizing the social graph to locate twitter users in a city. *Inf Sys*, 2016.

[5] C. Li, J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee. Twiner: named entity recognition in targeted twitter stream. In *SIGIR*. ACM, 2012.

[6] J. Lin, R. Snow, and W. Morgan. Smoothing techniques for adaptive online language models: topic tracking in tweet streams. In *Proc of the 17th ACM SIGKDD int conf on Knowledge discovery and data mining*, pages 422–429. ACM, 2011.

[7] R. Mihalcea and P. Tarau. Textrank: Bringing order into texts. ACL, 2004.

[8] J. Nichols, J. Mahmud, and C. Drews. Summarizing sporting events using twitter. In *IUI*, IUI '12, New York, NY, USA, 2012. ACM.

[9] S. Phuvipadawat and T. Murata. Breaking news detection and tracking in twitter. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM*, volume 3, pages 120–123. IEEE, 2010.

[10] E. J. Ruiz, V. Hristidis, and P. G. Ipeirotis. Efficient filtering on hidden document streams. In E. Adar, P. Resnick, M. D. Choudhury, B. Hogan, and A. Oh, editors, *Proc of the Eighth Int Conf on Weblogs and Social Media, ICWSM 2014, Ann Arbor, Michigan, USA, June 1-4, 2014*. The AAAI Press, 2014.

[11] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proc of the 19th int conf on World wide web*, WWW '10, New York, NY, USA, 2010. ACM.

[12] K. Starbird, L. Palen, A. L. Hughes, and S. Vieweg. Chatter on the red: what hazards threat reveals about the social life of microblogged information. In *Proc of the 2010 ACM conf on Computer supported cooperative work*, CSCW '10, New York, NY, USA, 2010.