Action-Aware Restricted Stream Influence Maximization Model to Identify Social Influencers

Meznah Almutairy(✉), Hailah Alaskar, Latifah Alhumaid, and Rawan Alkhalifah

Imam Muhammad Ibn Saud Islamic University, Riyadh, Saudi Arabia
mrmutairy@imamu.edu.sa, {halaskar,lalhumaid,rkalkhalifah}@sm.imamu.edu.sa

Abstract. The problem of *influencer identification* is an important problem in social network analysis, due to the impact of influential users one the opinions of their audience. Most of the existing approaches to identify influencers are developed for static networks, whereas the social networks are time-sensitive and evolving over time. Therefore, identifying influencers over a *dynamic*, or *stream*, social network is more adequate for such problem. However, the amount of work proposed for dynamic networks are limited. Recent work proposed that identifying influencers with respect to some analysis-specific *restrictions* (e.g. influencers' locations or Influence context) produces a more concrete analyses. Current models proposed to identify influencers are based on capturing the number of social actions triggered by an influencer's social action. These models do not differentiate between social actions' types and treat them indistinguishably. However, the type of an action a user select to do captures an important clues in how as user is influenced. In this paper we propose to solve *Action-Aware Restricted Stream Influence Maximization* (AR-SIM) problem that identifies the most influential social network users in real-time. We extend the *Action-based* dynamic model [5] to incorporate actions' types into the model. The model does not only differentiate between the actions' types, it gives the option to weight these actions differently; facilitating new approaches to identify influencers. We run the model with respect to a given set of commonly used restrictions. We adopted a sliding window to update efficiently the model in real time. The model is generic and can be used with any social network platform, actions types, and restrictions. We run our experiments using Twitter data where we differentiate between four action types: (tweet, retweet, reply and quote tweet) and with respect to location, topic and/or language restrictions. Our results shows that our new model is able to identify significantly different influencers based on the given actions wights. This should open the gate for more sophisticate and deeper understanding for influencers impact types over the social network. The model is generic and can be used in any type of social network.

Keywords: Action-Aware · Stream Influence Maximization · Real-time analysis · Social networks

G. Meiselwitz (Ed.): HCII 2021, LNCS 12775, pp. 15–28, 2021.
https://doi.org/10.1007/978-3-030-77685-5_2

1 Introduction

Online social networks, or social networks for short, have evolved into a place where public opinion is shaped by the influence of some users over others. Thus, analyzing this relation of influence between users is of a growing importance on many levels, some of which are for governmental purposes, such as political matters, and others are commercial, such as marketing and targeted advertising. Influence Analysis over social networks is the basis of these applications. The first step to achieve this analysis is to identify influencers in a large amount of social relations [4]. Traditionally, social network analysis is conducted over a static network [8], where the set of users and their relations are fixed. However, in real applications, these relations have significant different changes over time, and thus old influencer may no longer have significant impact on public opinion. On the other hand, new influencers may take over the change of public opinion. Therefore, when identifying influencer, real-time analysis is more appropriate. Although, the real-time analysis is the right way to go, many challenges are encountered at this setting. This includes the large size of the data, the continuous change in the users and their relations, and the need to efficiency, update the analysis as respond to the new data arriving and the pruning for old data.

In the recent years, there has been interest in finding social influencers given some required restrictions [4,6]. For example, influencers may be defined with respect to some geographical locations. Alternatively, influencers may be defined with respect to some context or topics of interest. Therefore there have been some attempts to incorporate these restrictions when finding social influencers. However, not all the systems support real-time identification and obey given restrictions in an efficient matter. The ones that could support that are all commercial and their approach to find influencers are not clear raising the question about software output the validity. All models proposed to identify influencers are based on the counting the number of a social actions (e.g. as tweeting, retweeting, replays) triggered by an influencer's social action. These proposed models do not differentiate between social actions types and treat them indistinguishably as a generic action type. We believe the type of an action a user selects to do over a social network captures an important semantic in how the user is influenced. Also, we believe the type of an action is not only important but also should be weighted differently allowing for further influencers identification from different perspectives.

In this paper we propose to solve *Action-Aware Restricted Stream Influence Maximization* (AR-SIM) problem that identifies the most influential social network users in real-time. We extend the *Action-based* dynamic model [5] to incorporate actions' types into the model. The proposed model do not only differentiate between the actions' types but it also weights these actions differently, which significantly expand the possible options to find influencers from various perspectives according to its impact type over a social network. In our system, we run the model with respect to a given set of user combined restrictions. We adopted a sliding window to run update the model in real time. We run our experiments using Twitter data with respect to location, topic and/or language

restrictions. Our results shows that our new model is able to identify significantly different influencers based on the given actions wights and combined restrictions. This should open the gate for more sophisticate and deeper understanding for influencers impact over the social network. The model is generic and can be used in any type of social network and tradition models can be derived from our model by simple set the weights to be equal and ignore the action types. For purposes of concentrating and simplicity, we mainly conduct the analysis using Twitter data.

The paper is organized as follows. First we describe relevant work. Next, we propose *AR-SIM* problem explain in details the system architecture and implementation. Finally, experiment results are presented and discussed.

2 Related Work

Much effort has been put into the problem of identifying influencers in social networks. In this section, we summarize the most relevant literature and how they approached this problem.

A social network is generally represented as a graph data structure $G = \{V, E\}$ [4,5,7,8]. Depending on the underlying analysis, V and E may capture different features of a social network. For example, V could represent the set of users and E represents the set of the actions between those users [7], or vice versa [5,8]. Alternatively, V could represent the set of social actions and E represents the causality of those social actions [5,8]. In both cases, the graph could be traversed and/or statistics could be collected to conduct the intended analysis.

A social networks is classically classified into *static* or *dynamic*. Both of them are usually represented as graphs. In static network, the assumption is that the graph structure is fixed and then any analysis is built on top of this fixed structure. On the other hand, in dynamic network, the graph structure could change where new nodes and edges can be added and/or deleted.

2.1 Influence Maximization Problem

In a social network, Influence Maximization (IM) aims to find the set of social network users that maximize the influence value in the network. In the classical version of IM problem, *Static Influence Maximization*, most of the existing solutions depend on the influence probabilities between users to compute the influence value from the social actions. In this case the set of users with high probabilities to influence a largest group of users are identified as a solution to IM problem. Classical IM methods have one major limitation, they assume the social influence between users is static, that makes influence probabilities determined based on the historical user actions, which do not support the fast evolving and the highly dynamic of the social network. Thus, we believe this static network assumption is not valid when solving IM problem in real-time.

IM methods that account for the dynamics of the social network are named *Stream Influence Maximization* (SIM). These methods are efficient and effective to track influencers in real-time. SIM methods track the social influences

along current available social actions and obtain updated list of influencers continuously. To solve SIM problem, some work adopt the *Sliding Window* model, which captures the short-term memory of social influences where past influences are replaced with new influences with respect to the allowed time window [5,6]. More information in how handle social stream is presented in Sect. 2.3.

Most of IM solutions identifying the influencers for general purpose [5,7,8]. Recently there have been an interest on solving IM using the set of actions that are only related to a *restricted topic*, e.g. hashtag or keyword [3,4], or originated by users in a *restricted location*, e.g. tweets/retweets with locations [6]. We refer to this version of as *Restricted Influence Maximization* problem.

In our system we support the two main types restrictions: topic-restricted and location-restricted. We also consider language-restricted since it useful in some scenarios like find the trending topics for the local language only in a location without restrict by topic. We allow the user of our system to create any combination of these restrictions. This problem can naturally can be extended to a stream setting since the restriction are used to filter the input used to build dynamic model.

2.2 Diffusion Models

Diffusion Models are used to capture the spread of an idea or a piece of information throughout a network. They are also known as an influence propagation models. Originally there were two basic diffusion models, *Linear Threshold* (LT) and *Independent Cascade* (IC) [2]. Under LT model, a node switches to being active if sufficient number of its neighbors become active and thus is considered receiver-oriented (i.e. neighbors target a node). Unlike the LT model, the process of IC model is considered sender-oriented (i.e. node targets neighbors). In both models, the process also terminates when no activations are possible and again the influence spread is the number of expected active nodes at the end of the process.

Recently, other diffusion models have been proposed to support efficient influence spread computation in dynamic settings such as *Flow-based* [4] and *Action-based* model [5]. In Flow-based model, the nodes are the users and the edges are the direction flow of the information. Nodes can influence one another by tracking the information flow paths. Since actions are reflective of the diffusion process, Action-based model defines influence propagation simply based on actions performed by users.

When working with a static network, the first two models are more appropriate, because they are designed based on given probabilities for a user influence, this can be a problem on its own and has been studied by Goyal et al. [1]. In a stream setting, these probabilities are highly dynamic and change rapidly. Recomputing these probabilities is computationally expensive. Thus, the last two models are more appropriate as they are more efficient for a dynamic setting, and also support context-aware and location-aware queries.

2.3 Handling a Social Stream

There are some approaches that have been proposed to handle the social stream. Because diffusion models capture the spread of information throughout a network, the approach to handle the stream must be consistent with the diffusion model used. Below is a short summary for the approaches that alined with diffusion model described above in Sect. 2.2

Yang et al. [7] proposed *Polling-based* under LT\IC diffusion model. They consider a weighted graph where the nodes are users and the edges are actions performed by users (e.g. a user v has an edge directed to u if v retweeted from u). The edge weight represents the number of retweets between v and u under the LT model, or the probability that v retweets from u under the IC model. A social stream then can be represented as a series of updates on the edge weights.

Zhuang et al. [8] proposed *Probing-based* under IC diffusion model. They consider the graph where the nodes are users while the edges are the actions. They keep a summary of the stream by saving the topological updates on the graph (e.g. edge/node insertions/deletions).

Subbian et al. 2016a and 2016b [3,4] proposed *Tree-based* method under the Flow-based diffusion model. They propose to keep a summery of the stream by creating a tree data structure of flow paths with relevant weights that are computed according to the transmission of massages containing the keyword. The update this summery when a new contents arrives. Subbian et al. [3] recomputes the summary from scratch every time for each content arrives, making their approach computationally expensive. When the updated version by Subbian et al. 2016a [4], updates the summary incrementally by updating the flow paths weights only when a new social content gets transmitted, then add it to the flow path in the tree.

Wang et al. 2017 and Wang et al. 2018 [5,6] proposed by *Sliding Window* method under the Action-based diffusion model. They consider the graph where the nodes are the actions performed by the users, an edge exists between actions if one action triggers the other. They model the social stream as social actions which are generated by actor activities, a summary is created by keeping the most N recent actions in the sliding window. The Sliding Window is used in our system and is explain in detail in Sect. 3

3 Action-Aware Restricted Stream Influence Maximization Problem (AR-SIM)

In this section we start with illustration for the need for incorporating the action types when solving Influence Maximization Problem. Next we formalize the new problem and the model.

3.1 Motivation for the Action-Aware Influence

Recently, there has been an interest to find top influential users given a set of restrictions. The restrictions serve in identifying influencers for certain purposes

instead of the general case. We refer to this problem as Restricted Stream Influence Maximization Problem R-SIM. In our system we consider three types of restrictions keyword, language and location. In this work we solve an extended version of R-SIM problem that differentiate the types of actions in the data stream. We call this version *Action-Aware Restricted Stream Influence Maximization Problem* (AR-SIM).

The new version distinguishes between the actions types since these actions represent, for an influenced user, different feelings and needs, and thus should be captured differently. To describe this consider a Twitter social stream data, which has four major action types: *Tweet, Retweet, Quote and Reply.* The user may select one of the four types based on some reasons. A user may use Retweet action to spread a tweet to his/her audiences or show an agreement and support with a posted tweet. It is possible that a user pick to Reply to a tweet to be active, positively or negatively, on a tweet thread to present their friendship and loyalty or to gain followers from that account. On the other hand The Quote usually show an interest to discuss further the tweet. In all cases, there is large room for social behavioral specialist to understand deeper the behavior and the impact of influencer.

3.2 AR-SIM Model Formalization

Below we describe AR-SIM and the model formally along illustration examples. We extend the formalization use in [5] to fit out actions-aware model.

Action Stream: We consider a social action stream over a social network with a user set U. We *model* the social stream using unbounded *time-sequence social actions* that are generated by user activities. More formally, we represent an action of user $u \in U$ at time t who is influenced by an earlier action $a_{t'}$ at time t' ($t' < t$), and an action weight V that represented by a value from 0 to 1 based on the action type as $a_t = (u, a_{t'}, V)$. Then, a social action stream S is modeled by $S = <...., a_t = (u, a_{t'}, V), ..>$.

Figure 1 shows a social stream with action types where each symbols means: Retweet(R), Tweet(T), Quote(Q) and Reply(P). For example $a_1 = (u_1, nil, T)$ as a_1 an action that performed by user u_1 who has Tweet(T) action type with no previous action and $a_2 = (u_2, a_1, R)$ as a_2 an action that performed by user u_2 who has Retweet(R) action type with respond by previous action a_1. Then the social action stream S modeled as $S = <a_1 = (u_1, nil, T), a_2 = (u_2, a_1, R), ...>$.

Time Bounded Stream: To capture only recent stream actions we adopt *sliding window approach.* We consider N as a window size and W_t as sliding window at time t. Figure 1 shows how to adopt sliding window using previous social stream example. The window size N is set to be $N = 8$, and two examples of sliding windows on the stream are shown W_8 at time 8 and W_10 at time 10, highlighted in blue and red boxes respectively.

Weighted Influence Set: We define the influence set of a user as follows: The influence set of a user $u \in U$ at time t, denoted as $I_t(u)$, is the set of users who are

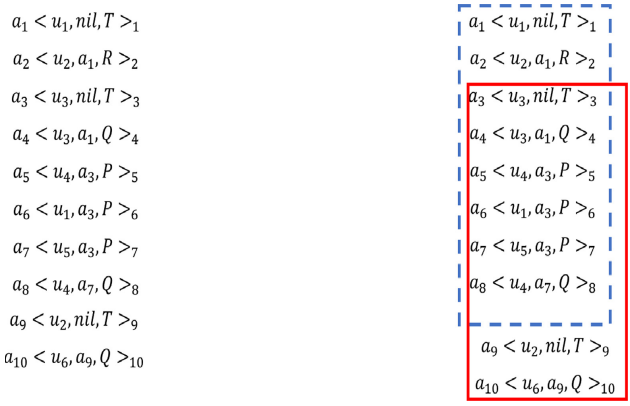

Fig. 1. An example of a social stream with distinguished action types with and without sliding windows (Color figure online)

influenced by u wrt. the sliding window at time t (i.e., W_t). Equivalently, $I_t(u) = \{v|(u,v)_t\}$. Traditional Influence set treats all actions equally. In our work, we augmented each action with a weight, and thus, we extend traditional Influence set to be weighted influence set. Action type weight are used to calculate the influence scores based on the action type values. The influence set contains a set of tuples, each tuple has user u and the highest action weight that user u has received wrt. the window at a time. In our experiments we the weight could be any real value from 0 to 1. When the weights are all set to 1, then we treat action equally and we get the tradition influence set.

Figures 2 and 3 show two influence sets at time 8 and 10 that display the difference between influence score for top_k (k = 1) users with action type and without action type. We assume the weights of the action types as the follow: 1, 0.75, 0.50 and 0.25 for Retweet, Quote, Tweet and Reply respectively. The tables shows the difference between the influence set values before and after considering the action weights. For Example, at W_8 $I_8(u_1)$ with type weight is equal to 2.25, on the other hand, the W_8 $I_8(u_1)$ without type weight is equal to 3, where each action is equal to 1. And at W_{10} $I_10(u_3)$ with type weight is equal to 1.25, where without the type weight is equal to 4. This difference will change significantly the result of the top influencers based on the action types.

Influence Value: Influence value measured by $f(I_t(S))$ where $I_t()$ is the "weighted" cardinality of the set of users at time t. In particular we take the sum of weighted cardinalities for all the influences sets of all users. More formally, for a given $k > 0$, to find the set of at most k users who collectively achieve the largest influence value S_t^{opt}. We compute the *influence value* for both single user and set of users to identify the most k influential users. For example the *influence value* for single user u_1 is $I_t(u_1) = \{u_1, u_2, u_3\} = 2.25$. As shown in Fig. 4, the *influence value* for set of users u_1 and u_3 is $I(u_1, u_3) = |I(u_1)| + |I(u_3)| = |\{u_1, u_2, u_3\}| + |\{u_1, u_3, u_4, u_5\}| = 3.50$. And the most k users when $k = 2$ is

User	$I_8(u)$	$I_8(u)$, without type weight	$I_8(u)$, with type weight
u_1	$\{(u_1,3), (u_2,5), (u_3,4)\}$	3	2.25
u_2	$\{(u_2,5)\}$	1	1
u_3	$\{(u_1,2), (u_3,3), (u_4,2), (u_5,2)\}$	4	1.25
u_4	$\{(u_4,2)\}$	1	0.25
u_5	$\{(u_4,2), (u_5,2)\}$	2	0.50
u_6	{}	0	0

Fig. 2. Examples of weighted influence sets at time 8

User	$I_{10}(u)$	$I_8(u)$, without type weight	$I_8(u)$, with type weight
u_1	$\{(u_1,3), (u_3,4)\}$	2	1.25
u_2	$\{(u_2,3), (u_6,4)\}$	2	1.25
u_3	$\{(u_1,2), (u_3,3), (u_4,2), (u_5,2)\}$	4	1.25
u_4	$\{(u_5,2)\}$	1	0.25
u_5	$\{(u_4,4), (u_5,2)\}$	2	1
u_6	$\{(u_6,4)\}$	1	0.75

Fig. 3. Examples of weighted influence sets at time 10

User	$I_8(u)$	$I_8(u)$, with type weight
u_1	$\{(u_1,3), (u_2,5), (u_3,4)\}$	2.25
u_2	$\{(u_2,5)\}$	1
u_3	$\{(u_1,2), (u_3,3), (u_4,2), (u_5,2)\}$	1.25
u_4	$\{(u_4,2)\}$	0.25
u_5	$\{(u_4,2), (u_5,2)\}$	0.50
u_6	{}	0

Fig. 4. An example of top k influence sets at time 8 where $k = 2$

$S_8^{opt} = \{u_1, u_3\}$, $f(I_8(S_8^{opt})) = 3.5$. To find the set of top k users, we simple sort the users based on their influence value and then take top k one to form the final list.

4 System Architecture

To identifying the most influential users in Twitter, the software passes through five main stages as shown in Fig. 5. The stages are listed in details below.

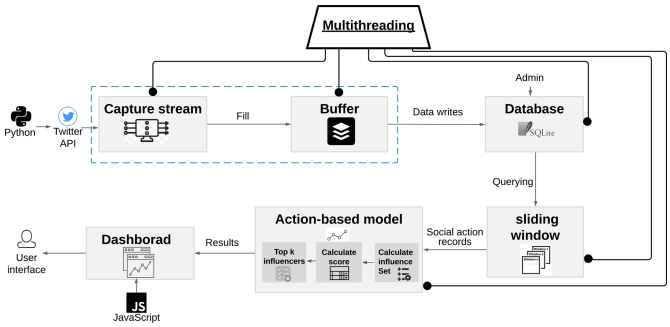

Fig. 5. System architecture.

Stage I - Capturing Data Stream: In this stage the system capture the stream by using Twitter API with Tweepy python library. The stream is filtered based a given restriction or/and list of restrictions (e.g. keyword or/and list of keywords) that is provided by the user. Next, the filtered stream is buffered continuously until either the maximum number of actions is buffered or a maximum time update time limit is exceeded. At this time stream information is flushed to the system database for later processing.

Stage II - Saving Data Stream: The system saves the data stream on SQLite database. The data in the database is organized into three parts of data (as shown in Fig. 6): the older data that used to build the model, the current data within a window, and, the new arriving data from the stream. To keep the size of the database as small as possible, the old data is usually removed when it is no longer has an impact on identifying influencers (illustrated in late stages). To identify the current influencers, the current data is used by model where the current data us defined by the set of data within a window of time. Finally, the new data is saved as we need to balance between handling the stream at different speed and the required computations to identify influencers.

Stage III - Recognizing Current Data: To maintain sufficiently low computational complexity, the system Identify influencers on only some reasonably recent data. The system create a sliding window that contains the most recent actions (possibly leaving out some very recent ones). The window is defined by two user-specified parameters: a window size N and an update interval M, where both N and M are units of time. The window size N controls how many data need to be considered to find influencers. The update interval M controls how frequent the window slides to a new position, leaving out old data points and creating space for new ones.

Stage IV -Identify Influencers: For every new shift in the window, the system updates the model as follows. It inserts the new actions to the model, removes the old ones from the model and their influence set, and compute the influence score.

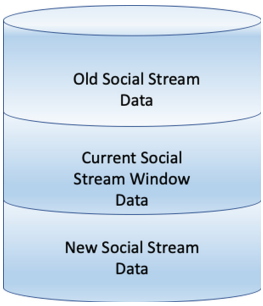

Fig. 6. Database conceptual orgnization.

Stage V - Visualize the Influence Results: The system displays a dashboard that measure the influence of identified influencers. A JavaScript library is used to implement a dashboard that contain a graph, chart, and diagrams representing the influencers relation with their *influencee* and influence scores.

Stage VI - Parallel Processing and Threading: Since the system is real time system, the system utilizes the parallel processing for efficiency. The system run each stage in the software in parallel with the other stages. There are two ways to implement parallelism: Multiprocessing and Multi-threading, the main difference is multi-threading enables sharing the memory between all threads, while with processes that's not the case.

5 Experiment Results and Discussions

As stated earlier, *AR-SIM* is an extended version of the Stream Influence Maximization problem *SIM*. Wang et al. [5], *SIM* developers, have defined an influencer as a user who performs an action that triggers many actions by other users. In our case, when the weights assigned are equal for all different actions, then our *AR-SIM* is identical to *SIM* model.

Later, Wang et al. 2018 [6] added a location restriction to firstly minimize the influencer results and hence gives a more targeted analysis based on the selected location, and secondly to handle the stream more efficiently. For our solution, we add more restrictions (keyword, location, and language) to the SIM problem. Further, we allow a user to create a complex restrictions from the basic ones using logical operators such as "and", "or" over more than on possible values for a restrictions. We also include other none traditional restrictions, such as a lower threshold on an influencer number of followers, and whether an account is verified.

We evaluate the effectiveness of our proposed solution *AR-SIM* with several real-world scenarios. We focus on two aspects: the impact of incorporate actions types into the model and the ability to handle complex restrictions.

5.1 The Impact of Actions' Types

First, we test the action types consideration and the difference results between identifying the influencers using our proposed model *AR-SIM* and the general *SIM* for the same dataset and the same restrictions. Figure 7 (a) shows the results of the general *SIM* model which all the actions have the same priority "action weight: 1". In contrast, Fig. 7 (b) shows the results of *AR-SIM* model, which considering the action weights as the following: Tweet: 0.25, Retweet: 1, Reply: 0.75, and Quote: 0.50. Note that our selection of the weights suggests, for example, that "Retweet" action shows a strong influence level while "Quote" action is about less influence by half compared to "Retweet". To understand the impact of *AR-SIM* model, we compare the discovered top four influencers from *SIM* model and *AR-SIM* while fixing the time interval, i.e., fixing the dataset. Next we study how the models are handling the stream in real time.

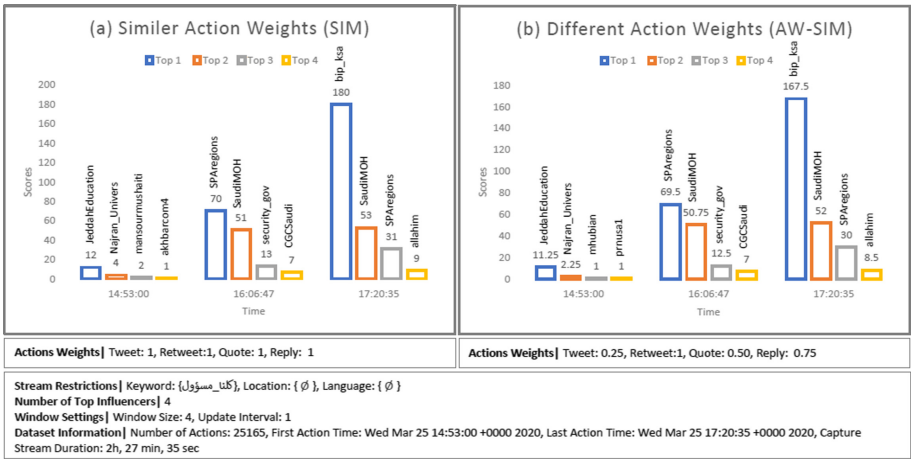

Fig. 7. The impact of action types experiment results.

When time interval is fixed, we can see that all the computed influence scores using *AR-SIM* model are less than the computed score using *SIM* for the same influencer in all three time intervals considered. This is expected since our selections for the actions weights are no more than 1. By investigating the first time interval, we can see clearly how *AR-SIM* changes the significant of an influence level. For example, the gap in the score between 2nd influencer and 3rd influencer has decreases from 2 to 1.25. This suggest that their influence is more closer in our model that simple model. In addition, the gap in the score between 3rd influencer and 4th influencer has decreases from 1 to 0, suggesting those influencer have similar influence level. In some cases it is possible that the gap is large to the point where the order of influencer is changed significantly and some may not be any more in the top list. The results confirm that differentiating

the types of social actions, by considering different weight, leads in quantifying and evaluating significantly the social influence. This suggesting that the list of discovered influencers could be significantly different.

For consecutive period of times, the influence scores changed causing influencers to appear or disappear. For example in *SIM* model Fig. 7 (a) the influencers (SaudiMOH) has in second period (B) score (51), and third period (C) has score (53). This suggest that three is new social actions related to SaudiMOH Tweet, causing it to be still a real time influencer in interval (C). On the other hand, the influencer (JeddahEducation) disappeared in the second period (B). This suggest that there is no more new social actions related to JeddahEducation's original tweet, and thus is no more an influencer. As in *SIM*, using *AR-SIM* model, Fig. 7 (b), theses scores also changes over time, but obviously using the computed scores by *AR-SIM* model. The results confirm that the model response effectively to the changes over time and report only active influences.

5.2 Implementing Restrictions

We test *AR-SIM* model/system using different stream restrictions (location, language, and keyword) and other parameters (See Fig. 8). These restrictions are aim to tune the influence analysis to a target needs, and to reduce computational cost. These restriction serve as stream filtering and appears to use the user as search parameters. Beside the mentioned restrictions, the search can be tuned further with user defined action weights, select only verified influencers accounts, set the number of top influencers, and the window settings (which defined the duration of real time analysis).

In the top-left figure, we use a single keyword, use different weights, pick verified account with at least 1000 followers, and set the window size is 4 where the updates is in every 1 min. The stream is captured for about 2 h and 27 min. We build the model and update it with a total of 25,165 tweets.

In the top-right figure, we use multi keywords, use equal weights, and the set window size is 8 where the updates is in every 1.5 min. The stream is captured for about 3 h and 17 min. We build the model and update it with a total of 7,465 tweets.

In the bottom-left figure, we use multi keywords, set the language to Arabic, use equal weights, and set the window size is 4 where the updates is in every 0.5 min. The stream is captured for about 8 h and 38 min. We build the model and update it with a total of 4,140 tweets.

In the bottom-right figure, we use multi keywords, set the langue to Arabic, set the location to "Riyadh", pick verified account with at least 2000 followers, use equal weights, and set the window size is 6 where the updates is in every 3 min. The stream is captured for about 8 min. We build the model and update it with a total of 9,732 tweets.

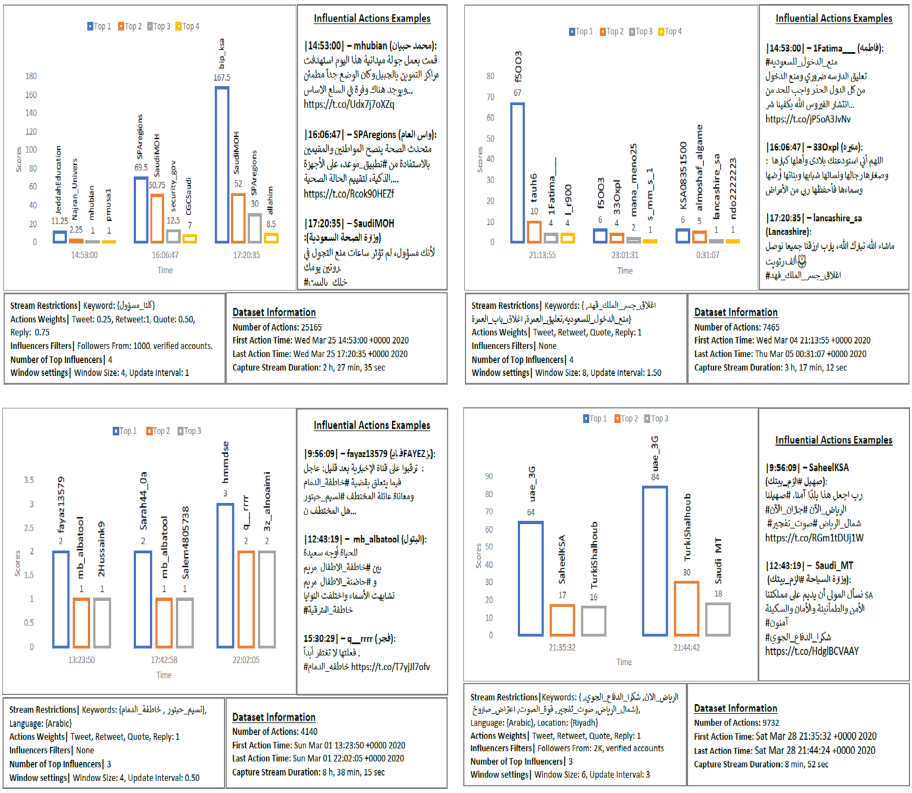

Fig. 8. Examples of *AR-SIM* model/system outputs using different stream restrictions and other user defined options.

6 Conclusion

In this paper, we extend and solve a new version of Stream Influence Maximization *SIM* problem, that is *Action-Aware Restricted-Stream Influence Maximization* problem. We extend the *Action-based* model to represent and weight different social actions. We adopted *Sliding window* model to a capture only the recent actions for real-time analysis. We designed and implemented a system that identifies the most influential users in real-time. The system allows the user to restrict the search using keyword, language and location.

It will be useful to extend this work to include model other than action-aware. For simplicity, we focused on solving the problem with handling sliding windows with a single action shift at a time. It is possible to extend the work with handling multiple action shifts at a time. We aim to explore better approaches to save and handle the stream. We focused only in Twitter data, however, a complete system should also consider other popular social network platforms.

References

1. Goyal, A., Bonchi, F., Lakshmanan, L.V.: Learning influence probabilities in social networks. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 241–250. ACM (2010)
2. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 137–146. ACM (2003)
3. Subbian, K., Aggarwal, C., Srivastava, J.: Mining influencers using information flows in social streams. ACM Trans. Knowl. Discov. Data (TKDD) **10**(3), 26 (2016)
4. Subbian, K., Aggarwal, C.C., Srivastava, J.: Querying and tracking influencers in social streams. In: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, pp. 493–502. ACM (2016)
5. Wang, Y., Fan, Q., Li, Y., Tan, K.L.: Real-time influence maximization on dynamic social streams. Proc. VLDB Endow. **10**(7), 805–816 (2017)
6. Wang, Y., Li, Y., Fan, J., Tan, K.L.: Location-aware influence maximization over dynamic social streams. ACM Trans. Inf. Syst. (TOIS) **36**(4), 43 (2018)
7. Yang, Y., Wang, Z., Pei, J., Chen, E.: Tracking influential individuals in dynamic networks. IEEE Trans. Knowl. Data Eng. **29**(11), 2615–2628 (2017)
8. Zhuang, H., Sun, Y., Tang, J., Zhang, J., Sun, X.: Influence maximization in dynamic social networks. In: 2013 IEEE 13th International Conference on Data Mining, pp. 1313–1318. IEEE (2013)