Geoinformatica (2016) 20:285–326
DOI 10.1007/s10707-015-0236-8

A framework for intelligence analysis using spatio-temporal storytelling

Raimundo F. Dos Santos Jr.[1] · **Sumit Shah**[2] · **Arnold Boedihardjo**[1] · **Feng Chen**[3] · **Chang-Tien Lu**[2] · **Patrick Butler**[2] · **Naren Ramakrishnan**[2]

Received: 8 May 2014 / Revised: 9 August 2015 /
Accepted: 14 September 2015 / Published online: 21 October 2015

Abstract Social media have ushered in alternative modalities to propagate news and developments rapidly. Just as traditional IR matured to modeling storylines from search results, we are now at a point to study how stories organize and evolve in additional mediums such as *Twitter*, a new frontier for intelligence analysis. This study takes as input news articles as well as social media feeds and extracts and connects entities into interesting storylines not explicitly stated in the underlying data. First, it proposes a novel method of spatio-temporal

✉ Raimundo F. Dos Santos Jr.
raimundo.f.dossantos@usace.army.mil;
rdossant@vt.edu

Sumit Shah
sshah@vt.edu

Arnold Boedihardjo
arnold.p.boedihardjo@usace.army.mil

Feng Chen
fchen5@albany.edu

Chang-Tien Lu
ctlu@vt.edu

Patrick Butler
pabutler@vt.edu

Naren Ramakrishnan
naren@vt.edu

[1] U.S. Army Corps of Engineers - Geospatial Research Laboratory, Alexandria, VA, USA

[2] Virginia Tech - Computer Science Department, 7054 Haycock Rd, Falls Church, VA 22043, USA

[3] State University of New York (SUNY) at Albany, Albany, NY, USA

analysis on induced concept graphs that models storylines propagating through spatial regions in a time sequence. Second, it describes a method to control search space complexity by providing regions of exploration. And third, it describes *ConceptRank* as a ranking strategy that differentiates strongly-typed connections from weakly-bound ones. Extensive experiments on the *Boston Marathon Bombings* of April 15, 2013 as well as socio-political and medical events in Latin America, the Middle East, and the United States demonstrate storytelling's high application potential, showcasing its use in event summarization and association analysis that identifies events before they hit the newswire.

Keywords Spatial-temporal systems · Entity relationship modeling · Social media networks · Spatial and physical reasoning · Semantic networks

1 Introduction

Social media, e.g., *Twitter*, have provided us an unprecedented opportunity to observe events unfolding in real-time. The intelligence community has embraced its power, but has an ongoing struggle on how to incorporate its vast resourcefulness. The reason is that the rapid pace at which situations evolve on social media necessitates new tools for capturing the spatio-temporal progression of entities (i.e., people, organizations, events, and objects). Take for instance the *Boston Marathon* bombings of April 15, 2013. In the immediate days afterward, law enforcement officers collected a significant number of eyewitness accounts, photo and video footage, and background information on several suspects who were spatially and temporally tagged. What followed was a succession of outcomes: several people were detained near the blast spots; the residence of a Saudi national was searched; MIT police officer S. Collier was killed; the Tsarnaev brothers were identified as two suspects. All these developments could be observed on *Twitter*, but to the best of our knowledge there exists no tool that can spatio-temaporally and semantically chain these events automatically.

The underlying problem is one of *storytelling*, the process of connecting entities through their behavior and actions [32]. In this work, unlike other traditional methods, an event is simply treated as a special type of entity that represents actions, such as a "riot" or a "protest". *Information retrieval* and web research have studied this problem, i.e., modeling storylines from search results, and linking documents into stories [8, 10, 12] (the terms *stories* and *storylines* are used interchangeably). Textual *storytelling* attempts to link disparate entities that are known ahead of time, such as the connections between two individuals. In this study, however, the focus is not traditional text analysis. Rather, we explore spatio-temporal entity analysis, which can fill some of the gaps left by traditional approaches. Our goal is to not only find meaningful connections, but also to derive new stories for which we do not know the endpoint, if one exists. For example, we would be interested in examining the passing of a new law and the reactions it provokes, such as protests in nearby areas. This falls in the field of exploratory analysis where the main focus is discovering new patterns in the data. We target spatio-temporal techniques on short, ill-formed text of *Twitter* data as well as news articles for which deriving stories has proven to be a difficult task.

Textual storytelling has been mostly successful on news articles, blogs, as well as structured databases. In general, it makes one strong assumption: the availability of comprehensive data sources, where textual content is robust and ideas are well presented. In this manner, it is able to perform document analysis using several techniques, some of which include vector-space measures such as *cosine similarity*, natural language processing (NLP) for *parts-of-speech tagging*, and keyword matching, among others.

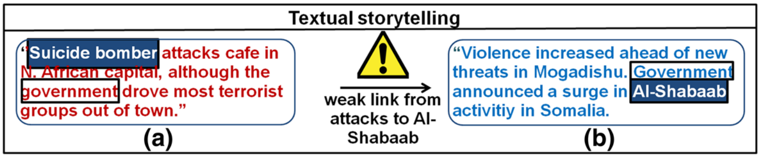

Textual storytelling
"Suicide bomber attacks cafe in N. African capital, although the government drove most terrorist groups out of town."
(a)
weak link from attacks to Al-Shabaab
"Violence increased ahead of new threats in Mogadishu. Government announced a surge in Al-Shabaab activitiy in Somalia.
(b)

Fig. 1 Under *textual storytelling*, **a** and **b** represent two partial NY Times articles [2013]. The two documents are weakly connected because no patterns other than two "government" entities relate the two documents, making the link between the "suicide bomber attacks" and the "Al-Shabaab" terrorist group of difficult identification

A common problem with such methods is that inferences may be missed whenever linkage among documents cannot be strongly asserted. Consider the example of Fig. 1. In **(a)**, a partial *NY Times* news articles describes a suicide bomber attack in Somalia in 2013, whereas **(b)** tells about a surge in terrorism activity. If the goal is to determine correlation between the suicide bomber in document (a) and the terrorist group Al-Shabaab of document (b), we would first have to link the two documents. Deriving this link is difficult for the following reason: except for government-government, no other terms are shared between the two documents. A simple *cosine similarity* calculation would yield a low score, and the suicide bomber-Al-Shabaab link would most likely be missed due to weak connectivity between the two sources.

The above example illustrates why techniques that apply to textual storytelling tend to perform poorly on social media content, such as *Twitter*, where text lacks proper form and function, and word matching can be challenging. For this reason, social media storytelling demands new techniques that can benefit not only from its textual content, but also from embedded tweet features. These features come in two flavors: (1) spatio-temporal knowledge of the entities described in text; (2) and intrinsic characteristics of social media represented in the form of metadata. An example of how these features can be helpful is given by Fig. 2 c d e and f, which shows four hypothetical tweets modeled after the *NY Times* documents of **(a)** and **(b)**, but written in a more "*Twitter*-like style" (showing the emitting users and some hashtags). Just as in the *NY Times* example, performing *cosine similarity* on any pair of the four tweets would also yield meaningless results, given that very few terms are shared. At closer investigation, however, *Twitter* data allow us to link all four documents through different means. First, tweets **(c)** and **(d)** can be linked because they were issued by

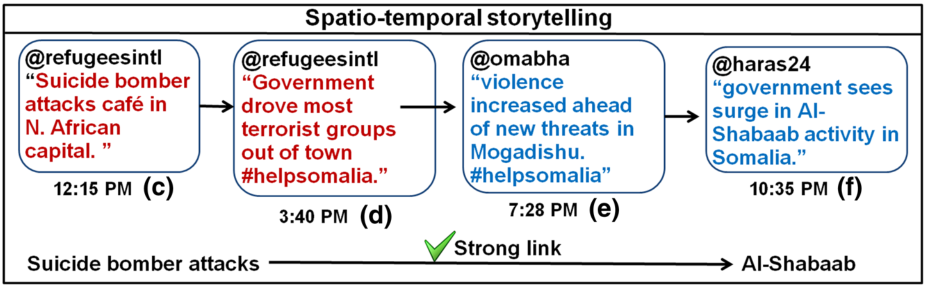

Spatio-temporal storytelling
@refugeesintl
"Suicide bomber attacks café in N. African capital. "
12:15 PM (c)
@refugeesintl
"Government drove most terrorist groups out of town #helpsomalia."
3:40 PM (d)
@omabha
"violence increased ahead of new threats in Mogadishu. #helpsomalia"
7:28 PM (e)
@haras24
"government sees surge in Al-Shabaab activity in Somalia."
10:35 PM (f)
Strong link
Suicide bomber attacks
Al-Shabaab

Fig. 2 Under *spatio-temporal storytelling*, **c d e** and **f** show four tweets with similar content to the NY Times articles of Fig 1. These tweets are strongly connected through the following features: *Twitter* user "@refugeesintl" in (**c**) and (**d**), hashtag "#helpsomalia" in (**d**) and (**e**), and locations "Mogadishu" and "Somalia" in (**e**) and (**f**). Together, the four tweets provide a stronger belief that the suicide bomber attacks are indeed linked to the Al-Shabaab terrorist group

the same *Twitter* user (@**refugeesintl**). Second, tweets **(d)** and **(e)** are connected through the *hashtag* **#helpsomalia**, a strong indication that they address the same general topics. To close the gap, tweets **(e)** and **(f)** are connected by location: geocoding Mogadishu and Somalia allows one to determine that the *latitude/longitude* of the former is enclosed in the latter, and thus making them geospatially related. Now that all four tweets are linked, it becomes possible to discover a connection between the desired $\boxed{\text{suicide bomber}}$-$\boxed{\text{Al-Shabaab}}$ entities, one attractive aspect of this approach that textual storytelling did not cover.

The above shows that tweets can be linked in many ways, such as by users, locations, and hashtags. This paper strongly emphasizes the **spatio-temporal** aspect of the data, considering only tweets that have locations and timestamps. Other features, which we explain later, are also available. Five aspects of this approach can be observed. First, it allows one to create a short storyline that, as concisely as possible, represents the four tweets without replicating them. The storyline that we envision has the format as shown in Fig. 3. It is composed of a sequence of entities identified in the tweets, such as $\boxed{\text{suicide bomber}}$ and $\boxed{\text{government}}$, and relationships, such as $\xrightarrow{attacks}$ and $\overset{surge}{\rightarrow}$, also from the tweets, which serve to make connections between the entities. The first entity in the sequence ($\boxed{\text{suicide bomber}}$) is the storyline's *entrypoint*, whereas the last one ($\boxed{\text{Al-Shabaab}}$) is the *endpoint*. Note that storylines do not necessarily follow grammar rules since they are meant to capture the semantics of the data stream rather than the syntax of the language. Later sections will explain how to create storylines and discuss other mechanical aspects, such as why some entities are included while others are ignored, and how to use the relationships. Second, storylines can be made as elastic as necessary by injecting new tweets in an incremental approach. Third, when represented as a graph, a theoretically-unlimited number of tweets can be collapsed into fewer entities and their corresponding relationships. For example, $\boxed{\text{government}}$ or $\boxed{\text{Al-Shabaab}}$ may appear thousands of times in the raw dataset, but in this approach, they are only represented once each, minimizing resource usage. In this manner, the number of generated storylines tends to be several orders of magnitude smaller than the number of tweets that generate them; fourth, they enforce time sequencing, which promotes storyline coherence by preserving the order of facts. In Fig. 2c, the storyline begins at 12:15 PM when the "suicide bomber attack" takes place, and ends with Fig. 2e at 10:35 PM when the "government announces the surge in Al-Shabaab activity". Fifth, graph structures are more machine-friendly than file systems, allowing efficient searches, spatial operations, and automated data mining.

The importance of location and time Applying traditional network analysis tools to find and link entities across tweets can lead to 'runaway' stories. Three important problems have to be surmounted. First, to ensure meaningfulness, we must use spatio-temporal coherence as both a desirable aspect of stories and as a way to control computational complexity. It is desirable because entities might be related to one another only under certain circumstances, and modeling spatio-temporal coherence ensures explainable stories. It is a way to control

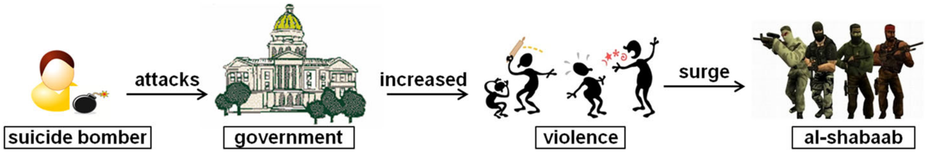


Fig. 3 A storyline composed of four entities linked by three relationships

computational complexity because it avoids searching for stories that might not be central to the topic under consideration. For instance, tweets that refer to *suicide bombing* in *South America* are most likely not related to *suicide bombing* cases in *Somalia*. Thus **spatial** is a fundamental consideration. Second, time and space must support the notion of typing to connections. For instance, a $\boxed{\text{suicide bombing}} \xrightarrow{met-with} \boxed{\text{Al-Shabaab}}$ link can potentially be inferred by the intelligence analyst if these entities are both in **proximal areas** and **close in time**. Otherwise, stating that one is related to the other in different places and times is mere speculation. Again such a notion of typing aids in both explainability and scalability objectives. Third, we require algorithms that can operate without specific provision of start and end points as long as entities can be coherently identified **in a location** and **within a timeframe**. The ability to support these dynamic aspects of storylines as they evolve is critical to modeling fast-moving social media streams such as *Twitter*. The goal of this paper is to address the above issues and enhance the current state of storytelling. The key contributions are:

1. **Modeling short text over space and time**: This research describes arguably the first algorithm to conduct storytelling without specific endpoints (i.e., without supervision) over short text (tweets), represented as an entity graph, and provides strategies to enforce coherence, precision, and the influence of spatial entity types on the generated storylines.
2. **Reasoning over spatio-temporal features**: Key to obtaining coherent stories is to identify regions of spatial propagation where related entities cluster. We demonstrate the use of *Ripley's K* function for this purpose and its use in conjunction with temporal propagation where time windows help keep stories succinct and coherent. In combination, they limit the search space from possibly millions down to the thousands of entities.
3. **Devising spatio-temporal storylines based on connectivity**: We provide a parameter-free relevance measure based on *ConceptRank*, which differentiates relationship types, boosts strongly-connected spatial entities, and helps eliminate large numbers of poorly-connected ones. In addition, storylines are found "on the fly", demonstrating our ability to generate lines of exploration that span across space and time.
4. **Performing extensive experiments on social media**: To show the effectiveness of spatio-temporal storytelling on both *Twitter* data and news articles, this approach is evaluated on current events related to the evolution of the *Boston Marathon Bombings* of 2013. Included is a comparison of this approach to others based on an event summarization task, and the discussion of a case study related to association analysis. The experiments showed that spatio-temporal storytelling was able to mimic other event summarization methods with as much as 80 % success rate in terms of event matching, and determine association among events 2 days before they hit the newswire.

Throughout this study, various components needed for storytelling are introduced. Section 2 elaborates on existing work, highlighting differences. Sections 3 and 4 explain the spatio-temporal mechanics of entity discovery, ranking, and storyline generation. Experiments are presented in Section 5 and a conclusion is given in Section 6.

2 Related works

Storytelling is not a single analytical task confined to a singular purpose. It can be better understood as a framework of intelligence analysis in which various tasks can be accomplished by different means. Very broadly, entities must be extracted, ranked, and connected, in order to make storylines visible. In this sense, storytelling involves a mix of quantitative analysis and semantic reasoning over which the boundaries are flexible. Similarly, the work proposed in this paper spans many areas of expertise, from clustering to geographic networks. This research best lines up with the approaches described below.

2.1 Storytelling

The phrase ‘storytelling’ has been introduced in an algorithmic context by Kumar et al. [12] who proposed it as a generalization of *redescription mining*. At a high level, *redescription mining* takes as input a set of objects and a collection of subsets defined over those objects with the goal of identifying objects described in two or more different ways. Such objects are interesting because they may signal shared characteristics and similar behavior, which can be a powerful tool in the context of *storytelling*. One such algorithm is *CARTWheels* [24] which utilizes induced classification trees to model redescriptions along with the *A* Algorithm* for least-cost path traversal. Hossain et al. [10] develop this idea to connect two unrelated PubMed documents where connectivity is defined based on a graph structure, using the notions of hammocks (similarity) and cliques (neighborhoods). This work was generalized to entity networks in [9] and specifically targeted for use in intelligence analysis. Their motivation is that current technology lacks better support for entity linkage, explanation of relationships, exploration of user-specified entities, and automated reasoning in general. The tools used in this work include concept lattices as a network where candidate entities are identified with three nearest neighbor approaches (Cover Tree, k-Clique, and NN Approximation). The *Soergel Distance* measures the strength between entities, while *coreferencing* serves to identify entities mentioned in various parts of the text using differing terms. All these works require specific start and endpoints, and link entities according to a desired neighborhood size and distance threshold. In many of these works, edge weight has been based on a variation of term frequency × inverse-document-frequency (*TF-IDF*). This class of works represent *traditional storytelling* approaches even though neighborhood distances are considered, albeit not from a geospatial perspective.

2.2 Connecting the dots

The primary focus of these works is on document linkage rather than entity connectivity. For this reason, textual reasoning is a strong facet of the targeted methods, which departs from a spatio-temporal view of events. Endpoints must (again) be specified and link strength utilizes the notion of *coherence* across documents, which is proposed by [27]. In this work, stories are modeled as chains of articles, where the appearance of shared words across documents help establish their relatedness. Another important aspect is the determination of *influence* between documents based on the presence of a given word. For this purpose, a bipartite graph is built using documents and words as nodes, where edge strength among them can be obtained by third-party tools or with *TF-IDF* scores. Extending that work, they also propose related methods to generate document summaries, i.e. *Metro Maps*, in [29] and [28], which target scientific literature. Some of the goals are to measure the importance of a paper in relation to the corpus, find the probability that two papers originate from the same source, and identify research lines. The basic data structure is also a directed graph, where for each map that has been generated, its *coverage* is calculated using each document as a vector of word features. The *coverage* is then defined for a set of words as *TF-IDF* values, which can be extended to sets of documents. Connectivity between maps is measured by the number of paths that intersect two maps. Overall, *connecting the dots* methods rely heavily on the abundance of robust content such that the aforementioned calculations (coherence, influence, coverage, etc.) can be calculated acceptably. Social media, however, breaks the assumption of robust content, limiting the amount of textual reasoning that can be performed. Thus, *connecting the dots* is less than ideal for environments that utilize *Twitter* data feeds.

2.3 Event detection and summarization

The goal of storytelling is to find meaningful streams of information that are neither spelled out in text nor apparent to the naked eye. As such, storytelling should not be labeled as an event detection technique or a summarization tool. However, because storytelling captures the underlying relationships among entities, it can serve broadly to summarize real-world developments or to aid in event forecasting.

In terms of event detection, event expansion and topic trending are two commonly-studied aspects. Event expansion starts with limited bits of information about an event and seeks to expand it using social media data. Topic trending, on the other hand, monitors large volumes of social streams to find the most popular themes of discussion. The work of Sakaki et al. [26] targets the detection of earthquakes in Japan using common classification techniques. Events are defined by the user by selecting keywords. TEDAS [15] describes a system for detecting new events related to crime and natural disasters, and identify their importance. It first crawls tweets, classify them as event-related or not, and stores spatio-temporal information. Users then issue queries that contain location, time, and keywords, which the system uses to retrieve and display related events. The importance of event reporting over *Twitter* is questioned by the work of Petrovic et al. [22]. The authors claim that the benefit of tweets comes from increased coverage, not timeliness. They devise a system that clusters both tweets and news articles, and measure their overlap to discover the coverage of one versus the other. Comparisons can then be done on their spread over time. Twevent [14], a different approach, proposes segment-based event identification. Initially, it detects bursty events and clusters them using frequency and content similarity. The similarity between segments is computed using their associated tweets, while Wikipedia is searched to verify which events are realistic or not. In [34], Walther and Kaisser monitor specific locations of high tweeting activity. They further analyze clusters of those tweets, using machine learning to detect if the identified posts during high activity represent real events or not.

Textual summarization has been well studied in IR, using a wide variety of techniques, such as latent semantic analysis and machine learning [3, 5]. Event summarization, as an extension, has gained strength in recent years due to social networks. *TwitInfo* describes a system that allows users to navigate a repository of tweets, where the system discovers high peaks of twitter activity [19]. In addition, the system allows geolocation and sentiment visualization. A more comprehensive approach to event summarization is detailed in [2]. The authors propose a segment-based approach where summarization takes places within each segment. This technique can take on different variations. The first uses cosine similarity as a straightforward method. The second applies a similar approach, but considers tweets that fall within a specific time window. A third approach uses a Hidden Markov Model (HMM) where each state can be a sub-class of events (e.g., “touchdown” in a football event). An alternative technique also based on time segmentation is given by [20], but with the added assistance of synonym expansion for keywords. For each of these approaches, the output is the set of tweets that best summarizes the events.

Differences Each of the above approaches have different goals and apply vastly different techniques to accomplish their objectives. As a result, direct comparisons to this proposal must be done with care. This study does not seek to summarize or detect events as the end goal, but to show them as potential applications. A better description would be to determine how entities are involved in particular events, and if so, show them as a meaningful storyline. This approach relies on a spatio-temporal model in which both geographical proximity and time ordering are favored over textual content. Most of the other approaches, instead, rely on the textual nature of documents. For these reasons, this article does not propose a competing method. Rather, it shows complementary approaches that fill a niche which has remained mostly understudied. The experiments section compares this proposed approach to three methods described in [2] and [20], explaining the differences along the way.

3 Spatial modeling

This section provides a visual representation of the proposed methods and explain the technical aspects of spatio-temporal storytelling. Figure 4 shows the three stages taken: (1) in the pre-processing stage, entities such as people and events, as well as concepts (i.e., relationships), are extracted from *Twitter* data and news articles. Combining the extracted entities and their relationships allows a concept graph to be constructed; (2) in the spatio-temporal modeling stage, entities are discovered in regions through which a storyline is most likely to propagate, using the concept graph to further rank those entities, and temporally order them; (3) Storylines are then generated using the highest-ranked entities and their observed relationships. First, the definitions used throughout the remainder of this paper are provided.

3.1 Definitions

In the scope of this study, an entity network is a graph $G(E, R)$ where entities $E = \{e_1, \ldots, e_n\}$ can be linked to one another through relationships $R = \{r_1, \ldots, r_n\}$ defined by conceptual interactions, and thus called a *concept graph*. Given a set of documents $D = \{d_1, d_2, \ldots, d_n\}$, the following definitions apply:

Definition 1 An entity e represents a person, location, organization, event, or object described in at least one document $d_i \in D$. Only entities for which a location and a timestamp can be obtained are considered in this study.

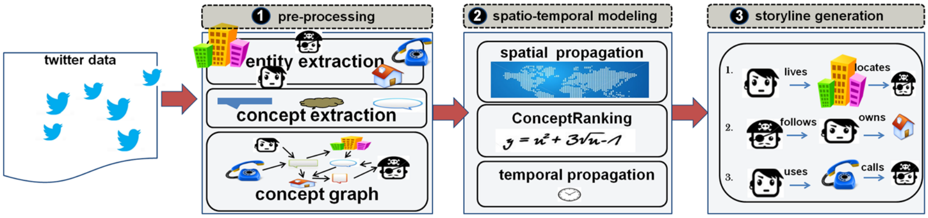


Fig. 4 Three-step process for spatio-temporal storyline generation using *Twitter* data. In the pre-processing stage, entities and concepts (relationships) are extracted and used to build the concept graph. Under spatio-temporal modeling, spatial propagation first discovers entities in nearby locations. For each entity, *ConceptRanking* determines its relevance in the graph, and the entities are subsequently time-ordered for proper temporal propagation. Storylines are then generated by linking the *top-k* ranked entities in time order

Definition 2 An event represents a special type of entity denoted by an action. Our previous examples mentioned several events such as an "attack" and an "explosion".

Definition 3 A semantic constraint is a user-defined data delimiter similar to a query parameter. For example, if one seeks stories related to "explosion" and other related terms (e.g., "bombing" or "blast"), he/she may use these terms as semantic constraints to guide the storytelling process toward those concepts.

Definition 4 A relationship, connection, or link defines a unit of interaction between two entities and is denoted by $e_i \xrightarrow{interaction} e_j$. It is deemed *explicit* if it is extracted from tweet text, such as in $D.Tsarnaev \overset{talks-to}{\rightarrow} T.Tsarnaev$. A relationship is *implicit* if it comes from metadata, as in the *Twitter* case of "follows". Note that all relationships $e_i \xrightarrow{interaction} e_j$ are intended to be directional.

Definition 5 An entrypoint is any entity *e* in the dataset and the point from where the story evolves. It is application-dependent from the perspective of the intelligence analyst. For instance, in the *Boston Marathon Bombings* scenario, the entrypoint can be the blast site (i.e., a location), an individual seen in the vicinity (i.e., a person), or any other entity of interest. The endpoint is the entity where the story ends.

Definition 6 A storyline is a time-ordered sequence of n entities $\{e_1, \ldots, e_n\}$ where consecutive pairs (e_i, e_j) are linked by one relationship. The number of entities n is the length of the storyline.

Twitter features In order to capture the importance of entities, both tweet metadata and textual content are used in the following manner:

1. **Users** are person entities and the subject and objects of **mentions**, **reply-to**, **following**, and **follower** relationships. They help establish implicit relationships, as defined above in 4.
2. **Countries**, **states/provinces**, **cities**, and **addresses** are geocoded and become location entities, both coming from metadata and text. Tweets without location are not considered.
3. **Hashtags** implicitly link entities either in the same or across tweets.
4. **Created At** (from tweet metadata) and **dates** (when available from tweet text) are both used for temporal analysis. Whenever an entity is extracted from text, a timestamp is associated to it. If the tweet text has an inline timestamp that can be associated to the entity, this timestamp will be used. Otherwise, the timestamp of the tweet metadata is used instead. Dates extracted from text are always given preference, if available.
5. **Organizations** are extracted from text (i.e., not metadata).

Figure 5 shows a simple concept graph related to the *Boston Marathon Bombings* where the entities were extracted from several tweets. Solid lines represent explicit relationships, while dashed lines denote implicit ones. We have the following: $\boxed{\text{D. Tsarnaev}}$ (D.T.) and $\boxed{\text{T. Tsarnaev}}$ (T.T.) are connected through a "talk" relationship, which was extracted from *Twitter* text (not *Twitter* metadata), and is thus defined as explicit. The same is true for the "meets" link between $\boxed{\text{T.T}}$ and $\boxed{\text{S. Collier}}$ (S.C.), the "works" link from $\boxed{\text{S.C.}}$ to $\boxed{\text{MIT}}$, and the "drives" link from $\boxed{\text{D.T.}}$ to $\boxed{\text{MIT}}$. The various links to other unknown entities (small triangles) come from *Twitter* metadata ("follows", "following"), and therefore are implicit.

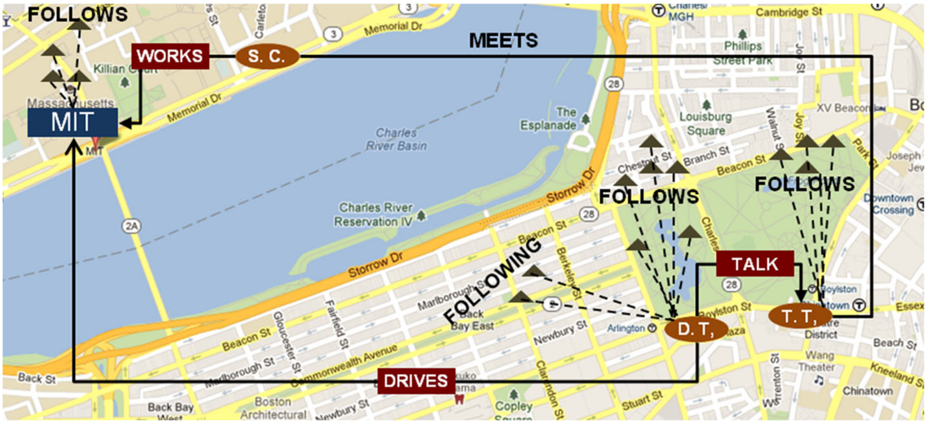


Fig. 5 Concept graph example. The *solid lines* between entities represent *explicit* relationships extracted from tweet textual content. The *dashed lines* denote *implicit* relationships from tweet metadata

The reason to differentiate the above relationships comes from a simple notion: in entity networks such as *Twitter*, semantic closeness in the form of social interactions is probabilistically correlated to spatio-temporal proximity [7] akin to Tobler's first law of Geography, in which similar things tend to be near one another. Intuitively, this notion has several implications to storytelling:

- **Relationship Typing**: explicit connections can be more helpful than implicit ones. Knowing that "D.Tsarnaev spoke to T. Tsarnaev" is potentially more powerful than simply learning that "D.Tsarnaev mentions (in the *Twitter* sense) T. Tsarnaev". This idea is explored in Section 3.3 about Concept Ranking.
- **Relationship Propagation**: a story can be modeled as a graph of entities and semantic relationships propagating through spatially-close regions in a temporal sequence. Consider Fig. 6a which depicts several locations related to the *Boston Marathon Bombings*. Most of its developments took place in an 8-day interval (Apr 15–22, 2013) and in proximal areas: Boston - MIT Campus - Watertown. Developments in Canada or Chechnya are an evolving part of the story, but do not necessarily play a major role. Based on these ideas, Section 3 defines spatio-temporal propagation in order to explore constrained regions of entity connectivity where stories can evolve from.
- **Relationship Boundaries**: stories do not necessarily have endpoints. Entities come and go, relationships develop, and locations vary. In the *Boston Marathon Bombings*, the entry point could be any one of thousands of persons. The end could propagate through Canada, Russia, and other places. This idea is applied in the experiments section to further justify the use of evolving stories.

3.2 Spatial entity discovery

In the process of telling a story, the entrypoint can be any entity such as a person or event, as in the "bombing" scenario. Given an entrypoint, the goal is to delimit a region where the "most amount of information" can be found, and grow that region until seemingly relevant information becomes sparse. To find this region, several techniques could be explored, but

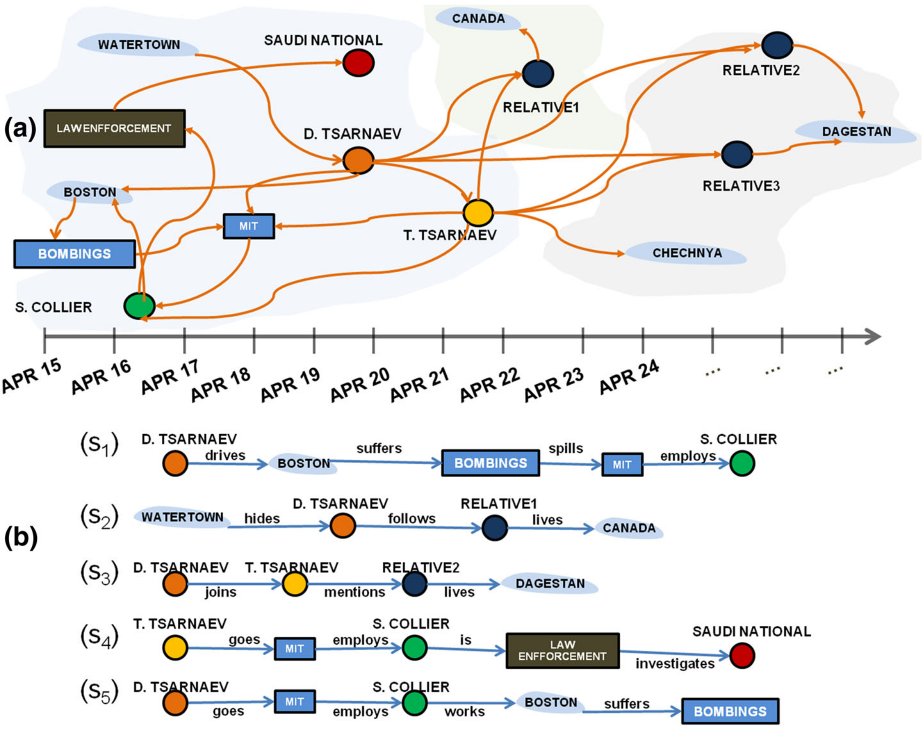

(a)
WATERTOWN
SAUDI NATIONAL
CANADA
RELATIVE2
LAWENFFORCEMENT
D. TSARNAEV
RELATIVE1
DAGESTAN
BOSTON
RELATIVE3
MIT
T. TSARNAEV
BOMBINGS
CHECHNYA
S. COLLIER
APR 15
APR 16
APR 17
APR 18
APR 19
APR 20
APR 21
APR 22
APR 23
APR 24
...
...
...
(b)
(S1) D. TSARNAEV drives BOSTON suffers BOMBINGS spills MIT employs S. COLLIER
(S2) WATERTOWN hides D. TSARNAEV follows RELATIVE1 lives CANADA
(S3) D. TSARNAEV joins T. TSARNAEV mentions RELATIVE2 lives DAGESTAN
(S4) T. TSARNAEV goes MIT employs S. COLLIER is LAW ENFFORCEMENT investigates SAUDI NATIONAL
(S5) D. TSARNAEV goes MIT employs S. COLLIER works BOSTON suffers BOMBINGS

Fig. 6 Boston Marathon Bombings spatio-temporal sequence. In **a**, each shape represents an entity observed in a tweet. The edges denote relationships between the entities. In **b**, S_1 through S_5 represent five storylines connecting different entities. The English verbs define their relationships and correspond to the edges of the concept graph in (**a**)

not all of them fit spatio-temporal *storytelling* adequately. One of them would be to perform a simple *Nearest Neighbor*(*NN*) search on the area of study and collect the discovered entities. *NN* searches, however, are "blind" to the dataspace, i.e, they find entities without relaying information about how they disperse, and thus are not used here. Another alternative method is *Pair Correlation Function* (*PCR*) [25], which divides the data space into spatial segments, allowing each segment to be weighted higher (lower) for closer (farther) entities. Spatio-temporal *storytelling*, however, only needs nearby regions, thus segmenting them does not serve a useful purpose. *PCR*, therefore, is not an ideal choice. Other possible methods are the variations of partitional clustering, such as *K-means*, which could serve to group related entities before linking them. While feasible, this type of clustering demands several initialization centroids, which *storytelling* does not provide (in our approach, only one entrypoint is initially given). In addition, this early in the process, performing any type of clustering adds complexity that can be avoided by other approaches. Below, we explain a preferable method.

Consider Fig. 7a where each point represents a person who tweeted during the *Boston Marathon Bombings* near the blast sites. Circle A designates an area of 1 km around the entrypoint (i.e., blast site) with a high concentration of person entities. If we consider 2 km, as in circle B, the density decreases, while circle C becomes even more sparse. Intuitively, the investigation should focus on the 1 or 2-km radii where most of the information resides. In theory, this is the modeling of a point process (i.e., a collection of persons who sent tweets) in terms of a randomly chosen event E (i.e., bombing) with an estimator distance function for a given density λ, which is given by *Ripley's K-coefficient* $K(r) = \lambda^{-1}E$. Mathematically, $K(r)$ can be stated as:

$$K(r) = \sqrt{\frac{A\sum_{i=1}^{n}\sum_{j=1}^{n} w(i,j)}{\pi n(n-1)}}, i \neq j \tag{1}$$

where r is a desired radius originating at a chosen entrypoint, n is the total number of entities in the data space, A is the entire area of study, and $w(i, j)$ represents a weight. $w(i, j) = 1$ if distance$(e_i, e_j) < r$, and 0 otherwise. In effect, $K(r)$ performs a nearest-neighbor search and can be viewed as a clustering coefficient for a desired type of entity (e.g., persons sending tweets) within a limited radius. The coefficient can be evaluated at different scales, such as $r = 1$ km or $r = 1.5$ km. Figure 7b and c show two simple calculations of the *K-coefficient* for 3 persons $\{P_1, P_2$, and $P_3\}$ located in a (3 km × 3 km) area A. In Fig. 7b, the chosen radius is 1 km. The calculation follows: using each entity P_i as the center of a 1 km circle, count the number of other entities P_j within that radius, adding 1 if their distance is less than the radius, zero otherwise. In that range, P_1 "can see" 2 others (P_2 and P_3), since their respective distances ($dist(P_1, P_2)$ and $dist(P_1, P_3)$) are both less than $r = 1$. Using P_2 as the center of a 1km-radius, P_2 "sees" only P_1. The same is true for P_3, which yields $K(1) = 0.53$. In Fig. 7b, the radius is increased to 1.5 km, and the calculations are repeated, yielding a $K(1.5) = 0.65$.

Comparing the two calculations indicates that the larger radius picked up more points and resulted in more clustering, with the same density. Increasing the radius can potentially find more empty space, which is undesirable. *Ripley's K-coefficient* is an elegant method of discovering related nearby things, but does not tell what a good radius should be or whether lower/higher density is better or worse. *Ripley's* gives us an opportunity to present a set of heuristics that calculates a feasible $K(r)$ in the discussion below.

3.2.1 Finding a feasible K(r)

In the previous analysis, one needs a systematic way to determine if the 1-km radius is better than 1.5 km, or vice-versa. The region delimited by the radius that yields the highest K-function score is where the storytelling process will initiate. Given that a real-world dataset may contain millions of entities, a feasible region is one that includes enough data points,

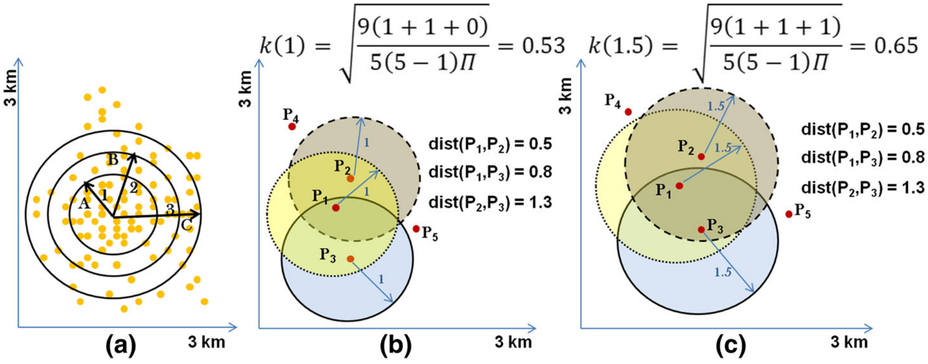

$k(1) = \sqrt{\frac{9(1+1+0)}{5(5-1)\Pi}} = 0.53$
$k(1.5) = \sqrt{\frac{9(1+1+1)}{5(5-1)\Pi}} = 0.65$
3 km
A
B
C
1
2
3
3 km
(a)
P4
P2
P1
P3
P5
1
dist(P1,P2) = 0.5
dist(P1,P3) = 0.8
dist(P2,P3) = 1.3
3 km
(b)
1.5
(c)

Fig. 7 Spatial scaling for different radii. **a** *Circle* A depicts high entity density, becoming more sparse in circles B and C. **b** and **c** shows the calculation of *Ripley's K* function for a 1 and 1.5-km radius respectively

but not all of them. Looking at Fig. 7a, Circle B covers most of the entities in that dataset, which may be excessive for many applications. The problem is that Circle B has a 2-km radius, which corresponds to most of the length of the entire study area of 9 km^2. A better approach in this case can be done according to Algorithm 1, which is explained below.

The first step is to select an initial random radius to work with. Since this ideal initial radius is not known, the algorithm takes a "half-decrement" approach, in which the analyzed radius is cut by half of the length of the dataset iteratively until a reasonable radius is found. This is initiated where the algorithm specifies r_i as 1/2 the length of the data set (r_i in Line 1). This initial radius can be manipulated higher or lower to comply with application needs or when better knowledge of the dataset is known apriori. Using radius r_i from the story's entrypoint, a list of entities is obtained by performing a range query over the spatially-indexed entities in the database L (Line 2). A simple check is then made: if the ratio of retrieved entities (|Ents|) and total number of entities ($|e_A|$) is equal to or greater than a certain threshold, say 10 %, then too many entities have been retrieved (Line 3) and they are discarded (Line 5). The algorithm halves the initial radius (Line 6) and tries again (Line 7). Once the calculation hits a point below the threshold, the algorithm has found r_{Limit}, i.e., a radius that covers an adequate number of entities (Line 8).

Algorithm 1 Distance Computation

```
inputs: spatially-indexed entity database $L$, area $A$, entity count threshold $T_e$, number of entities in $A$ $|e_A|$, story entrypoint $e$
output: radius $r_i$
1: initialize: i=1; k = i-1; $r_i = \frac{length(A)}{2}$ ; // set the initial radius as half of the length of the study area (customizable).
2: List {Ents} ← rangeQuery(L, $e$, $r_i$) ; // create a list of entities by performing a range query from the radius.
3: if $\left(\frac{|Ents|}{|e_A|} \geq T_e\right)$ // compare the list of entities against a desired threshold
4: then
5:     discard {Ents} ; // if too many entities are found , discard them all.
6:     set $r_i = \frac{r_i}{2}$ ; // shorten the radius by half of the previous size.
7:     iterate Line 2 ; // and run a new iteration with the new radius
8: set $r_{Limit} = r_i$ ; // save the newly found radius.
9: $K(r_i) = calculateK(r_{Limit})$ ; // calculate Ripley's K function for the new radius.
10: initialize $K(r_k) = 0$;
11: while $\left(K(r_i) > K(r_k)\ and\ \frac{|Ents|}{|e_A|} < T_e\right)$ // run more iterations until K stops increasing and threshold is not met, then output $r_i$
12: do
13:     {Ents} ← rangeQuery(L, entrypoint, $r_i$) ;
14:     $K(r_k) = K(r_i)$ ;
15:     set $r_i = r_i + \frac{r_{Limit}}{2}$ ; // as long as $K(r_1)$ keeps increasing, increase $r_i$ by half its previous value.
16:     set $r_{Limit} = \frac{r_{Limit}}{2}$ ; // save the new radius temporarily.
17:     $K(r_i) = calculateK(r_i)$ ; // calculate Ripley's K function for the increased radius.
18: end
19: output $r_i$;
```

On its own, r_{Limit} is possibly good enough, but not necessarily the best radius. For example, it is possible that r_{Limit} corresponds to Circle B of Fig. 7a. Ideally, however, it would be better to find Circle A, or even a smaller circle inside of A, as they seem to concentrate most of the entities. The goal, then, is to find the highest clustering coefficient beginning with r_{Limit}, which is stored as $K(r_i)$, through an iterative process, but one which does not exceed threshold T_e. Using r_{Limit}, $K(r_i)$ is computed (Line 9). In successive steps, r_i is incremented by half the value of r_{Limit} and its K is recomputed (Lines 11–17). As soon as $K(r_i)$ stops growing from its previous value or the number of retrieved entities reaches threshold T_e, the process stops. $K(r_i)$ has reached an adequate coefficient for this specific radius, which is output in Line 19. In theory, there is no guarantee the "truly best" radius has been found, but since increments of r_{Limit} become smaller and smaller over many iterations, we hit the law of "diminishing returns" and stop the process for the sake of efficiency. It now can be stated that the storytelling process will include all entities located within range r_i of e.

3.3 Concept ranking

In Section 3.2, r_i is calculated as the radius originating at the entrypoint from where the storyline should propagate. Within that range, many entities can be present, which requires a ranking strategy to determine an order in which entities should be investigated. For this purpose, there are alternative approaches, as in performing textual similarity based on methods such as *cosine similarity* [23] or comparing the values of attributes from each entity [17]. These approaches, however, are efficient on textually-rich sources, but not adequate for *Twitter* data, which are more often than not poorly described. Since this work uses a graph of connected entities as data representation, ranking is proposed as a variation of *PageRank* [1], extended as *ConceptRank*, and explained below.

Given a network of web pages, *PageRank* assigns the highest(lowest) importance to the most(least) referenced page(s), offset by the relevance of the referring page. It is given by:

$$PR_{(p_k)} = \left(\frac{1-\Gamma}{N}\right) + \Gamma \sum_{p \in Links(p_k)} \left(\frac{PR_{(p_i)}}{OL_{(p_i)}}\right) \tag{2}$$

where $PR(p_k)$ is the *PageRank* of page p_k, N is the total number of web pages, Γ is a user-defined damping factor in [0..1], $Links(p_k)$ is the set of links to page p_k, and $OL(p_i)$ is the number of outbound links from page p_i. Consider the concept graph of Fig. 8, where each node, instead of a web page, is assumed to be a spatially-tagged entity. It can be seen that T.TSARNAEV has the most **inbound links** (5), MIT has four, and S.COLLIER has only one. The other entities have none. Under *PageRank*, the most important entities (i.e., entities with the highest *PageRanks*) would be T.TSARNAEV, MIT, and S.COLLIER since they are the most connected entities.

One notable aspect of *PageRank* is that it does not differentiate relationships. Thus, in Fig. 8, "stop" and "drive" have the same influence in the *PageRank* calculation as does "following" or any other relationships. In terms of storytelling, this represents a deficiency because the types of interaction among entities relay strong information and should be accounted for. For example, persons seen around the blast site may hold clues to the bombing. However, students commuting to the MIT Campus from other directions most likely play no role in the bombing. Therefore the types of links influence the story and should be discriminated appropriately.

Given the above discussion, we propose *ConceptRank* not on web pages, but rather on entities, as follows. In a concept graph, the relevance of an entity is determined by a combination of both *implicit* and *explicit* relationships, as stated in *Definition* 4, but differentiated by their respective frequencies. Mathematically, *ConceptRank* is defined as follows:

$$CR_{(e_k)} = \left(\frac{1-\Gamma}{N}\right) + \Gamma \sum_{p \in Links(p_i)} \left(\frac{CR_{(e_i)}}{\psi_{e_i}} + \frac{CR_{(e_i)}}{\Phi_{e_i}}\right) \tag{3}$$

where $CR_{(e_k)}$ is the *ConceptRank* of entity e_k, N is the total number of entities in the concept graph, Γ is the same damping factor as before, $Links(p_i)$ is the set of links to page p_i, ψ_{e_i} is the number of explicit outbound relationships of entity e_i, and Φ_{e_i} is the number of implicit outbound relationships of e_i. For all purposes, Φ_{e_i} can be viewed as a *Twitter*-specific parameter obtained from metadata relationships as outlined by the *Twitter features* of Section 3.1. In real datasets, explicit relationships are less prevalent while implicit relationships tend to abound, making them less useful in a ranking strategy. An illustration follows.

Consider the case in which law enforcement is investigating persons who were **stopped** by a cop, or anybody **driving** to the MIT Campus. The underlined words are the semantic constraints sought on text. The concept graph of Fig. 8 depicts a few interactions related to $N = 10$ entities. We set $\Gamma = 0.75$, which can be viewed as the initial *ConceptRank* value that every entity receives regardless of its connections. This parameter can be manipulated. For each entity i, we must first determine its number of implicit (Φ) and explicit (ψ) outbound relationships. $\boxed{\text{S.COLLIER}}$ has one outbound relationship ($\overset{stop}{\rightarrow}$), which is explicit since it comes from *Twitter* text (not *Twitter* metadata), and no implicit ones. Thus its $\psi = 1$ and $\Phi = 0$. $\boxed{\text{FATHER}}$ has only one outbound relationship ($\overset{mentions}{\rightarrow}$), which comes from *Twitter* metadata, and so is considered implicit. Thus its $\psi = 0$ and $\Phi = 1$. Table 1 summarizes the data for all entities, along with their *ConceptRank* (calculations not shown). What the *ConceptRank* values contribute is a ranked list such that the most relevant entities and their relationships can be weaved into a storyline. The ordering goes from highest to lowest values of *ConceptRank*, yielding the following ranking:[] $\boxed{\text{T.TSARNAEV}}$ $\boxed{\text{MIT}}$ $\boxed{\text{S.COLLIER}}$ $\boxed{\text{MIT students}}$, since these entities have the highest values. The next four entities, ($\boxed{\text{FATHER}}$, $\boxed{\text{MOTHER}}$, $\boxed{\text{AUNT}}$, and $\boxed{\text{D.TSARNAEV}}$) have the same *ConceptRank*, in which case they can be inserted in any order. Given a different mix of implicit and explicit relationships, the ordering may change. In practical terms, *ConceptRank* favors the most well-connected entities, punishing the ones that are thinly-referenced in its spatial region. In the next section, we explain that only the top ranked entities (according to a threshold) are considered. All others are disregarded, preventing them from taking part in the story generation process.

4 Spatio-temporal propagation

In this section, entities that were previously extracted from the datasources are organized such that they are not only spatially-correlated, but also time-ordered in a way that makes sense to the human mind. The key concept here is that entities evolve along space and time,

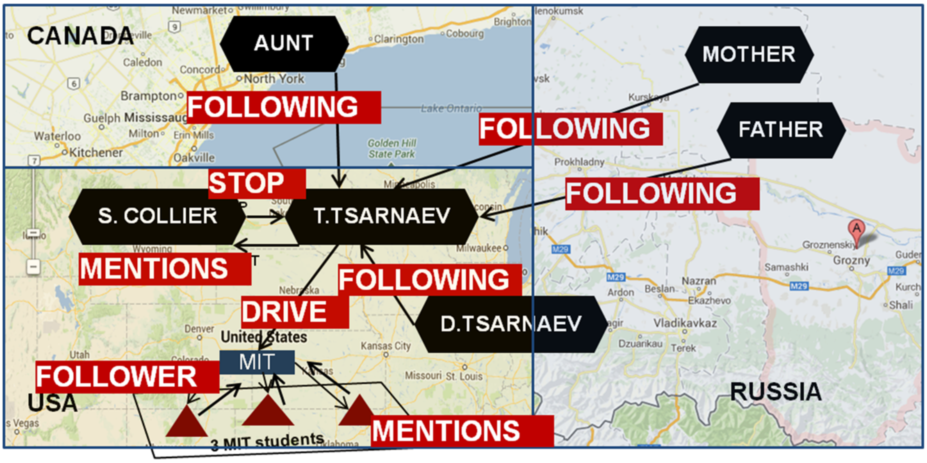


Fig. 8 Concept graph with mixed relationships. Twitter features such as *following*, *follower*, and *mentions* are considered *implicit relationships*. Others, such as *stop* and *drive* are deemed *explicit*

Table 1 ConceptRank illustration for the network of $N = 10$ entities in Fig. 8 with a starting damping factor of $\Gamma = 0.75$

q=**stop,drive**		$N = 10$	$\Gamma = 0.75$	
i	Entity (e_i)	ψ	Φ	$CR(e_i)$
1	T.TSARNAEV	$1\ (\overset{drive}{\rightarrow})$	$1\ (\overset{mentions}{\rightarrow})$	0.0282
2	MIT	0	$3\ (\overset{follower}{\rightarrow})$	0.0276
3	S.COLLIER	$1\ (\overset{stop}{\rightarrow})$	0	0.0264
4	MIT Students (each)	0	$1\ (\overset{mentions}{\rightarrow})$	0.0250
5	MOTHER, AUNT, FATHER, D.TSARNAEV (each)	0	$1\ (\overset{following}{\rightarrow})$	0.0241

Entities are ranked from highest to lowest values of *ConceptRank* $CR(e_i)$

and thus the notion of *spatio-temporal propagation* becomes an integral part of the storytelling process. Spatio-temporal propagation requires a strategy that prevents sequential contradiction, which is addressed with the use of *time windows* in Section 4.1. Time windows must be treated carefully as it raises several design questions, which are addressed in Section 4.2.

4.1 Devising time windows

One important aspect of intelligence analysis is the sequence in which real-life developments take place. In the *Boston Marathon Bombings* scenario, for instance, it is clear that the $\boxed{\text{BOMBING}}$ event should precede the arrest of suspect $\boxed{\text{D.TSARNAEV}}$, and not the other way around. Temporal propagation over *Twitter* data is challenging for three reasons:

1. **Varying lengths**: in many instances, entities are spread throughout long periods of time (e.g., a war), while in others, the time span can be very short (e.g., a terrorism act). Therefore, varying-length time intervals must be accounted for;
2. **Bursty behavior**: often, entities display disparate frequencies in arrival rates. In an initial time period, for example, millions of tweets can be issued due to a high-visibility event (e.g., Barack Obama's election). But that same event may subside over time when it is no longer considered "news". Thus, distribution becomes important;
3. **Time synchronization**: many entities may be observed at the same time, in which case ordering them is not intuitive. Therefore, ties must be somehow dealt with.

One way to get around the above problems is to utilize a *time matrix*, which provides an intuitive way of aggregating spatial entities in flexible time intervals. In a *time matrix*, each column is a *time unit* and each row is a fraction of the *time unit*. Each cell of the matrix holds the entities observed at specific times. Figure 9a shows an example where each column represents one day of the week (i.e., the *time unit*), and each row represents the time of the day. A *time matrix* permits entities to be observed as a sequence of interactions and can be made as short or as long as the situation dictates. One can then perform data analysis on the entire matrix or on a subset of rows and columns, which we denote as a *time window*. In the scope of this study, a *time window* is defined with a simple rule:

Definition 7 Given a *time matrix* of interest (TM) composed of n *time units* (TU), a time window (*TW*) is composed of one or more TU_i where $0 < i \leq n$. In other words, a time window corresponds to a pre-defined time interval or a subset of it.

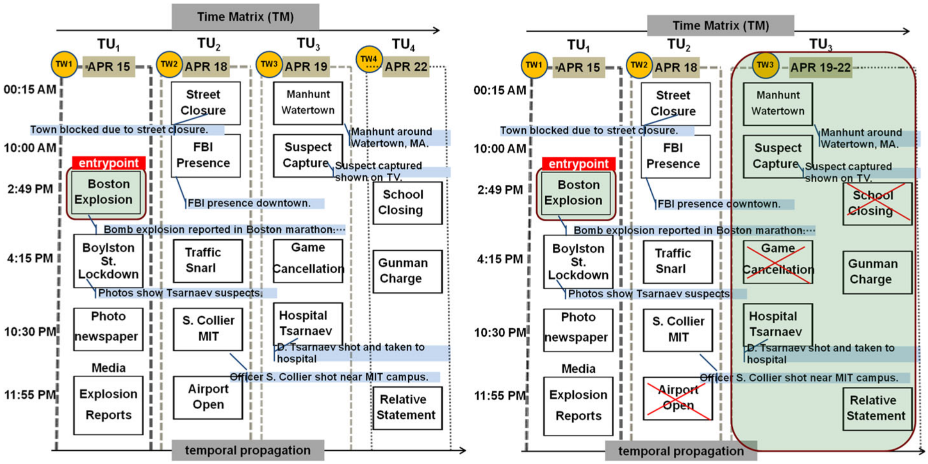


Fig. 9 Visualization of a *Time Matrix*. **a** Temporal propagation of entities in 4 time windows TW1-TW4. Each entity is designated by a box and allocated to a *time unit* TU_i according to the entity's timestamp. **b** The *crossed* entities indicate that they have been pruned. Time units TU_3 and TU_4 are merged as a new single *time unit* TU_3

For example, consider the *Boston Marathon Bombings*, where some of the developments took place over 7 days starting on April 15, 2013. We can establish its time matrix as **TM** = ***one week***, each *time unit* TU_i = ***one day***, and each column as the hours/minutes of the day, with $\boldsymbol{n} = \mathbf{7}$. Figure 9 shows the corresponding time matrix, where $TU_1 = 15Apr$, $TU_2 = 16Apr$, etc... For more granular applications, the time matrix can be adjusted to one day and each *time unit* can be the minutes/seconds of the day. The point is that the user must determine the time units that make sense for the task at hand. Having established the time units, we must now define the length of each time window. A simple approach is to make each time window the same as a *time unit*. In Fig. 9a, for instance, each time window *TW* corresponds to one *time unit* (e.g., $TW_1 = APR15$ or $TW_2 = APR18$). Alternatively, a time window can be a combination of several time units, as is shown in Fig. 9b where $TW_3 = APR19 - 22$.

The time window parameters above are decided on a per-application basis. Once established, each time window can be populated with the entities found according to the method in Section 3.2. This is easily accomplished by allocating each entity to the appropriate TW_i based on the entity's timestamp. On *Twitter* data, the timestamp is ideally extracted from text. Since that is not always available, the tweet's metadata timestamp can be used as a good-faith approximation. One additional caveat must be made: only entities that meet a minimum value of *ConceptRank* are inserted (*ConceptRank* is explained in Section 3.3). A visual example follows.

Figure 9a depicts a partial time interval of four discrete days (Apr 15,18,19,22) related to the *Boston Marathon Bombings*. Some textual description is included for illustration purposes. It is assumed no data is available for the missing days (Apr 16,17,21). Here, we set TM = 4 days and set each each TU_i = 1 day on an hourly basis. Knowing that the $\boxed{\text{Boston Explosion}}$, which is set as the storyline's entrypoint, occurred on April 15 at 2:49 PM, we place that entity in TU_1. It is followed by the $\boxed{\text{Boylston St. Lockdown}}$ at 4:15 PM, and so forth. The same is done for the rest of the days until all entities in the data space have been addressed for that time matrix. This organizational model is not only attractive for its simplicity, but it also serves as a look-up data structure where sequences of developments can be easily found. In Section 4.3, time windows are revisited and put to use after the computation of entity connectivity.

4.2 Time windows considerations

The model explained above provides an efficient view of time-ordered entities and events, which facilitates reasoning. However, it raises design questions for which decisions must be made and are explained below:

- **Time span**: entities may cover more than one time window. It is possible, for instance, that the Street Closure of April 18 last several days. In this case, allocation to a time window is done according to the entity's earliest observation time. That entity, therefore, is placed in TW_2 since its earliest occurrence is indeed April 18.
- **Concurrency**: entities may have the same timestamp, in which case there is no clear way to order them. In such scenarios, the following differentiation can be made: preference is given to the entities that contain either a *semantic constraint* (see *Definition* 3) or the most specific location. If the tie still cannot be broken, arbitrary ordering is taken as the last option. For example, if the user seeks semantic constraint "explosion", then entities with such a mention are placed in its time window before another entity that has the same timestamp, but with no such mention. Similarly, an entity located at *Boylston St.* precedes any simultaneous entity located in *Boston*, since the former location is more specific than the latter.
- **Frequency**: rare entities can be pruned since they provide little connectivity strength (connectivity, an important feature of this approach, is explained in Section 3.3). For example, assume that the Airport Open in TW_2, the Game Cancellation in TW_3, and the School Closing in TW_4 appear very few times. In this case, they are removed from the analysis, which is indicated by the red crosses in Fig. 9b. Pruning removes non-interesting entities, thus saving processing cycles.
- **Merging**: two time windows TW_j and TW_k can be merged when they are deemed too sparse. For example, in Fig. 9b, *Apr 19* and *Apr 22* had some entities pruned, leaving them relatively unpopulated as compared to the other TW_i. To save computing resources, they are combined into a single window, namely "*Apr19–22*", denoted by the shaded area. The time sequence of the remaining entities are still preserved.
- **Size**: in theory, a time window TW can hold any number of entities and can be composed of any number of time units, only limited by the length n of the time matrix. In addition, they do not have to have uniform lengths. However, long time windows, whether uniform or not, may generate excessively long storylines, which in turn tends to become less intelligible. The experiments of this study reveal that short time windows of one or two time units are not only more computationally efficient, but also allow more coherent storylines than longer time windows.

4.3 Spatio-Temporal storyline generation

This discussion puts together the ideas in Sections 3.2, 3.3, and 4.1 to generate storylines. Algorithm 2 takes as input the user's desired entrypoint, and an appropriately pre-defined Time Matrix. The essential steps are as follows: obtain the radius of study and identify the entities in that radius (Lines 1 and 2); compute the *ConceptRank* of the found entities and allocate the most important ones to an appropriate time window according to their timestamps (Lines 3 and 4); using each time window, build the storylines with temporal ordering (Line 6); for each pair of entities, select a relationship to insert in between them (the most frequent relatioship is often appropriate) (Line 8); if the storyline is too short or incomplete, a new entrypoint is established as the next highest ranking entity above the top-k ones (Line 10). The process iterates (Lines 9 to 11), otherwise, the storyline is output (Line 13).

Algorithm 2 Storyline Generation

```
inputs : Entity entrypoint, pruned and consolidated Time Matrix $TM$
output: List storyline
1:  using entrypoint $\rightarrow$ get radius $r$ from spatial propagation ;
2:  entities $\leftarrow$ identify entity set from radius $r$ ;
3:  compute ConceptRank for each $e_k$ in entities ;
4:  segregate each $e_k$ into the appropriate $tu_i$ of $TM$ ;
5:  foreach $tu_i$ and if |entities| $< k$ do
6:      storyline $\leftarrow$ add top-k entities in time order
7:  end
8:  storyline $\leftarrow$ for each pair of entities, establish their relationship as their most frequent one ;
9:  if (storyline should proceed) then
10:     set new entrypoint $= e_{k+1}$;
11:     iterate $\rightarrow$ step 1
12: end
13: output storyline;
```

The above process may generate long storylines, which may become less intelligible. However, the point at which this iterative process should stop depends on one's own understanding of fact completeness. Entity extraction operates on O(log N) where N is the number of entities. Distance computation must be done two at a time, which takes O(N^2), which is one of the most costly steps, but may be optimized by avoiding computing far away entities. Insertion of new entities into the graph requires a check to see if the entity already exists, which is done in constant time. Range searches may perform from O(log N) to O(N) depending on the number of location overlaps. Computation of the *ConceptRank* affects only the inserted entities and the ones they link to either directly or indirectly. Figure 10 shows the propagation of a storyline across four different regions in four iterations [t_i, t_{i+3}] of Algorithm 2. The entrypoints are represented by squares and the other entities by circles. At each iteration, the top 2 entities are linked followed by a new entrypoint,

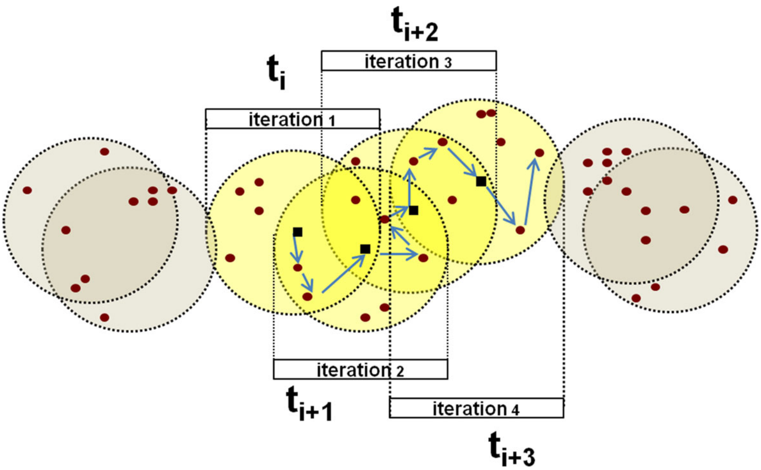


Fig. 10 Hypothetical generation of a storyline through four iterations of the algorithm (t_i through t_{i+3}). Each *circle* corresponds to one iteration. *Squares* represent entrypoints and dots represent entities. Each iteration begins at an entrypoint and connects two other entities, before a new entrypoint is considered

from where a new iteration begins. In this simple example, the four iterations generate one storyline composed of 12 entities (4 entrypoints + 8 other entities) and their relationships.

4.4 Complexity analysis

As mentioned previously, spatio-temporal storytelling involves the execution of many tasks before results can be displayed. Below, the major tasks are listed in terms of time performance and some options are considered:

- **Ripley's K function**: One of the uses of Ripley's coefficient (Eq. 1 in Section 3.2) is to find regions where entities concentrate in high numbers. Therefore, this function is very sensitive to the size of the study area. This process entails calculating the distance between every pair of entities located within a given radius, which runs in O(n^2) provided that no prior distance information (other than latitude and longitude) is known. This may change in situations where points are within negligible distance of one another. In this case, which can be done as an optimization step, the total number of distance calculations can be reduced significantly. In many applications, Ripley's K function is performed several times for different values of radii. The true running time, thus, becomes $t \times$ O(n^2) in the worst case, where t is the number of radius values to be investigated and new data points have been introduced. Looking at Eq. 1, it can also be seen that Ripley's function can be weight-based. In this study, that weight is simply 1 if the entities fall within the radius of study, and 0 otherwise. Under this condition, the weight is simply a look-up, which operates in constant time O(1). Another possible approach would be to vary the weight with different segments, such that entities in the same (different) segment would receive a higher (lower) weight. This would have the effect of increasing spatial accuracy, but comes with a performance cost. The running time would increase to O(n^2) + O($s \times n^2$), the first factor to compare all entities within the radius, and the second to compare entities within each segment s (combining the operations is possible provided that segment sizes are known ahead of time). Applying an index at the segment level (so that entities within the same segment would not need comparison) would bring the complexity back down to O(n^2). A spatial index such as an *R-Tree* often incurs in O(n) for the initial build, where n is the number of entities assuming a clean dataset without overlaps. Its benefit, however, comes in terms of searches, which can be done in O(log $\frac{n}{m}$), where m is the branching factor, and can well speed up distance computations.
- **Finding a feasible K(r)**: This refers to Algorithm 1 in Section 3.2.1, which determines an ideal radius of investigation. The first step is to create a list of entities from the results of a range query. This range query is delimited by a given radius, and constrained by a user-defined threshold representing the maximun number of entities that should be included. It runs on O(n), where n is the number of entities. This process can get more costly if the threshold is set too high. In such cases, the list may need to be recreated with a longer radius if not enough entities are located. Conversely, if the threshold is set to low, fewer entities will be analyzed, but may not generate enough storylines. Again, the process may need to be repeated. In an iterative process where t is the number of iterations, the running time would increase to O($t \times n$), where n is the number of found entities. This process also requires computing Ripley's K function, which was explained in the item above.
- **ConceptRanking**: Before linking entities into storylines, a *ConceptRank* (Eq. 3 in Section 3.3) is computed for each entity in the graph. *ConceptRank* requires for every entity in the graph: the number of incoming links (in), the number of outgoing links

(out), and the types of each link (typ). Given a graph of n entities, determining all *ConceptRank* requires O(*in*) + O(*out*) + O(*typ*). But since these items are simply properties of each entity, they can be combined on a single per-entity step, and thus run at O(n). This a worst-case greedy scenario. In fact, this process can be highly optimized under certain conditions: disconnected entities (there can be many of them) may not need processing depending on the application; when not all relationship types are relevant, only the nodes with the relevant ones can be considered; the application may be able to discard certain entities types (e.g., hashtags) from the analysis. Adding more nodes incrementally tends to cause little change to this process.

- **Establishing time windows**: This task is described in Section 4.2, and generally speaking, can be considered lightweight given similar optimizations as described in the *ConceptRank* discussion above. In a worst-case scenario, every entity must be tagged with a timestamp, which runs in O(n) for n entities. Determining entity frequencies requires pairwise comparison at O(n(n − 1)). Frequencies, however, can be determined from previous steps, and may be disregarded. Time window creation is simply a set-up step with constant time. Merging sparse windows requires two look-ups: the first to find which windows are poorly populated; and the second to determine which windows are adjacent (only adjacent ones are eligible for a merge). Both of these steps can be performed concurrently at $O(w) + O(adj^2)$, where w is the number of time windows and adj is the number of sparse windows that have a sparse neighbor as well. Note that even under quadratic complexity, this step is often efficient due to the fairly small number of time windows that most applications require as compared to the number of entities that go into them.
- **Storyline Generation**: This is the final step of the process, which encompasses all items discussed above, and is detailed in Section 4.3. Putting all steps together, the most appropriate way to describe the entire complexity is $O(n^2)$. Indeed, spatio-temporal storytelling is mostly based on pairwise comparisons. While there are many optimizations that can be done, some of which are mentioned above, it is surprisingly challenging to achieve lower running time, even in the O(n) range. The best workaround is to operate with less, more relevant data filters in the preprocessing stages so to combine efficiency and accuracy early on. Alternatively, one may want to consider distributed processing on platforms such as *Mapreduce* [18], which has been used in the experiments, but is not discussed in the scope of this document.

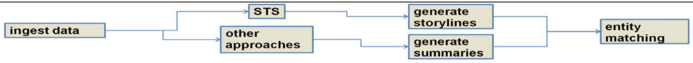
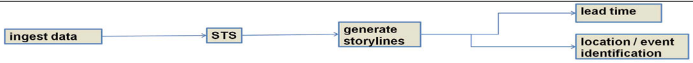
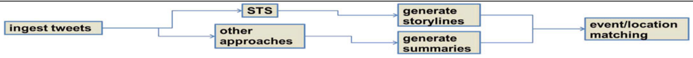
5 Empirical evaluation and technical discussion

Spatio-temporal storytelling can be gainfully applied to everyday analytical tasks. To follow through with that statement, the experiments are divided in three parts. Section 5.2 compares the approach in this paper to three existing methods of event summarization. Section 5.3 presents association analysis as the chosen task to verify how far in advance the generated storylines find an event before its first published occurrence in the news. And Section 5.4 adds similar experiments as the first, but with other datasets for greater topic variety. To begin, the general experiment setup is given below.

5.1 Experiment setup

To demonstrate various insights that can be garnered from spatio-temporal storytelling, the experiments are decomposed into three parts as listed in Table 2. The tasks are related to

Table 2 Methodology and data specification of the experiments

Task	Nature of Events	Time Span	Number of Storylines	Measure	Validation Set
Event Summarization	Boston Marathon Bombings	Mar 25 - Apr 30 2013	4,500,000	number of entity matches	*TREC-KBA*
 ingest data STS other approaches generate storylines generate summaries entity matching					
Event Association	Boston Marathon Bombings	Mar 25 - Apr 30 2013	4,500,000	lead time	*TREC-KBA*
 ingest data STS generate storylines lead time location / event identification					
Event & Location Identification	Financial, elections, *drug abuse*	2009, 2011, 2012, 2013	5,500,000	event/location matches	*EMBERS, GDELT DAWN*
 ingest tweets STS other approaches generate storylines generate summaries event/location matching					

different events ranging from the Boston Marathon Bombings of April 15, 2013 to social issues in Latin America (employment, salary, illnesses, elections) to drug abuse in the United States. The first task demonstrates how storytelling can be used as a means of event summarization. The second task, event association, investigates if storytelling is able to capture the relatedness among events without knowing that an event will actually take place. The third task concerns the identification of events and their associated locations. For each part, the table shows a general flow of the steps taken to perform that task. And because each task involves different data, pertinent details are provided in the corresponding sections below.

Data specification The TREC-KBA and Microblog data (we designate both as “TREC-KBA” going forward) are two extensive collections of different documents that include microblogs, blogs, and news articles (among others) spanning the years of 2011 to 2013, as of this writing [31]. For these experiments, the data sources encompass the period between March 25 to Apr 30, 2013. This period was selected as it immediately covers the Boston Bombings of April 15, 2013, which is the case study in question (the data, however, is mostly composed of general topics, not just Boston Bombings content). The nature of the data reflects items of interest to many different communities, certainly to include intelligence analysts.

A subset of the TREC-KBA articles have been annotated by the TREC contributors: some of their entities are tagged with a type (e.g., location, time, person, organization, misc, etc.). Items in each document are also given a timestamp that is helpful in temporal analysis (each document can have several items, with each item being an article). Location names are sometimes available, though they are not geocoded, which must be done separately. One of the purposes of TREC-KBA is to strengthen entity knowledge by gathering extra features about those entities from available news sources and web files (among others document types). In this sense, TREC-KBA operates at the entity level, but not at the storyline level. For this study, we have utilized the TREC-KBA files to compose storylines using their entities and relationships. We have also used a subset of those annotated files for verifying time-ordered events, and entity types.

For added variety, the experiments also use three other datasets: EMBERS [11], an extensive repository of tweets mostly from Latin America, and related to social factors such as employment, salary, and election issues; GDELT [13], a world-wide database of events of various natures from where topics of financial aid, cooperation, mass violence, and terrorism investigation were selected; and DAWN [4], a collection of medical reports that detail hospitalization events of people who suffered drug reactions. It is important to note the following: all of these three datasets (EMBERS, GDELT, and DAWN) contain fields that can be used for validation of the experiments presented later. For example, whenever a record describes a certain development, there is also a specific field that denotes the event to which that development relates. In addition, there is also a separate field that lists the location(s) of that event. These locations sometimes are textual names, a point of interest (POI), a zip code, or a latitude and longitude. As will be seen in the experiments, summaries are generated by different methods with the above datasets. To evaluate whether the summaries are valid, we use the provided event and location fields to verify if the summaries reflect them, which is one of the metrics. For our purposes, therefore, the dataset itself provides their own ground truth.

Comparative methods For event summarization, the objective was to find out how well the proposed approach of *Spatio-temporal Storytelling* (STS) performed as a summarization tool. The choice of event summarization, as opposed to other potential methods such as pattern mining or clustering, was due to two reasons. First, event summarization does not depend on high frequencies of the same patterns to occur. It performs just as well on disparate (and many times rare) entities, which is a characteristic of the datasets applied in the tasks. Second, the spatial aspect of the proposed approaches requires the computation of distances, and events provide spatial differences that can be reasoned about. Currently, there is an extensive body of works related to text summarization [21] from where many options are available. Note that the objective was not to compare existing summarization approaches. Rather, it was to evaluate if spatio-temporal storytelling could be utilized as a summarization tool when the analyst was performing entity analysis.

Three methods were selected that allowed for an adequate mix of textual analysis, time reasoning, and synonymy that overlaps with this paper's proposed methods. The first approach, *SUMMALLTEXT* (denoted as *summ-text*) uses a variation of TF × IDF to compare storylines. It takes storylines as inputs (4.5 million generated from 2 million entities), the set of words in all storylines, and the desired number of records. The second approach, *SUMMTIMEINT* (*summ-time*), uses a similar technique, but segments the storylines in different time windows and does processing based on each time window. Its inputs are all the storylines (again, 4.5 million), a minimum activity threshold (to filter out segments with less than 10,000 storylines), and the desired time segment (1 hour). They are described by Chakrabarti in [2] using tweets as storylines. Both output the top n storylines of maximum score to represent summaries. The third approach, described by Medvet [20] and denoted as *EDCS-summ*, identifies highly-frequent words in storylines, builds a set of synonyms from them, and outputs the storylines for sets that are also highly frequent. They also accept the storylines as input, and apply time segmentation for which 1 hour is set.

For event association, an evaluation was performed on the same dataset as above to verify whether the proposed algorithms could point to an event before that event was published in the newswire. Event association is portrayed as one more potential use of spatio-temporal storytelling.

For event and location identification, the purpose was to evaluate how well each of the approaches could single out an event from each record and find the location(s) associated to that event.

Performance measures For the first task, the question to be answered is the following: how well can STS perform if it is utilized as a summarization approach? For example, assume that a given summarization tool generates 10 summaries. In addition, also assume that STS generates 8 out of those same 10 summaries using the same input data. The match between them is thus 80 %, and the difference is 20 %. Note that these percentages do not relay any information on which approach is the most robust. However, they do allow the analyst to be aware of the differences, and decide on his/her own which method to use. If the STS summaries perform to the analyst's satisfaction, it would allow the analyst to use STS as a summarization tool instead of having to acquire an extra technology just for summarization. To be considered a match, two summaries must share a minimum number of entities as well as a location. The experiments evaluate matches under 10, 30, and 60 entities. Thus, Match $= \frac{|matching\ summaries\ of\ size\ x|}{|all\ summaries\ of\ size\ x|}$ is the performance measure. Note that summaries must have the same size in terms of entities, and that matches have been relaxed: a "protest" is considered the same as a "demonstration" or a "rally". This is done according to Wordnet [35]. Similarly, locations are also coalesced as long as one place geographically encloses the other (e.g. "Boston" and "Faneuil Hall MarketPlace"), which can be determined with geocoding tools [6, 30]. The reader should be aware that this definition makes sense for spatio-temporal storytelling. However, for other domains of application, relatedness might make more sense with different criteria, such as cosine similarity, among others. For simplicity of discussion, these experiments are limited to one location at a time.

For event association, the same storylines as above were used. Those storylines were then compared to a subset of the *TREC-KBA Corpora*, which had not been used previously (dated later than the previous files, and referred here as the "unseen files"). The performance measure, *lead time*, is simply the time difference between when a storyline was generated (i.e., the latest timestamp of any document used in the generation of that storyline) and the earliest occurrence of the events in those storylines in the *TREC-KBA* unseen files. For example, assume that storyline A was generated with data up to April 20, having such entities as a "protest", a "demonstration", "injury" or similar items in Boston. If on April 22, someone was reported injured in a protest in Boston (event B) as shown in one of the unseen files, than storyline A has been linked to event B with a 2-day lead time. Please note that storyline A does not predict event B. It only provides signals of things that could happen, which could be helpful to an analyst.

For event and location identification, the task is simple: if the data record details a "protest in Mexico City" and the output of the approach also makes any mention of "Mexico City" (or a nearby location) and a "protest" (or a similar concept such as "demonstration", as determined by Wordnet [35]), then a match has been made. The percentage of successful matches is then the evaluation criterion.

5.2 Comparison of event summarization approaches on the Boston Marathon Bombings (2013)

In this subsection, the three event summarization approaches mentioned in the experiment setup (Section 5.1) are used to evaluate the proposed method, spatio-temporal storytelling. One line of research complementary to this work, but which often does not include spatial *storytelling*, is event detection, which is left for future work, and indicated to the reader for further consideration [16, 33].

Table 3 lists a set of 5 locations, labeled $E1$ through $E5$, that were used as storytelling entrypoints. 2 million entities were obtained from a subset of the TREC-KBA Corpora and, using each of the four methods, 4.5 million summaries were generated (in the case of STS, a summary is equal to a storyline, while for the other approaches, it is several tweets). The dataset had no bounds on geographic coverage (could include any location in the world), but was guaranteed to have instances of the locations in the table. The comparative methods were: *STS*, which is the proposed work; *Summ-Text* [2], a cosine similarity variant of summaries; *Summ-Time* [2], a cosine variation with time-based segments; and

Table 3 Match comparison between Spatio-Temporal Storytelling (STS) and summarization of three different methods: *Baseline Summ-Text*, *Baseline Summ-Time*, and *EDCS-Summ*

	Summ-Text			Summ-Time			EDCS-Summ		
Event	10	30	60	10	30	60	10	30	60
E1-Boston	**0.78**	**0.59**	0.42	0.77	0.51	0.42	0.64	0.35	**0.57**
E2-Cambridge	0.61	0.34	**0.45**	0.53	0.36	0.29	**0.65**	**0.71**	0.28
E3-Quincy	0.52	0.38	0.57	0.69	**0.40**	**0.60**	**0.83**	0.34	0.39
E4-Newton	**0.76**	0.54	0.70	0.37	**0.80**	0.40	0.62	0.79	**0.76**
E5-Somerville	0.57	0.49	0.48	0.60	**0.67**	**0.61**	**0.80**	0.66	0.58

Summaries are aggregated by minimum number of matches of 10, 30, and 60 entities for a given location. The values in the table indicate the percentage of the summaries in the three approaches that are also included in STS. Highest values for each row and group are shown in bold

EDCS-Summ [20], which uses segmentation applied to synonym sets. The output summaries from each of the three methods were then compared against the output of STS. In the table, the higher the percentage, the better STS reflects the summarization method.

The way to interpret the table, exemplified for row 1, is as follows. From the initial 4.5 million summaries that were generated, all of the ones with maximum size of 200 entities were gathered. The summaries from STS were then compared to the summaries of the other three approaches. The question then becomes: what percentage of the summaries in STS matches summaries in the other three approaches by at least 10, 30, 60 entities, as well as by one or more locations? In other words, this task evaluated whether STS would be strong enough to serve as a summarization method in case the storytelling analyst did not have a fully-dedicated summarization tool to use.

For example, if one output of *Summ-Text* were “Make-shift memorial honors Boston Marathon victims”, and if *STS* output “Memorial dedicated to Boston Marathon victims”, then these two summaries would be considered a match on three entities and one location: “memorial”, “honors and dedicated”, and “marathon victims”, in addition to the location (Boston). As mentioned earlier, matches such as *honors* and *dedicated* are relaxed by *WordNet*. The table then lists other locations related to the Boston bombings that may be useful to an analyst.

Discussion At first glance, one can notice the fairly high variation of values across all approaches. This is especially true for summaries with 60-entity matches under *EDCS-Summ*, where matches go from 28 % to 76 %. High variation is also observed with *Summ-Time* in the range of 29 % to 61 %, but slightly less with *Summ-Text*. This high variation is not unexpected. In general, matching 60 entities at a time is very challenging, given the extremely unstructured characteristic of TREC-KBA data. Under 30 entities, there are generally more instances of matches than 60, though this could mean a lower level of reading coherence (the stories tend to look “stranger”). The strength of *STS* is shown in E4 (Newton), which displays very high match percentages of all sizes for all three approaches (76 %, 80 %, 76 %). *STS* performed well when the summaries themselves were not very lengthy (we show for 200 entities), matching on 10 and 30 entities. But not nearly as well on longer summaries, especially when trying to match 60 entities. One would expect that the longer the summaries, the more matches would occur, since a higher number of entities would be included. This is often quite misleading: it is true that longer storylines have the potential for more matches, but may include entities that are rare, having the opposite effect of preventing matches.

The *Summm-Time* approach clusters events based on time intervals, disregarding the clusters where events are not highly frequent. It explains its low matching on $E2$ (Cambridge) regardless of the number of entities in the storylines (Cambridge was significantly less frequent than other nearby locations). $E1$ (Boston) and $E4$ (Newton), on the other hand, were the most common occurrence of places with described events, which boosted their matchings. In addition, matching levels varied considerably for different reasons: *STS* can correctly capture people and organizations as highly-connected entities according to their *ConceptRank* measure. It was also helped by the fact that many locations mentioned in TREC-KBA were already inside Boston, such as street names, and its neighborhoods (Beacon Hill, Dorchester, West Roxbury). They aided in the location matches. But, at times, the other approaches generate summaries with unrelated locations, such as the mention of a “Cleveland” marathon. This makes STS miss the match. In addition, *Summ-Text* and *EDCS-Summ* generally suffered under storylines of 30 entities because many of the events were not always accompanied by specific locations names (these two approaches do not account for spatial operations, and thus locations names must match).

It must be noted that the fourth technique, *EDCS-Summ*, is interesting for its use of a dictionary approach to identify events. While *STS* also used one (WordNet), the other two approaches did not. Thus, both *EDCS-Summ* and *STS* could merge “attack” with “assault” or “aggression”, among other terms. This feature explains why *STS* appeared to come closer to *EDCS-Summ* than to the other two approaches. It was observed, however, that *Summ-Text* did display good stability, that is, the low-to-high range of matching was in general less than the other approaches, which may benefit applications that require predictability.

STS is a spatial technique in which places are regarded as geocodes (i.e., latitude and longitude coordinates), not plain keywords. In essence, this has the effect of capturing a wider variation of events across many areas, regardless of how they are described in the dataset. These results are encouraging for three reasons: they reinforce the importance of the spatial aspect which the other methods do not target; they indicate that the other methods could use the output of our approach (storylines) as the input to theirs in order to incorporate the spatial contribution; they confirm our initial claim that storylines can be a valuable tool in many different activities, summarization being a case in point.

The main goal of summarization is to capture essential ideas from the underlying text while disregarding unrelated points, what one would call noise. To this end, another claim can be made. Spatio-temporal storytelling is able to effectively capture two facets of the underlying data: the important locations and relevant events in the input data. Take, for example, Table 4 which lists a set of 20 short summaries (S1–S20), five for each of the comparative methods. *STS* shows five storylines (S1–S5) which captures four locations related to the topic of discussion (the Boston Marathon Bombings). These locations are Boston, Vassar Street, Copley Square, and Trinity Place. The other approaches not only find less locations, but some of them are unrelated, such as Manhattan. The reason *STS* is able to find a more coherent set of locations has to do with *Ripley’s K* function. It helps localize the investigated entities in short radii, which prevents far-away entities (such as the ones in Manhattan) from showing up. In addition, *STS* is also able to identify relevant events, such as “raid”, “explosion”, “destruction”, and “competition”. This is because of *ConceptRank* which promotes these highly-connected entities into the storylines. The other approaches are also able to identify some of the same events, but many times do so with less success.

Table 4 Illustrative summaries (S1–S20) for the four comparative methods

Comparative methods	Summaries	Locations found	Events identified
STS†	S1 - boston feds raid apartment in Vassar street seeking bombings attacked by explosive devices landed Copley Square and Trinity Pl. S2 - marathon bombs release debris mark pavement destroy sidewalk boylston street. S3 - Nantucket brothers damage legs boston running competition. S4 - people running boston marathon devise theories explain bombings. S5 - boston marathon victim identified woman krystle campbell.	**4** (Boston, Vassar Street, Copley Square, Trinity Place)	**5** (raid, bombings, explosion, destruction, competition)
Summ-Text	S6 - spirited elder becomes boston man victim S7 - 78 old runner behind boston photo. S8 - mayors menino giuliani praise brave first responders in boston. S9 - lawmakers seek answers boston 224321554 politics. S10 - obama evil boston bombings identified act terror.	**2** (boston, logan airport)	**3** (response, terror, removal)
Summ-Time	S11 - passengers removed ual flight logan airport sector. S12 - thatcher funeral see security tight massachusetts. S13 - boston marathon explosions right wing terrorists al-qaeda extremists. S14 - boston bombs first pictures devices released. S15 - boston marathon headquarters locked down blasts heard.	**2** (massachusetts, boston)	**3** (funeral, explosion, lock down)
EDCS-Summ	S16 - marathon explosions boston city mourning. S17 - boston marathon explosions man roof speculation twitter picture mystery. S18 - winner marathon boston suspended celebration time. S19 - vigil endemic explosion Brookline avenue recorded S20 - downtown Manhattan finance bank protect fight.	**2** (manhattan, brookline)	**3** celebration, explosion, fight

The table shows the number of events and locations that each approach is able to identify in their storylines.

†Proposed approach

5.3 Event association in the Boston Marathon Bombings (2013)

Storylines can be useful in many different applications. This section takes a lightweight view of *association analysis* and discusses how storylines are able to identify real-world developments before they are published in the news. The goal of this task is to generate storylines, identify which real-world stories they relate to, and determine how far in advance the storyline can be linked to the real event.

The dataset is the same as previously explained. The targeted events are related to the Boston Marathon Bombings of April 15, 2013, which provoked a myriad of events in the Boston area. At this stage, the strategy is the following: generate storylines using a subset of TREC-KBA data and validate the events identified in those storylines using a different subset of TREC-KBA data that has a later date. In other words, if a storyline is generated using TREC-KBA files up to April 18, than validation is done on TREC-KBA files dated April 19 or later. If a storyline points to a certain event that was published at a later date than the storyline's generation date, it indicates that a valid association was found. A valid association means that they match on at least two entities (for storylines of max 10 entities) plus one location (longer stories are possible, but not practical to display). This verification process has been done by searching the contents in the storylines (entities and locations) in the TREC-KBA files.

Table 5 shows a sample of 10 such events that are used for discussion. Each *TREC-KBA event* has an associated *reported by* source, an *event location*, and *published* date. For each event, the table shows a short *generated storyline*, which was generated by the proposed algorithms from source TREC-KBA files. The *generated date* of the *generated storyline* is the timestamp of the most recent file used as input data. In the *generated storyline*, entities are bolded in uppercase, relationships are in lowercase. The *lead time* is the time difference to the *published* date. The starting location is *Boston* from where we consider a radius of 50 km that includes nearby neighborhoods as shown in Fig. 11.

Discussion Item 1 has a *generated storyline* of four entities (*FBI*, *SUSPECT*, *MARATHON*, *BACKPACK*). These four entities are the ones of highest *ConceptRank*, and thus selected for the storyline. The relationships (*investigate*, *walk*, *carrying*) are the most frequent ones between the adjoining entities. Note that storylines do not reflect stylized English language. Because they are linked based on spatial connectivity and time order, grammar rules cannot be easily enforced, though they often come out in an intelligible format. The *generated storyline* closely resembles the associated *TREC-KBA event*: semantically, they have similar entities, such as "FBI" and "authorities" or "investigate" and "question", indicating that an individual (the "Saudi national") has been deemed a person of interest. Both the storyline and the TREC-KBA event relay a similar message. All source files that helped generate this storyline were dated April 15, 2013. The FBI investigation, however, was published by *cbsnews.com* on April 16, 2013,[1] indicating that the storyline could be associated to the TREC-KBA 1 day in advance.

While the storyline of item 1 relates to Cambridge, item 2 takes place in Boston, which has different latitude and longitude. Thus, they are considered different locations. At first glance, the *generated storyline* bears little relatedness to the *TREC-KBA event*. They are strongly connected, however, in two manners: the semantic closeness from the "conflicting"

[1] In the real world, it is possible another news source may have published this event even earlier. However, only the sources contained in the input files are considered here.

Table 5 Ten events reported in the TREC-KBA source files related to the Boston Marathon Bombings

	TREC-KBA event	Reported by	Event location	Published	Generated storyline	Generated date	Lead time
1	Authorities in Cambridge question Saudi National in Boston bombings attack.	cbsnews.com	Cambridge	Apr-16-2013	**FBI** investigate **SUSPECT** walk **MARATHON** carrying **BACKPACK**.	APR-15-2013	1 day
2	Conflicting reports about possible arrest in the Boston Marathon bombing.	theoakland press.com	Boston	Apr-17-2013	**MAN** held **FBI** suspicious **BEHAVIOR** conflicting **WITNESS**.	Apr-16-2013	1 day
3	Bomb suspect identified on security video at Bowdoin Station.	boston globe.com	Bowdoin St.	Apr-18-2013	**PERSON** appear **CAMERA** show **TEENAGERS** walk **SUBWAY**.	Apr-16-2013	2 days
4	Boston bombs said to be made from pressure cookers.	starbea con.com	Dorchester	Apr-16-2013	**DEVICE** explode **METAL BOX** found **SHOP** identifed **BOMBS**.	Apr-15-2013	1 day
5	Official video shows suspect walking down Boyslton St. with bag.	usatoday.com	Boylston St	Apr-18-2013	**PERSON** carry **BAG** left **SIDEWALK** along **BOSTON MARATHON** reach **FINISH LINE**.	Apr-17-2013	1 day
6	Investigators poring over photos and video from the Boston Marathon bombing of a man dropping off a bag at the scene of the one of the blasts.	cleveland.com	Boston	Apr-18-2013	**DEVICE** blast **MARATHON** investigated **WOMAN** shown **PHOTO** display **video**.	Apr-16-2013	2 days
7	Recently married couple injured in Boston bombing.	news.yahoo.com	Boston	Apr-17-2013	**TRAGEDY** affect **WEDDING** plans **RUNNER** injured **BLAST** treated **AMBULANCE**.	Apr-16-2013	1 day

Table 5 (continued)

8	Feds deny reports of suspect in custody at Dedham St. police department.	mercury news.com	Dedham St.	Apr-17-2013	**JOURNALIST** write **SUSPECT** identified **POLICE** held **STATION**.	Apr-15-2013	2 days
	TREC-KBA event	Reported by	Event location	Published	Generated storyline	Generated date	Lead time
9	Bombing arrest imminent after suspect was idd in Massachusetts.	mercury news.com	Massachu- setts	Apr-19-2013	**ARREST** made **CITY** loses **COLLIER** work **LAW ENFORCEMENT** identify **PEDESTRIAN**.	Apr-18-2013	1 day
10	Fund set up for mother and daughter injured during Boston Marathon bombings and treated at Boston Medical Center	nydaily news.com	Boston Medical Center	Apr-17-2013	**HOSPITAL** receive **DONATION** participate **FUNDRAISER** help **VICTIM**.	Apr-16-2013	1 day

For each event, the table shows its *Reported By* source, *Event Location*, and date it was first *Published* in the news. Storylines are then generated using only data that precedes the TREC-KBA event. The *Lead Time* column shows that the storylines have a *Generated Date* before the news article was published

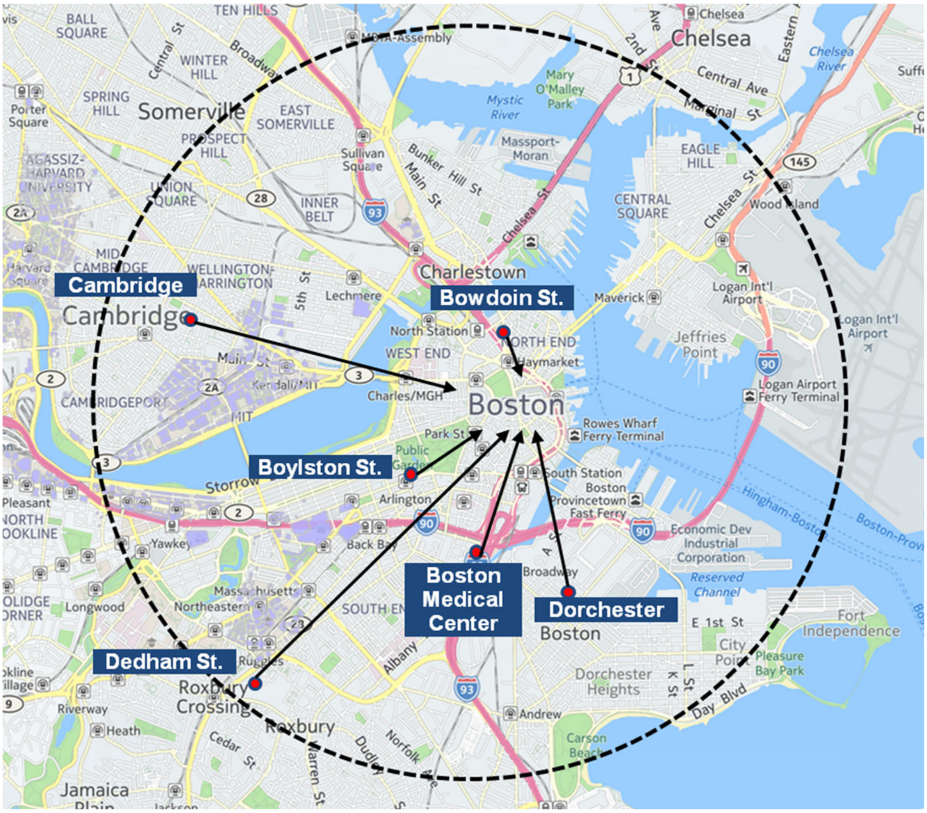


Fig. 11 Spatial propagation of developments due to the *Boston Marathon Bombings*. Starting from the city of Boston, related events listed in Table 5 are observed in nearby areas. The map shows 6 of approximately 1,200 affected locations. Locations are approximate

relationships, as well as by location. All source files that helped generate this storyline were verified to have a location inside *Boston*. The *lead time* is one day, showing that the algorithm was able to reconcile the event prior to when it was reported by the media. This example underscores the importance of location, which would otherwise make this linking difficult to justify.

The remaining items from 3 to 10 are also highly-dependent on location. Note that none of those *generated storylines* (with the exception of number 5) have a location entity explicitly stated. However, their generating files do contain at least one metadata location that matches the location of the *TREC-KBA event*, and a timestamp that closely pre-dates the event's published date. This is particularly interesting in the case of item 10, whose *TREC-KBA event* is shown at *Boston Medical Center*, but whose *generated storyline* does not reflect that location, mentioning only that is a hospital. In fact, that storyline's generating files have a latitude/longitude that closely matches that of the *Boston Medical Center*. Also worth mentioning is the fact that there are a few entities very popular across the generating files, and as a consequence, appear commonly in the storylines. Three of them are *MARATHON*, *BLAST*, and *FBI*, which are commonly observed in *Boston* itself, and *Massachusetts* for their large areas. The prominence of these entities as part of the storylines is significant for a simple reason: it indicates that the spatio-temporal methodology is able to find storylines about events related to the Marathon Bombings in Boston using location as a decisive factor. Even though *Boston* is the most prominent area, the algorithm also identifies other important locations where events occurred with similar entities, such as the ones in *Bowdoin St* (item 3), *Dorchester* (item 4), and *Dedham St.* (item 8).

5.4 Event and location analysis on other datasets

In the previous subsections, experiments utilized a single dataset (Boston Marathon Bombings) for consistency. In this subsection, other datasets are used to evaluate the algorithms' performance under higher data variety.

Just as previously done, this stage first uses the same approaches as before to generate summaries. But instead of showing the summaries, these experiments detail the success percentage of each approach in generating summaries that contain the event(s) and location(s) from the input data. For example, if 10,000 input records mentioned "cooperation" as an event between "Turkey" and "Afghanistan", and the output summary also contained those three items (relaxed by Wordnet synonyms as before), then that output summary would be considered a successful match for all 10,000 records. The percentage of records that matched over all records is thus the measure in question (every record is matchable, i.e., every record has an event and at least one location).

Figure 12 displays four plots from the EMBERS dataset, which represents tweets about various topics from several countries of Latin America. Given an increasing number of records, the goal was to verify if STS could perform summarization at a level comparable to the other approaches, which were specifically designed to generate summaries. If so, the analyst might have been able to accomplish the summarization task with just STS, saving the extra time and money expenditure in acquiring other tools. Figure 12A was generated with up to 1,000,000 tweets at the high range. The plot indicates the following: starting with 250,000 tweets, STS storylines generated 97 % successful matches with the events and

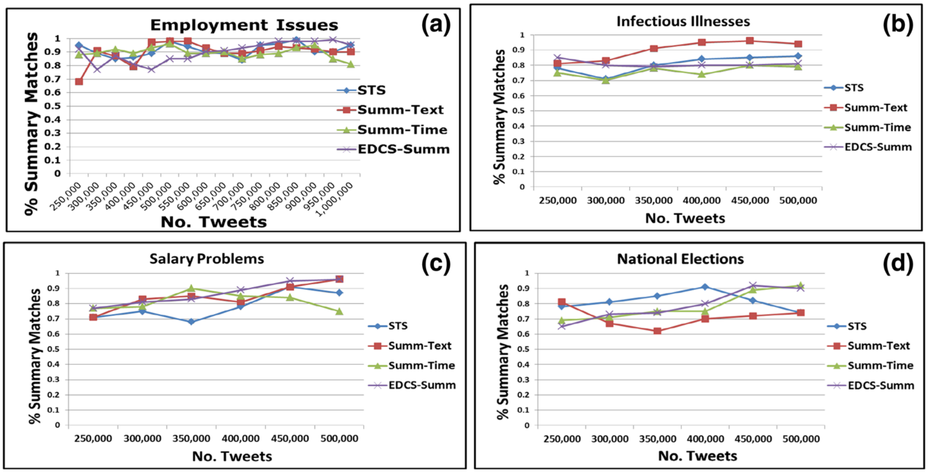


Fig. 12 Comparison of four summarization approaches on the EMBERS dataset. **A** events related to employment issues. **B** Events with sources related to infectious illnesses. **C** Events related to salary and finance. **D** Events related to elections. The plots depict the percentage of summaries correctly generated by each approach for the different event types. The four plots represent four distinct datasets

locations of the input data (meaning the storylines contained the event name and location names of the input tweets). On that same datapoint (250,000) the other approaches had a slightly lower success rate (96 % and 88 % for Summ-Text and EDCS-Summ respectively) and quite lower for Summ-Text (68 %). Looking further into the plot, there exists a high variation of results. Different methods performed better at different points. Summ-Text, for instance, started low, but ended higher, with its best performance in the range of 400,000 to 500,000 input records. STS started high (97 %) and ended high (96 %), but showed several lows in between 300,000 and 400,000. Based on events related to employment issues, there is no clear winner: all four approaches were comparable in terms of summarization potential. It reinforces the idea that STS, even though it was not designed for summarization, is able to reach very similar results provided by the other tools, and at times better. In the case of events related to infectious illnesses (Fig. 12B), and salary problems (Fig. 12C), the distinction among the methods is somewhat better defined. In B, and using up to 500,000 records, Summ-Text demonstrated generally higher match percentage than the others, with STS lower, but still better than EDCS-Summ and Summ-Time. STS fared better with the national elections events (Fig. 12D), outperforming the other approaches for most of the data. Further below, some of the reasons for the high variation in results will be given.

Figure 13 represents experiments done on the GDELT dataset. In contrast to tweets, which are highly unstructured and very noisy, GDELT records are generally clean, with a combination of textual description as well as other structured fields such as time, location, and entities (among others). This set of experiments differs from the previous on the data sources (semi-structured instead of unstructured), the nature of the data (events that are not necessarily violent, such as cooperation, investigation, and aid), and match relaxation (a match was considered successful if any location was identified, not all of them, along with the event). With a few exceptions, all plots show higher match performance for STS than the other approaches. Under “financial cooperation” and “investigate” the higher difference in favor or STS is more accentuated. Under “provide aid” and “mass violence”, STS has

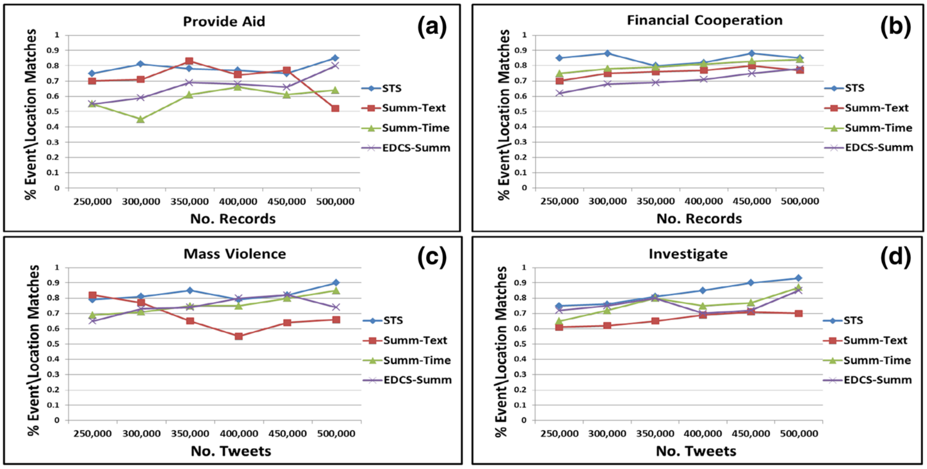


Fig. 13 Comparison of four summarization approaches on the GDELT dataset. Each plot represents a category of events related to **A** aid transactions between organizations and countries. **B** Financial cooperation among different nations. **C** Mass violence events across the Middle East and Africa. **D** Investigation related to acts of terrorism. The plots show the percentage of summaries that correctly identified the event and its location. The four plots represent four distinct datasets

generally better performance with a few dips below Summ-Text and EDCS-Summ. To explain why the above performance levels fluctuate (at times, significantly), the following factors can be stated. First, the nature of the data, whether it is financial, social, or political, appears to have very little influence on the number of matches that any of the four approaches make. Other disparate types, such as sports or real estate, would have experienced just as similar results. Factors that appear to be more relevant, and expectedly so, are the cleanliness of the data (which negatively affects identification of events and locations for all approaches), and the size of the text. Small textual sources (twitter-size) works best for Summ-Time, Summ-Text, EDCS-Summ, while longer text (news articles, reports) can benefit STS. The most impacting factor is that of location: while STS always looks for names of locations, and can disambiguate them by latitude and longitude, the other approaches do not natively include that capability. Thus, in many instances, the other approaches fail to include the location in their summaries, which causes many matches to be missed (this was also observed with timestamps, though we did not consider time as a matching criterion here). It is important to notice that events are simply textual concepts, such as “riot”. While STS and EDCS-Summ are embedded with concept expansion (so to equate “riot” with “fight” or “battle” via Wordnet), the other approaches do not have that capability. We, however, added that feature to them. Otherwise, their ability to find events of similar natures that are described differently would have lowered in the magnitude of 40 %, making the comparison to STS doubtful. STS also performs more robustly than the other methods when events are described with many locations (e.g., a celebration that takes place in many different cities at the same time). STS will commonly list all locations as part of the storyline, while the other approaches may list none or one at best. The underlying reason is that when the input records mention location names with high frequency, they impact TFxIDF (used by some of the other approaches), which tends to lower the importance of a concept that is repeated over and over.

The plots of Fig. 14 are presented under a very different dataset (DAWN) coming from U.S. hospitals. Whenever a patient checks into an emergency room because of drug reaction(s), a report is filed describing the drug name(s), comments of the situation, description of the reactions, and demographics about the patient. This dataset is interesting because of the high number of very uncommon terms related to drugs (“CIPROFLOXACIN”, “PHENYLEPHRINE”, “VORICONAZOLE”, “DICHLOROBENZENE”, etc.). The event in this case is the use of the drug listed on the report, and the location of the event is the location of the hospital, both of which are given by the dataset for validation purposes.

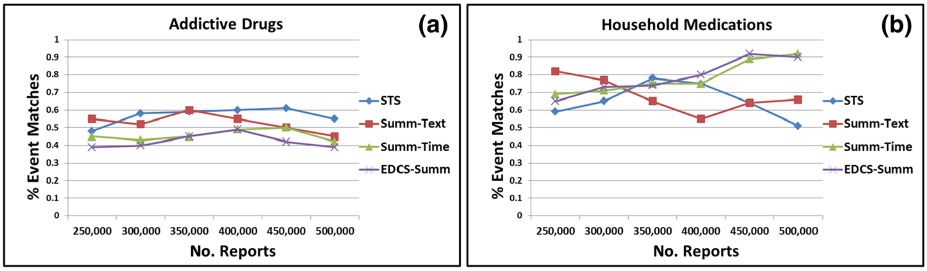


Fig. 14 Comparison of four summarization approaches on the DAWN dataset. Each plot represents an emergency room event (hospitalization) that occurred as a consequence of the use of **A** addictive drugs. **B** common household medications. The plots show the percentage of summaries that correctly identified the name of the drug that caused the hospitalization. The two plots represent two distinct datasets

In this sense, events here are medical, and not political or social, as was for the previous experiments. The uncommonality of drug names highly favors TFxIDF approaches, which places higher importance on infrequent terms. However, it makes Wordnet query expansion unusable, since these drug terms do not generate synonyms, and impacts STS unfavorably. In addition, the locations in this dataset are represented simply as zip codes (due to the patient's privacy, no addresses or better geolocation are provided). The effects can be seen in Fig. 14A, where percentage matches are in general much lower than in the previous plots for all four approaches. The matches are never above 60 % for addictive drugs (uncommon sophisticated names), but higher for household medications (Fig. 14B) which tend to be more frequent (e.g., "ASPIRIN","ALCOHOL", "IBUPROFEN"). Notice, however, that STS's performance is still strong under addictive drugs, and often comparable to the other methods under household medications. What saves STS in this case are the zip codes (which are often ignored by the other summarization methods), while drug names on their own are a negative factor for STS. The other approaches behave in an opposite manner, benefiting from the drug names, but suffering from zip codes.

Lessons learned In general, spatio-temporal storytelling is sensitive to many different factors that can affect results both positively and negatively:

- **Scope targeting**: targeted events which are very prominent in one area (e.g., Boston) are better candidates for spatio-temporal storytelling than events that span wide regions evenly. This gives us the first lesson: storytelling benefits when the scope is targeted. In other words, the examined topic should be specific enough to a region in order to maximize preciseness.
- **Match relaxation**: when entities are matched in a loose manner (e.g., "riot = fight"), such as with a dictionary or ontology, there is a significant increase in one's ability to find related events. Making the match more strict prevents understandability. The second lesson is that relaxing the matches tends to find very coherent storylines, which can also be helped by increasing the number of investigated locations.
- **Region granularity**: these experiments consider a radius of 50 km from downtown Boston. It should be apparent that a shorter or longer radius can have different connotations: for country-wide applications, it could mean data explosion. For geographically-constrained applications, it could mean missing important data that resides far away. For example, elections across the country would make sense when viewed in large areas, whereas a terrorism act makes more sense when treated locally. Therefore, the third lesson is that the length of the radii may require careful consideration for spatio-temporal storytelling to be practical.
- **Data variation**: data volumes and variation are important factors that bring up the fourth lesson: low data volumes and poor variation creates storylines that lack meaning and appear disconnected from real events.

5.5 Tasks and execution times

The purpose of this subsection is to provide a quick snapshot of the elapsed times that were observed during the execution of the experiments. Table 6 lists the eight major tasks that encompass spatio-temporal storytelling. The tasks were run in two different ways: geocoding, distance calculation between entities, and time stamping were conducted under *Mapreduce*[18] in a cluster of 5 machines (Intel Atom 1.66GHz 4 GB RAM); the other tasks were conducted in a standalone machine (Intel Xeon 3.33GHz 24 GB RAM). A few important notes:

- The values in the table represent elapsed times in minutes to process the given number of entities. Thus, for entity extraction, it took 151 minutes to process 500 K entities, while 1,000 K entities took 310 minutes.
- Tasks were conducted independent of one another. In other words, processing 1,000 K entities did not use previous results from processing 500 K. Clearly, these two steps could have been made cumulative for greater efficiency. However, it was the intent of this study to measure tasks in a greedy manner, and allow for a better understanding of what optimizations could be done.
- Data parsing in general (tasks T1, T2, T3, T4, and T5) is computationally intensive. This is particularly true in the case of relationship extractions that come from noisy text (e.g., determining if two people "talk"). For example, it took 280 minutes to extract 500 K entities, and 2,265 minutes (about 38 hours) for 2,000 K entities. This progression can be linear at times, but very frequently deteriorates to much worse levels. Relationships that come in an API, on the other hand, such as a *Twitter* user that "follows" another one, are not as significant a cost factor.
- While the parsing tasks (T1–T5) can be computationally costly, the algorithmic parts (T6–T8) are less so. This is because the algorithmic steps already benefit from the earlier pre-processing. Ripley's K function, for example, relies on the computation of distances of T4 in order to find a good radius of investigation. ConceptRank, on the other hand, requires counting every type of relationship (inbound and outbound) for every entity. This is by large the heaviest algorithmic task, and should be done in a distributed platform for high data volumes in excess of what Table 6 displays.
- Each of the 8 tasks of Table 6 have subtasks which are not listed, but included as part of the timings. When an entity is extracted, for example, it is saved both in a database (for spatial indexing and distance calculations) and in a graph (to query for storylines between entrypoints and end nodes). In this process, it is also checked to avoid duplicates and meaningless formats (such as numbers), and examined for

Table 6 Elapsed time observed to generate storylines from raw data

Task	500 K†	1,000 K	1,500 K	2,000 K
T1 - entity extraction	151 ‡	310	743	1,875
T2 - relationship extraction	280	643	1,143	2,265
T3 - geocoding	90	252	400	772
T4 - distance calculation between entities	34	51	89	412
T5 - time stamping	12	44	70	90
T6 - optimal radius calculation	48	71	190	502
T7 - conceptRank	41	99	274	611
T8 - storyline generation	656	1,470	2,909	6,527

The values in the table represent time in minutes required to process the given task from 500,000 to 2 million entities

†Number of entities

‡Time in minutes

non-English characters. Another problem is *retweets*, which cause duplication of entities and relationships, and are discarded in these experiments.

- Storyline generation (T8) is simply the time addition of the previous tasks (T1-T7). It can be seen that most of the time is spent on pre-processing, and less on running the algorithms.
- It should be obvious that the numbers of Table 6 are only representative of this paper's data and experiments. Results will vary significantly with more data, different hardware, and the introduction of optimization steps.

5.6 Experiment summary

The experiments in Section 5.2 demonstrated the potentially-high applicability of spatio-temporal storytelling, exemplified in an event summarization case study. *STS* yielded high summarization matching as compared to existing methods (up to 80 %) on mostly-unstructured documents (i.e, *TREC-KBA Corpora*). Rather than relying on textual content, *STS* introspects entities that are spatio-temporally tagged so to identify the ones with high levels of connectivity. Those entities are targeted for storyline generation regardless of how they are described in the underlying data source. In addition, *ConceptRank* helped differentiate the important relationships from the less relevant ones. This is an essential contribution to intelligence analysis, which often faces large data volumes, but have little ability to automatically segregate important connections among millions of possibilities.

Spatio-temporal storytelling's ability to capture the underlying links among entities is complemented by its flexible method of temporal propagation. Analysis with time windows promotes coherent storylines, which has the potential to uncover developments before they materialize. This was shown in the experiments of Section 5.3, where a set of ten events related to the Boston Marathon Bombings of 2013 was identified. *STS* proved highly successful in associating events up to two days in advance of their publication in the news under a highly-noisy dataset.

Experiments on three extra datasets were presented in Section 5.4 to show that *STS* is capable of handling records of highly infrequent vocabulary (drug names) and poor geospatial resolutions (zip codes). It also pointed to the fact that the nature of data (political, social, medical) had little impact on the results, attesting to STS's generality, and its ability to performing other seemingly disparate tasks for which it was not designed.

6 Conclusion

In studying socio-political interactions from spatio-temporal propagation, this study has been able to generate dynamic real-world storylines that are of great significance to the intelligence community. Because spatial distribution is treated as an integral factor of the described algorithms, dense regions where storylines developed were identified. Further, this approach established time-coherent entity connections that otherwise might have been more challenging from purely textual approaches that do not consider the myriad locations as the ones affected by the *Boston Marathon Bombings* as well as financial, social, political, and medical events in Latin America, Middle East, and the United States. Ranking was devised based on different relationship types, and proved effective on ill-formed datasets. The experiments demonstrated a high potential for applicability in tasks such as summarization and association of current events, indicating the versatility of STS for intelligence analysis.

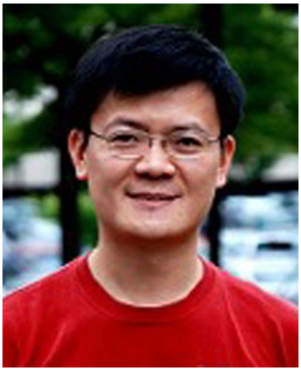
References

1. Brin S, Page L (1998) The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems 30:107–117
2. Chakrabarti D, Punera K (2011) Event summarization using tweets. In: Proceedings 6th AAAI international conference on weblogs and social media
3. Das D, Martins A (2007), A survey on automatic text summarization
4. Dawn (2009) Drug abuse warning network - u.s. dept of health and human services - inter-university consortium for political and social research ann arbor. mi - http://www.samhsa.gov/data/
5. Gong Y, Liu X (2001) Generic text summarization using relevance measure and latent semantic analysis. In: 24th international ACM SIGIR conference on research and development in information retrieval, pp 19–25
6. https://developers.google.com/maps/ (2015)
7. Groh G, Straub F, Koster B (2012) Spatio-temporal small worlds for decentralized information retrieval in social networking. In: ACM GIS'12, pp 418–421
8. Hossain MS, Andrews C, Ramakrishnan N, North C (2011) Helping intelligence analysts make connections. In: Workshop on scalable integration of analytics and visualization, AAAI '11, pp 22–31
9. Hossain MS, Butler P, Ramakrishnan N, Boedihardjo A (2012) Storytelling in entity networks to support intelligence analysts. In: Conference on knowledge discovery and data mining (KDD'12), pp 1375–1383
10. Hossain MS, Gresock J, Edmonds Y, Helm R, Potts M, Ramakrishnan N (2012) Connecting the dots between pubmed abstracts. Public Libr Sci (PLoS ONE) 7(1)
11. Iarpa - open source indicators program (osi) (2014). http://www.iarpa.gov/solicitations_osi.html
12. Kumar D, Ramakrishnan N, Helm RF, Potts M (2008) Algorithms for storytelling. IEEE TKDE 20(6):736–751
13. Leetaru K, Schrodt P (2013) Gdelt: global database of events, language, and tone, 1979–2014. In: Proceedings intl. studies assoc. annual conference (ISA)
14. Li C, Sun A, Datta A (2012) Twevent: segment-based event detection from tweets. In: Conference on information and knowledge management, pp 155–164
15. Li R, Lei KH, Khadiwala R, Chang K (2012) Tedas: a twitter-based event detection and analysis system. In: Proceedings 28th IEEE conference on data engineering (ICDE), pp 1273–1276
16. Li Z, Wang B, Li M, Ma WY (2005) A probabilistic model for retrospective news event detection. In: ACM SIGIR conference on research and development in information retrieval, SIGIR '05, pp 106–113
17. Lin D (2008) An information-theoretic definition of similarity. In: ICML '08, pp 296–304
18. http://hadoop.apache.org/ (2015)
19. Marcus A, Bernstein M, Badar O, Karger D, Madden S, Miller R (2011) Twitinfo: aggregating and visualizing microblogs for event exploration. In: ACM conference on human factors in computing systems (CHI)
20. Medvet E, Bartoli A (2012) Brand-related events detection, classification and summarization on twitter. In: Proceedings of the the 2012 IEEE/WIC/ACM international joint conferences on web intelligence and intelligent agent technology, WI-IAT '12, pp 297–302
21. Nenkova A, McKeown K (2011) Automatic summarization. Foundations and Trends in Information Retrieval 5(2):103–233
22. Petrovic S, Osborne M, McCreadie R, Macdonald C, Ounis I, Shrimpton L (2013) Can twitter replace newswire for breaking news? In: 7th international AAAI conference on weblogs and social media (ICWSM)
23. Radinsky K, Horvitz E (2013) Mining the web to predict future events. In: WSDM '13, pp 255–264
24. Ramakrishnan N, Kumar D, Mishra B, Potts M, Helm RF (2004) Turning cartwheels: an alternating algorithm for mining redescriptions. In: KDD '04, pp 266–275
25. Reed T, Gubbins K (1973) Applied statistical mechanics: thermodynamic and transport properties of fluids. Butterworth-Heinemann, Boston
26. Sakaki T, Okazaki M, Matsuo Y (2010) Earthquake shakes twitter users: Real-time event detection by social sensors. In: Proceedings of the 19th international conference on world wide web, WWW '10, pp 851–860
27. Shahaf D, Guestrin C (2010) Connecting the dots between news articles. In: ACM conf. on knowledge, discovery, and data mining (KDD '10), pp 745–770
28. Shahaf D, Guestrin C, Horvitz E (2012) Metro maps of science. In: Conference on knowledge discovery and data mining, KDD'12, pp 1122–1130
29. Shahaf D, Guestrin C, Horvitz E (2012) Trains of thought: generating information maps. In: World wide web conference, WWW'12, pp 899–908

30. http://www.alchemyapi.com/ (2013)
31. (2015). trec.nist.gov/data/kba.html/
32. Turner S (1994) The creative process: a computer model of storytelling and creativity. Psychology Press, pp 122–123
33. Vavliakis KN, Symeonidis AL, Mitkas PA (2013) Event identification in web social media through named entity recognition and topic modeling. Data Knowl Eng 88:1–24
34. Walther M, Kaisser M (2013) Geo-spatial event detection in the twitter stream. In: Advances in information retrieval, Lecture notes in computer science, vol 7814. Springer, Berlin Heidelberg, pp 356–367
35. (2013). http://wordnet.princeton.edu/

Raimundo Dos Santos received a Bachelor's Degree in Computer Science from the University of South Florida. He received a Master's and a PhD in Computer Science from Virginia Tech. He has published in several venues including ACM-GIS, IEEE-ICTAI, IJTAI, and Geoinformatica. His research focuses on semantic entity analysis and Spatial Data Management, including retrieval, exchange, and processing of information for Geographic Information Systems and location-based services. Other interests include data integration, graph mining, and analytical methods for storytelling.

Sumit Shah received a BS and MS in Computer Science from Virginia Tech. He has worked extensively in software engineering and systems architecture and is currently a PhD. candidate in the Department of Computer Science at Virginia Tech. He has published work at ACM GIS and other academic venues. His research focuses on large scale data mining, Big Data, information retrieval, and location-based services. Other interests include mobility and data visualization.



Arnold P. Boedihardjo received his BS degrees in Mathematics and Computer Science from Virginia Tech in 2001. He received his MS and PhD degrees in Computer Science from Virginia Tech in 2006 and 2010, respectively. He has published in various scholarly venues such as IEEE International Conference on Data Engineering, IEEE International Conference on Data Mining, ACM Conference on Information and Knowledge Management, Knowledge and Information Systems Journal, and IET Communications Journal. His research interests include data stream systems, spatial databases, information retrieval, optimizations, networking, and statistical learning. He is currently a research scientist at the U.S. Army

Feng Chen is an assistant professor at State University of New York at Albany. He received his B.S. from Hunan University, China, in 2001, M.S. degree from Beihang University, China, in 2004, and Ph.D. degree from Virginia Polytechnic Institute and State University in 2012, all in Computer Science. He has published 25 refereed articles in major data mining venues, including ACM-SIGKDD, ACM-CIKM, ACM-GIS, IEEE-ICDM, and IEEE-INFOCOM. He holds two U.S. patents on human activity analysis filed by IBM's T.J. Watson Research Center. His research interests are in the areas of statistical machine learning and data mining, with an emphasis on spatiotemporal analysis, social media analysis, and energy disaggregation.

Chang-Tien Lu received the MS degree in computer science from the Georgia Institute of Technology in 1996 and the PhD degree in computer science from the University of Minnesota in 2001. He is an associate professor in the Department of Computer Science, Virginia Polytechnic Institute and State University and is founding director of the Spatial Lab. He served as Program Co-Chair of the 18th IEEE International Conference on Tools with Artificial Intelligence in 2006, and General Co-Chair of the 20th IEEE International Conference on Tools with Artificial Intelligence in 2008 and 17th ACM International Conference on Advances in Geographic Information Systems in 2009. He is also serving as Vice Chair of the ACM Special Interest Group on Spatial Information (ACM SIGSPATIAL). His research interests include spatial databases, data mining, geographic information systems, and intelligent transportation systems.

Patrick Butler received his BS degrees in Physics and Computer Science from Virginia Tech in 2005. He received his PhD degree in Computer Science from Virginia Tech in 2014. He has published work in ACM-SIGKDD, ACM-TIST, SIAM-SDM, AAAI, IEEE-TDSC, VAST and other academic venues. His research interests include mining open source data sources such as twitter, temporal data mining, statistical learning, computer security, and visual analytics.

Naren Ramakrishnan is the Thomas L. Phillips Professor of Engineering at Virginia Tech. He directs the Discovery Analytics Center (DAC; http://dac.cs.vt.edu) at Virginia Tech, a university-wide effort that brings together researchers from computer science, statistics, mathematics, and electrical and computer engineering to tackle knowledge discovery problems in important areas of national interest, including intelligence analysis, sustainability, neuroscience, and systems biology. Ramakrishnan's research has been supported by NSF, DHS, NIH, NEH, DARPA, IARPA, ONR, General Motors, HP Labs, NEC Labs, and Advance Auto Parts. He serves on the editorial boards of IEEE Computer, Data Mining and Knowledge Discovery, and other journals. Ramakrishnan was an invited co-organizer of the NAE Frontiers of Engineering symposium in 2009. He is an ACM Distinguished Scientist (2009).