StoryPivot: Comparing and Contrasting Story Evolution

Anja Gruenheid† Theodoros Rekatsinas∗ Donald Kossmann† Divesh Srivastava†

1Systems Group
Department of Computer Science
ETH Zurich
{agruen,donalddk}@inf.ethz.ch

∗University of Maryland
thodrek@cs.umd.edu
divesh@research.att.com

ABSTRACT

As the world evolves around us, so does the digital coverage of it. Events of diverse types, associated with different actors and various locations, are continuously captured by multiple information sources such as news articles, blogs, social media etc. day by day. In the digital world, these events are represented through information snippets that contain information on the involved entities, a description of the event, when the event occurred etc. In our work, we observe that events and thus also their corresponding digital representation are often inter-connected, i.e., they form stories which represent evolving relationships between entities over time. Take as an example the plane crash in the Ukraine in July 2014 which involved multiple entities such as the Ukraine, Malaysia, and Russia and ranging in events from the actual crash to the incident investigation and the presentation of their findings. In this demonstration, we present a framework to detect evolving stories in event datasets over time. To resolve stories, we differentiate between story identification, the problem of connecting events over time within a source, and story alignment, the problem of integrating stories across sources. The goal of this demonstration is to show an interactive exploration of both these problems and how events can be dynamically interpreted and put into context. Furthermore, we present large-scale results from extensive experiments with real-world datasets that show the scalability and applicability of our techniques.

1. INTRODUCTION

Lately there has been an increasing interest in monitoring news media, blogs and social media from all over the world and extracting geo-referenced records that correspond to different events and interactions between diverse groups of people, international organizations, countries etc. The extracted event records are usually stored in repositories that get updated regularly as new extractions are obtained from a diverse collection of data sources. Example repositories of such extractions include GDELT [11] and EventRegistry [9] which are updated over fixed time intervals (e.g., daily).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 20XX ACM X-XXXXXX-X/XX/XX ...$15.00.

Extracted data is stored in a tuple format containing information about its origin, the type of the corresponding event, describing an activity linked with the event, the entities associated with the corresponding activity, a short description and a timestamp. An example of such a tuple is <New York Times, Accident, {Ukraine, Malaysian Airlines}, “Plane Crash”, 07/17/2014>.

Extracted pieces of information over a fixed time-window represent only a snapshot of what is happening in the world. Real-world events are often connected over time forming stories that represent evolving relationships between different entities. Recovering the evolution and the dynamics of news stories across time is of tremendous value in different application domains, ranging from trend detection to economic analysis, and has been a focus of recent research literature [2, 10]. Nevertheless, the majority of proposed approaches for story detection focus on identifying current and thus often mentioned stories in streaming news. Thus, they do not provide means for tracking the evolution and dynamics of stories even though this type of story tracking has applications in different domains. For example analysts rely on temporal patterns of event occurrences to discover supporting evidence and validate their hypotheses. The most common paradigm to consider is political scientists who rely on historical data to forecast political crises [15, 8] and conflicts [17]. A different application domain that is increasingly popular is that of financial analytics based on political event extractions [7, 6].

Another limitation of current systems is that they are oblivious to the presence of multiple sources and their story reporting perspectives. Data sources have different perspectives on stories because they report the same story with varying content and with varying levels of timeliness [18]. However, it has been recently shown that leveraging these individual source characteristics can lead to a significant accuracy improvement for difficult prediction tasks such as predicting civil unrest incidents [12]. Moreover, there has been an increasing interest in the database community to understand and exploit the characteristics of different data sources to either provide affordable analytics or improve the quality of integrated data [9, 16, 13].

In this demonstration, we present StoryPivot, a novel story detection framework for detecting stories involving different entities over time. StoryPivot allows users to explore the evolution and dynamics of stories but also to understand the perspectives of different data sources when reporting these stories. To address the limitations mentioned above, we decompose the story detection problem into two subproblems (1) story identification and (2) story alignment.
Story identification is the problem of finding those events that represent the same real-world story within a single source, and captures the individual story perspectives of each data source. Story alignment then integrates stories across multiple sources to enhance completeness and correctness. Due to the sheer number of available sources, one of the main challenges here is combining stories across data sources efficiently. In summary, we present the following features in our demonstration:

**Interactive Framework.** We provide an interactive interface for the users that allows them to examine the development of stories over time. It will provide visualization for both, story identification and story alignment, components of **StoryPivot**.

**Large-Scale Experiments.** To provide evidence of the scalability and efficiency of **StoryPivot**, we perform extensive experiments to be presented during the demo. Furthermore, the user can use the interactive visualizations to explore the results of these experiments as well.

The remainder of this demonstration description is structured as follows: First, we introduce the problem of story detection and a 2-phase mechanism to solve it in Section 2. It is applicable for a variety of use cases that we briefly describe in Section 3. Afterwards, we present the details on the technical demonstration in Section 4 and conclude this demonstration paper in Section 5.

2. **STORY EVOLUTION**

Identifying the evolution of stories requires detecting related events over multiple sources and over time and combining them into a single cohesive story. In this section, we present an overview of **StoryPivot** which detects the evolution and dynamics of stories through processing event data. Furthermore, we present ideas and techniques on how to address story identification and story alignment efficiently.

2.1 **Overview**

An overview of **StoryPivot** is illustrated in Figure 1. The input to our system corresponds to elemental units of information, called **information snippets** and may correspond to single events or small sets of events. Let $V$ be the set of all available information snippets. We make the following assumptions for the structure of these snippets: First, every snippet $v_i \in V$ is associated with a timestamp $t_i$ which records when the event(s) in the snippet occurred in the real-world. Second, it is associated with a data source $s_i$. Data sources may correspond to newspapers such as the New York Times or the Wall Street Journal but can encompass all digital media that provide event-based information. The problem of finding stories is thus to search $V$ for all snippets $v_j \in V$ that form a story $c_j$. To extract those snippets, any data extractor for web documents can be applied as shown in Figure 1(a). Our system does not require data snippets to follow a specific format. Instead their granularity is determined by the choice of extractor. The only requirement is that snippets follow a consistent format such as the tuple format introduced in Section 1.

Let $C$ be the set of all stories that can be extracted from the snippets in $V$. To process information snippets, we adopt a two-phase strategy which first determines the stories $C_i \subseteq C$ that can be found in a single source $s_i$. To compute these stories, $V$ is partitioned into subsets $V_i \subseteq V$ where subscript $i$ corresponds to the data source $s_i$. These sets of information snippets are then processed independently of each other to identify the stories within $s_i$. A visualization of this phase is shown in Figure 1(b). For every snippet $v_i^j \in V_i$, story identification proceeds as follows: It first determines the most likely story that $v_i^j$ can be matched to and assigns it to this story. If no such story exists, it generates a new story around this information snippet. Second, the story evolution depends on the evolution of the corresponding real-world events which have to be captured correctly. In that context, we observe that it is possible for stories to split into multiple substories or to merge into a bigger story. For example political and economic events were interwoven during the height of the Ukraine crisis while they started to separate after the situation had stabilized. Instead of using single pass story detection, we therefore construct stories within a data source incrementally. Once story identification for all available sources has been completed, we align stories across data sources. The alignment step is crucial for story detection for two reasons. First, it allows users of **StoryPivot** to retrieve unbiased information, i.e., a complete view of all available information on a story from a variety of sources instead of a single perspective which is the case if data is retrieved from a single source. Second, it allows us to correct errors made during the story identification phase. In Figure 1, we show that $v_i^j$ has been wrongly assigned to story $c_1$. Story alignment helps to determine this mistake by correlating events across sources and finding irregularities.
in the matching process which are then corrected. We call this correction step story refinement (Figure 1(d)). A more detailed description of the story identification and story alignment phases are presented next. Finally, we point out that the proposed two-level approach allows us to handle changes to the set of available data sources efficiently. As new sources become available, we first identify the stories associated with them and then align them with existing stories extracted from previously available sources. This enables an efficient integration of new data sources into the established information flow.

2.2 Story Identification

To enable efficient information aggregation into a story, we propose an incremental solution for the problem of story identification. As explained previously, story integration is the task of finding those information snippets $v_j^i$ in $V_i$ that refer to the same story $c_j$. Recall that an information snippet can be described by the entity mentions associated with it and its timestamp. Given this structure of an information snippet $v_j^i$, it is easy to see that there are two components that describe the evolution of a story and thus the connectivity of snippets: The content of $v_j^i$, i.e., the entities and the description that represent the unique information contained in $v_j^i$, as well as its temporal information which puts $v_j^i$ into context with other candidate snippets $v_k^i \in V_i$ which are possibly part of the same story.

For this framework, we explore two different matching techniques for two snippets $v_j^i$ and $v_k^i$: The first mechanism follows a complete matching scheme, while the second one pursues a temporal scheme as illustrated in Figure 2. The complete mechanism serves as a baseline for our system as we observe that due to the evolving nature of stories, complete mechanisms overfit stories because they tend to add related snippets to the same story independent of the evolution of the story in between. In reality, story evolution means that characteristics of a story change over time. Consider the example shown in Figure 2. Imagine that snippet $v_j^1$ is compared with the existing story set $C_1$: Here, it is not useful to compare it against all events $v_1^1, \ldots, v_1^n$ associated with previously found stories (Figure 2(a)), because an event that happened a long time before $v_j^1$ can have completely different content, even if they both refer to the same entities. Take as an example the Ukraine crisis which started with civil protests and evolved into a military conflict. Comparing snippets describing the fights around Donpez and protests on the Independence Square will not likely yield good results. In contrast, if we consider events closer to the fights such as the plane crash or the split of Crimea, there will be a higher overlap in entities and description between the single data snippets. To leverage temporal information, we propose to use a sliding window based technique for story identification. Instead of comparing $v_j^i$ to all other snippets, we thus reduce the window of comparison to all snippets $v_k^i \in V_i$ for which the corresponding timestamp $t_k^j$ is in a range of $t_j^i - \omega_i \leq t_k^j \leq t_j^i + \omega_i$ (Figure 2(b)). Additionally to providing a more realistic presentation of story evolution, sliding windows guarantee efficiency in information processing as the search space for related information snippets is reduced effectively.

2.3 Story Alignment

The story identification phase concludes with producing a set of stories $C_i$ that contain all stories that have been identified for a data source $s_i$. These stories can then be further used for cross-source story alignment, i.e., finding those stories in a data source $s_j$ that contain the same semantic information as the stories in a different data source $s_j$. Imagine that there are two sources $s_1$, corresponding to the New York Times, and $s_2$, corresponding to the Wall Street Journal. Though both of these information sources provide details on events happening in the United States, they have different perspectives on topics expressed in their published documents due to political orientation or different concepts for content presentation in the newspaper amongst others. Story alignment thus helps to enrich the stories identified for each source and to provide end users with a more encompassing view of a story. This also means that users are not only provided with source-specific, and thus potentially biased, information but have a structured way of exploring all perspectives of a story.

For story alignment, we observe that two stories are likely to refer to the same real-world story if their evolution is similar as well as their content. It is highly unlikely that two stories $c_1$ and $c_2$ are similar if $c_1$ ends at time $t_1$ and $c_2$ starts at $t_2$ with $t_1 \ll t_2$. We observe that in the alignment phase, we can partition information snippets into two categories considering their purpose for alignment: they can (a) help to align stories or (b) enrich stories. The first type is used for the actual alignment step as two snippets $v_j^i$ and $v_j^i$ from sources $s_i$ and $s_j$ provide aligning information if they contain similar entities, description, and are temporally (approximately) similar. If data snippets enrich a story instead, they provide additional information that is contained in data source $s_i$ but not in $s_j$. Relevant examples include special reports, background information etc. that are produced by journalists working for one data source but not the other.

Once stories have been compared across sources, the last step of story alignment is to resolve conflicts between the results of story identification and story alignment. Consider, the example shown in Figure 1. Here, story alignment reveals a mismatch in the story results of data sources $s_1$ and $s_n$: While $v_1^1, v_1^2, v_1^3$, and $v_1^4$ belong to the same story in $s_1$, their related snippets are in different stories in $s_n$. Resolving such conflicts helps to refine the results of the story identification phase as the decisions can be propagated back if desired.
2.4 Dynamic Interaction between Story Identification and Alignment

So far, we talked about batch processing a set of information snippets abstractly. If these snippets are extracted from online newspapers, blogs, etc., they are generated dynamically every time a news document is published online. Furthermore, there exist different types of data sources which do not necessarily publish their information in a temporarily ordered manner. Local events for example usually get picked up faster by local newspapers and might be delayed in international media because of locality and interest in those events. To address these challenges, story identification and alignment need to be dynamically integrated and realized efficiently as to provide users with live information on ongoing stories. This integration needs to consider an efficient representation of stories to reduce the search space and thus provide near real-time integration and out-of-order integration of events into evolving stories. Last, consider that data sources are not always known to end users but it should be possible to add or remove data sources from the event repository as necessary. The goal of our work is to address all of these problems with temporal story evolution techniques that adhere to the principles and ideas mentioned in the previous sections.

3. USE CASES

Our StoryPivot system has practical utility for a variety of users. In this section, we present two target groups for our system and describe scenarios in which they can employ StoryPivot to obtain information about stories that they are interested in. For both of these user types, we envision an interface where they can explore stories where our system supports different patterns of usage.

Expert User. Given its two-phase construction of stories, our system can provide users with insights on a story from either within a source or aggregated across sources. This functionality is essential for example for political scientists or journalists who want to determine the evolution of conflicts or interactions: Examining data within a data source reveals source bias and explicitly contrasts the different points of view that arise around a specific story. The results from the story alignment phase then help to construct a complete view of a story and to put single pieces of information from different data sources into context.

News Reader. As an occasional news reader, it is often difficult to obtain a clear picture of the reason for stories to evolve the way they do. For example, young news readers are often not fully aware of the historical developments and trends that influence politics. Also, readers might get interested in a story only after it has gained some traction in the media. Investigating the timeline of a story can help such users to get a comprehensive view of which events contribute to a story and to understand the implications certain events have on the development of this story.

In addition to these inherent features of our system, we can further extend it with interfaces to existing knowledge bases such as DBpedia[3]. Connecting StoryPivot to knowledge bases helps experts and casual users to obtain more context information and to explore the data provided by our framework more extensively.

4. DEMONSTRATION

In this section, we describe how we structure our demonstration and present the functionalities that will be available when interacting with the StoryPivot system.

4.1 Implementation

Our demonstration introduces StoryPivot, a system that efficiently processes event datasets to find semantically correlated events that form stories, i.e., that describe relationships between entities with evolving content. Users of StoryPivot can explore two different views on stories following the differentiation between story identification within a data source and alignment across data sources presented in Section 2 which we will describe in the next sub-sections. The backend processing of information extracted from documents is done as shown in Figure 1. First, we use an extraction mechanism that retrieves snippets of information which is followed by our implementation of the story identification and alignment steps. To realize story identification, we implemented the two different identification models shown in Figure 2, temporal story identification and complete story identification. The story alignment phase uses similar techniques that can be combined on the fly by the user to test advantages and disadvantages as well as result quality for these varying techniques. The event data explored for this demonstration is taken from real-world online newspapers and blogs such as the New York Times or the Wall Street Journal analogous to other event repositories such as GDELT[11] or EventRegistry[9].

4.2 Functionalities

For the demonstration, we will provide dynamic and static execution components. First, we will show how stories evolve on a small scale with a predefined example that the user can explore by adding and removing documents from the system. This will enable users to understand how extracted information from a document changes the outcome of the story identification and alignment phases. Furthermore, this part of our demonstration serves to illustrate the applicability and output quality of our system. Next, we will provide
examples of large scale story detection and show the scalability and efficiency of STORYPIVOT when processing larger amounts of data. The purpose for this part of the demonstration is to show that real-time event integration can be achieved through efficient story identification and alignment mechanisms. Users will be able to explore the results of the larger integration run and can query STORYPIVOT to see the evolution of a story over time within and across sources.

### 4.2.1 Interactive Story Detection

For this part of our demonstration, we will provide the user with an interface where a set of documents can be selected. Each of these documents is taken from real-world event data streams. An example for the document selection mechanism is shown in Figure 4. The data sources here are two well-known newspapers and documents correspond to news articles published on their corresponding websites. For this example specifically, the articles mainly focus on the Downing of a plane in the Ukraine in July 2014. Once the user has selected a subset of these, an initial set of stories within the selected data sources is computed and then stories are aligned across data sources, as in Figure 4. After the computation has finished, the user will be able to explore the data in two different ways corresponding to the view (i) within and (ii) across data sources.

#### Stories per Source Module.

The “Stories per Source” module, Figure 5, shows the computed stories within a source. Information snippets are sorted according to their story affiliation and can be browsed to understand why they are associated with the same or different stories. This module represents the story identification part of our work. Next to visualizing and explaining the correlation of snippets within a story, we also show the relationship of snippets across stories but within a data source. Take $v_2$ in Figure 5 as an example: We show that it is connected to both $v_1$ and $v_3$ but also to $v_4$ which contains information on a call for investigation on some problems in the Israeli conflict around the same time of the jet’s downing. As the entities for both snippets and the content are similar, there exists such a connection intuitively though the events can be separated through their context. Here, $v_2$ is similar to $v_1$ and $v_3$ which indicates that it is in fact part of a different story than $v_1$. In addition to providing this overview information per snippet, we allow users of STORYPIVOT to explore snippets within a data source in this module. Changing the selected snippet will automatically change the displayed content.

#### Snippets per Story Module.

As the second exploration module, we provide a “Snippets per Story” module, as in Figure 6, which shows a user-selected story that has been aligned across data sources. Again, this module will help the user to understand why our algorithms make certain decisions through visualizing the story alignment phase. Users can select a story that they are interested in and again explore single information snippets within the story to determine their connectivity. In the example of Figure 6, the selected snippet, $v_5$, is well-connected within story $c_1$ which consists of information on the downing of the jet ranging from the initial news to results of the investigation.

Both Figure 5 and Figure 6 show snippet-centric views of story detection. We choose this form of visualization because it puts emphasis on the evolving nature of stories as a set of semantically connected pieces of information. To enable users to explore this evolution, additional documents can be added after the initial import step. Furthermore, we allow users to remove documents from STORYPIVOT to explore how missing information modifies the displayed stories.
Statistics within a data source after story identification has been performed will provide an intuition of which techniques are more advantageous than others in terms of data analysis but also performance over time. Users are shown information and statistics on a set of extensive experiments through an overview module as visualized in Figure 7. Here, they can select all available datasets as well as different execution mechanisms that were run for these datasets. This will provide an intuition of which techniques are more advantageous than others over time.

Next to these static ways of exploring these datasets and the meta information on performance and quality, we also allow users to use the modules presented previously for an interactive exploration of these datasets. As a result, users develop an understanding of how stories evolve over time and how our system captures this evolution. It will not be possible to add or remove documents for this setup as the datasets will be fixed beforehand.

### 4.3 Take-away Points

With our work, we want to emphasize that story detection is an important and interesting problem that can help users to obtain information on real-world stories in a structured way. We use a two-step mechanism of story identification and story alignment for two reasons: First, it inherently guarantees that users can choose the granularity of stories presented to them. For example, they can decide to see a story from within a data source after story identification has been performed or as a (partial) integration across sources through our story alignment methods. With our interactive design, we want to show system users how these two mechanisms interact and help to build a better understanding of story evolution. Second, this two-step mechanism provides efficient implementation mechanisms and fast information processing because it makes effective use of incremental rather than complete computation of results. Remember that stories evolve over time which means that using and integrating all information snippets completely is more harmful than information processing in a time-aware manner. The stability and performance of our mechanisms are shown in large-scale experiments which help to obtain a better understanding of the efficiency of StoryPivot.

### 5. CONCLUSION

In this demonstration proposal, we have introduced a system called StoryPivot that constructs stories from information snippets that were extracted from news documents such as online blog entries, articles, reports and others. We proposed to process this data in two steps through story identification and subsequently story alignment. Both of these can be interactively explored in our demonstration with a small predefined example which allows users to manipulate stories by adding and removing information at will. In the second part of our demonstration, we examine large scale datasets to show that the mechanisms devised for this system are stable over time in terms of performance and result quality. Furthermore, we allow users to explore the results of these large-scale experiments interactively to understand how StoryPivot forms stories.

### 6. REFERENCES