Automatic Story Construction from News Articles in an Online Fashion

Özgür Can
Dept. of Computer Engineering
İzmir Institute of Technology
İzmir, Turkey
ozgurcan.can@gmail.com

Selma Tekir
Dept. of Computer Engineering
İzmir Institute of Technology
İzmir, Turkey
selmatekir@iyte.edu.tr

Abstract

This paper presents a novel story construction system to track the evolution of stories in an online fashion. The proposed system uses a novel sliding window solution, named Inching Window, allowing the processing of each new data element on-the-fly. To assign a new data element into a community in a fast and memory-efficient manner, we apply the modularity maximization idea of Louvain method on-the-fly. As part of the experimental validation, we provide step by step construction of a meaningful news story and support the case with a set of visualizations.

1 Introduction

Every day, thousands of local and global news become online. Each arriving news piece gets its meaning through its connections with some other news. Understanding the present situation requires an analysis in light of past. Thus, organizing news content in a coherent set of articles to form a story is a fundamental requirement.

Automatic story construction is a challenging task subject to some key issues. Firstly, data may come from multiple sources and could include overlaps. Moreover, it must be processed in an online fashion. Time-span may become another difficulty as long ranging stories must be supported as well as short-ranging ones. Visualization is another concern because stories gain meaning in the eyes of the beholder.

Topic Detection and Tracking (TDT) Allan (2002) covers identifying and following events from a constantly arriving stream of text from multiple sources. TDT defines an event as something that happens at a particular time and place and aims to group related documents that discuss the same event. At a higher conceptual level, a topic is defined as a theme that is related with a set of events. Topics are combined to form stories that represent a coherent structure of news. To illustrate story structures from a document stream, a sample mini-corpus of 15 news articles on four stories; **Turkey's Coup**, **Nice Attack**, **Munich Shooting** and **Brexit** is processed using our proposed system as depicted in Figure 1. As can be seen from the figure, **Turkey Coup** story appears at the beginning of the period. At the final stages, it disappears. While **Nice Attack** and **Munich Shooting** remain steady all along the time period, **Brexit** becomes evident eventually.

In literature, story structures are represented in different ways, as story graphs Yang et al. (2009), Subašić and Berendt (2013), information maps Shahaf et al. (2012), Shahaf et al. (2015) or story trees Liu et al. (2017).

In their study, Laban and Hearst (2017) propose a sliding window-based approach to story construction. Articles are represented with a set of TF-IDF filtered keywords. Their system operates based on three phenomena: Linking, splitting, and merging.
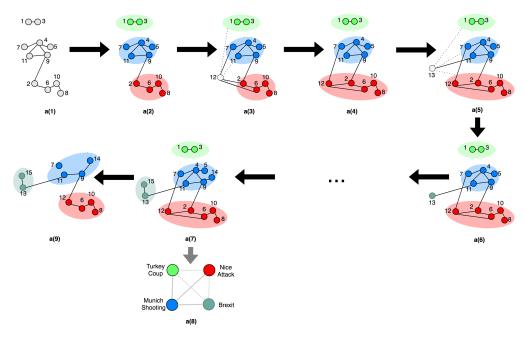
Figure 1: A sample evolving set of stories using our proposed system.

Ansah et al. (2019) present a novel graph-based timeline summarization system, named StoryGraph, to track the evolution of the stories from online Twitter communities.

Hu et al. (2017) propose a word embeddings-based document representation method and an online event detection algorithm, which uses time slicing and event merging.

In this work, we propose a system to generate continuously updated news stories. The main contributions of this work can be summarized as follows:

1. Novel way to represent stories as vectors.
2. Novel sliding window approach (Inching Window) that reduces the time complexity of processing each item in the context of the recent data.
3. Online clustering method (Louvain on-the-fly), which can automatically detect new stories from a continuous stream.

2 Updating News Stories

2.1 Document Representation

We use doc2vec Le and Mikolov (2014) embeddings to represent documents.

2.2 The Clustering Algorithm

In this work, we represent each news article as a node in a fully connected undirected weighted network, where the edge weights between the nodes are set based on the cosine similarities between the document vectors.

To detect the stories, we use the Louvain method Blondel et al. (2008). The Louvain community detection method is an algorithm for finding communities in the network by trying to maximize the modularity in a repeated process. One of the major advantages of the Louvain method is that it reveals a hierarchy of the communities at different scales and this hierarchical perspective helps us understand the story construction and observe root stories.
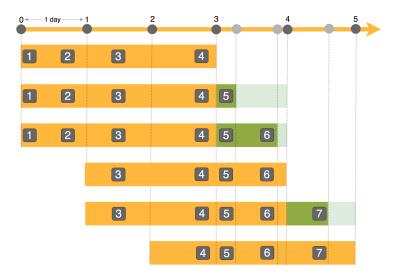
2

Figure 2: Illustration of the inching window process.

2.3 Story Representation

Learned document representations can capture syntactic and semantic regularities and they also preserve these regularities on algebraic operations. Thus, in this work, we propose to use this feature to come up with story representations. To be specific, we add up all the document vectors in the story to calculate the story vector.

2.4 Online Clustering

Stream clustering algorithms are used for extracting useful knowledge from data streams.

One common technique for online clustering is evaluating the new data point only over sliding windows of recent data. For instance, only data from the last week can be used to cluster new data instead of querying over entire past history. Sliding window approach emphasizes only the recent data, and for many applications, recent data are more important and relevant than the old one.

Majority of the news articles belonging to the same story are published in a dense way in a short period of time. Considering this fact, many existing works on online news story detection feed the news stream in the chronological order to their systems and use sliding window method, and time-sensitive queries to cluster the documents de Andrade Silva et al. (2013). In such systems, defining an optimal sliding window interval is a challenge.

2.5 Inching Window

Generally, the main concern of a data streaming system is processing new data elements on-the-fly, but processing with respect to a time interval requires gathering the data until interval condition is met. For instance, a sliding window of five days with a one-day sliding interval requires to wait until the end of the last day in the window to gather all data. To overcome this issue, we propose a novel sliding window technique, named *"Inching Window"* as the window slides like an inchworm. Inchworms move in a characteristic *inching* or *looping* gait by extending the front part of the body and bringing the rear up to meet it" [1]. An *inching window* moves in a similar fashion. Given a window size of 3 days, and a sliding interval of 1 day, similar to sliding window, *inching window* groups the incoming data in batches every 3 day, but rather than sliding after processing the batch, it continues to process each incoming data one by one until the sliding interval (1 day), then it slides. This *inching window* process is illustrated in Figure 2.

The main advantage of the *inching window* approach over *sliding window* is that *inching window* enables processing of the new data elements on-the-fly within the recent data context.

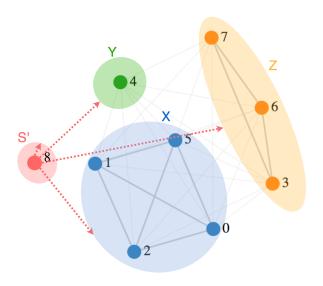[1]https://www.merriam-webster.com/dictionary/inchworm

Figure 3: On-the-fly clustering process.

After feeding a news stream in the chronological order to an *inching window* set up, we propose an online story detection system that is composed of three main steps: Topic creation, on-the-fly document clustering, and story construction.

2.6 Topic Creation

A set of news articles in a small time interval (e.g. 5 days) have a potential to create a densely-connected local group around a *topic* which, can then be merged in a *story*. To detect the topics in a window, we initiate a news article network and apply the clustering algorithm explained in Section 2.2. The resultant communities are topics and they can be represented with topic vectors, which are computed by adding all document vectors belonging to the topic.

2.7 On-The-Fly Document Clustering

After the initial inching window, news articles are received one-by-one until the specified window interval. This enables us to process a newcomer article without waiting for the end of the day. However, it is not computationally feasible to apply the clustering algorithm again and again for each newcomer article. Furthermore, it is often acceptable to assign a newcomer article to the most suitable community in the given context, and later on, review the temporary assignments and then make permanent assignments.

The clustering algorithm that we use in Section 2.2, relies on Louvain community detection method Blondel et al. (2008) that tries to identify communities in a network by repeating optimization steps iteratively until a maximum value of modularity is attained. To assign a new document into a community in a fast and memory-efficient manner, we propose to compare modularity changes in the network after assigning the new document into one of the existing communities or creating a new community for the document. The on-the-fly news clustering process is illustrated in Figure 3. Given a network with communities found (X-blue, Y-green, and Z-yellow), for the newcomer $Article_8$ a new community named $Community'_S$ is created. Then, we measure modularities on the graph for the scenarios where $Article_8$ joins $Community_X$ or $Community_Y$ or $Community_Z$. Then, we select the scenario where the graph modularity is maximized. For the given sample, we found that community X maximizes the modularity, so $Article_8$ is assigned to $Community_X$.

Before sliding a window, we apply the clustering algorithm all over the window and override the temporary assignments.

4

2.8 Story Construction

A *topic* that we detect might be; 1. an initial seed of a new story, 2. belonging to an active story, 3. a continuation of a story that is interrupted in time, for instance, a crucial piece of evidence for *Malaysia Airlines Flight 370 Disappearance* is found after years.

To properly address these characteristics of a topic, we propose to create another fully connected network where the nodes are stories detected, and the edges' weights are cosine similarities between the nodes. First, we represent topics found in a window with the sum of the vectors representation of the articles belonging to that topic. Then, we added the topics as a node to the story network that we initiated. Then, we apply the community detection algorithm. Running community detection over the story network creates interesting dynamics:

1. A topic might create a community with an existing story in the network. In this case, we merge the topic with the story, then remove the topic node from the network. Merging is executed by summing up the story vector and the topic vector after handling the document migrations from one story to another. Sliding windows overlap in time and data. Thus, a topic found in a window can have the same document with a story found in the previous window. Thus, when a story and topic is merged, if a document is already belonging to another story, it should be removed from that story first.

2. A topic might create a new individual community. After handling the document migrations, the topic is directly casted into a story.

3. A topic might create a community with another topic found in the same window. Communities are decided based on the modularity of the network. Two document communities found in the local article network context might be merged into one in the global story network. Merging is executed in the same fashion with topic-story merging.

4. An existing story might create a community with another story. This case rarely happens, but as the stories grow in time, their relations with each other evolve as well. Thus, two distinct stories might start to have connections and finally be merged. Two stories are merged on the oldest one by directly summing up their vectors.

In our system, stories do not need to keep individual document data. In creating a story from a topic, keeping only the document ids and sum of the document vectors to represent the story is sufficient. Document data and their individual vector representations can be deleted from the memory.

3 Experimental Results

To illustrate the story construction process, we executed our proposed method in a sample mini-corpus of 15 news articles between 24.06.2016 and 29.06.2016. The corpus consists of news articles on four stories; *Turkey's Coup*, *Nice Attack*, *Munich Shooting* and *Brexit*. We set up an inching window for 4 days with the inching interval of 1 day (Figure 1). In Figure 1, in order to simplify the tracking of the stories, we do not depict the network as a fully connected one.

1. Subfigure a(1) shows the network of the first window articles between 24.06.2016 and 28.06.2016.

2. In the Subfigure a(2), the clustering algorithm is run to detect the topics.

3. Forward moving process of the inching window is started in Subfigure a(3). To assign the newcomer $Article_{12}$ to a community, the graph modularity is iteratively checked for each possible community assignments.

4. In the Subfigure a(4), $Article_{12}$ is assigned to the community that produces the maximum modularity.

5. Subfigure a(5) and Subfigure a(6) apply the same procedure as the Subfigure a(3) and Subfigure a(4) for the $Article_{13}$, but this time, creating a new community maximizes the graph modularity.

6. On-the-fly clustering is executed for each new document until $Article_{15}$, which is the latest article of the window. This last step of the window is shown in Subfigure a(7).

7. Topic clustering algorithm is run again to review and correct the on-the-fly community assignment in Subfigure a(8).

8. In the Subfigure a(8), the final set of topics is added to the story network and proposed flow for the story construction is executed.

9. In the Subfigure a(9), we shrink the inching window by removing the first day of the window; 24.06.2016.

In order to further evaluate the performance of the proposed system, we created a sample corpus of 400 documents which contains a short-ranging story, a long-ranging story, and a story that is interrupted in time and fed the set of documents as a stream. As a result, we achieved to get high $F1(0.829)$ and $NMI(0.823)$ scores that indicate a good clustering performance.

4 Conclusion

In this work, we present a method for aggregating news articles into topics and then merge them into story clusters in an online fashion.

To the best of our knowledge, this is the first work that uses doc2vec Le and Mikolov (2014) for news article representation in topic detection and story construction. To perform the task in an online fashion, we present a modified sliding window approach called *"Inching Window"*. Moreover, in accordance with the Inching Window concept, we use the modularity maximization idea of Louvain method to perform on-the-fly clustering. We will publish our code so that the research community can build on top of our work.

5 Future Work

Revealing communities in document article network allows for analysis of relations between individual articles and classifications of the objects based on their characteristics. In some real-life networks such as social networks, objects can simultaneously belong to multiple communities at once. These communities are called overlapping communities. In this study, we follow the general traditional sense that a news article can belong to only one news story at a time. However, analysis of the overlapping news stories might create interesting dynamics. We plan to change switch Louvain community detection algorithm with an overlapping community detection algorithm and evaluate the performance of the system.

Although the focus of the work is to find stories from the news articles, we plan to test the proposed system with different types of data, such as social media news.

In this work, we evaluate the accuracy with a manually collected document set. Real world data might have more noise and different characteristics. We plan to evaluate the accuracy of the story creation algorithm in a large annotated corpus.

References

Allan, J. (2002). Topic detection and tracking: event-based information organization.

Ansah, J., L. Liu, W. Kang, S. Kwashie, J. Li, and J. Li (2019). A graph is worth a thousand words: Telling event stories using timeline summarization graphs. In *The World Wide Web Conference*, WWW '19, New York, NY, USA, pp. 2565–2571. ACM.

Blondel, V. D., J.-L. Guillaume, R. Lambiotte, and E. Lefebvre (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment 2008*(10), P10008.

de Andrade Silva, J., E. R. de Faria, R. C. Barros, E. R. Hruschka, A. C. P. de Leon Ferreira de Carvalho, and J. Gama (2013). Data stream clustering: A survey. *ACM Comput. Surv. 46*, 13:1–13:31.

Hu, L., B. Zhang, L. Hou, and J. Li (2017, December). Adaptive online event detection in news streams. *Know.-Based Syst. 138*(C), 105–112.

Laban, P. and M. A. Hearst (2017). newslens: building and visualizing long-ranging news stories. In *Proceedings of the Events and Stories in the News Workshop@ACL 2017, Vancouver, Canada, August 4, 2017*, pp. 1–9.

Le, Q. and T. Mikolov (2014). Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pp. II–1188–II–1196. JMLR.org.

Liu, B., D. Niu, K. Lai, L. Kong, and Y. Xu (2017). Growing story forest online from massive breaking news. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, pp. 777–785.

Shahaf, D., C. Guestrin, and E. Horvitz (2012). Trains of thought: generating information maps. In *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*, pp. 899–908.

Shahaf, D., C. Guestrin, E. Horvitz, and J. Leskovec (2015, October). Information cartography. *Commun. ACM 58*(11), 62–73.

Subašić, I. and B. Berendt (2013, January). Story graphs: Tracking document set evolution using dynamic graphs. *Intell. Data Anal. 17*(1), 125–147.

Yang, C. C., X. Shi, and C. Wei (2009, July). Discovering event evolution graphs from news corpora. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans 39*(4), 850–863.