ABSTRACT

Wikipedia is the biggest ever created encyclopedia and the fifth most visited website in the world. Tens of millions of people surf it every day, seeking answers to various questions. Collective user activity on the pages leaves publicly available footprints of human behavior, making Wikipedia an excellent source of the data for large-scale analysis of collective dynamical patterns. The dynamic nature of the Wikipedia graph is the primary challenge for the analysis.

In this work, we propose a graph-based dynamical pattern extraction model, inspired by the Hebbian learning theory. We focus on data-streams with underlying graph structure and perform several large-scale experiments on the Wikipedia visitor activity data. We extract dynamical patterns of collective activity and show that they correspond to meaningful clusters of associated events, reflected in the Wikipedia articles. We demonstrate evolutionary dynamics of the graphs over time to highlight changing nature of visitors’ interests. We discuss clusters of events that model collective recall process and represent collective memories – common memories shared by a group of people. In the experiments, we show that the presented model is scalable regarding time-series length and graph density, providing a distributed implementation of the proposed algorithm.

1 INTRODUCTION

Over the recent years, the Web has significantly affected the way people learn, interact in social groups, store and share information. Apart from being an essential part of modern life, social networks, online services, and knowledge bases generate a massive amount of logs, containing traces of online activity on the Web. Large-scale examples of such publicly available information are the pages in WWW, linked by the hyperlinks, or Wikipedia articles, connected by the references. An essential part of these data sources is the underlying human-made graph structure that was initially introduced to facilitate navigation. This data is a great source for collective human behavior analysis at scale.

Online activity of users enriches static graph-structured data with temporal components. The combination of dynamics and structure of Web graphs inspires an idea of their similarity to biological neural networks. A good example of such network is the human brain. Numerous neurons in the brain constitute a biological dynamical neural network, where dynamics are expressed in terms of neural spikes. This network is in charge of perception, decision making, storing memories, and learning.

During learning, neurons in our brain self-organize and form strongly connected groups – neural assemblies. These groups express similar activation patterns in response to a specific stimulus. When learning is completed, and the stimulus applied once again, reactions of the assemblies correspond to consistent dynamical activity patterns – memories. Synaptic plasticity mechanisms govern this self-organization process. Hebbian learning theory [20] proposes an explanation of this self-organization and describes basic rules of synaptic plasticity. The theory implies that simultaneous activation of a pair of neurons leads to an increase in the strength of the connection between the two.

In general, the Hebbian theory implies that the brain network is being transformed under the influence of neural activity. This assumption makes one think of an interesting question. Can temporal dynamics cause a self-organization process in a graph, similar to the one, driven by neural spikes in the brain?

In this work, we introduce a graph learning approach based on the ideas of neuronal self-organization in the brain. The approach is inspired by the theory of learning and memory formation in the brain. In the experiments, we consider the dynamic network of Wikipedia articles. The dynamics of the network is expressed by the visitors’ activity on each page – the number of visits per hour. We assume that the Wikipedia network can self-organize and reveal dynamic patterns under the influence of visitors activity similar to neurons in the brain. We formalize event detection problem into a graph learning problem. Section 3 provides the details of graph learning. The experiments demonstrate that the emerged patterns correspond to groups of events, containing clusters of linked pages that have closely related meaning.

Contributions. We propose a novel graph learning framework, inspired by Hebbian learning theory. We benefit from initial graph structure and learn dynamical patterns using time-series signals on the nodes. The learning algorithm is unsupervised and does not require prior knowledge of the number of clusters. Computations are local on the graph, allowing to build an efficient implementation. We provide a distributed implementation of the algorithm to show that it can handle dense graphs with long time-series on the nodes. Besides, we apply the approach to event detection problem and consider a particular type of an event – collective memory [17]. So far, collective memory has been considered just as a concept that lacks a general model of memory formation. In this work, we formalize our findings into a general graph-based model of collective memories, based on Wikipedia graph viewership dynamics. Finally, we present graph visualizations as an interactive tool for event detection and collective memory studies.

The rest of this paper organized as follows. In Section 2 we give an overview of the related works on graph learning, event detection, dynamical process modeling, and collective memory research and analysis. Graph learning and community detection are presented in Section 3. Section 4 describes the dataset that we use...
2 RELATED WORK

Dynamical process modeling and graph learning. The core part of the presented work is the proposed graph learning approach, described in Section 3. We use it as a tool for dynamical process modeling. Another popular approach for that is Hawkes process (HP) [18], [26] – self-exciting multivariate point process.

HP allows capturing dynamical correlation between events, defined as an intensity function or probability \( \lambda_i(t) \). In HP, the probability of future events depends on the history of past events, the base intensity of an event, its excitation rate and a kernel function that captures temporal dependencies.

A few works combine HP with Dirichlet process to build clustering models for document streams, using both temporal and content information [10], [19], [27]. The proposed algorithms achieve good results concerning clustering and topic modeling, but do not benefit from the given information about the graph structure, underlying the time-varying data. Although He et al. [19] infers hidden diffusion network, the authors neglect prior knowledge of the graph structure.

A recent work introduced a mathematical model named Hawkes graphs [11] – directed weighted graphs, summarizing multivariate HP. Nodes in the graph correspond to events. The edges, connecting the events, denote dynamic correlation. The proposed approach is helpful for preliminary analysis of HP with multi-dimensional streams of events. In case of high-dimensional situations with a large number of events, the authors mentioned a limitation of the model – the graph has to be sparse to achieve a feasible level of performance (less than five parents for each node in the graph).

For the settings, when the graph structure is unknown and slowly changes over time, another graph learning framework was proposed [22]. The framework splits time into windows, allowing to learn a sequence of graphs for each time-window. The authors take into account the variation of edge weights. This prior allows enhancing known graph learning algorithms. This approach does not utilize the given graph structure.

In this work, we propose the approach that allows exploiting known graph structure, combined with time-varying signals on the nodes. We show that the initial graph structure can be dense. In the experiments, the average degree of the graph is equal to 91. We demonstrate the scalability of the solution regarding the time-series length and graph density, providing the code for distributed computations.

Event detection and collective memory. Dynamic visitor activity data inspired the large-scale analysis of the collective human behavior concept. A few recent works use visitor activity data to extract collective dynamical patterns from Web networks. Specifically, Mavroforakis et al. proposed a model, extracting the learning patterns from StackOverflow user activity – Q&A service for software developers[27]. Shao et al. built a model for events detection and forecasting in dynamic networks, based on the activity on Twitter and Weibo microblogging social platforms [29]. Wikipedia edits prediction model was proposed by Konovalov et al[25]. The model uses streaming data from live news and social media text streams to identify transition events (e.g., weddings, elections) and to use them for Wikipedia edits prediction.

Another direction, similar to the collective human behavior research, is the study of collective memory. The term Collective memory first appeared in the book of Maurice Halbwachs in 1925 [17]. He proposed a concept of a social group that shares a set of collective memories that exist beyond the memory of each member and affects the understanding of the past by this social group. Halbwachs’s hypothesis influenced a range of studies in sociology [1], [3], psychology [9], [13], and, only recently, in computer science [2], where authors extract collective memories using LDA [7], applied to a collection of news articles.

Nonetheless, the general nature of collective memory formation and its modeling remain open questions. Can we model collective and individual memory formation similarly? Is it possible to find collective memories and behavior patterns inside a collaborative knowledge base? In this work, we adopt a data-driven approach to shed some light on these learning processes.

Despite the fact that Wikipedia is the largest ever created encyclopedia of public knowledge and the fifth most visited website in the world, the studies on collective memory started considering the Wikipedia visitor activity data only recently. Analyzing Wikipedia page views, the researchers proposed a collective memory model, investigating 5500 events from 11 categories [23]. The authors proposed a remembering score based on the combination of time-series analysis and location information. The focus of the work is on the four types of events: aviation accidents, earthquakes, hurricanes, and terrorist attacks. The work presents extensive experimental results, however, lacks the general collective memory extraction model. Traumatic events such as attacks and bombings have also been investigated in [12] based on the Wikipedia data.

Another case study proposes the view flow of the events [15] to define collective memories. The collective memory model is inferred from the Wikipedia visitors activity. The work considers only the case of aircraft incidents reported in English Wikipedia. The authors try to build a general mathematical model and explain the phenomenon of collective memory, extracted from Wikipedia, based on that single example.

Popularity and celebrities represent another focus point of public interest. The Wikipedia hourly visitor activity on the pages of the celebrities was used to investigate fame levels of the tennis players [32]. The authors, though, did not tackle collective memories problem and aimed to quantify the relationship between performance and popularity of the athletes.

To the best of our knowledge, we are the first to apply a graph-based computational model to the Wikipedia viewership data for event detection and collective memory analysis. Unlike the results of the previous works, our model is not limited to particular classes of events. In this work, we also try to present the general model for collective memory learning and extraction.
contains a time-varying signal \( x_i(t) \) of length \( T \), and \( W \in \mathbb{R}^{n \times n} \) is a weighted adjacency matrix. The combination of underlying graph topology and time-varying signals on the nodes constitute a temporal graph. The graph topology is fixed over time. Figure 1 illustrates an example of the temporal graph.

Based on the given definition, we build a graph-based model allowing to extract dynamical patterns, given the known graph structure and the signals of the temporal graph. Initially, we set all weights \( w_{ij} \) in \( W \) equal to 0.

**Hebbian learning.** We use a synaptic-plasticity-inspired computational model in the proposed graph learning algorithm to update weights. Donald Hebb[20] proposed the fundamental mechanism of synaptic plasticity. He introduced a model, explaining self-adaptation of neurons in the brain during learning and memory formation. The simplified idea of the theory is that the co-activation of two neurons results in reinforcement of the connection (synapse) between them. The original model takes into account causality of activations – a neuron (pre-synaptic) should cause the activation of the neighbor (post-synaptic). The weight update model can be formalized as follows:

\[
\Delta w_{ij} = \alpha \sum_{t=0}^{T} x_i(t)x_j(t),
\]

where \( \Delta w_{ij} \) is the weight update for the synapse (edge) between neurons (nodes) \( i \) and \( j \), \( \alpha \) is the learning rate, \( x_i \), \( x_j \) are the activations of neurons \( i \) and \( j \), and \( r \) is the learning time.

Our model neglects the causality of co-activations. The condition to reinforce a connection between a pair of nodes only depends on the simultaneous activity on both nodes, during a defined period.

**Graph learning.** We refer to self-organization of the graph as learning. The activation of a node is the value of a signal, residing on the node, at time \( t \). To learn a graph of dynamical patterns, we iteratively compare activations on the connected nodes for each time step \( t \) and update the weights according to a similarity function. We use known information about the structure to build an initial graph.

The computations are tractable because 1) they are local on the graph – weight updates depend on a node and its first order neighborhood, 2) weight updates are iterative, and 3) a weight update occurs only between the connected nodes and not among all possible combinations of nodes. These three facts allow us to build a distributed model to speed up computations. For this purpose, we use a graph-parallel Pregel-like abstraction, implemented in GraphX framework [16], [31].

**Similarity function and weight update.** We do not use the original Hebbian update rule since it is not robust to unnormalized signals. Normalized cross-correlation measures of similarity are not sufficient either because they do not reflect burstiness of signals. As a result, cross-correlation similarity measures are biased towards general correlation trends over time so that short-term dependencies die out.

To normalize weight updates and capture time-specific dependencies between signals on the nodes we introduce the following non-linear similarity function:

\[
Sim(i, j) = \frac{\min(x_i(t), x_j(t))}{\max(x_i(t), x_j(t))} \in [0, 1]
\]

It computes the ratio of two activity levels \( x_i(t) \) and \( x_j(t) \) on the connected nodes \( i \) and \( j \) at time \( t \). The function is bounded between 0 and 1.

Weight updates \( \Delta w_{ij} \) depend on a threshold parameter \( \lambda \in [0, 1] \) and the similarity \( Sim(i, j) \):

\[
\Delta w_{ij} = \begin{cases} 
+Sim(i, j), & \text{if } Sim(i, j) > \lambda, \\
-Sim(i, j), & \text{otherwise}.
\end{cases}
\]

An update \( \Delta w_{ij} \) is positive if the similarity is higher than the threshold \( \lambda \), and negative otherwise.

**3.1 Graph visualization and community detection**

Dynamical patterns correspond to communities of correlated nodes in a learned graph. To interpret and analyze resulting graphs we have to visualize them and extract communities.

We use a heuristic method based on modularity optimization (Louvain Method) [8] for community detection. Colors in the graph depict and highlight the detected communities. A resolution parameter controls the size of communities.

To represent the graph in 2D space for visualization, we use a force-directed layout [21]. Spatial representation is force-based and takes into account density of a neighborhood of the nodes. The forces in the graph correspond to edges and neighborhoods of the nodes. Resulting layout reflects spatial placing of a node depending on its neighbors.

The presented solution allows to extract the dynamical patterns as the groups of associated nodes using the community detection algorithm and visually assess the results on the force-directed layout.

**4 DATASET**

The dataset is based on two Wikipedia SQL dumps: English language articles and user visit counts per page per hour. The original datasets are publicly available on the Wikimedia website [14].

We create Wikipedia graph using the data from article dumps that contain information about the references (edges) between the pages (nodes). Time-series data contains the user visit counts from 02:00, 23 September 2014 to 23:00, 30 April 2015. Note that Wikipedia is continuously updating. Some links that existed at the moment we made the dump may have been removed from current
versions of the pages. To check consistency with past versions, use the dedicated search tool\footnote{http://wikipedia.verbalehofs.de/wiki blame.php).

Preprocessing. Most of the Wikipedia pages remain unvisited during their entire lifecycle. Some pages are continuously active and have the high rate of visits per hour. Before building the Wikipedia graph, we introduce thresholds on the minimum number of visits per hour and maximum in-degree. We remove the pages that have less than 500 visits per hour at least once during the specified period.

We remove nodes (pages) with in-degree higher than 8 000 since such nodes are connected to thousands of semantically unrelated pages. This preprocessing step helps achieve better initial modularity of the dynamical patterns without losing relevant information.

To alleviate the corruptive influence of nodes that are always active, we penalize high-frequency time-series. For each of time-series, we count the number of time steps where the value of activation is higher than 0. To penalize high-activity nodes, we divide the time-series by that value.

Another issue of the original dataset of user visit counts is redirected pages. Initially, redirected and real pages have individual logs. We merge the data for both and sum page counts, keeping only the real pages in the dataset.

The preprocessed dataset \cite{dataset}, \cite{dataset2} contains 116 016 pages (out of total 4 856 639 pages), 6 573 475 hyperlinks, and time-series visits per page per hour from September 2014 till April 2015. We use this dataset for all the experiments in the following section.

5 EXPERIMENTS

To learn dynamical patterns, we use the approach, presented in Section 3. We set the threshold parameter $\lambda = 0.5$. We represent the Wikipedia network as an undirected weighted graph $G = (V, E)$, where $V$ is the set of Wikipedia pages, $E$ is the set of references between the pages, and $W$ is the set of weights, reflecting the similarity of the articles. Two nodes are connected in the graph if their associated Wikipedia pages are connected by at least one hyperlink. During the learning, we only consider those edges and their weights. It reduces the number of potential correlations to compute and makes the learning tractable.

We refer to hourly visits of a Wikipedia article as activations. Activation time-series correspond to the signals $x_i(t)$ on the nodes of the graph.

For the sake of simplicity, we neglect directions of the references between the pages. This measure does not affect the results of the experiments since we are only interested in the coherent activity of the pages, which does not depend on directions of the Wikipedia references.

The most popular pages are constantly active. The high-activity penalization in the preprocessing step allows avoiding corruption of the graph by popular nodes. This approach lets low-activity nodes to form smaller, independent, and topic-specific communities.

Evolutionary models separate short-term and long-term scale of the dynamics \cite{evolutionary}. We apply this separation to extract short-term (1 month) and long-term (7 months) collective memories.

We aim to learn collective memories and detect events, taking place during a chosen period. First, we give detailed measures of the graph, learned by the long-term 7-months dynamics. Monthly and weekly graphs have similar trends and properties. Then, we pick and discuss in details one cluster among 172 detected (NFL championship) and show monthly evolution dynamics of the graph. Lastly, we pick three detected events and explore collective memory formation and associativity over one week period. These events are Ferguson unrest (2nd wave, 24 November 2014), Charlie Hebdo shooting (7 January 2015), and Germanwings 9525 flight incident (24 March 2015). This experiment allows learning event-specific memories. We can refer to it as zooming in time since the learning algorithm in this experiment focuses the attention on specific events.

Properties of the graphs. The initial and learned Wikipedia graphs have statistically heterogeneous connectivity (figure 5) and weight patterns (figure 4) that correspond to skewed heavy-tailed
The larger the exponent, the higher the scale of fluctuations. (figure 5) distributions.

The degree distribution of both graphs has power-law behavior $P(k) \sim k^{-\gamma}$ (figure 5). The magnitude of power-law exponent $\gamma$ reflects the steepness of the slope or scale of degree fluctuations. The larger the exponent, the higher the scale of fluctuations.

The degree distribution in log-log scale corresponds to power-law behavior $P(k) \sim k^{-\gamma}$. Power-law exponent $\gamma = 3.81$ for the initial graph (blue), and $\gamma = 2.85$ for the learned graph (red). The decrease of the exponent in the learned graph indicates that the weight distribution became smoother and the number of greedy hubs dropped.

The initial Wikipedia graph (figure 2a) is dense and cluttered with a significant number of unused references. The degree distribution has a large power-law exponent $\gamma = 3.81$ (figure 5, blue). This shows us that the initial Wikipedia graph has a small number of large hubs that attract most of the connections to numerous low-degree neighbors. These hubs correspond to general topics in the Wikipedia network. We aim to extract smaller communities, corresponding to localized events in terms of a topic breadth.

The proposed method allows to obtain the learned graph with lower power law exponent $\gamma = 2.85$ (figure 5, red). Besides, the number of communities jumps from 32 in the initial graph (figure 2a) to 172 after learning in the resulting graph (figure 2b). The size of communities in the learned graph is one order of magnitude smaller than in the initial one, indicating that the proposed approach allows extracting small localized patterns from the dynamic graph.

Visualization. We visualize all the graphs presented in the experiments to assess the results of the learning algorithm. We use the visualization approach described in Section 3.1. For community detection algorithm [8] we set resolution parameter to 0.7 for all the visualizations. The initial graph (figure 2a) before learning is denser, comparing to the learned one. Modularity value is equal to 0.566. It consists of a small number of large communities, which represent clusters of general knowledge on Wikipedia. The learned Wikipedia graph (figure 2b) is much sparser. After learning, modularity value increased to 0.775, indicating that communities became much separated, comparing to the initial graph. The learned graph contains collective memories corresponding to communities of the nodes. The hub nodes with high betweenness centrality typically belong to multiple communities. Each community is either a potential set of memories that was triggered by a particular current event or a semantic set of associated events and people.

Long-term dynamics. Event detection. To extract the long-term collective memories we learn the graph, using 7-month activity data on Wikipedia. The learning algorithm described in Section 3. After graph learning, we can observe the set of important edges. These edges reflect a strong dynamical correlation between the pages. After learning, the majority of weights are either 0 (5 135 338, 78%) or negative (1 162 639, 17.8%). Only 275 498 edges (4.2%) have positive weights and reflect the dynamic correlation between the pages. To see the dynamical patterns in the Wikipedia network we prune out the edges that have 0 or negative weight. Figure 2 shows snapshots of the initial Wikipedia graph before learning (a) and the learned graph after weight update and pruning (b). Edge weight distribution after learning is presented on figure 4. We also delete disconnected nodes that remain after edge pruning and remove small disconnected components of the graph. The number of remaining important nodes is 35 839 (31% of the initial number). Figure 5 illustrates weighted degree distributions in the initial (blue) and learned (red) graphs respectively.

The long-term dynamic graph allows extracting events that took place during the considered period of 7 months: championships,
tournaments, awards ceremonies, world-level contests, attacks, incidents, and most popular holidays. The graph contains the summary of global events happened during the period of interest – the global collective memory over time. The resulting graph shows only the outcome without any details related to the dynamics and timeline of its evolution. To study the dynamics of the graph and the process of the memory formation, we look at the short-term dynamics.

**Dynamics of learning.** To learn the graphs, reflecting short-term dynamics, we split our dataset into months. Short-term collective memories correspond to the clusters, created under the influence of monthly dynamics of Wikipedia. Monthly graphs are smaller, comparing to the graph with long-term patterns and contain 10,000 nodes on average. The properties and distributions of the short-term dynamic graphs are similar to the long-term ones, described above.

Short-term graphs allow to understand and visualize the dynamics of the memory formation in the long-term graph. The short-term graphs highlight the events that are specific to the short period of one month. To give an example of the memory evolution, we demonstrate and discuss the cluster of the USA National Football League championship (figure 6).

NFL championship is one of the most popular sports events in the USA. The final game of the 2014 season, Super Bowl XLIX, had been played on February 1, 2015. Figure 6 shows the monthly evolution of the event, represented as a cluster of connected nodes in the graph. For the plot interpretability, we extracted 30 NFL team pages from the original cluster (485 pages) to show the details of the dynamics over the cluster. This fraction of nodes reflects overall dynamics in the entire cluster. The timeline (red line) shows the collective activity of the detected cluster over time. Each row describes the hourly activity of a page. The columns split the plot into months. The top part of the figure contains monthly snapshots of the graph. The NFL cluster highlighted in red.

Dynamics of the detected cluster reflects the real timeline of the NFL championship. Spiking nature of the overall activity corresponds to weekends when most of the games were played. The peaks become stronger closer to the end of the championship, following the increasing interest of fans. We see the highest splash of the activity on 1 February, when the final game was played.

Evolution of the cluster on top of the figure 6 gives intuition on how the proposed algorithm detects and follows a particular event. In the beginning, the activity of the cluster is low, so the size and density of the connections are minimal. Closer to the final of the championship the cluster evolves, grows, and becomes denser, highlighting the peak of the interest to the event. After the final game, the visitors’ interest drops, causing vanishing of the cluster. The cluster activity precisely reflects the NFL fans behavior. They become the most active users of Wikipedia during the final
The memory effect. Collective memories have associative nature. We pick three events among 172 detected and discuss them to show the details of memory detection by the algorithm. Figure 7 shows the extracted collective memories as communities of the nodes (top) and the overall timeline of the clusters’ activity (bottom). From the memory perspective, the core events trigger past events, connected by the edges in the cluster. Table 1 presents examples of the collective memories.

In the following, we discuss the noise that appears in the detected collective memory clusters. In our case, the noise represents pages that are unrelated to the topic of the detected cluster. Charlie Hebdo shooting. 7 January 2015. This event is an example of a terrorist attack. The cluster emerged over a week, following the attack. All the pages in the cluster are related to the core event. There is a sharp peak of activity on the first day of the attack, slowly decaying over the following week. There is no noise in the cluster because all the memory-pages are inactive most of the time and were triggered by the core event.

Germanwings flight 9525 crash. 24 March 2015. This cluster is the example of a noisy memory. The main source of the noise is the node of the day of the crash – 24 March. This page connects all the events, happened that day in the past. Even though noisy spikes exceed the spike of the core event, their duration is short (4-8 hours). We distinctly see the main activity burst due to its longer duration. This also shows that the whole cluster becomes active at the moment of the accident. There is a smaller period of the activity rise at the end of December. This minor, but steady period of increased activity reflects another aircraft crash – Indonesia AirAsia Flight 8501 – that is associated with the core event.

Ferguson unrest. Second wave. November 24, 2014 – December 2, 2014. This cluster is another example of a noisy memory. The source of the noise is similar to the previous example. The event is associated with the Anonymous group whose Wikipedia page is connected to the largest tech companies’ pages. This connection is the major cause of the noise. The page “Online shopping” causes a first steady burst of activity. This page appears in the activation

Figure 7: Collective memory clusters (top) and overall activity of the visitors in the clusters (bottom).
The venue is to be announced,

V. Miz et al.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we presented a novel graph-based approach for dynamic pattern detection. We demonstrated an example of the event and collective memory detection using large-scale Wikipedia viewership data and hyperlinks graph. We showed that the approach is scalable regarding the initial graph density and the length of time-series.

In this work, we preprocessed the dataset to penalize high-frequency signals. Next step is to include a graph filter [30] into the learning algorithm to benefit from structural information when dealing with the problem of continuously active nodes. This solution should improve the robustness of the algorithm to the high-frequency noise that we discussed at the end of Section 5.

Our work opens new avenues for future work in dynamic graph-structured data analysis. For example, the proposed approach could be used for automatic event detection and monitoring in crowdsourced streaming data.

The resulting framework could be used as a tool in social sciences for collective memory and behavior understanding. The distributed implementation allows analyzing a significant amount of crowd-generated data to include more subjects into the experiments and reduce bias to certain social groups.

7 TOOLS, IMPLEMENTATION, CODE AND ONLINE VISUALIZATIONS

All the learned graphs (overall September-April, monthly activity, and localized events) are available online to influence further exploration and analysis. For graph visualization we used open source software package gephi [5] and layout ForceAtlas2[21]. We used Apache Spark GraphX [31], [16] for graph learning implementation and graph analysis. The presented results can be reproduced using the code for Wikipedi dynamic graph learning, written in Scala. In case of the need for another time period analysis of the Wikipedia, the data pre-processing can be done using the Python code.

ACKNOWLEDGMENTS

TBD: SWISS FUNDING

We would like to thank Michael Defferrard for fruitful discussions and useful suggestions.

The research leading to these results has received funding from the European Union’s H2020 Framework Programme (H2020-MSCA-ITN-2014) under grant agreement no 642685 MacSeNet.

REFERENCES

[1] Jan Assmann and John Czaplicka. 1995. Collective memory and cultural identity. New German Critique 65 (1995), 125–133.
[2] Ching-ming Au Yeung and Adam Jatowt. 2011. Studying how the past is remembered: towards computational history through large scale text mining. In Proceedings of the 26th ACM international conference on Information and knowledge management. ACM, 1231–1240.
[3] Jeffrey Andrew Barrash. 2016. Collective Memory and the Historical Past. University of Chicago Press.
[4] Alain Barrat, Marc Barthelemy, and Alessandro Vespignani. 2008. Dynamical processes on complex networks. Cambridge university press, Chapter 12, 291–293.
[5] Mathieu Bastian, Sebastien Heymann, Mathieu Jacomy, et al. 2009. Gephi: an open source software for exploring and manipulating networks. Icwsm 8 (2009), 361–362.
[6] Kirell Mael Benzi. 2017. From recommender systems to spatio-temporal dynamics with network science. EPFL, Chapter 5, 97–98.
[7] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. Journal of machine learning research 3, Jan (2003), 993–1022.
[8] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment 2008, 10 (2008), P10008.
[9] Alyn Coman, Adam D Brown, Jonathan Koppel, and William Hirst. 2009. Collective memory from a psychological perspective. International Journal of Politics, Culture, and Society IJPS 22, 2 (2009), 125–141.
[10] Nan Du, Mehrdad Farajtabar, Amr Ahmed, Alexander J Smola, and Le Song. 2015. Dirichlet-hawkes processes with applications to clustering continuous-time document streams. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 219–228.
[11] Paul Embrechts and Matthias Kirchner. 2016. Hawkes graphs. arXiv preprint arXiv:1601.01879 (2016).
[12] Michela Ferron and Paolo Massa. 2011. Studying collective memories in Wikipedia. Journal of Social Theory 3, 4 (2011), 449–466.
[13] Michela Ferron and Paolo Massa. 2012. Psychological processes underlying Wikipedia representations of natural and manmade disasters. In Proceedings of the Eighth Annual International Symposium on Wikis and Open Collaboration. ACM, 2.
[14] Wikmedia Foundation. 2016. Wikimedia Downloads: Database tables as sql.gz and content as XML files. (May 2016). https://dumps.wikimedia.org/other/ Accessed: 2016-May-16.

1https://github.com/mizvol/WikiBrain
2https://github.com/kikohs/wiki
3https://github.com/kikohs/wiki
4https://github.com/mizvol/WikiBrain
5https://github.com/mizvol/WikiBrain
[15] Ruth García-Gavilanes, Anders Mollgaard, Milena Tsvetkova, and Taha Yasseri. 2017. The memory remains: Understanding collective memory in the digital age. *Science Advances* 3, 4 (2017), e1602368.

[16] Joseph E Gonzalez, Reynold S Xin, Ankur Dave, Daniel Crankshaw, Michael J Franklin, and Ion Stoica. 2014. GraphX: Graph Processing in a Distributed Dataflow Framework. In *OSDI*, Vol. 14. 599–613.

[17] Maurice Halbwachs. 2013. *Les cadres sociaux de la mémoire*. Albin Michel.

[18] Alan G Hawkes. 1971. Spectra of some self-exciting and mutually exciting point processes. *Biometrika* 58, 1 (1971), 83–90.

[19] Xinran He, Theodoros Rekatsinas, James Foulds, Lise Getoor, and Yan Liu. 2015. Hawkestopic: A joint model for network inference and topic modeling from text-based cascades. In *International Conference on Machine Learning*. 871–880.

[20] Donald Olding Hebb. 2005. *The organization of behavior: A neuropsychological theory*. Psychology Press.

[21] Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. 2014. ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software. *PloS one* 9, 6 (2014), e98679.

[22] Vassilis Kalofolias, Andreas Loukas, Dorina Thanou, and Pascal Frossard. 2017. Learning time varying graphs. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[23] Nattiya Kanhabua, Tu Ngoc Nguyen, and Claudia Niederée. 2014. What triggers human remembering of events? A large-scale analysis of catalysts for collective memory in Wikipedia. In *Digital Libraries (JCDL)*. 2014 IEEE/ACM Joint Conference on. IEEE, 341–350.

[24] Alex Kirell, Míz Volodomyric, Ricaud Benjamin, and Vanderghyestn P. 2017. Wikipedia time-series graph. (Sept. 2017). https://doi.org/10.5281/zenodo.886484

[25] Alexander Konovalov, Benjamin Strauss, Alan Ritter, and Brendan O'Connor. 2017. Learning to Extract Events from Knowledge Base Revisions. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1007–1014.

[26] Thomas Josef Liminger. 2009. Multivariate hawkes processes. (2009).

[27] Charalampos Mascolo, Isabel Valera, and Manuel Gomez-Rodriguez. 2017. Modeling the Dynamics of Learning Activity on the Web. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1421–1430.

[28] Ashwin Pananjape, Robert West, Leila Zia, and Jure Leskovec. 2016. Improving website hyperlink structure using server logs. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 615–624.

[29] Minglai Shao, Jianxin Li, Peng Chen, Hongyi Huang, Shuai Zhang, and Xunxun Chen. 2017. An Efficient Approach to Event Detection and Forecasting in Dynamic Multivariate Social Media Networks. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1631–1639.

[30] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine* 30, 3 (2013), 83–98.

[31] Reynold S Xin, Joseph E Gonzalez, Michael J Franklin, and Ion Stoica. 2013. Graphx: A resilient distributed graph system on spark. In *First International Workshop on Graph Data Management Experiences and Systems*. ACM, 2.

[32] Burcu Yucesoy and Albert-László Barabási. 2016. Untangling performance from success. *EPJ Data Science* 5, 1 (2016), 17.