



Capturing Social Data Evolution Using Graph Clustering

Maria Giatsoglou and Athena Vakali • Aristotle University

The fast and unpredictable evolution of social data poses challenges for capturing user activities and complex associations. Evolving social graph clustering promises to uncover the dynamics of latent user and content patterns.

Massive user involvement in Web applications renders the Web a huge, constantly evolving social data repository that embeds associations among different modes. Such modes represent types of social application “actors,” such as users, resources, metadata, and groups. Evolving social data clustering has two seminal objectives: to identify clusters of latent entities and to track their evolution by efficiently modeling and analyzing associations across time. Achieving these objectives requires that we address the challenges inherent in massive data sizes, heterogeneous coevolving data, complex structure data associations, and frequent data updates.

Uncovering hidden relations and entity clusters based on their interaction patterns is challenging on its own and is typically addressed via static social data clustering. This approach aggregates interactions from a specific time period in a unique dataset and detects groups of entities with similar activity or usage patterns based on some optimization criterion. Given today’s fast social (Web) data streams, we need efficient methods for exploiting more fine-grained and ultimately richer information to capture such patterns’ changes and important aspects about their evolution. The problem encompasses a wide audience, including computer scientists; developers and market entrepreneurs, who can leverage results for more accurate recommendations, user behavior prediction, and social media monitoring for strategic marketing; and policy makers and journalists, who can harvest

valuable insights about the evolution of user interaction patterns with regard to real-world events. In parallel, drastic cluster changes can indicate special events (for example, a shift in users’ activity patterns might reflect the emergence of a new popular topic).

Here, we briefly survey approaches for the evolving social data clustering problem, focusing on how they interpret social data evolution and highlighting their seminal characteristics, rather than providing a detailed clustering algorithm survey. Based on this main axis, we identify four seminal evolving clustering methodologies: *sequential mapping*, *temporal smoothing*, *milestone detection*, and *incremental adaptation*. Because we focus on social relational data, we discuss graph-based clustering algorithms – known as *community-detection algorithms* – that represent associations on a graph or tensor model and also identify evolving communities as latent groups that usually span a finite time frame while undergoing various structural and contextual changes.

Social Associations, Time Granularity, and Graph Structures

Social networking applications engage users in multiple activities with other entities, generating associations among them that are either explicit (for example, a “friend of” relationship between users) or implicit (such as an “uploaded by same user” relationship among resources). Associations are often multipartite – that is, they involve multiple entities (user u_1 comments

on user u_2 's post p). For simplicity, however, these associations are often projected onto typical bipartite associations between pairs of entities from the same or different modes.

We can best capture social data with a graph $G(V, E)$ that represents social entities with a set of nodes (V) and their associations with a set of edges (E). Connectivity can be encoded in an adjacency or similarity matrix. *Hypergraphs* – generalized graphs representing multipartite relationships with hyper-edges connecting more than two nodes – and their projections on usually unipartite or bipartite simple graphs are common static structures for complex social data modeling.¹

Social data change states across time through processes such as the introduction of new social entities or associations. Successive states can aggregate such information, but we often model data under the hypothesis that entities disappear if they're not involved in recent activities or, similarly, that associations disappear or are less intense. Evolving social data representation structures, unlike static ones, model different states in successive time steps, often defined by the data's sampling rate. As Figure 1 shows, we can model evolving social data as a three-layered stream G of successive snapshot graphs representing data associations at a given time-granularity setting.

The *snapshot* layer captures evolving social data as a sequence of (static) graph snapshots. We can analyze each snapshot individually and then perform joint analysis on all the snapshot results to infer how data is evolving.^{2,3}

The *segment* layer has a more compact representation structure, with an online graph stream partitioned into segments⁴ comprising successive similar snapshots. These snapshots' similarity depends on factors such as the number of common

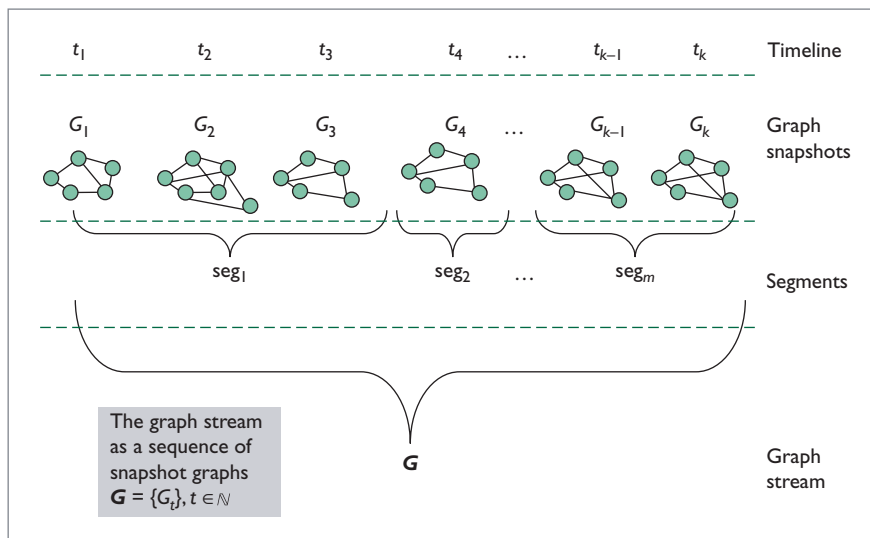


Figure 1. Layers for evolving social data graph structures. We can model evolving social data as a three-layered stream G of successive snapshot graphs representing data associations at a given time-granularity setting.

nodes or edges and the encoding cost for graph stream compression. Segment connectivity is often captured by *tensors* – that is, generalized matrices with more than two dimensions. In Figure 1, for example, representing segment *seg1* with snapshots $G_1, G_2,$ and G_3 as members, we can use a 3D tensor to store associations among a given snapshot's nodes in two dimensions and represent associations across different time step snapshots with the third dimension. More coarse-grained 2D *adjacency/similarity matrices* can represent segments, but could sacrifice snapshot individuality in favor of storage space efficiency and analysis complexity. In such matrices, two nodes are connected once they're associated in any segment's snapshot, with edges weighted based on their aggregated activities' intensity. We can place different emphasis on links on the basis of their creation date, thus promoting the most recent link and leading to a *time-aggregate adjacency matrix*.⁵

Finally, the *stream* layer directly represents the graph stream. Using appropriate edge aggregation techniques,⁵ we can model a graph stream as a tensor updated on an

upcoming snapshot or as a simple matrix. Another relevant structure is the *multigraph*, whose edges encode both associations and temporal information.⁶ The multigraph is updated upon a new association's arrival (*single update*) rather than on a graph snapshot (*batch update*). Because updates arrive arbitrarily, no predefined time steps exist in a multigraph.

An alternative streaming approach many incremental adaptation-driven algorithms follow involves modeling associations up to a point with a graph, deriving a data's clustering, and then directly updating the clustering structure instead of the graph on an unbounded individual update stream (*change stream*).

Evolving Social Data Clustering

Researchers have addressed evolving social data clustering via different methodologies that share some common ground. That is, they all approach social data clustering as an evolving process across time, assuming that social data continuously undergo changes that generate successive correlated clustering states and aiming to extract some

Table 1. Overview of data clustering methodologies.

Methodology	Data model	Updating scheme	Evolution aspect	Temporal dependency	Recomputation requirement
Sequential mapping-driven evolving clustering	Snapshot graphs/tensors	Batch	Tracks individual cluster evolution	Independent clustering at each time step	Required
Temporal-smoothing-driven evolving clustering	Snapshot graphs/tensors	Batch	Assumes clusters evolve smoothly from the previous model and incorporates deviation as a regularization term	Captures long-term changes in clusters' composition and ignores short-term concept drifts	Required
Milestone detection-driven evolving clustering	Snapshot graphs/tensors and segments	Batch	Generates a new clustering model on an identified drastic change in the data structure	Identifies time points when the clustering structure drastically changes	Required
Incremental adaptation-driven evolving clustering	Snapshot graphs/tensors and change stream	Single/batch	Builds new clusters by adapting the previous ones	Identifies individual clusters' evolution through new data-dependent adaptation processes	Not required

kind of knowledge about identified patterns' evolution.

To address different aspects of evolution, we can formulate the evolving social data clustering problem in a generic context as follows: In a given social application's context, considering a set of relevant entities and an unbounded stream of new entity associations, we define an appropriate data model and updating scheme (for example, single/batch). The problem is to support an "evolving" data clustering structure by capturing latent data patterns at any given time based on some optimization criterion, while "exploiting" the data's evolution under some hypothesis.

To address this seminal problem for each methodology, we look at the following characteristics: data model, updating scheme, evolution aspect, the temporal dependency of clusters, and clusters' recomputation requirement (see Table 1). We can use these characteristics to create the following classifications for the methodologies we surveyed.

Sequential Mapping-Driven Evolving Clustering

SM-EC occurs in two steps: the first clusters data on individual

graph snapshots, and the second uncovers transitions across time by analyzing correlations between successive snapshot clusters. This method addresses data evolution by mapping each community to its predecessor and successor using special similarity measures and assuming relatively "smooth" community evolution. SM-EC adopts traditional static community-detection algorithms, such as *clique percolation*² (CP) and *random walks* (RW).³ It identifies transitions in communities (such as birth, death, merge, or split) using node-centric, cross-community similarity measures (for example, node overlap). Because real-world data are often ambiguous and noisy, SM-EC's process of extracting communities at each time step independently results in community structures with high temporal variation.

Temporal-Smoothing-Driven Evolving Clustering

TS-EC (also known as *evolutionary clustering*) assumes that the clustering of new data shouldn't deviate much from previous clustering structures. Thus, it leverages clustering history to maximize temporal

smoothness and results in a "smooth" cluster evolution that's more robust to noise. This methodology encompasses some significant indicative approaches.

Traditional clustering revisited for an evolutionary setting (TCR) refers to initial evolutionary clustering efforts that modify existing static clustering algorithms (such as *k*-means) to remember previous data states.⁷

Spectral clustering (SC) introduces and minimizes clustering cost functions by combining graph-based measures (such as a normalized graph cut) with temporal smoothness regularization terms.⁸ This approach derives soft (that is, having continuous values) clustering matrices that comprise the eigenvectors of some history-aware similarity matrix version, and applies *k*-means clustering in a lower-dimensional space.

Block-model approximation (BM) acts in multimode networks with cross-mode interactions to approximate the interactions among nodes as interactions among blocks comprising nodes of the same mode (that is, single-mode communities).⁹ This is similar to block modeling. Nodes' soft (probabilistic) community membership is expressed in the

community indicator matrix, which is iteratively approximated using the previous time step's matrix as a regularization term.

Nonnegative-matrix/tensor factorization (NTF) discovers communities as latent variables by jointly maximizing the fit to the observed data and the temporal evolution.¹⁰ It operates on multiple tensors that capture cross-mode entities' interactions. Unlike the BM approach, it simultaneously approximates these interactions using a common nonnegative core tensor (that is, multimode communities) and nonnegative factors – one for each mode – thus expressing soft community membership.

Recomputing a clustering structure for each new graph snapshot, however, could result in numerous intermediate clustering models, some with insignificant differences.

Milestone Detection-Driven Evolving Clustering

MD-EC operates on a graph stream by trying to find characteristic change points in time to segment it and identify communities within each segment. This lets the algorithm maintain clustering structures for each segment rather than for each snapshot. Detecting a new change point indicates a discontinuity in time, signaling a new segment that comprises snapshots with similar connectivity according to some criterion. Important relevant approaches include *information-theory-based clustering* (ITC⁴), which operates on a bipartite graph stream, and *modularity-optimization clustering* (MOC¹¹), which targets weighted directed graphs.

Previous methodologies require recomputing each snapshot's community structure, which could prove inefficient considering social data's frequent update rates and large sizes, not to mention the data model's complexity (for example, high-order tensors).

Incremental Adaptation-Driven Evolving Clustering

IA-EC addresses the aforementioned challenge under a streaming data scenario by incrementally creating and maintaining a time-aware model that best approximates the original data. IA-EC updates clusters by exploiting the current clustering structure and newly observed data under a single or batch update scheme. IA-EC encompasses several approaches.

Incremental-tensor factorization (ITF¹²) summarizes complex data streams under a batch update scheme. Such data streams are characterized by multipartite relations in some updatable, lower-order core tensors, along with their respective transformation projections.

Incremental-spectral clustering (ISC¹³) maintains and incrementally updates a generalized eigenvalue system for graph partitioning based on normalized cut. At a microscopic level, it focuses on the effect single updates (node addition/deletion or similarity changes) have on eigenvalues and eigenvectors, and on how these can be approximated to decrease cluster recalculation costs.

Incremental k-clique clustering (ICC¹⁴) detects communities by finding the maximal k -clique set of the data's association graph and constructing a *depth-first search* (DFS) forest on a k -clique adjacency-derived graph. Based on a change stream, this approach incrementally updates the clique set and DFS forest locally.

Table A in the Web appendix at [//URL to come//](#) gives a more comprehensive view of these evolving social data clustering approaches. It highlights the crucial issues of algorithms and scope, structures and models, required parameters, supported update types, and datasets used for demonstration.

Current Practices and Future Trends

Each of the existing approaches is more effective at addressing different evolving social data clustering challenges. Here, we briefly discuss their current state and future trends based on some seminal challenge axes.

The complexity of social data in today's Web demands methods that can efficiently exploit multiple modes and multipartite associations.^{9,10,12} Moreover, engaging in joint analysis for different coevolving associations seems to be a promising approach for uncovering more realistic patterns, as indicated by the NMF approach's application to a user-interest prediction task.¹⁰ Derived communities' interpretations vary, given that associations can be jointly analyzed to detect single⁹ or multimode^{10,12} and hard or soft membership communities. The tensor model emerges as a graph's natural extension for multipartite associations, and tensor decomposition is often employed for complex data clustering. Identifying the most suitable explicit and implicit associations in a social application context and jointly analyzing them are challenging future steps.

The large sizes of social data make clustering algorithms' scalability critical. From the datasets used in the referenced works, we observe that the largest number of edges corresponds to the NMF approach (up to a million relations).¹⁰ However, this doesn't prove its superiority over the other approaches, because some approaches provided information only on the number of nodes (and didn't discuss time complexity). Initial TS-EC approaches⁷ seem inefficient for large datasets owing to their $O(n^2)$ complexity, whereas some recent iterative tensor/matrix decomposition approaches are efficient for sparse tensors or matrices. For example, in the NMF approach,¹⁰ the complexity per iteration is linear

as regards the number of nonzero entries in all tensors. For settings in which the sparseness hypothesis is unrealistic, iteration steps might prove too time consuming, and different approaches are needed. Possible future directions involve calculating approximate solutions, such as an extension to the NMF approach in which the size of entities' interaction tensors is bounded by some fixed threshold, based on a sampling scheme devised to "forget" long-term inactive entities.¹⁵ Also, some researchers have proposed matrix factorization on a small subset of high-degree nodes exploiting social networks' power-law property.⁹ As we discuss next, IA-EC is also generally scalable.

Frequent data updates are best addressed by IA-EC, which aims mainly to efficiently adapt clusters rather than recalculate them, while sometimes sacrificing clustering quality.¹³ The scalability of IA-EC approaches varies. The ISC approach,¹³ for example, has a cluster-updating complexity of $O(n)$, linear to the number of nodes. The complexity of the more efficient ICC approach¹⁴ varies – depending on the update's type – from $O(1)$ to $O(\max(\log L, |CN|^{3^{CN/3}}))$, where L is the maximal number of cliques and CN is the number of common neighbors between the new edge's source and destination nodes. Unlike these approaches, the ITF approach supports heterogeneous networks, but its complexity depends on the product of the number of nodes for each mode. To improve scalability, the approach's authors propose an interaction sampling strategy.¹²

The detected evolution of communities differs significantly in snapshot-based and change-stream-based approaches. Snapshot-based approaches require setting the snapshot's time interval to track communities at a more macroscopic level. In change-stream-based approaches,

individual changes directly affect clustering, thus identifying fine-grained cluster dynamics on observing successive clustering structures (for example, a new association could cause a cluster to split, whereas other changes might have insignificant effects). MD-EC offers a compromise between the two approaches in that, although time-step duration is prespecified, granularities are "adjusted" over time into more meaningful ones (via segmentation) based on observed data fluctuations.

Finally, algorithm evaluation as regards real-world datasets is challenging even for static clustering owing to a lack of explicit labels (ground truth). Communities are often evaluated in terms of cohesion, applying metrics such as *conductance*,³ *coverage*,^{2,3} and *modularity*³ based on the assumption that interactions are much denser among community members than between different communities' members. It's even more challenging to evaluate an algorithm's ability to identify evolution in real-world communities; the most common approach is to verify the validity of identified evolution patterns by manually inspecting a community's members or combining predefined facts. IA-EC approaches are evaluated against static approaches in terms of time efficiency per cluster adaptation, while approximate IA-EC approaches are also evaluated in terms of clustering quality. TS-EC approaches are generally evaluated by proving their robustness to noise, by calculating measures such as the entropy between successive clustering distributions.³

Another common TS-EC evaluation technique involves generating synthetic datasets with predefined community structure, simulating nodes' community membership changes and new interactions across different time steps, and inserting random noise.⁸ Clusters identified on these synthetic datasets are evaluated

against ground truth based on accuracy metrics such as normalized mutual information,⁹ which aims to differentiate between short-term concept drifts and long-term community changes. A promising future direction is to combine knowledge acquired through social network dynamics research to produce more realistic synthetic datasets for evaluation, based on typical network evolution patterns.

Raw social Web data encompass valuable latent information that's relevant with users' societal perceptions and reactions to regular as well as to groundbreaking events. Here, we've formulated the problem of evolving social data clustering and presented the data's evolution aspects as revealed by different relevant existing approaches. The concept of evolving communities is dominant in the presented work, and an overview of community detection algorithms highlights advantages and shortcomings in the different approaches.


Several issues should be considered in terms of evolving social data clustering algorithms, including user activity fluctuations across time, large data volumes and storage space constraints, data complexity, and demand for real-time-streaming-data-oriented applications. Evolving social data clustering via graph models can greatly assist application providers in offering end users valuable services that are adaptive to trends (for example, recommendations). Moreover, evolving social data clustering can greatly enhance users' interests and opinion mining and support opinion makers and authorities in understanding the public pulse. Future work in this area could leverage social data from different types of social Web applications jointly to obtain a global and more reliable conclusion in terms of Web and real-world phenomena and events. □

References

1. P. Mika, "Ontologies Are Us: A Unified Model of Social Networks and Semantics," *Proc. Int'l Semantic Web Conf. (ISWC 05)*, Springer, 2005, pp. 522–536.
2. G. Palla et al., "Quantifying Social Group Evolution," *Nature*, vol. 446, no. 7136, 2007, pp. 664–667.
3. Y. Lin et al., "Blog Community Discovery and Evolution Based on Mutual Awareness Expansion," *Proc. 2007 IEEE/WIC/ACM Int'l Conf. Web Intelligence (WI 07)*, IEEE CS, 2007, pp. 48–56.
4. J. Sun et al., "GraphScope: Parameter-Free Mining of Large Time-Evolving Graphs," *Proc. 13th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD 07)*, ACM, 2007, pp. 687–696.
5. H. Tong et al., "Proximity Tracking on Time-Evolving Bipartite Graphs," *Proc. SIAM Int'l Conf. Data Mining (SDM 08)*, SIAM, 2008, pp. 704–715.
6. Q. Zhao et al., "Temporal and Information Flow-Based Event Detection from Social Text Streams," *Proc. 22nd Conf. Artificial Intelligence (AAAI 07)*, vol. 2, AAAI Press 2007, pp. 1501–1506.
7. D. Chakrabarti et al., "Evolutionary Clustering," *Proc. 12th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD 06)*, ACM, 2006, pp. 554–560.
8. Y. Chi et al., "Evolutionary Spectral Clustering by Incorporating Temporal Smoothness," *Proc. 13th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD 07)*, ACM, 2007, pp. 153–162.
9. L. Tang et al., "Identifying Evolving Groups in Dynamic Multi-Mode Networks," *IEEE Trans. Knowledge and Data Eng.*, vol. 24, no. 1, 2011, pp. 72–85.
10. Y. Lin et al., "MetaFac: Community Discovery via Relational Hypergraph Factorization," *Proc. 15th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD 09)*, ACM, 2009, pp. 527–536.
11. D. Duan et al., "Community Mining on Dynamic Weighted Directed Graphs," *Proc. Workshop Complex Networks in Information & Knowledge Management (CNIKM 09)*, ACM, 2009, pp. 11–18.
12. J. Sun et al., "Incremental Tensor Analysis: Theory and Applications," *ACM Trans. Knowledge Discovery from Data (TKDD)*, vol. 2, no. 3, 2008, article no. 11, pp. 1–37.
13. H. Ning et al., "Incremental Spectral Clustering by Efficiently Updating the Eigen-System," *Pattern Recognition*, vol. 43, no. 1, pp. 113–127.
14. D. Duan et al., "Incremental K-Clique Clustering in Dynamic Social Networks," *Artificial Intelligence Rev.*, vol. 38, no. 2, 2011, pp. 129–147.
15. C. Bockermann et al., "Stream-Based Community Discovery via Relational Hypergraph Factorization on Evolving Networks," *Proc. Workshop on Dynamic Networks and Knowledge Discovery (DyNaK 10)*, CEUR Workshop Proc., 2010, pp. 41–52.

Maria Giatsoglou is a PhD student in the Department of Informatics at the Aristotle University of Thessaloniki, Greece. Her research interests include social network analysis, recommender systems, and community detection. Giatsoglou has a diploma in electrical and computer engineering and an MSc in information systems from Aristotle University. Contact her at mgiatsoglou@csd.auth.gr.

Athena Vakali is an associate professor in the Department of Informatics at the Aristotle University of Thessaloniki, Greece. Her research interests include Web usage mining, content-delivery networks, and social data mining and analysis. Vakali has a PhD in computer science from Aristotle University. Contact her at avakali@csd.auth.gr.

 Selected CS articles and columns are also available for free at <http://ComputingNow.computer.org>.