A Combinatorial Multi-Armed Bandit based method for Dynamic Consensus Community Detection in Temporal Networks

Domenico Mandaglio and Andrea Tagarelli

Dept. Computer Engineering, Modeling, Electronics, and Systems Engineering (DIMES), University of Calabria, Italy
`d.mandaglio@dimes.unical.it, andrea.tagarelli@unical.it`

Abstract. Community detection in temporal networks is an active field of research, which can be leveraged for several strategic decisions, including enhanced group-recommendation, user behavior prediction, and evolution of user interaction patterns in relation to real-world events. Recent research has shown that combinatorial multi-armed bandit (CMAB) is a suitable methodology to address the problem of dynamic consensus community detection (DCCD), i.e., to compute a single community structure that is conceived to be representative of the knowledge available from community structures observed at the different time steps.
In this paper, we propose a CMAB-based method, called CreDENCE, to solve the DCCD problem. Unlike existing approaches, our algorithm is designed to provide a solution, i.e., dynamic consensus community structure, that embeds both long-term changes in the community formation and newly observed community structures. Experimental evaluation based on publicly available real-world and ground-truth-oriented synthetic networks, with different structure and evolution rate, has confirmed the meaningfulness and key benefits of the proposed method, also against competitors based on evolutionary or consensus approaches.

1 Introduction

Community detection and evolution in temporal networks has been largely studied in the last few years, mainly focusing on graph-based unsupervised learning paradigms (e.g., [3,10,8,24]). Nonetheless, detecting and tracking the evolution of the change events that occur in the communities remains challenging [5], which is partly due to the uncertainty and dynamicity underlying the different types (e.g., birth/death, growth/decay, merge/split) and evolution rates of structural changes in time-evolving network systems.

In this regard, we have recently explored the opportunity of adopting the multi-armed bandit (MAB) paradigm, which is conceived to learn how to perform actions in an uncertain environment [17]. Indeed, in the problem under consideration, the uncertainty is inherent to the temporal network system and the structural changes of its communities, while actions correspond to node assignments to communities. Moreover, each action is associated with a notion of "reward" that determines how much benefit is gained by (a set of) node assignments to communities. Within this view, MAB methods are well-suited to model the *exploitation-exploration* trade-off [21,13], i.e., balancing between making decisions that yield high current rewards or making decisions that sacrifice current gains with the prospect of better future rewards. Moreover, to deal with choosing *a set of* actions, i.e., a set of community assignments that constitute a whole community-structure, a particular extension of MAB problems is needed, which is called *combinatorial multi-armed bandit* (CMAB) [4,7].

In this work, we focus on the *dynamic consensus community detection* (DCCD) problem, that is, given a sequence of temporal snapshots of a time-evolving network, we want to compute a single community structure to be representative of the knowledge available from community structures detected in the different snapshot networks. Remarkably, unlike in *consensus community detection* [14,22], the knowledge on the community structures from which a consensus must to be inferred is not available at a given initial time, but it evolves over time along with the associated temporal network. In this respect, here we follow the directions outlined in [17] for the CMAB-based DCCD problem, and propose a fully defined instantiation of the algorithmic scheme.

Note that existing approaches to related problems involving a notion of community representative in temporal networks [6,12] may suffer from restrictions on the network model, such as fixed set of nodes and number of communities for each snapshot of the temporal network [12], or on selected types of community dynamics [6]. By contrast, our proposed approach does not incur such issues.

Our contributions can be summarized as follows:

- We develop CreDENCE – **C**MAB-based **D**ynamic Cons**EN**sus **C**ommunity D**E**tection method. To achieve the exploration-exploitation trade-off, our algorithm is designed to balance over time between the need for embedding long-term changes observed in the community formation and the need for capturing short-term effects and newly observed community structures. Moreover, CreDENCE is conceived to be versatile in terms of the static community detection approach used to identify the communities at each snapshot, and robust in terms of a number of parameters that control the CMAB-learning rate, temporal smoothness factors, and the node-relocation bias.
- We provide insights into technical as well as computational complexity aspects of CreDENCE; upon this, we propose an enhancement of CreDENCE to ensure its linear complexity in the size of the temporal network.
- Our experimental evaluation was conducted using 5 real-world networks and ground-truth-oriented synthetically generated networks, including comparison with 3 competing methods. Results have provided useful indications about the quality of the consensus solutions obtained by CreDENCE, which is able to cope with temporal networks having different evolution rates.

2 Problem statement

We are given a set $\mathcal{V}$ of *entities* (e.g., users in a social environment) and a *temporal network* $\mathcal{G}$ as a series of graphs over discrete time steps $(G_1, G_2, \ldots, G_t, \ldots)$, where $G_t = \langle V_t, E_t \rangle$ is the graph at time t, with set of nodes V_t and set of undirected edges E_t. We denote with $\mathcal{G}_{\leq t}$ a *series of graphs observed until time t*. Each node in V_t corresponds to a specific instance from the set $\mathcal{V}_t \subseteq \mathcal{V}$ of entities that occur at time t. The snapshot graphs can share different subsets of entities.

Given any G_t, we denote with $\mathcal{C}^{(t)}$ a *community structure* for G_t, which is a set of non-overlapping communities, and is assumed to be unrelated to any other $\mathcal{C}^{(t')}$ $(t' \neq t)$, both in terms of number of communities and set of entities involved. Let $\mathcal{E}_{\leq t} = \{\mathcal{C}^{(1)}, \ldots, \mathcal{C}^{(t)}\}$ be a *dynamic ensemble* at time t, i.e., a set of community structures associated to the snapshot graphs. We consider the following problem:

*Problem 1 (***Dynamic Consensus Community Detection (DCCD)***).* *Given a temporal graph sequence $\mathcal{G}_{\leq t}$ and associated dynamic ensemble $\mathcal{E}_{\leq t}$, for any time $t \geq 1$ compute a community structure, called dynamic consensus community structure and denoted as $\mathcal{C}^*_{\leq t}$, which is designed to encompass the information from $\mathcal{G}_{\leq t}$ to be representative of the knowledge available in $\mathcal{E}_{\leq t}$.*

Given $\mathcal{G}_{\leq t}$ and $\mathcal{E}_{\leq t}$, the dynamic consensus being discovered over time can be represented as a matrix $\mathbf{M}$ we call ***dynamic co-association*** (or ***consensus***) ***matrix*** (DCM). Its size is initially $\mathcal{V}_t \times \mathcal{V}_t$ with $t = 1$, and at a generic time t is $|\mathcal{V}| \times |\mathcal{V}|$. The (i, j)-th entry of $\mathbf{M}$, denoted as m_{ij}, stores the probability of co-association for entities $v_i, v_j \in \mathcal{V}$, i.e., the probability that v_i and v_j are assigned to the same community, in the observed timespan.

Given the incremental nature of Problem 1, unlike in conventional consensus community detection [14,22], we want to avoid (re)computation of the consensus from scratch at any time t. We also do not want to depend on any mechanism of tracking of the evolution of communities [5]. More importantly, the dynamic consensus community structure should be able to embed long-term changes in the community formation as well as to capture short-term effects and newly observed community structures. To address Problem 1, in [17] we proposed a CMAB-based methodology, whose principles are recalled in the next section.

2.1 Dynamic consensus community detection as a CMAB problem

Review of CMAB. We are given m *base arms*, where each arm i is associated with a set of random variables $\{X_{i,t} \mid 1 \leq i \leq m, t \geq 1\}$, where $X_{i,t} \in [0, 1]$ indicates the random outcome of *triggering*, or playing, the i-th arm in the t-th round. The random variables $\{X_{i,t} \mid t \geq 1\}$ of the i-th arm are independent and identically distributed. Moreover, in a *non-stationary context*, those variables may change [9]. Variables of different arms may not be independent.

At each round t, a *superarm* (a set of base arms) A is chosen and the outcomes of the random variables $X_{i,t}$, for all $a_i \in A$, are revealed. Moreover, the base arms belonging to A may probabilistically trigger other base arms not in A [4,7], thus revealing their associated outcomes. Playing a superarm A at round t gives a reward $R_t(A)$ modeled as a random variable, which is a function of the outcomes of the triggered base arms. The objective of a CMAB method is to select at each round t the superarm A that maximizes the *expected reward* $\mathbb{E}[R_t(A)]$, in order to maximize the *cumulative expected reward* over all rounds. At each round, the bandit may decide to choose the superarm with the highest expected reward (exploitation) or to select a superarm discarding information from earlier rounds (exploration) with the aim of discovering the benefit from adopting some previously unexplored arm(s) [7,4].

Adaptation to DCCD. In our context, each pair of entities $\langle v_i, v_j \rangle$ in $\mathcal{G}_{\leq t}$ is a base arm and it is hypothetically associated with an unknown distribution (with unknown mean μ_{ij}) for the probabilities of co-association over time, whose mean estimate is the entry m_{ij} in DCM. Each observation of a community structure of a snapshot network can be considered as a sample from such distributions. Moreover, since the network and community structures can vary, the co-association distributions may also change their mean over time, thus the DCCD setting is non-stationary. Including a base arm $\langle v_i, v_j \rangle$ in a superarm corresponds to "*assign v_i and v_j to the same community at a given time*". If we denote with $c_i^{(t)}$ the community of v_i at round t, a superarm A at round t is a set of pairs $\langle v_i, v_j \rangle$ such that $c_i^{(t)} = c_j^{(t)}$.

According to the framework in [17], playing a superarm A at each round t consists of stochastic optimization that considers node relocations to neighbor communities. The stochastic nature of the process depends on both the random order with which we consider the node relocations and on the fact that, according to the optimization of a quality criterion, an improvement due to relocation is accepted with a certain probability. Intuitively, this allows us to account for uncertainty in the long-term overall quality improvement of the consensus due to local relocations at a given time; for instance, it is unknown if the relation that explains two users share the same community at a given time could become meaningless in subsequent times. After playing a superarm A, the rewards associated to the entity pairs (base arms) corresponding to the status of communities after the relocation phase, are revealed; these pairs include both the nodes that did not move from their community and the arms $\langle v_i, v_j \rangle$ triggered with the accepted relocations, i.e., such that node v_i was moved to the community of v_j. For the base arms that were neither selected nor triggered (i.e., pairs of nodes that were not in the same community before and after the relocation phase), we assume an implicit reward of zero that corresponds to the observation of the "no-coassociation" event. (This is in line with the possibility in CMAB of enabling the probabilistic triggering of *all* base arms.)

The reward of a superarm corresponds to the *quality* of the community structure at the end of the relocation phase, which is a non-linear function of the base arms' rewards. More specifically, we resort to *modularity* as quality criterion for a community structure. Let X_{ij}^t be the reward associated to the base arm corresponding to node pair $\langle v_i, v_j \rangle$ at time step t. The reward of the played superarm A (leading to a consensus structure $\mathcal{C}_{\leq t}^*$ after the stochastic relocation of nodes) can be defined in terms of the base arms' rewards as follows:

$$R_t(A) = \frac{1}{d(\mathcal{V}_{[1..t]})} \sum_{i,j} \sum_{\ell=1}^{t} \beta^{t-\ell} \Big(A_{ij}^l - \frac{k_i^\ell k_j^\ell}{d(\mathcal{V}_{[1..t]})} \Big) \delta(X_{ij}^t) \qquad (1)$$

where k_i^ℓ is the degree of v_i in the ℓ-th snapshot, A_{ij}^l is the (i,j)-th entry of the adjacency matrix of the ℓ-th snapshot graph, $d(\mathcal{V}_{[1..t]})$ is the total degree of the multiplex graph including snapshots from the first one to the t-th (i.e., $d(\mathcal{V}_{[1..t]}) = \sum_{\ell=1}^{t} \sum_{v \in V_\ell} d(v)$), $\beta \in (0,1)$, and $\delta(X_{ij}^t) = 1$ if $X_{ij}^t > 0$, 0 otherwise. The stochastic nature of the above defined reward is determined by the random variables X_{ij}^t. In Sect. 3.2, we will define a generalization of the above reward equation that allows us to focus on selected snapshots of the networks.

3 The *CreDENCE* method

To solve the dynamic consensus community detection problem, we develop a CMAB-based method called CreDENCE – **C**MAB-based **D**ynamic Cons**EN**sus **C**ommunity D**E**tection, which is sketched in Algorithm 1.

Initially, the dynamic consensus matrix **M** is set as an identity matrix (Line 1), which reflects that no information has been processed yet, and hence each entity-node has co-association with itself only. At each round t, the algorithm chooses to perform either exploration or exploitation, according to a given bandit strategy ($\mathcal{B}$). Intuitively, in the exploitation phase, we seed an oracle (i.e., a conventional method for community detection) with the mean estimates of co-association of the current DCM to infer the communities in the new snapshot graph observed at time t; by contrast, in the exploration phase, the new communities are identified using the t-th graph only. In either phase, the community structure generated at time t is finally used to produce a superarm that will correspond to the dynamic consensus community structure up to t ($\mathcal{C}_{\leq t}^*$).

Besides the involvement of a conventional community detection method $\mathcal{A}$ and a bandit strategy $\mathcal{B}$ to control the exploration-exploitation trade-off, we introduce a few parameters to ensure robustness in the algorithmic scheme of CreDENCE: (i) the learning rate α for the update of the mean estimates (i.e., m_{ij} entries), (ii) the relocation bias λ, and (iii) the temporal smoothness factor β and window size ω to control the amount of past knowledge for the step of node-relocations. Nonetheless, some of these parameters are interrelated, or reasonable values can be chosen as default.

Another remark on CreDENCE concerns its *incremental* nature: whenever a new step of evolution is observed, say at $T+1$, the last-update status of the DCM matrix along with $\mathcal{G}_{\leq T+1}$ will become the input for a further CMAB round.

3.1 Finding communities

At each round t, CreDENCE invokes a community detection method $\mathcal{A}$. This is just required to deal with (*static*) *simple graphs*. While in the exploration phase it directly applies to the snapshot graph G_t (Line 4), to handle the exploitation phase, the method should also be able to deal with *weighted* graphs: in this case, $\mathcal{A}$ is executed on the graph $G_{\mathbf{M}}$ (Line 7), which is built from the current DCM matrix in such a way that the edge weights in $G_{\mathbf{M}}$ correspond to the entries of **M** (Line 6). Next, from the obtained partitioning $\mathcal{C}_{\mathbf{M}}$ of $G_{\mathbf{M}}$ (Line 7), the knowledge about the community memberships of entity nodes in $\mathcal{C}_{\mathbf{M}}$ is used to

Algorithm 1 **C**MAB-based **D**ynamic Cons**EN**sus **C**ommunity D**E**tection (CreDENCE)

```
Input: Temporal graph sequence $\mathcal{G}_{\leq T}$ ($T \geq 1$), (static) community detection method
    $\mathcal{A}$, bandit strategy $\mathcal{B}$, learning rate $\alpha \in (0,1)$, relocation bias $\lambda \in [0,1]$, temporal
    smoothness $\beta \in (0,1)$, temporal window width $\omega \geq 1$.
Output: Dynamic consensus community structure $\mathcal{C}^*_{\leq T}$.
 1: $\mathbf{M} \leftarrow I_{|\mathcal{V}_1| \times |\mathcal{V}_1|}$
 2: for $t = 1$ to $T$ do
 3:   if $\mathcal{B}$ decides for EXPLORATION then
 4:     $\mathcal{C}^{(t)} \leftarrow$ findCommunities($G_t, \mathcal{A}$)
 5:   else {EXPLOITATION}
 6:     $G_{\mathbf{M}} \leftarrow$ buildDCMGraph($\mathbf{M}$)
 7:     $\mathcal{C}_{\mathbf{M}} \leftarrow$ partitionDCMGraph($G_{\mathbf{M}}, \mathcal{A}$)
 8:     $\mathcal{C}^{(t)} \leftarrow$ inferCommunities($G_t, \mathcal{C}_{\mathbf{M}}$)
 9:   end if
10:   $\mathcal{C}^*_{\leq t} \leftarrow$ project($\mathcal{C}^{(t)}, \mathcal{G}_{\leq t}$)
11:   $\mathcal{C}^*_{\leq t} \leftarrow$ evalRelocations($\mathcal{G}_{\leq t}, \mathcal{C}^*_{\leq t}, \lambda, \beta, \omega$)        {Using Eq. (3)}
12:   $\mathbf{M} \leftarrow$ updateDCM($\mathbf{M}, \mathcal{C}^*_{\leq t}, \alpha$)                  {Using Eq. (4)}
13: end for
14: return $\mathcal{C}^*_{\leq T}$
```

infer a community structure $\mathcal{C}^{(t)}$ on G_t (Line 8). Each community in $\mathcal{C}^{(t)}$ will have node set corresponding to exactly one community in $\mathcal{C}_{\mathbf{M}}$, and edge set consistent with the topology of G_t. Any entity v that newly appears in G_t (i.e., $v \in \mathcal{V}_t \wedge v \notin \mathcal{V}_{t'}$, $\forall t' < t$) and is disconnected will form a community in its own.

It should be noted that, although *any* method can in principle be used as $\mathcal{A}$, our preferred choice is towards efficient, modularity-optimization-based methods, such as [1]. This is motivated for consistency with our choice of using (multiplex) modularity as quality criterion in the (consensus) community structure refinement, as discussed next in Sect. 3.2.

3.2 Generating the dynamic consensus community structure

The dynamic consensus community structure $\mathcal{C}^*_{\leq t}$, for each t, is generated in two steps. The first step (Line 10) corresponds to a simple projection of the community memberships from $\mathcal{C}^{(t)}$ onto $\mathcal{G}_{\leq t}$. The second step (Line 11) corresponds to *stochastic refinement* of the candidate $\mathcal{C}^*_{\leq t}$ obtained at the previous step. This stochastic refinement is performed through local search optimization, which is designed to relocate some nodes from their assigned community in $\mathcal{C}^*_{\leq t}$ to a neighboring one by acting greedily w.r.t. a quality criterion.

As previously anticipated, one appropriate choice refers to *modularity*. However, to account for the multiplexity of $\mathcal{G}_{\leq t}$ as well as the dynamic aspects, we modify the definition of modularity to include the temporal window by which the modularity context is set. The reason behind this choice is twofold: (i) to focus on a limited number of latest snapshots of the network, and (ii) to reduce the computational burden in the local search optimization.

Given $\mathcal{C}^*_{\leq t}$, temporal-window width ω and temporal smoothness factor β, we denote with $d(\mathcal{V}_{[t-\omega+1..t]})$ the total degree of the multiplex graph including snapshots from the $(t-\omega+1)$-th to the t-th and, for any community c, $d_\ell(c)$ and $d_\ell^{\text{int}}(c)$ are the total degree and the internal degree of c, respectively, measured w.r.t. edges of the ℓ-th snapshot network only. We define the (ω, β)-*multiplex modularity* of $\mathcal{C}^*_{\leq t}$ as follows:

$$Q(\mathcal{C}^*_{\leq t}, \omega, \beta) = \frac{1}{d(\mathcal{V}_{[t-\omega+1..t]})} \sum_{c \in \mathcal{C}^*_{\leq t}} \sum_{\ell=t}^{t-\omega+1} \beta^{t-\ell} \left(d_\ell^{\text{int}}(c) - \frac{(d_\ell(c))^2}{d(\mathcal{V}_{[t-\omega+1..t]})} \right) \quad (2)$$

As previously mentioned, the meaning of β is to smooth the contribution of earlier snapshots in the computation of the quality of the dynamic consensus, i.e., lower values of β will penalize older snapshots. It is worth noting that β may take a role that is opposite to that of the learning rate α in Algorithm 1. Therefore, by default, we set $\beta = 1 - \alpha$.

The local search optimization, at any time t, evaluates the possible improvement in terms of modularity due to the relocation of nodes v_i that lay on the boundary of their assigned communities towards one of the communities that at time t contain nodes linked to v_i. By denoting with c_i the initial community of a boundary node v_i, and simplifying the modularity notation with function symbol Q, the *modularity variation*, denoted as ΔQ_i, corresponding to moving v_i to a neighbor community is as follows:

$$\Delta Q_i = Q(c_i \setminus \{v_i\}) - Q(c_i) + \max_{c_j \in NC_i^{(t)}} (Q(c_j \cup \{v_i\}) - Q(c_j)) \quad (3)$$

where $NC_i^{(t)}$ is the set of neighbor communities for node v_i at time t. If $\Delta Q_i > 0$, then there is a single chance to accept the relocation of v_i to c_j with probability $1 - \lambda e^{-\lambda \Delta Q_i}$, with $\lambda \in [0, 1]$ to control the bias towards relocations.

3.3 Updating the dynamic consensus

The DCM-update scheme in Algorithm 1 (Lines 12) follows a standard principle in reinforcement learning, whereby as the agent explores further, it is capable of updating its current estimate according to a general scheme of the form $newEstimate \leftarrow oldEstimate + \alpha(target - oldEstimate)$, which intuitively consists in moving the current estimate in the direction of a "target" value, with slope α. In our setting, we want to control the update of co-associations by subtracting a quantity α of resource from the co-associations of each node, at time t, and redistributing this quantity among the nodes in $c_i^{(t)}$, for each v_i. This redistribution corresponds to the *reward* of a single co-association, i.e., given v_i, the reward of assigning any v_j to the same community of v_i. Upon this, given $\alpha \in [0, 1]$ and any (i, j)-th entry of $\mathbf{M}$, we define the *update* equation as:

$$m_{ij}^{(t+1)} = m_{ij}^{(t)} + \alpha \left(\frac{1}{|c_i^{(t)}|} [v_j \in c_i^{(t)}] - m_{ij}^{(t)} \right) = \frac{\alpha}{|c_i^{(t)}|} [v_j \in c_i^{(t)}] + (1 - \alpha) m_{ij}^{(t)} \quad (4)$$

where $[x \in X]$ denotes the Iverson-bracket notation for the indicator function.

Properties of the update equation. It should be noted that the reward $1/|c_i^{(t)}|$ produces the effect of making it stronger the co-association between nodes belonging to smaller communities. This is consistent with a major finding in a recent study proposed in [11] whereby co-memberships of nodes in larger communities are statistically less significant (than in smaller ones), because members in such communities have limited influence upon each other in the network. A further reason to favor co-associations in smaller communities is to compensate for a typical bias relating to a tendency of producing large communities (e.g., resolution limit in modularity-optimization based methods).

Another property of the update rule in Eq. (4) is the *exponential smoothing of earlier actions*, with constant α [21], i.e., the update scheme leads to weight recently obtained rewards more heavily than earlier ones, and the reward of a past co-association between two nodes decreases exponentially in time.

Proposition 1. *Eq.* (4) *ensures that the rewards of past co-association between any two nodes* v_i, v_j *decreases by a factor* $(1-\alpha)^{t-s}$*, with* $s \leq t$*.*

PROOF. Let us assume that nodes v_i, v_j are assigned to the same community and remain therein over time. By repeated substitutions, we derive that:

$$\begin{aligned} m_{ij}^{(t+1)} &= \frac{\alpha}{|c_i^{(t)}|} + (1-\alpha)m_{ij}^{(t)} = \frac{\alpha}{|c_i^{(t)}|} + \Big(1-\alpha\Big)\left[\frac{\alpha}{|c_i^{(t-1)}|} + \Big(1-\alpha\Big)m_{ij}^{(t-1)}\right] = \\ &= \frac{\alpha}{|c_i^{(t)}|} + \frac{(1-\alpha)\alpha}{|c_i^{(t-1)}|} + (1-\alpha)^2 m_{ij}^{(t-1)} = \\ &= \frac{\alpha}{|c_i^{(t)}|} + \frac{(1-\alpha)\alpha}{|c_i^{(t-1)}|} + \ldots + \frac{(1-\alpha)^{t-1}\alpha}{|c_i^{(1)}|} + (1-\alpha)^t m_{ij}^{(1)} = \\ &= (1-\alpha)^t m_{ij}^{(1)} + \sum_{s=1}^{t}(1-\alpha)^{t-s}\frac{\alpha}{|c_i^{(s)}|}. \end{aligned}$$

Also, it can easily be shown that Eq. (4) ensures that $\mathbf{M}$ is a *stochastic matrix.*

3.4 Speeding up *CreDENCE*

The time complexity of the basic version of CreDENCE is determined by the update operations on $\mathbf{M}$ given by Eq. (4) and by the community detection step. As concerns the update step, we first observe that the relocation of nodes can be executed in $O(|V_t| + \omega|E_t|) = O(|\mathcal{V}| + \omega|E_t|)$, since for each node we look at its neighbor communities, which are bounded by the degree of the node. Evaluating the modularity improvement (Eq. (3)) is $O(\omega)$, provided that ω indexes are maintained to store the degree of communities for each of the last ω time steps, and to store the number of links of v with nodes in community c at time t, for each node v, time t and community c. Therefore, since we constrain the number of relocation trials to be of the order of the number of nodes, the overall time cost of relocation of nodes is $O(|\mathcal{V}| + \omega|E_t|)$. However, the update of $\mathbf{M}$ involves a number of entries that is at least equal to $\sum_{c_i \in \mathcal{C}^{(t)}} |c_i|^2$. This could lead to a cost that becomes quadratic in the number of entities as soon as some of the communities have size of the order of $\mathcal{V}$. Moreover, the spatial complexity of CreDENCE is determined by the number of non-zero entries of $\mathbf{M}$, which again could be quadratic in the number of entities.

As discussed above, maintaining and updating the DCM matrix represents a computational bottleneck of CreDENCE. By definition, $\mathbf{M}$ can easily become dense, yet noisy, since many co-associations may be weak (e.g., outdated co-associations), thus corresponding to poorly significant consensus memberships. One way to alleviate this issue is to prune the matrix by zeroing those entries that are below a predefined threshold; in practice, this will unlikely be enough to solve the issue. Rather, we notice that it is more appropriate to introduce a *constraint of linkage between nodes* when evaluating Eq. (4): this is not only consistent with the requirement of having as high density as possible within a (consensus) community (as studied in [22]), but it will also impact on making $\mathbf{M}$ sparser. However, one drawback would be the loss of symmetry in $\mathbf{M}$.

We hence propose a modification to the update equation that both integrates the linkage constraint and preserves the stochasticity property of the matrix:

$$m_{ij}^{(t+1)} = \frac{\alpha}{|c_i^{(t)} \cap N_i^{(t)}|}[v_j \in c_i^{(t)} \cap N_i^{(t)}] + (1-\alpha) m_{ij}^{(t)}, \qquad (5)$$

where $N_i^{(t)}$ denotes the set of neighbors of v_i in G_t. The entry m_{ij} now is meant to store the strength of co-association of v_i conditionally to the topological link with v_j. Moreover, the graph representation of $\mathbf{M}$ becomes directed: to keep the scheme presented in Algorithm 1, we simply modify the definition of the consensus graph $G_{\mathbf{M}}$ so that the weight of an edge (v_i, v_j) is set to $\max\{m_{ij}, m_{ji}\}$. This allows us to preserve the importance of a co-association between any two entities when finding a community structure in $G_{\mathbf{M}}$.

We incorporate the above modifications into Algorithm 1 to obtain an enhanced, efficient version of CreDENCE. It can be noticed that the time complexity of CreDENCE now becomes $O(T \times (|\mathcal{V}| + |E_{\leq T}|))$, while the spatial cost is determined by the size of $\mathbf{M}$, i.e., $O(|E_{\leq T}|)$, with $E_{\leq T} = \bigcup_{t=1}^{T} E_t$.

4 Evaluation methodology

Data. We used 5 real-world, publicly available temporal networks: *Epinions* [18], *Facebook* [23], *Wiki-Conflict* [2], *Wiki-Elections* [15], *YouTube* [19]. Table 1 reports statistics for each evaluation network. Note that with terms 'static', 'hapax', and 'dynamic' we mean nodes/edges that are present in all snapshots, present in only one snapshot, and present in multiple, not necessarily contiguous snapshots, respectively. Also, symbols e_t^+ and e_t^- refer to the fraction of new edges and disappeared edges, respectively, when transitioning from the t-1-th to the t-th snapshot; analogously for nodes corresponding to symbols v_t^+, v_t^-. Note also that, while the friendship-based networks (i.e., Facebook and YouTube) evolve very smoothly, the other selected networks undergo to drastic changes in terms of disappearing/appearing edges and nodes. Preprocessing of

Table 1: Main characteristics of our evaluation data. Mean ± standard deviation values refer to all snapshots in a network.

	#entities ($\lvert\mathcal{V}\rvert$)	#edges	#time steps	node set coverage	edge semantics	% static (nodes, edges)	% hapax (nodes, edges)	% dynamic (nodes, edges)
Epinions	131 828	727 344	32	0.05	trust/distrust	(0.1, 0)	(80.8, 95.6)	(19, 2.2)
Facebook	63 731	17 676 817	30	0.87	friendship birth	(82.9, 2.7)	(0.2, 0)	(16.9, 1.9)
Wiki-Conflict	118 100	2 272 276	82	0.05	wikipage editing	(0, 0)	(60.1, 83.4)	(38.9, 5.8)
Wiki-Election	7 118	102 906	44	0.08	vote assignment	(0, 0)	(49.7, 95.7)	(50.3, 2.2)
YouTube	3 223 589	41 955 741	8	0.62	friendship birth	(33.4, 6.7)	(12.4, 4)	(54.2, 11.6)

	network evolution rate			
	$e_t^+ = \frac{\lvert E_t \setminus E_{t-1}\rvert}{\lvert E_t\rvert}$	$e_t^- = \frac{\lvert E_{t-1} \setminus E_t\rvert}{\lvert E_{t-1}\rvert}$	$v_t^+ = \frac{\lvert V_t \setminus V_{t-1}\rvert}{\lvert V_t\rvert}$	$v_t^- = \frac{\lvert V_{t-1} \setminus V_t\rvert}{\lvert V_{t-1}\rvert}$
Epinions	0.97 ± 0.007	0.98 ± 0.008	0.65 ± 0.08	0.69 ± 0.06
Facebook	0.02 ± 0.01	0	0.006 ± 0.006	0
Wiki-Conflict	0.95 ± 0.02	0.95 ± 0.02	0.52 ± 0.1	0.51 ± 0.12
Wiki-Election	0.99 ± 0.004	0.99 ± 0.005	0.5 ± 0.07	0.49 ± 0.08
YouTube	0.16 ± 0.06	0	0.14 ± 0.06	0

the networks and statistics about the temporal width resolution are available at `http://people.dimes.unical.it/andreatagarelli/cmab-dccd`.

We also used synthetic networks generated through *RDyn* [20], which is designed to handle community dynamics and change events (merge/split). RDyn adopts the notion of stable iteration to mimic ***ground-truth*** communities; in particular, when a community structure reaches a minimum quality (i.e., conductance), then it is recognized as ground-truth. We believe that the latter property of RDyn is important since it fills a lack in the literature about the unavailablilty of ground-truth data for (large) time-evolving multilayer networks.

Competing methods. We conducted a comparative evaluation of CreDENCE with the following three methods, which are also based on modularity optimization and do not require an input number of communities:

– DynLouvain [10]: it applies Louvain method [1] to a condensed network based on the topology of the snapshot at current time t and community structure at time t-1.
– EvoAutoLeaders [8]: this is an evolutionary method based on a notion of community as a set of follower nodes congregating close to a potential leader (i.e., the most central node in the community).
– M-EMCD* [16]: this is a parameter-free enhanced version of the consensus-based method in [22], which filters noisy co-associations via marginal likelihood filter and optimize the multilayer modularity of the consensus w.r.t. a static ensemble of community structures.

Evaluation settings. We varied the learning rate α in $\{0.15, 0.5, 0.85\} \cup \{\alpha^*\}$, where α^* is an *adaptive* learning rate set to the fraction of times a base arms is used, and the temporal-window width ω from 2 to 10; however, unless otherwise specified, we used the setting $\omega = 2$, $\beta = 1 - \alpha$ to emphasize the importance of few, more recent snapshots. We set the relocation bias λ to 0, i.e., a relocation is accepted if it leads to an improvement in modularity (Eq. (3)). To reduce sensitivity issues due to the randomness in the exploration-exploitation interleaving, we averaged the CreDENCE perfomance scores over 100 runs.

To detect communities from each snapshot (i.e., $\mathcal{A}$ in Algorithm 1), we used the classic Louvain method [1]. This choice is not only consistent with our modularity-optimization-based relocation phase, but also with the choice of static algorithm in most approaches for dynamic community detection [5].

As for the bandit strategy $\mathcal{B}$, we resorted to ϵ-*greedy*, i.e., with a small probability ϵ we take an exploration step, otherwise (i.e., with probability $1 - \epsilon$) an exploitation step. We set $\epsilon = 0.1$, which revealed to lead to a performance stability trade-off for networks having different evolution rates.

5 Results

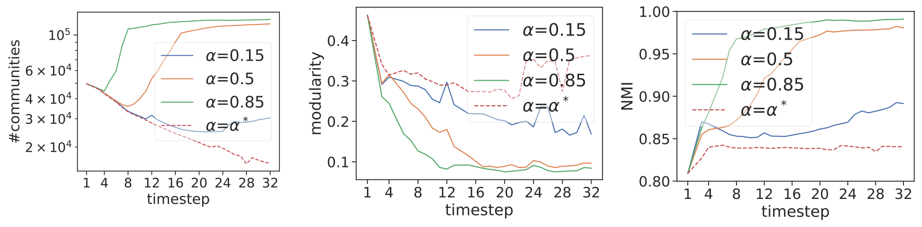
Impact of learning rate. As shown in Fig. 1, the number of detected consensus communities generally increases for higher values of α, because this more quickly leads to lose memory of past co-associations, thus causing proliferation of communities in the consensus solution. Moreover, on the networks having high rate of structural change, the trends for the various settings of α tend to deviate in correspondence of the time steps associated with most change events; by contrast, in the networks characterized by a smooth evolution (i.e., Facebook and YouTube), the consensus sizes are very similar while varying α.

Figure 1 also shows *multilayer modularity* [22] results by varying α. Lower values of α generally lead to higher modularity except for Facebook and YouTube networks. This is explained since, in networks having high rate of structural change, a lower learning rate helps remember past co-associations, thus information about older snapshots. Moreover, in such networks we observe a decreasing trend in modularity since the consensus must embed an increasing number of snapshots, each very different from the others (cf. Table 1). By contrast, for Facebook and YouTube, a high learning rate reveals to be beneficial to discovering consensus communities with higher modularity.

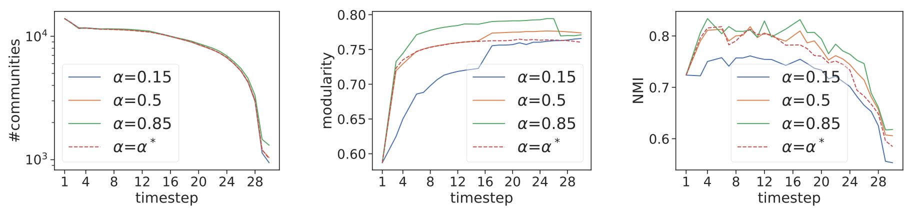
We also measured the Strehl and Ghosh's *NMI* between the dynamic consensus and the community structure of snapshot, for each time step (Fig. 1). As expected, the two structures are more similar (i.e., higher NMI values) as α increases, which implies weighting more the current snapshot in the consensus generation. Analogous remarks were drawn for the *average cumulative NMI*, which is computed at each t by averaging the NMI between the dynamic consensus at t and the community structures over all snapshots at any time $t' \leq t$.

Impact of temporal-window width. Higher values of ω will lead to better modularity performance, which is explained since the criterion function optimized in the relocation phase becomes closer to the measured modularity as ω increases. We indeed observed modularity improvements up to 0.04 (at any time step) already for $\omega = 4$, while negligible increments occurred as $\omega > 4$. Generally, no evident differences were observed in terms of NMI and consensus size, which indicates relative robustness of CreDENCE with variation in ω. Results can be found at `http://people.dimes.unical.it/andreatagarelli/cmab-dccd`.

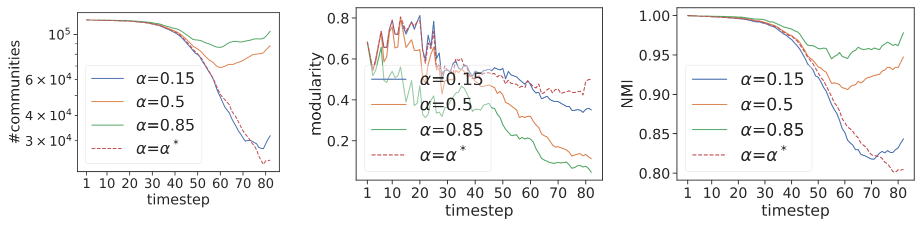
Impact of the exploration step probability. The default setting $\epsilon = 0.1$ revealed to lead to a suitable trade-off for our networks, which have different evolution rates. In fact, as shown in Fig. 2, with $\alpha = 0.5$ and default setting for



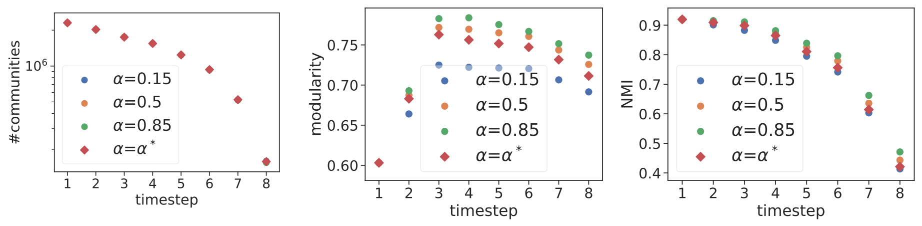
(a) Epinions



(b) Facebook



(c) Wiki-Conflict



(d) YouTube

Fig. 1: Size of the dynamic consensus by CreDENCE (left), multilayer modularity of the CreDENCE solutions (mid), and NMI between the CreDENCE consensus community structure and the snapshot's community structure, at each t (right).

the other parameters, higher values for the exploration probability lead to more "unstable" results since more exploration steps are performed, thus information derived from a newly observed snaphot impact more on the consensus update. This is particularly evident in networks with high rate of structural changes, such as Epinions and Wiki-Conflict. On the contrary, for networks with a smooth evolution (e.g., Facebook), a higher number of exploration steps is beneficial in terms of modularity and NMI.

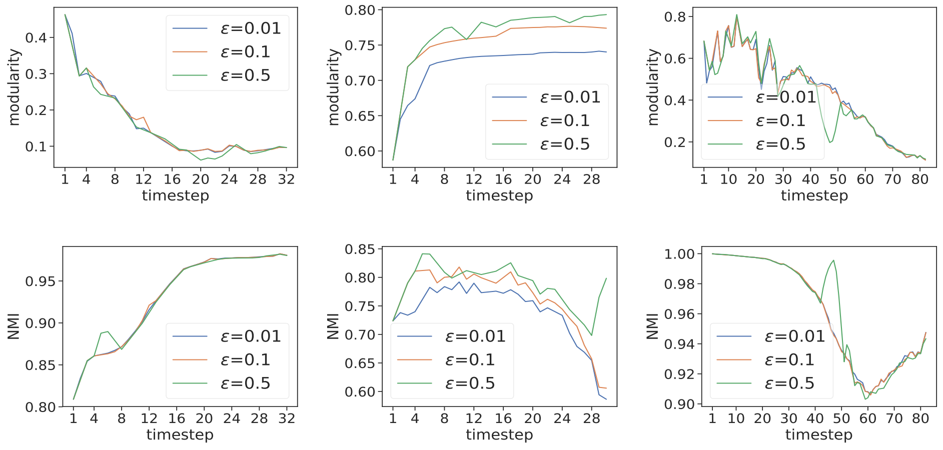
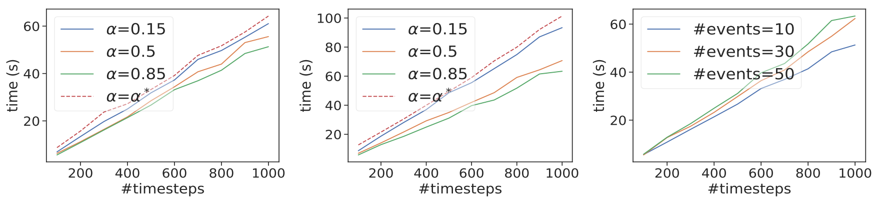


Fig. 2: Multilayer modularity and NMI by varying exploration-step probability ϵ, on Epinions (left), Facebook (mid), and Wiki-Conflict (right).



(a) 10 events (b) 50 events (c) $\alpha = 0.85$

Fig. 3: Time performance on RDyn synthetic networks.

Efficiency evaluation. To assess the scalability of CreDENCE, we used *RDyn* [20] to generate different synthetic networks by varying the number of snapshots and community events.[1] Figure 3 reports the execution times for different settings of α, over a temporal network with 1K entities and 1K time steps.

We observe that, for different change rate of community events (Fig. 3(a)–(b)), CreDENCE always scales linearly with the number of considered timesteps, which is consistent with our complexity analysis (cf. Sect. 3.4). Also, the execution time is generally higher for the adaptive learning rate α^* as well as for lower values of α (i.e., as the past co-associations are preserved longer), thus making the DCM matrix denser and more costly to process. Figure 3(c) shows the execution times of our method with $\alpha = 0.85$, on three synthetic networks with 10, 30, 50 community events, respectively. As expected, the higher the evolution rate, the higher the execution time; nonetheless, CreDENCE again shows to scale linearly with the size of the network.

[1] Experiments were carried out on a Linux (Mint 18) machine with 2.6 GHz Intel Core i7-4720HQ processor and 16GB ram

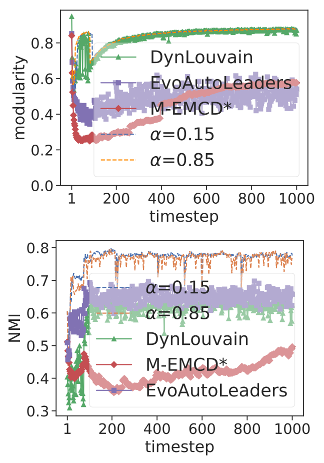


Fig. 4: Competitors vs. CreDENCE on RDyn: modularity (top), NMI (bottom).

Table 2: Increment percentages of CreDENCE w.r.t. DynLouvain and M-EMCD*. Values correspond to the increment percentages averaged over all snapshots in a network, using the average best-performing α.

	DynLouvain		M-EMCD*	
	Modularity	NMI	Modularity	NMI
Epinions	1789.0 %	-2.2 %	13.9 %	37.6%
Facebook	3.5 %	9.4 %	60.0 %	37.5 %
Wiki-Conflict	> 1.0 E+05 %	-1.8 %	-6.8 %	37.6 %
Wiki-Election	660.5 %	-2.1 %	32.0 %	58.5 %
YouTube	-0.1 %	8.4 %	21.1 %	11.6 %
RDyn	2.0 %	24.97 %	103.22 %	81.1 %

Comparison with competing methods. Table 2 and Fig. 3 compare CreDENCE with the other methods. Concerning modularity results, our method outperforms both DynLouvain and M-EMCD*, where performance gains vs. the former (resp. latter) are outstanding for networks with high (resp. low) rate of structural change. NMI by CreDENCE is always significantly higher than the competitors' ones, especially against M-EMCD*; one exception is represented by a gap of just 2% w.r.t. DynLouvain for three networks with high evolution rate. Moreover, we emphasize that CreDENCE also outperforms the evolutionary EvoAutoLeaders, as long as the competitor results were available—indeed, it incurred in processing-time issues (tens hours) in all networks but the smallest ones, i.e., Wiki-Election and RDyn.

6 Conclusion

In this paper, we proposed CreDENCE, a CMAB-based method for the problem of dynamic consensus community detection in temporal networks. Experimental evidence on real and synthetic networks has shown the meaningfulness of the consensus solutions produced by CreDENCE, also revealing its unique ability of dealing with temporal networks that can have different evolution rate.

We plan to further investigate on the impact of different bandit strategies (e.g., UCB, Thompson sampling), and on learning our model parameters to best fit the community structure and evolution in a given temporal network.

References

1. V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *J. Stat. Mech.*, 10, 2008.
2. U. Brandes, P. Kenis, J. Lerner, and D. van Raaij. Network analysis of collaboration structure in Wikipedia. In *Proc. ACM WWW*, pages 731–740, 2009.
3. P. Brodka, S. Saganowski, and P. Kazienko. GED: the method for group evolution discovery in social networks. *Social Netw. Analys. Mining*, 3(1):1–14, 2013.

4. W. Chen, Y. Wang, and Y. Yuan. Combinatorial Multi-Armed Bandit: General Framework and Applications. In *Proc. ICML*, pages 151–159, 2013.
5. N. Dakiche, F. B.-S. Tayeb, Y. Slimani, and K. Benatchba. Tracking community evolution in social networks: A survey. *Inf. Process. Manag.*, 56(3):1084–1102, 2019.
6. T. La Fond, G. Sanders, C. Klymko, and H. Van Emden. An ensemble framework for detecting community changes in dynamic networks. In *Proc. IEEE HPEC*, pages 1–6, 2017.
7. Y. Gai, B. Krishnamachari, and R. Jain. Combinatorial Network Optimization With Unknown Variables: Multi-Armed Bandits With Linear Rewards and Individual Observations. *IEEE/ACM Trans. Netw.*, 20(5):1466–1478, 2012.
8. W. Gao, W. Luo, and C. Bu. Adapting the TopLeaders algorithm for dynamic social networks. *The Journal of Supercomputing*, 2017.
9. Y. Gur, A. J. Zeevi, and O. Besbes. Stochastic Multi-Armed-Bandit Problem with Non-stationary Rewards. In *Proc. NIPS*, pages 199–207, 2014.
10. J. He and D. Chen. A fast algorithm for community detection in temporal network. *Physica A: Stat. Mech. Appl.*, 429:87–94, 2015.
11. P. Wagenseller, F. Wang, and W. Wu. Size matters: A comparative analysis of community detection algorithms. *IEEE Trans. Comput. Soc. Syst.*, 5:951–960, 2018.
12. P. Jiao, W. Wang, and D. Jin. Constrained common cluster based model for community detection in temporal and multiplex networks. *Neurocomputing*, 275:768–780, 2018.
13. M. N. Katehakis and A. F. Veinott Jr. Multi-armed bandit problem: Decomposition and computation. *Math. Oper. Res.*, 12:262–268, 1987.
14. A. Lancichinetti and S. Fortunato. Consensus clustering in complex networks. *Sci. Rep.*, 2:336, 2012.
15. J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg. Governance in Social Media: A Case Study of the Wikipedia Promotion Process. In *Proc. ICWSM*, 2010.
16. D. Mandaglio, A. Amelio, and A. Tagarelli. Consensus Community Detection in Multilayer Networks Using Parameter-Free Graph Pruning. In *Proc. PAKDD*, pages 193–205, 2018.
17. D. Mandaglio and A. Tagarelli. Dynamic Consensus Community Detection and Combinatorial Multi-Armed Bandit. In *Proc. IEEE/ACM ASONAM 2019*, `http://dx.doi.org/10.1145/3341161.3342910`.
18. P. Massa and P. Avesani. Controversial Users Demand Local Trust Metrics: An Experimental Study on Epinions.com Community. In *Proc. AAAI*, pages 121–126, 2005.
19. A. E. Mislove. *Online social networks: measurement, analysis, and applications to distributed information systems.* PhD thesis, Rice University, 2009.
20. G. Rossetti. RDyn: graph benchmark handling community dynamics. *Journal of Complex Networks*, 2017.
21. R. S. Sutton and A. G. Barto. *Introduction to reinforcement learning.* Vol. 135. Cambridge: MIT press, 1998.
22. A. Tagarelli, A. Amelio, and F. Gullo. Ensemble-based community detection in multilayer networks. *Data Min. Knowl. Discov.*, 31(5):1506–1543, Sep 2017.
23. B. Viswanath, A. Mislove, M. Cha, and P. Krishna Gummadi. On the evolution of user interaction in Facebook. In *Proc. ACM WOSN*, pages 37–42, 2009.
24. Z. Wang, Z. Li, G. Yuan, Y. Sun, X. Rui, and X. Xiang. Tracking the evolution of overlapping communities in dynamic social networks. *Knowl. Based Syst.*, 157:81–97, 2018.