Representation Learning in Continuous-Time Dynamic Signed Networks

Kartik Sharma*
Georgia Institute of Technology
ksartik@gatech.edu

Mohit Raghavendra†*
NIT Karnataka
mohithmitra@gmail.com

Yeon-Chang Lee
Georgia Institute of Technology
yeonchang@gatech.edu

Anand Kumar M
NIT Karnataka
m_anandkumar@nitk.edu.in

Srijan Kumar
Georgia Institute of Technology
srijan@gatech.edu

arXiv:2207.03408v3 [cs.SI] 5 Feb 2023

ABSTRACT

Signed networks allow us to model conflicting relationships and interactions, such as friend/enemy and support/oppose. These signed interactions happen in real time. Modeling such dynamics of signed networks is crucial to understanding the evolution of polarization in the network and enabling effective prediction of the signed structure (i.e., link signs and signed weights) in the future. However, existing works have modeled either (static) signed networks or dynamic (unsigned) networks but not dynamic signed networks. Since both sign and dynamics inform the graph structure in different ways, it is non-trivial to model how to combine the two features. In this work, we propose a new Graph Neural Network (GNN)-based approach to model dynamic signed networks, named **SEMBA**: **S**igned link's **E**volution using **M**emory modules and **B**alanced **A**ggregation. Here, the idea is to incorporate the signs of temporal interactions using separate modules guided by balance theory and to evolve the embeddings from a higher-order neighborhood. Experiments on 4 real-world datasets and 4 different tasks demonstrate that SEMBA consistently and significantly outperforms the baselines by up to 80% on the tasks of predicting signs of future links while matching the state-of-the-art performance on predicting existence of these links in the future. We find that this improvement is due specifically to superior performance of SEMBA on the minority negative class.

1 INTRODUCTION

Background. Social interactions are often signed and dynamic, representing the evolving nature of trust/distrust, support/oppose, agreement/disagreement, or positive/negative sentiments among people. Signed networks have been used to measure polarization in social networks [2, 5], performance in financial networks [3], and predicting international conflicts [10]. Dynamic signed networks model conflicting relationships that evolve over time, as shown in Figure 1. For instance, online social media discussions involve people supporting/opposing a claim, which continuously evolve as new users and new replies are made in real time. Dynamic signed networks have several societal applications such as predicting veracity of posts [36], toxicity in conversations [34], conflict across communities [21], and analyzing polarization in debates [2, 25, 26, 32]. Thus, it is important to create predictive models for dynamic signed networks so that one can devise intervention strategies against future unwanted behavior, maintaining a healthy online environment [4, 31, 39].

*MR and KS contributed equally to this work.
†This work was done during MR's internship at Georgia Tech

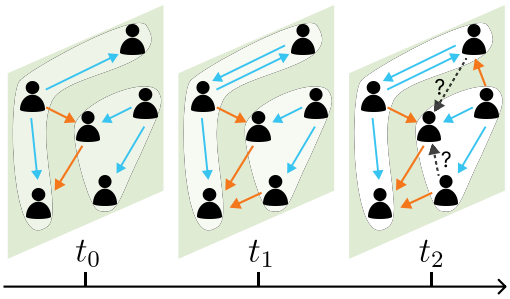

Figure 1: Visualizing Dynamic Signed Networks.

Structural balance theory models the organization of signed networks [6]. It states that signed relationships tend to form balanced triads such that "a friend of my friend is my friend," "a friend of my enemy is my enemy," "an enemy of my friend is my enemy," and "an enemy of my enemy is my friend." Several Signed Graph Neural Networks (GNNs) have been proposed to learn effective representations of static signed graphs, while incorporating balance theory [9, 13–15, 28, 29, 37].

Challenges. Modeling dynamic signed networks with existing (static) signed GNNs faces two challenges.

(C1) Temporal-Awareness, i.e., modeling the sequential order and evolution of signed interactions to learn node representations. Static signed GNNs only learn the network structure at a given timestamp [9, 13, 15], ignoring the temporal order of the interactions. Additionally, while signed GNNs can incorporate balance theory in static networks, they are unable to follow balance theory when the network is dynamic [1, 10, 16, 30, 41].

(C2) Staleness, i.e., in the absence of recent direct interactions of a node, its representation becomes stale. Staleness is a pervasive issue since real-world networks are long-tailed and sparse, i.e., most nodes only have a few direct interactions.

Meanwhile, there has been a surge of interest in learning temporal representations of nodes in dynamic networks by developing dynamic GNN models [24, 33, 38]. However, these model architectures are not **(C3) Sign-Aware**. In other words, since these models have been developed specifically for unsigned networks, they treat negative and positive edges the same and thus, fail to incorporate the higher-order organization induced by the link signs (i.e., structural balance theory). Due to their homophily assumptions in learning [9], these models fail to model heterophilic network organizations such as negative links between polarized communities.

Proposed Ideas. In this work, we aim to effectively learn representations of dynamic signed networks while addressing temporal-awareness, staleness, and sign-awareness problems. Towards this goal, we propose a new dynamic signed GNN model, named as **SEMBA**: **S**igned link **E**volution using **M**emory modules and **B**alanced **A**ggregation. SEMBA learns a short-term 'memory' encoding and a long-term 'embedding' per node. SEMBA consists of three key modules:

(1) *Signed Memories:* SEMBA learns two signed memories (positive and negative) per node to distinctly encode past interactions of a node with its friends and enemies, respectively.
(2) *Balanced Aggregation:* when the network evolves by adding a link (u, v), SEMBA updates signed memories of nodes u and v while explicitly incorporating balance theory.
(3) *Long-term Propagation:* SEMBA learns a node's embedding by attentively aggregating signed memories over all its neighbors until the current timestep. This overcomes staleness by learning not only from direct interactions at that time but any interactions in any of its past neighbors.

Representations thus formed can be used to solve multiple different tasks on dynamic signed networks: predicting the existence of a future link (i.e., **dynamic link existence prediction**), predicting its sign (i.e., **dynamic link sign prediction**), predicting both existence and sign in future (i.e., **dynamic link signed existence prediction**), and predicting its signed weight (i.e., **dynamic link signed weight prediction**). While dynamic link existence prediction has been studied extensively for unsigned dynamic networks in the literature [7, 33, 41], no work has studied it for signed networks. Similarly, while sign and signed weight prediction have been studied for static signed networks [9, 13, 15, 17, 23], no prior work exists that studies the dynamic setting. Due to the inductive nature of the dynamic case, SEMBA generalizes to new nodes and links, which has never been addressed in the signed network literature. We show experimentally that (1) modeling both balance and network evolution is essential to accurately predict signs and signed weights, and (2) SEMBA consistently and significantly outperforms state-of-the-art baselines for signed networks and for dynamic networks.

Contributions. Our contributions are summarized as follows:

- **Novel Problem:** We devise a novel problem of representing dynamic signed networks via 4 different prediction tasks.
- **Dynamic Signed GNN:** We propose SEMBA, a GNN-based model to represent dynamic signed networks. To the best of our knowledge, we are the first to devise a model to jointly address temporal-awareness, staleness, and sign-awareness problems.
- **Comprehensive Validation:** We validate the effectiveness of SEMBA by comparing it with 5 baselines on 4 real-world datasets under the problems of dynamic link existence, sign, and signed weight prediction. Specifically, SEMBA matches or outperforms the baselines in all tasks across datasets with an improvement of up to 80% in predicting signs.

2 RELATED WORK

In this section, we briefly review existing works for static signed GNNs and dynamic unsigned GNNs. Table 1 compares our model with the existing works. Unlike other models, SEMBA uses an end-to-end GNN-based model to generate node representations of dynamic signed networks.

Table 1: Comparison of existing models with the proposed approach SEMBA based on the challenges that they address.

Method	Temporal (C1)	Staleness (C2)	Signed (C3)
GCN [20]			
SGCN [9]			✓
SiGAT [13]			✓
TGN [33]	✓	✓	
RW-LSTM [8]	✓		✓
SEMBA (proposed)	✓	✓	✓

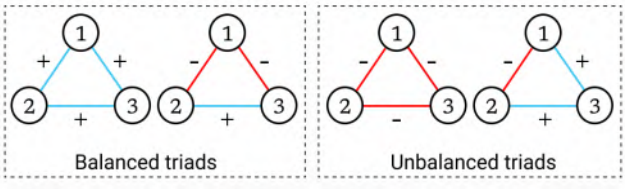

Figure 2: Balanced and unbalanced triads as dictated by balance theory [6].

2.1 Static Signed GNN

Machine learning on graphs has seen widespread interest, especially since the advent of message-passing GNNs. Plenty of GNNs have been adapted to perform a variety of graph-related tasks [43]. Signed networks allow us to model polarity of the relationship which becomes particularly important in a variety of applications [22, 23, 42]. Tasks of predicting the sign and signed weights of the links are often studied in the literature [9, 23]. With the advent of GNNs, several works have developed GNNs for signed networks [9, 13, 15, 17]. They make use of the inductive bias induced by the neighborhood message passing mechanism, but augments it with ideas from social science like balance (as shown in Figure 2) [6] and status [27]. However, most of these signed GNNs ignore the temporal aspect of the signed network and are limited to static networks. As mentioned in Section 1, this leads to temporal-awareness and staleness problems. Since majority of networks in the real world are dynamic and the temporal interactions are rich sources of data, we present a model that exploits network evolution along with the signed information to make prediction on future signed links.

2.2 Dynamic Unsigned GNN

Dynamic networks are popularly modeled as a stream of continuously time-stamped links, and allow us to capture the network evolution in real time. Methods like DyRep [38], JODIE [24] and TGN [33] have successfully modeled dynamic networks in continuous time, enabling them to solve multiple network-based problems. In particular, the problem of predicting whether a link would arrive in future has been studied extensively in the literature. GNNs are effective in this task as it is often the case that link dynamics is affected by the past interactions in the local neighborhood of the involved nodes.

Table 2: Notations used in this paper

Notation	Description
$\mathcal{G}(t)$	graph snapshot at t
$\mathcal{V}(t)$	vertex set at t
$\mathcal{E}^+(t), \mathcal{E}^-(t)$	positive and negative edge set at t
$\mathcal{E}(t)$	$\mathcal{E}^+(t) \cup \mathcal{E}^-(t)$
$\mathcal{G}_0$, $\mathcal{G}_n$	initial and final snapshot
ev	event like $\textsc{Add}_{edge}(u, v, \mathbf{e}_{uv})$
$\mathbb{E}[t_i, t_f]$	sequence of time-stamped events from t_i till t_f
$\mathbf{e}_{uv}(t)$	signed weight of (u, v) at t
$\mathbf{x}_u(t)$	node features of u at t
$\mathcal{N}_u^k([0, t])$	k-hop neighbor set of u from time 0 to t
$\mathbf{s}_u^+(t), \mathbf{s}_u^-(t)$	positive/negative memory of node u at t
$\mathbf{m}_u^+(t), \mathbf{m}_u^-(t)$	positive/negative message encoding of u at t
$\mathbf{M}_u^+(t), \mathbf{M}_u^-(t)$	aggregated positive/negative messages on u at t
B	Batch size
$T_B(t)$	First time after t with B interactions in between
$T_B^{-1}(t)$	Last time from t that was batched
$p(u, v, t; \Theta)$	probability of edge (u, v) for specified task at t
$\mathcal{E}_{train}, \mathcal{E}_{val}, \mathcal{E}_{test}$	training, validation, test set of edges

However, these models do not use the signed information, which we pointed out as the sign-awareness problem in Section 1. In particular, signed interactions are shown to reorganize in well-defined higher-order structures such as balanced triads. However, the existing works based on standard message-passing assume homophily and fail to capture the heterophilic nature of signed networks (e.g., polarized social networks). In this work, we incorporate balance theory to effectively learn representations of continuous-time dynamic networks. To the best of our knowledge, there has been only one prior work [8] in this direction, but not using GNNs. It uses random walks to model the network as an evolving time-series which is then fed into a recurrent neural network for sign prediction. However, it does not make use of balance theory, GNNs for processing the graph structure, or attention mechanism for processing temporal data. We shall build on these recent advancements to create a more accurate model.

3 PROBLEM DEFINITION

A directed continuous-time dynamic signed network at time t is denoted by $\mathcal{G}(t) = (\mathcal{V}(t), \mathcal{E}^+(t), \mathcal{E}^-(t))$, where $\mathcal{V}(t)$ denotes the set of nodes, and $\mathcal{E}^+(t)$ (resp. $\mathcal{E}^-(t)$) denotes the positive (resp. negative) links at time t. It can be modeled using an initial network $\mathcal{G}_0 = (\mathcal{V}_0, \mathcal{E}_0^+, \mathcal{E}_0^-)$ along with a sequence of time-stamped events (ordered by time), $\mathbb{E} := \mathbb{E}[t_0, t_n] = \{(\tau, \mathsf{ev}) : t_0 \le \tau \le t_n\}$, such that t_n is the final time step and ev represents a link addition at timestep τ.[1] More precisely, one can construct a network $\mathcal{G}(t)$ at snapshot t by augmenting links to $\mathcal{G}_0$ such that $\mathcal{G}(t_n) = \mathcal{G}_0 \sqcup \{\mathsf{ev} : (t, \mathsf{ev}) \in \mathbb{E}, t_0 \le t \le t_n\}$. The operator $\sqcup$ sequentially updates the network based on the events $\{\mathsf{ev}\}$. For example, if the event ev is addition of the link $(u, v, \mathbf{e}_{uv})$, i.e., $\textsc{Add}_{edge}(u, v, \mathbf{e}_{uv})$, and the link signed weight $\mathbf{e}_{uv}(t) > 0$, then network $\mathcal{G} = (\mathcal{V}, \mathcal{E}^+, \mathcal{E}^-) \to (\mathcal{V}, \mathcal{E}^+ \cup \{(u, v, \mathbf{e}_{uv}(t))\}, \mathcal{E}^-)$. For the sake of simplicity, we will use $\mathcal{E}(t) := \mathcal{E}^+(t) \cup \mathcal{E}^-(t)$ and $\mathcal{G}_n$ to denote $\mathcal{G}(t_n)$.

In this work, we study the effectiveness of signed links' representation in time through the following four problems:

Problem 1 (Dynamic Link Existence Prediction). *Given $\mathcal{G}_0, \mathbb{E}[t_1, t_j]$, the goal is to predict whether a link exists between u and v in future, i.e., $(u, v, \cdot) \in \mathcal{E}(t_k)$ for $t_k > t_j$.*

Problem 2 (Dynamic Link Sign Prediction). *Given $\mathcal{G}_0, \mathbb{E}[t_1, t_j]$, and a link $(u, v, \mathbf{e}_{uv}) \in \mathcal{E}(t_k)$ for $t_k > t_j$, the goal is to predict its sign, i.e., whether $\mathbf{e}_{uv}(t_k) > 0$ or $\mathbf{e}_{uv}(t_k) < 0$.*

Problem 3 (Dynamic Link Signed Existence Prediction). *Given $\mathcal{G}_0, \mathbb{E}[t_1, t_j]$, the goal is to predict whether a positive, negative or no link exists between u and v in future, i.e., whether $(u, v, \cdot) \in \mathcal{E}^+(t_k)$, $(u, v, \cdot) \in \mathcal{E}^-(t_k)$, or $(u, v, \cdot) \notin \mathcal{E}(t_k)$ for $t_k > t_j$.*

Problem 4 (Dynamic Link Signed Weight Prediction). *Given $\mathcal{G}_0, \mathbb{E}[t_1, t_j]$, and a link $(u, v, \mathbf{e}_{uv}) \in \mathcal{E}(t_k)$ for $t_k > t_j$, the goal is to predict the edge weight $\mathbf{e}_{uv}(t_k)$.*

4 SEMBA: OUR APPROACH

In this section, we propose SEMBA for representation learning in dynamic signed networks, which stands for **S**igned links **E**volution using **M**emory modules and **B**alanced **A**ggregation. We learn two representations per node: node memories to encode temporal signed interactions of the node, and node embeddings to incorporate interactions in its long-term neighbors. Finally, we use the learned embeddings to make the prediction.

In Section 4.1, we present an overview of SEMBA. In Section 4.2, we describe the four steps of SEMBA in detail. In Section 4.3, we discuss the process of learning and inference for SEMBA. Lastly, in Section 4.4, we clarify our contributions in comparison to existing static signed and dynamic unsigned GNNs.

4.1 Overview

SEMBA is motivated by the following three observations that help to understand signed links evolution in a dynamic signed network:

(1) **Links Evolution**: In a dynamic network, new links of a node u at time t tend to appear depending on the past interactions in the ego-network of u [33, 38], which consists of the focal node u and its neighborhood to whom u had a connection at some path length before t.
(2) **Structural Balance**: Balance theory [6] guides signed network evolution. If $(u, v) \notin \mathcal{E}$ but $(u, w), (w, v) \in \mathcal{E}$, then the sign of (u, w) and (w, v) would determine how information from v reaches u and consequently, the sign of future link (u, v). As a result, u should consider the information coming from a friend of a friend (resp. a friend of an enemy) differently than the information coming from an enemy of a friend (resp. an enemy of an enemy).
(3) **Long-tailed Distribution**: It is well known that real-world networks tend to have a power-law degree distribution, which indicates only a few nodes in the network have a high degree,

[1] We only consider link additions in this work. Link removals and node-level events will be considered in future work.

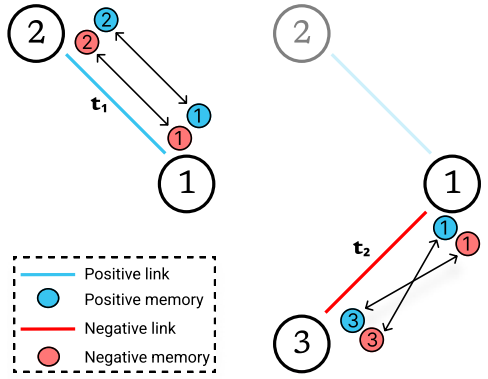

Figure 3: Balanced aggregation: Schema representing how link addition events (t_1, $(1, 2, +)$) and (t_2, $(1, 3, -)$) update the corresponding signed memories at the end-nodes.

while most of the other nodes have a very low degree. In a dynamic network, it makes the representation of long-tailed nodes that have no *recent* interactions more challenging. In other words, their representations can easily become stale.

Following these observations, our model has three key architectural components — signed memories, balanced aggregation, and long-term propagation.

Signed Memories. A node's memory serves the purpose of representing the node's recent interactions with its immediate positive and negative neighbors, which is useful for predicting new links according to Observation 1. To be able to distinguish between positive versus negative neighbors as per Observation 2, our architecture uses two different memory modules, one positive and one negative, to accumulate the past interactions according to their sign.

Balanced Aggregation. Upon addition of a signed link (u, v), both nodes' positive and negative memories must be updated based on Observation 2:

(1) If link (u, v) is positive, memories of the *same polarity* influence each other. Thus, the positive memory of u is updated using the positive memory of v, while u's negative memory is updated using the negative memory of v.
(2) If link (u, v) is negative, memories of *opposite polarity* provide influence. Thus, the positive memory of u is updated using the negative memory of v, while u's negative memory is updated using the positive memory of v.

Figure 3 visualizes the balanced aggregation mechanism to update a node's memory due to incident positive and negative links.

Furthermore, it should be noted that *balanced aggregation naturally follows higher-order structural rules of balance theory*. Figure 4 shows how updating node memories using neighbor-to-neighbor balanced aggregation results in complying with higher-order balance theory relationships. At the first time step t_1, node 2's positive memory updates node 1's positive memory, which then updates node 3's negative memory at the next time step t_2. Hence, one can note that positive memory from a friend of an enemy (i.e., node 2 is a friend of an enemy of node 3) updates the negative memory, which stores the information from enemies, thus, in line with balance theory. Similarly, other rules of balance theory can also be realized by local updates in future time steps. For example, node

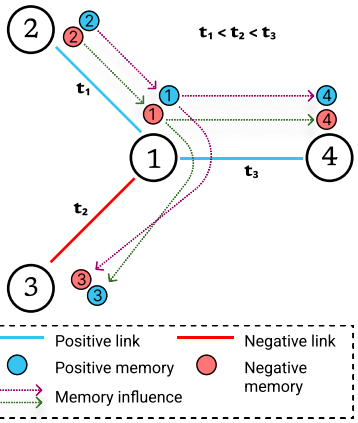

Figure 4: Schema showing how updating memories based on balanced aggregation enforces balance theory in triadic relations $2-1-3$ and $2-1-4$ when $t_2 < t_1, t_2 < t_3$.

2's positive memory updates node 4's positive memory via node 1's positive memory by the 'friend of a friend is a friend' rule.

Long-Term Propagation. A node's embedding serves the purpose of propagating signed memories of a node to the neighbors with no active interaction, which is useful for addressing staleness for long-tailed nodes according to Observation 3. To generate a node's embedding, we attentively aggregate signed memories of a node over all of its neighbors in the past and not just the neighbors in the current batch.

4.2 SEMBA Model Pipeline

Figure 5 demonstrates the pipeline of our model. It consists of the following four steps:

(1) Encode the events as signed messages following balance theory
(2) Aggregate signed messages in a batch (for efficient processing)
(3) Update node memories with the aggregated messages
(4) Generate node embeddings using memories and other features

Temporal Batching. Since networks evolve rapidly, it is computationally inefficient to update node embeddings after each change in the network. Thus, we combine a fixed number B of sequential events happening after a time t into a temporal batch. Node embeddings are then updated based on all the events included in such a batch. This also allows us to parallelize the computation of events within the batches and make the training process efficient. We create uniform batches of size B starting from t_0. The first batch stores these B interactions $\mathbb{E}[t_0, T_B(t_0)]$ such that $T_B(x)$ is the smallest timestep after x such that there are exactly B events in $\mathbb{E}[x, T_B(x)]$. Consequently, the kth batch consists of the events $\mathbb{E}\left[T_B^{(k-1)}(t_0), T_B^{(k)}(t_0)\right]$ with $T_B^{(j)}(x) = T_B \circ T_B^{(j-1)}(x)$.

Now, we describe the details of each step. To this end, we consider the updates due to an event of the form $\text{ev} = (t, \textsc{Add}(u, v, \mathbf{e}_{uv}))$. Let us also consider $T_B^{-1}(t)$ to be the latest batch time until t, i.e., $T_B^{-1}(t) = \max_k T_B^{(k)}(t_0)$ such that $T_B^{(k)} \leq t$.

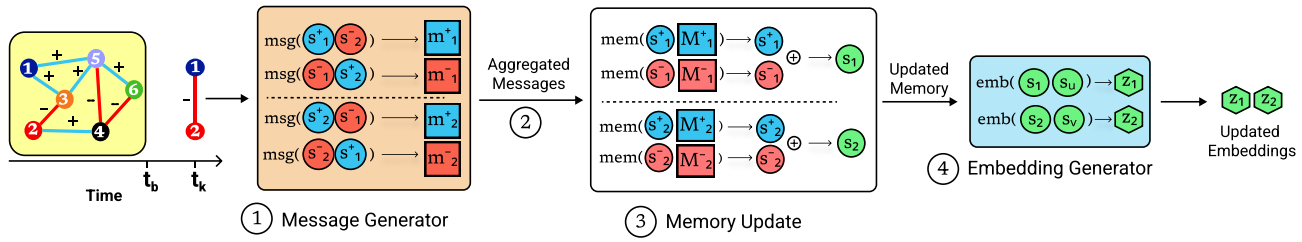

Figure 5: Pipeline of our method, SEMBA. The encircled numbers dictate the sequence in which each step happens. The model first adds the new link to the network, followed by generating messages as dictated by balance theory, in step 1. It then aggregates the messages and updates its memory using these new messages in steps 2 and 3. A graph attention based neural network is then used to update the embeddings of the nodes involved in step 4, which are used for downstream prediction tasks.

Step 1: Message Generation

Given a new interaction between u and v by $\text{ev} = (t, \textsc{Add}(u, v, \mathbf{e}_{uv}))$, we generate signed messages from v to u, which contain v's information passed to u, according to balance theory. The signed messages from u to v are also generated similarly. Thus, we here explain how to generate the signed messages from v to u.

Specifically, if e_{uv} is positive (resp. negative), then the message from positive (resp. negative) memories of node v will be used by node u to update u's positive (resp. negative) memory. More formally, positive and negative messages from v to u at time t, $\mathbf{m}^+_{u\leftarrow v}(t)$ and $\mathbf{m}^-_{u\leftarrow v}(t)$, are generated as follows[2]:

$$\mathbf{m}^+_{u\leftarrow v}(t) = \begin{cases} \texttt{msg}(\mathbf{s}^+_u(t-\Delta t_u) \oplus \mathbf{s}^+_v(t-\Delta t_v) \oplus \Delta t_u \oplus |\mathbf{e}_{uv}(t)|), \mathbf{e}_{uv}(t) > 0, \\ \texttt{msg}(\mathbf{s}^+_u(t-\Delta t_u) \oplus \mathbf{s}^-_v(t-\Delta t_v) \oplus \Delta t_u \oplus |\mathbf{e}_{uv}(t)|), \mathbf{e}_{uv}(t) < 0, \end{cases} \tag{1}$$

$$\mathbf{m}^-_{u\leftarrow v}(t) = \begin{cases} \texttt{msg}(\mathbf{s}^-_u(t-\Delta t_u) \oplus \mathbf{s}^-_v(t-\Delta t_v) \oplus \Delta t_u \oplus |\mathbf{e}_{uv}(t)|), \mathbf{e}_{uv}(t) > 0, \\ \texttt{msg}(\mathbf{s}^-_u(t-\Delta t_u) \oplus \mathbf{s}^+_v(t-\Delta t_v) \oplus \Delta t_u \oplus |\mathbf{e}_{uv}(t)|), \mathbf{e}_{uv}(t) < 0, \end{cases} \tag{2}$$

where $\oplus$ indicates the concatenation operator and $\mathbf{s}^+_u$ (resp. $\mathbf{s}^-_u$) denotes positive (resp. negative) memory of u. Also, $t - \Delta t_u$ denotes the time when the signed memories of node u were last updated (i.e., the last time when there was a direct interaction involving the node u). In addition, $|\mathbf{e}_{uv}(t)|$ indicates the link signed weight between u and v. Lastly, we employ a standard multi-layer perception (MLP) to implement the message function $\texttt{msg}$.

Intuitively, $\mathbf{m}^+_{u\leftarrow v}(t)$ and $\mathbf{m}^-_{u\leftarrow v}(t)$ preserve the information received through recent positive and negative memories of both u and v while taking into consideration of the balance theory. Furthermore, it considers the strength of the interaction between u and v. By doing so, SEMBA can (1) differentiate the impact of recent versus older interactions, (2) treat signed memories differently based on the link sign, and (3) differentiate the influence per link. That is, signed messages are designed so that SEMBA can jointly address the temporal-awareness (C1) and sign-awareness (C3) problems.

[2] $\mathbf{m}^+_{v\leftarrow u}(t)$ and $\mathbf{m}^-_{v\leftarrow u}(t)$ are also formed similarly.

Step 2: Message Aggregation

Given a set of positive and negative messages, i.e., $\mathbf{m}^+_{u\leftarrow(\cdot)}(t)$ and $\mathbf{m}^-_{u\leftarrow(\cdot)}(t)$, received by each node u in the batch, we aggregate them separately as follows:

$$\begin{aligned} \mathbf{M}^+_u(t) &= \texttt{agg}(\{\mathbf{m}^+_{u\leftarrow i} : \ (\tau, \textsc{Add}(i, u, \mathbf{e}_{iu})) \in \mathbb{E}[T_B^{-1}(t), t]\}), \\ \mathbf{M}^-_u(t) &= \texttt{agg}(\{\mathbf{m}^-_{u\leftarrow i} : \ (\tau, \textsc{Add}(i, u, \mathbf{e}_{iu})) \in \mathbb{E}[T_B^{-1}(t), t]\}), \end{aligned} \tag{3}$$

To implement the aggregation function $\texttt{agg}$, we can employ any methods used in [33], such as mean aggregation, temporal aggregation using long short-term memories (LSTMs), and a function that returns the most recent message from its input set of messages. We found no significant difference among these and used the function returning the most recent message.

Step 3: Memory Update

Now, we have signed messages, i.e., $\mathbf{M}^+_u(t)$ and $\mathbf{M}^-_u(t)$, which encode new knowledge from the events involving each node u in the batch from $\mathbb{E}[T_B^{-1}(t), t]$ Given $\mathbf{M}^+_u(t)$ and $\mathbf{M}^-_u(t)$, we update u's positive and negative memories, $\mathbf{s}^+_u(t)$ and $\mathbf{s}^-_u(t)$, at time t and then form a joint memory $\mathbf{s}_u(t)$ for the node u as follows:

$$\begin{aligned} \mathbf{s}^+_u(t) &= \texttt{mem}(\mathbf{M}^+_u(t), \mathbf{s}^+_u(t-\Delta t)), \\ \mathbf{s}^-_u(t) &= \texttt{mem}(\mathbf{M}^-_u(t), \mathbf{s}^-_u(t-\Delta t)), \\ \mathbf{s}_u(t) &= \mathbf{s}^+(t) \oplus \mathbf{s}^-(t), \end{aligned} \tag{4}$$

where $\texttt{mem}$ indicates a memory update function, which we employ as an LSTM in this paper.

Step 4: Embedding Generation

Given a joint memory $\mathbf{s}_u(t)$ of u and memories $\mathbf{s}_i(t)$ of all of its neighbors i until that time t, i.e., $i \in \mathcal{N}_u([0, t])$, we find each node u's embedding $\mathbf{z}_u(t)$ at time t using Graph Transformer [35] as:

$$\mathbf{z}_u(t) = \mathbf{W}_1\mathbf{s}_u(t) + \sum_{\substack{(\tau,\text{ev})\in\mathbb{E}[0,t] \\ \text{ev}=\textsc{Add}(i,u,\mathbf{e}_{iu})}} \alpha_{i,u}(\mathbf{W}_2\mathbf{s}_i(t) + \mathbf{W}_4\tau + \mathbf{W}_4|\mathbf{e}_{iu}|), \tag{5}$$

where the attention coefficients $\alpha_{i,u}$ are found using multi-head dot product attention, and $\mathbf{W}_j$ denote learnable weights. If there are node features $\mathbf{x}_u(t)$, we concatenate them with the node memories $\mathbf{s}_u(t)$ and pass the concatenated value in place of $\mathbf{s}_u(t)$ in Eq. (5) to obtain the embeddings $\mathbf{z}_u(t)$.

Therefore, one can note that u's embedding $\mathbf{z}_u$ will be updated at time t even if there are no direct interactions with u in the batch. This is due to the fact that we attentively aggregate the memories of all of its neighbors formed until now and not just in this batch. Thus, if any of these nodes have direct interactions in this batch, their memories would be updated, subsequently updating u's embedding. In this way, SEMBA can generate robust node embeddings even in the lack of direct interactions during a time period, thereby successfully mitigating the staleness problem (C2).

4.3 Learning and Inference

To make effective predictions about events, the node embeddings are first updated using events until the previous temporal batch. The predictions are made using these latest embeddings, which ensures that the embeddings account for the immediate history. The node embeddings are used to make inference on a node pair (u, v) at a time t by concatenating the corresponding embeddings $\mathbf{z}_u(t)$ and $\mathbf{z}_v(t)$ and then passing it through an MLP, i.e.,

$$\widehat{y}(u, v, t; \mathbf{\Theta}) = \text{MLP}(\mathbf{z}_u(t) \oplus \mathbf{z}_v(t)), \tag{6}$$

where $\mathbf{\Theta}$ denotes the set of learnable model parameters.

We treat Dynamic Link Existence and Dynamic Link Sign Prediction (Problems 1, 2) as binary classification tasks. For Problem 1, negative sampling was employed to find an equal number of non-existing edges. To train the models for these tasks, we minimize the Binary Cross Entropy (BCE) loss in sequential batches as:

$$\mathcal{L}_{BCE}(\mathcal{E}_{tr}; \mathbf{\Theta}) = -\sum_{(u,v,t) \in \mathcal{E}_{tr}} \log p_y(u, v, T_B^{-1}(t); \mathbf{\Theta})), \tag{7}$$

where $\mathcal{E}_{tr}$ denotes the training set and $p_y := \sigma(\widehat{y})$ if $y_{uv}(t) = 1$ and $1 - \sigma(\widehat{y})$ otherwise. Note that $T_B^{-1}(t)$ is the latest batch time until t and $y_{uv}(t)$ is the actual label of the edge, which is 1 for link existence in Problem 1 and 1 for positive link in Problem 2.

Dynamic Signed Link Prediction (Problem 3) is treated as a multi-class classification task. Here, we minimize the Cross-Entropy Loss similar to Equation 7 with the true labels are in $\{0, 1, 2\}$ denoting positive, negative, and non-existing edges respectively. Corresponding p_y is found after applying softmax of $\widehat{y}$.

Finally, we consider the Dynamic Link Signed Weight Prediction (Problem 4) as a regression task for which we minimize the Root Mean Squared Error (RMSE) in sequential batches as follows:

$$\mathcal{L}_{RMSE}(\mathcal{E}_{tr}; \mathbf{\Theta}) = \sqrt{\frac{1}{|\mathcal{E}_{tr}|} \sum_{(u,v,t) \in \mathcal{E}_{tr}} (\mathbf{e}_{uv}(t) - \widehat{y}(u, v, T_B^{-1}(t); \mathbf{\Theta}))^2}. \tag{8}$$

4.4 Discussion

In this subsection, we summarize the unique architectural contribution of SEMBA as compared to closely related TGN [33]. TGN is a temporal graph learning module specifically designed for homophilic unsigned networks. On the other hand, the use of two different memories and their update rules inspired by balance theory, SEMBA makes the learning appropriate to heterophilic signed networks. As we shall see in the experiments, such simple modification can be very effective in learning representations of links in a continuous-time dynamic signed network, by jointly addressing the temporal awareness, staleness, and sign-awareness problems.

Table 3: Summary of the datasets. $f_+ = |\mathcal{E}_f^+|/|\mathcal{E}_f|$ denotes the fraction of positive links and $f_{ub} = \triangle_{ub}(\mathcal{G}_f)/\triangle(\mathcal{G}_f)$ denotes the fraction of unbalanced triangles in the signed network. Weight denotes the presence of link weight in a dataset.

Dataset	#nodes	#links	f_+	f_{ub}	#days	Weight
BTC-Otc	5.9K	21.5K	0.89	0.13	1,905	✓
BTC-Alpha	3.8K	14.1K	0.85	0.14	1,902	✓
WikiRfA	12.8K	170.5K	0.77	0.24	3,758	
Epinions	131.9K	871.4K	0.85	0.09	945	

5 EXPERIMENTS

We designed our experiments, aiming at answering the following key evaluation questions (EQs):

- **EQ1**: How does SEMBA perform compared to its competitors on the sign and link prediction tasks (Problems 1, 2, 4 and 3)?
- **EQ2**: How does the performance vary in transductive (nodes seen during training) and inductive (unseen nodes) settings?
- **EQ3**: How important is each component of SEMBA?
- **EQ4**: Is SEMBA time and memory efficient?

5.1 Experimental Setup

5.1.1 Datasets. We used four publicly available[3] real-world signed networks with temporal links: BTC-Otc, BTC-Alpha, WikiRfA, and Epinions. Table 3 shows some statistics of these datasets.

- **BTC-Otc** and **BTC-Alpha** [22, 23] are two who-trusts-whom networks of people trading Bitcoin on OTC and Alpha platforms, respectively. In addition to the sign and timestamp, these links are also weighted on a scale from −10 to +10.
- **WikiRfA** [42] is a voting network by fellow members for a Wikipedia member's request of adminship.
- **Epinions** is a trust network between members of an online consumer review site.

From Table 3, we note that all datasets are skewed towards positive links (see f_+) and follow balance theory (see f_{ub}). Note that WikiRfA originally has links with null signs that denote abstention, which we filtered out to make it into a binary signed network. Furthermore, we also removed any links with no given timestamps or no given weights from all the datasets. Since no datasets contain node features, we use a zeroed vector of dimension 8 as the features.

5.1.2 Baselines. We study a novel problem on dynamic signed networks that have been studied in only one prior work, i.e., RW-LSTM [8]; we made its variant **SGC-LSTM**, which can be used in the inductive setting for a fair compariosn. Also, we employ four state-of-the-art GNNs as additional baselines: one static unsigned GNN, i.e., **GCN** [20]; two static signed GNN, i.e., **SGCN** [9] and **SiGAT** [13]; and one dynamic unsigned GNN, i.e., **TGN** [33].

- **GCN** [20] uses neighborhood convolution to generate node embeddings for static unsigned networks.
- **SGCN** [9] incorporates the balance theory in the neighborhood aggregation for static signed networks.
- **SiGAT** [13] generalizes GATs to incorporate motifs guided by balance and status theory.

[3] https://snap.stanford.edu/data/index.html#signnets

- **SGC-LSTM**. Since the existing work RW-LSTM [8] does not use a GNN, it cannot be used in our inductive setting where unseen nodes may appear during the test phase. For a fair comparison, we thus employ an inductive SGCN module in place of the random walks. Thus, SGC-LSTM combines SGCN and LSTM modules sequentially such that graphs in each batch are encoded using SGCN and these embeddings are passed as a sequence to an LSTM to obtain the representations. Thus, SGC-LSTM would act like a discrete-time *time-and-graph* model [12].
- **TGN** [33] is the state-of-the-art model to make predictions on dynamic unsigned networks. It encodes unsigned temporal interactions in a memory encoding and uses these to effectively learn temporal representations of nodes.

5.1.3 Evaluation Metrics. We use appropriate metrics to evaluate the performance on the 4 different tasks, specified in Section 3.

For the binary classification tasks (*Dynamic Link Sign Prediction* and *Dynamic Link Existence Prediction*), we use the F1 and AUROC scores between the true and predicted labels/probabilities. F1 measures the harmonic mean of the precision and recall of the positive class, while AUROC computes the Area Under the Receiver Operating Characteristic (Precision vs Recall) curve drawn at different thresholds. While F1 handles the imbalance of classes in the data, AUROC ignores the class imbalance. Higher F1 and AUROC values are desired from a good classifier.

For multiclass classification, i.e., the *dynamic link signed existence Prediction*, we aggregate the F1 metric over different classes as an unweighted mean or macro averaging (F1mac), an average weighted by the support (F1wt), and a global calculation of precision/recall (F1mic or equivalently, accuracy Acc). F1wt accounts for imbalance in classes by weighing based on the number of samples while F1mac gives equal weight to all the classes. On the other hand, Acc may perform well by just predicting the majority class.

For the regression task, i.e., the *Dynamic Link Signed Weight Prediction*, we use the RMSE, R^2, and KL-div between the predicted and actual signed weights. RMSE measures the root mean squared errors between the actual and predicted values while R^2 measures the proportion of variation in the outcome that is explained by the predicted value. Finally, KL-div measures KL-divergence between the actual and predicted distributions. We expect a high R^2 and a low RMSE, KL-div values from a good model.

5.1.4 Implementation Details. To evaluate prediction quality in the future, we order all the links according to their timestamps and then spilt them into training (70%), validation (15%), and test (15%) sets. All models are trained using a common framework described in Section 4.3 with the Adam optimizer [19]. The predictions are made in a temporal batch, i.e., $t_k = T_B(t_j)$, where the batch size B is fixed to be 1,000 interactions for Bitcoin-OTC, Bitcoin-Alpha, and WikiRfA, and 16,000 interactions for the larger Epinions dataset. Note that the evaluation is also done in sequential batches, i.e., to predict a link at time t, all events (including those in the test set) before $T_B^{-1}(t)$ are included in the model's context for prediction. In other words, while the parameters remain fixed during the test phase, embeddings are updated in an online manner to incorporate past events to predict the future. For static models, i.e., GCN, SGCN, and SiGAT, we thus pass the graph formed by all the interactions until the current batch, i.e., $\mathcal{G}(t)$ for prediction.

Table 4: Performance on dynamic link existence prediction (rounded to 2 decimal digits). Higher AUROC and F1 are desired. Bold values are the best performing and underlined values are the second best. '-' denotes out of GPU memory.

Method	BTC-Otc		BTC-Alpha		WikiRfA		Epinions	
	F1	AUROC	F1	AUROC	F1	AUROC	F1	AUROC
GCN	0.00	0.49	0.00	0.50	0.00	0.49	0.00	0.49
SGCN	0.75	0.82	0.79	0.83	0.64	0.75	0.81	0.88
SiGAT	0.77	0.87	0.85	0.93	0.86	0.92	-	-
SGC-LSTM	0.25	0.57	0.31	0.59	0.50	0.33	-	-
TGN	**0.97**	**0.99**	**0.98**	**0.99**	**0.99**	**0.99**	0.94	0.98
SEMBA	0.96	**0.99**	**0.98**	**0.99**	**0.99**	**0.99**	**0.95**	**0.99**

Table 5: Performance on dynamic link sign prediction (rounded to 2 decimal digits). Higher AUROC and F1 are desired. Bold values are the best performing and underlined values are the second best. '-' denotes out of GPU memory.

Method	BTC-Otc		BTC-Alpha		WikiRfA		Epinions	
	F1	AUROC	F1	AUROC	F1	AUROC	F1	AUROC
GCN	0.03	0.54	0.06	0.51	0.00	0.52	0.00	0.47
SGCN	0.67	0.65	0.72	0.61	0.42	0.64	0.22	0.64
SiGAT	0.39	0.70	0.49	0.71	0.55	0.68	-	-
SGC-LSTM	0.12	0.54	0.12	0.56	0.00	0.05	-	-
TGN	0.79	0.79	0.73	0.70	0.45	0.62	0.38	0.67
SEMBA	**0.80**	**0.83**	**0.79**	**0.76**	**0.81**	**0.85**	**0.57**	**0.80**

We carefully tune the hyperparameters such as the learning rate and the number of epochs of baselines for each method and dataset via grid search on the validation set. The embedding dimension is fixed to be 64 and the transformer has 8 attention heads. All the experiments were conducted on the NVIDIA Tesla K80 GPUs with 12GB memory, and the models were implemented using the PyTorch Geometric framework.[4]

5.2 Prediction Performance

To test the effectiveness of representations learned by SEMBA, we evaluate its performance against representative baselines on the four tasks outlined earlier in Section 3.

5.2.1 Dynamic Link Existence Prediction. Table 4 shows the performance of different methods on the task of predicting whether a link exists between two nodes at a time step t given the graph until the previous batch $T_B^{-1}(t)$. This is a standard task in dynamic graphs literature [33, 44]. We find that SEMBA closely matches the performance of state-of-the-art TGN in this task for signed networks with a maximum difference of 0.01 in the F1 and AUROC metrics. This shows that SEMBA can effectively handle the evolution of link format in dynamic signed networks. SiGAT is the second closest baseline with a difference of at least 0.13 F1 and at least 0.06 AUROC points. Thus, we conclude that dynamic GNNs (such as TGN and SEMBA) are essential to model link existence in dynamic signed networks as they show an average of 14.2% and a minimum of 6.4% improvement over their static counterparts. We also find that SGC-LSTM is highly ineffective in modeling link formation dynamics

[4] https://github.com/pyg-team/pytorch_geometric

Table 6: Performance on the 3-class dynamic link signed existence prediction (rounded to 2 digits). Higher F1wt, F1mac, and Acc are desired. Bold values are the best performing and underlined values are the second best. '-' denotes out of GPU memory.

Method	BTC-Otc			BTC-Alpha			WikiRfa			Epinions		
	F1wt	F1mac	Acc	F1wt	F1mac	Acc	F1wt	F1mac	Acc	F1wt	F1mac	Acc
GCN	0.37	0.30	0.34	0.37	0.29	0.33	0.39	0.31	0.37	0.35	0.30	0.33
SGCN	0.65	0.53	0.63	0.70	0.57	0.69	0.57	0.48	0.59	0.71	0.55	0.74
SiGAT	0.67	0.55	0.62	0.70	0.57	0.65	0.65	0.55	0.61	-	-	-
SGC-LSTM	0.40	0.32	0.51	0.41	0.35	0.52	0.33	0.22	0.49	-	-	-
TGN	**0.88**	**0.76**	**0.87**	**0.86**	0.72	0.84	0.85	0.71	0.83	0.79	0.68	0.76
SEMBA	**0.88**	**0.76**	**0.87**	**0.86**	**0.73**	**0.85**	**0.91**	**0.80**	**0.90**	**0.82**	**0.74**	**0.81**

Table 7: Performance (rounded to 2 decimal digits) on dynamic link signed weight prediction. Lower RMSE and KL-div and higher R^2 are desired. Bold values are the best performing and underlined values are the second best.

Method	BTC-Otc			BTC-Alpha		
	RMSE	R^2	KL-div	RMSE	R^2	KL-div
GCN	1.85	-0.01	21.44	1.94	-0.02	11.45
SGCN	1.84	-0.00	21.44	1.93	-0.00	20.82
SiGAT	1.84	-0.00	21.44	1.94	-0.01	20.82
SGC-LSTM	1.84	-0.00	21.44	1.94	-0.00	20.82
TGN	**1.83**	**0.01**	9.42	**1.91**	**0.02**	9.34
SEMBA	**1.83**	**0.01**	**8.20**	1.92	**0.02**	**8.93**

and only shows a slight improvement over static unsigned GCN model, which produces random results. We know from the literature that *time-and-graph* models are theoretically less expressive than *time-then-graph* models [12]. Our results complement this theory by showing that even a simple *time-then-graph* model that aggregates all the temporal interactions in a single graph and pass the aggregated graph to a static signed GNN (as is done for SGCN and SiGAT baselines) is more effective than a *time-and-graph* model such as SGC-LSTM. More sophisticated *time-then-graph* models like TGN and SEMBA are found to be even more effective.

5.2.2 Dynamic Link Sign Prediction. Table 5 shows the performance of different methods on the task of predicting whether the sign of a future link at t is positive or negative given the graph until $T_B^{-1}(t)$. We find that SEMBA outperforms the baselines across all the datasets with an improvement of up to 80% in the F1 measure and up to 33% in the AUROC measure. Note that due to the scarcity of negative signs in the datasets, there is a class imbalance (see Table 3). Thus, F1 is a better measure to assess the quality of the classifier. TGN is the second best performing method, closely followed by the static signed SiGAT and SGCN models. This shows that sign classification requires effective modeling both temporal (recall C1) and sign (C3) features, which is achieved jointly by SEMBA. As shown by the suboptimality of SiGAT and TGN models through our results, neither balance theory nor temporal modeling alone can predict the signs of links in the future. Here too, we see poor performance of SGC-LSTM which can be attributed, as earlier, to its *time-and-graph* modeling.

5.2.3 Dynamic Link Signed Existence Prediction. Here, we combine the previous two tasks to do a 3-class classification, i.e., to predict whether a link exists between two nodes in the future and if it exists, predict its sign. Table 6 compares the performance of different methods for this task. SEMBA outperforms or matches the baselines across all the datasets with an improvement of up to 7% in the weighted F1 (F1wt), 12.6% in the macro F1 (F1mac), and 8.4% in the micro F1 (or accuracy Acc) scores. TGN is the closest baseline and matches SEMBA's performance on BTC-Otc and BTC-Alpha datasets. However, SEMBA shows significant improvements over TGN for the larger WikiRfa and Epinions datasets. Note that F1mac takes a mean by giving equal weights to all three classes while F1wt weighs each class by the number of samples it has. More improvement in F1mac than F1wt would then mean more improvement in the minority (i.e., negative signed links) class. Thus, we can note that SEMBA specifically improves over TGN by improving on classifying node pairs with a negative link, owing to the implicit prior of balance theory in its architecture. Here too, we note the poor performance of SGC-LSTM for similar reasons as mentioned above. Furthermore, we find that static signed methods (SGCN and SiGAT) are highly suboptimal for this task as the temporal methods (TGN and SEMBA) outperform them by a margin of at least 15.5% in F1wt, at least 28.1% in F1mac, and 9.5% in Acc measures. This shows that modeling temporal evolution (recall C1) is essential for predicting signed structure of graphs in future.

5.2.4 Dynamic Link Signed Weight Prediction. In this section, we test the efficacy of the methods in predicting the signed weights of links at t given the graph until $T_B^{-1}(t)$. Table 7 shows the performance of different methods on BTC-Otc and BTC-Alpha datasets (as only these have weighted edges). SEMBA outperforms or matches the baselines in getting a high R^2 and low RMSE and KL-div scores. While SEMBA consistently achieves a low KL-div, there isn't a significant improvement in RMSE and R^2 measures. In particular, the close-to-zero R^2 values indicate that all the models fail to learn the variability of the weights around its mean. This could be attributed to the lack of node features in these datasets or/and implicit conditional independence assumption of the weights given neighborhood features in GNNs [18]. Our results leave room for future research to improve upon our model in predicting signed weights of future links in dynamic signed networks.

Table 8: Performance (rounded to 2 decimal digits) on the 3-class dynamic link signed existence prediction for transductive (no new nodes) and inductive (both new nodes) settings. Higher F1wt and F1mac is desired. Bold values are the best performing and underlined values are the second best. '-' denotes out of GPU memory.

Method	BTC-Otc				BTC-Alpha				WikiRfa				Epinions			
	Transductive		Inductive		Transductive		Inductive		Transductive		Inductive		Transductive		Inductive	
	F1wt	F1mac	F1wt	F1mac	F1wt	F1mac	F1wt	F1mac	F1wt	F1mac	F1wt	F1mac	F1wt	F1mac	F1wt	F1mac
GCN	0.38	0.31	0.36	0.28	0.34	0.28	0.37	0.29	0.33	0.29	0.42	0.34	0.35	0.30	0.36	0.29
SGCN	0.69	0.59	0.77	0.65	0.75	0.69	0.79	0.67	0.60	0.50	0.67	0.57	0.75	0.57	**0.84**	**0.76**
SiGAT	0.68	0.58	0.76	0.63	0.70	0.65	0.74	0.61	0.60	0.52	0.71	0.62	-	-	-	-
SGC-LSTM	0.46	0.39	0.32	0.21	0.50	0.44	0.33	0.26	0.33	0.22	0.33	0.22	-	-	-	-
TGN	0.82	0.68	0.84	0.68	0.75	0.64	0.83	0.67	0.63	0.53	0.56	0.46	0.82	0.66	0.59	0.50
SEMBA	**0.85**	**0.73**	**0.85**	**0.72**	**0.83**	**0.75**	**0.85**	**0.70**	**0.75**	**0.68**	**0.83**	**0.75**	**0.84**	**0.76**	0.73	0.65

5.3 Transductive and Inductive Settings

In this section, we compare the performance of the methods in transductive and inductive settings. A *transductive* setting is one where all the links in the test set have nodes that were seen earlier during the training phase. On the other hand, in an *inductive* setting, all the links in the test set have nodes that were not seen earlier during training. We sample the original test set to form a transductive and an inductive test set for each dataset and test our previously trained models on each of these sets. Table 8 shows the performance on the *dynamic link signed existence prediction* on these two settings. We find that SEMBA outperforms the baselines in all but two cases with an improvement of up to 16% in F1wt and 22% in F1mac for the transductive setting and 17% and 21%, respectively, for the inductive setting. TGN is the closest baseline on average but interestingly, we find that it is often outperformed by static signed methods (SGCN and SiGAT) specifically in the inductive setting. In particular, we note that SGCN beats SEMBA in the inductive setting of Epinions by a margin of 15% and 17% in F1wt and F1mac values respectively. One reason for this observation could be the sparsity of the inductive nodes in the Epinions dataset. Around 75% of the inductive nodes have a degree equal to 1, i.e., they interact only once. SEMBA is designed specifically to model nodes from their interactions. Node features are used only in the final propagation step and may not affect the node embeddings too much. Future research can study more powerful mechanisms for learning node embeddings from memories to handle highly inductive cases.

5.4 Ablation Study

In this section, we conduct an ablation study to examine SEMBA's sensitivity to each component. In particular, we consider three variants of SEMBA. **(1) - BA**, which removes the balanced aggregation (BA) of memories. This is equivalent to the original TGN model. **(2) - emb**, where we remove the long-term propagation using a Graph Transformer to form embeddings (i.e., Step 4). The concatenated memories are used as embeddings. **(3) - mem**, where we remove the memories and message generation steps. We directly propagate the node features through the aggregated graph until that time via a transformer to find embeddings.

Table 9 shows the ablation results for the 3-class dynamic link signed existence prediction. We find that each component is essential for the superior performance of SEMBA as it consistently outperforms its ablation counterparts across datasets. Our results further show that memories are the most important component of the architecture, followed by long-term embedding propagation, and finally balanced aggregation. In fact, removing balanced aggregation can result in a performance comparable to removing the long-term embedding propagation (see Epinions). Thus, balanced aggregation is an essential advance over the original TGN model for representation learning for dynamic signed networks.

Table 9: Ablation study of SEMBA on the multiclass dynamic link signed existence prediction. Higher F1wt and F1mac are desired. Bold values show the best performing and underlined values show the second best method.

Method	BTC-Otc		BTC-Alpha		WikiRfa		Epinions	
	F1wt	F1mac	F1wt	F1mac	F1wt	F1mac	F1wt	F1mac
SEMBA	**0.88**	**0.76**	**0.86**	**0.73**	**0.91**	**0.80**	**0.82**	**0.74**
- BA	**0.88**	**0.76**	**0.86**	0.72	0.85	0.71	0.79	0.68
- emb	0.71	0.57	0.72	0.59	0.79	0.67	0.78	0.71
- mem	0.67	0.54	0.71	0.57	0.63	0.46	0.70	0.54

5.5 Efficiency

We compare the efficiency of SEMBA against baselines in terms of training time and memory usage. This is quantified by the per epoch training time in seconds and the allocated GPU memory in bytes for the largest Epinions dataset. First, we note that both SiGAT and SGC-LSTM fail to run on this dataset as they ran out of GPU memory while allocating space during training. SGCN takes 11.9 s on average to train one epoch while allocating 862 MB. On the other hand, more sophisticated TGN and SEMBA take more time and space resources to train. In particular, TGN takes 151.4 s per epoch and requires 11.9 GB, while SEMBA takes 335.2 s per epoch with 12.9 GB of GPU memory. Thus, the superior performance of SEMBA comes at a cost of more training time and GPU resources.

6 CONCLUSION

Our work provides a novel framework to learn and test representations of dynamic signed networks. Our proposed model SEMBA is shown to outperform the baselines across different prediction tasks over dynamic links. We hope that our model opens new future avenues to study signed dynamic GNNs. Our method aggregates only first-order signed information from the neighborhood. Recently proposed propagation-based methods such as APAN [40] can be

extended to encode signed information from the k-hop neighborhood. Other extensions can focus on enforcing other theories of signed network configuration such as status theory [11, 27]. Another drawback of our model is its incapability in predicting the weight distributions. Regression tasks on graphs are known to be notorious and are largely understudied for dynamic graphs. Future works can develop effective models in this vein to learn the entire distribution of signed ratings.

REFERENCES

[1] Peter Abell and Mark Ludwig. 2009. Structural balance: a dynamic perspective. *Journal of Mathematical Sociology* 33, 2 (2009), 129–155.

[2] Teresa Alsinet, Josep Argelich, Ramón Béjar, and Santi Martínez. 2021. Measuring Polarization in Online Debates. *Applied Sciences* 11, 24 (2021), 11879.

[3] Omid Askarisichani, Jacqueline Ng Lane, Francesco Bullo, Noah E Friedkin, Ambuj K Singh, and Brian Uzzi. 2019. Structural balance emerges and explains performance in risky decision-making. *Nature communications* 10, 1 (2019), 1–10.

[4] Robert Axelrod, Joshua J Daymude, and Stephanie Forrest. 2021. Preventing extreme polarization of political attitudes. *Proceedings of the National Academy of Sciences* 118, 50 (2021), e2102139118.

[5] Francesco Bonchi, Edoardo Galimberti, Aristides Gionis, Bruno Ordozgoiti, and Giancarlo Ruffo. 2019. Discovering polarized communities in signed networks. In *Proceedings of the 28th acm international conference on information and knowledge management*. 961–970.

[6] Dorwin Cartwright and Frank Harary. 1956. Structural balance: a generalization of Heider's theory. *Psychological review* 63, 5 (1956), 277.

[7] Jinyin Chen, Xueke Wang, and Xuanheng Xu. 2022. GC-LSTM: Graph convolution embedded LSTM for dynamic network link prediction. *Applied Intelligence* (2022), 1–16.

[8] Quang-Vinh Dang and Claudia-Lavinia Ignat. 2018. Link-sign prediction in dynamic signed directed networks. In *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*. IEEE, 36–45.

[9] Tyler Derr, Yao Ma, and Jiliang Tang. 2018. Signed graph convolutional networks. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 929–934.

[10] Patrick Doreian and Andrej Mrvar. 2019. Structural balance and signed international relations. *Journal of Social Structure* 16, 1 (2019).

[11] M Hamit Fişek, Joseph Berger, and Robert Z Norman. 1991. Participation in heterogeneous and homogeneous groups: A theoretical integration. *Amer. J. Sociology* 97, 1 (1991), 114–142.

[12] Jianfei Gao and Bruno Ribeiro. 2021. On the equivalence between temporal and static graph representations for observational predictions. *arXiv preprint arXiv:2103.07016* (2021).

[13] Junjie Huang, Huawei Shen, Liang Hou, and Xueqi Cheng. 2019. Signed graph attention networks. In *International Conference on Artificial Neural Networks*. Springer, 566–577.

[14] Junjie Huang, Huawei Shen, Liang Hou, and Xueqi Cheng. 2021. SDGNN: Learning Node Representation for Signed Directed Networks. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*. 196–203.

[15] Zexi Huang, Arlei Silva, and Ambuj Singh. 2021. POLE: Polarized Embedding for Signed Networks. *arXiv preprint arXiv:2110.09899* (2021).

[16] Norman P Hummon and Patrick Doreian. 2003. Some dynamics of social balance processes: bringing Heider back into balance theory. *Social Networks* 25, 1 (2003), 17–49.

[17] Amin Javari, Tyler Derr, Pouya Esmailian, Jiliang Tang, and Kevin Chen-Chuan Chang. 2020. Rose: Role-based signed network embedding. In *Proceedings of The Web Conference 2020*. 2782–2788.

[18] Junteng Jia and Austion R Benson. 2020. Residual correlation in graph neural network regression. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 588–598.

[19] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[20] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[21] Srijan Kumar, William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2018. Community interaction and conflict on the web. In *Proceedings of the 2018 world wide web conference*. 933–943.

[22] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and VS Subrahmanian. 2018. Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 333–341.

[23] Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. 2016. Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 221–230.

[24] Srijan Kumar, Xikun Zhang, and Jure Leskovec. 2019. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 1269–1278.

[25] Mirko Lai, Viviana Patti, Giancarlo Ruffo, and Paolo Rosso. 2018. Stance evolution and twitter interactions in an italian political debate. In *International Conference on Applications of Natural Language to Information Systems*. Springer, 15–27.

[26] Mirko Lai, Marcella Tambuscio, Viviana Patti, Giancarlo Ruffo, and Paolo Rosso. 2019. Stance polarity in political debates: A diachronic perspective of network homophily and conversations on Twitter. *Data & Knowledge Engineering* 124 (2019), 101738.

[27] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. 2010. Signed networks in social media. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 1361–1370.

[28] Yu Li, Yuan Tian, Jiawei Zhang, and Yi Chang. 2020. Learning Signed Network Embedding via Graph Attention. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*. 4772–4779.

[29] Haoxin Liu, Ziwei Zhang, Peng Cui, Yafeng Zhang, Qiang Cui, Jiashuo Liu, and Wenwu Zhu. 2021. Signed Graph Neural Network with Latent Groups. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1066–1075.

[30] Seth A Marvel, Jon Kleinberg, Robert D Kleinberg, and Steven H Strogatz. 2011. Continuous-time model of structural balance. *Proceedings of the National Academy of Sciences* 108, 5 (2011), 1771–1776.

[31] Ana Peleteiro, Juan C Burguillo, and Siang Yew Chong. 2014. Exploring indirect reciprocity in complex networks using coalitions and rewiring.. In *AAMAS*, Vol. 14. 669–676.

[32] Simonetta Primario, Dario Borrelli, Luca Iandoli, Giuseppe Zollo, and Carlo Lipizzi. 2017. Measuring polarization in Twitter enabled in online political conversation: The case of 2016 US presidential election. In *2017 IEEE international conference on information reuse and integration (IRI)*. IEEE, 607–613.

[33] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. 2020. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637* (2020).

[34] Martin Saveski, Brandon Roy, and Deb Roy. 2021. The structure of toxic conversations on Twitter. In *Proceedings of the Web Conference 2021*. 1086–1097.

[35] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. 2020. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509* (2020).

[36] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter* 19, 1 (2017), 22–36.

[37] Lin Shu, Erxin Du, Yaomin Chang, Chuan Chen, Zibin Zheng, Xingxing Xing, and Shaofeng Shen. 2021. SGCL: Contrastive Representation Learning for Signed Graphs. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*. 1671–1680.

[38] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. 2019. Dyrep: Learning representations over dynamic graphs. In *International conference on learning representations*.

[39] Nathan Walter, John J Brooks, Camille J Saucier, and Sapna Suresh. 2021. Evaluating the impact of attempts to correct health misinformation on social media: A meta-analysis. *Health Communication* 36, 13 (2021), 1776–1784.

[40] Xuhong Wang, Ding Lyu, Mengjian Li, Yang Xia, Qi Yang, Xinwen Wang, Xinguang Wang, Ping Cui, Yupu Yang, Bowen Sun, et al. 2021. APAN: Asynchronous propagation attention network for real-time temporal graph embedding. In *Proceedings of the 2021 International Conference on Management of Data*. 2628–2638.

[41] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. 2021. Inductive Representation Learning in Temporal Networks via Causal Anonymous Walks. In *International Conference on Learning Representations*.

[42] Robert West, Hristo S Paskov, Jure Leskovec, and Christopher Potts. 2014. Exploiting social network structure for person-to-person sentiment analysis. *Transactions of the Association for Computational Linguistics* 2 (2014), 297–310.

[43] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 32, 1 (2020), 4–24.

[44] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2020. Inductive representation learning on temporal graphs. *arXiv preprint arXiv:2002.07962* (2020).