**Embedding Temporal Network via Neighborhood Formation**

**CCS CONCEPTS**

**• Information systems →** Data mining; Network data models;

**• Computing methodologies →** Dimensionality reduction and manifold learning;

**Network embedding** aims at representing large-scale networks by mapping nodes to low-dimensional space. Perform network mining (eg. node classification, link prediction, community detection) The neighborhood of a node is not formed simultaneously, and the observed snapshot network structure is the accumulation of neighborhood in certain time periods because a network is formed by adding nodes and edges sequentially.

**Co-author temporal network (a network with edges annotated by chronological interactive events between nodes)** needs to connect the nodes have same **co-authored paper**. The process to formed that is called **neighborhood formation sequence** (each neighbor can appear repeatedly in the sequence to represent multiple interactions with the source node. The events in neighborhood formation sequence are not independent because the historical neighbor formation events can influence the current neighbor formation).

By the time goes by, old co-author relation will vanish, and new relation will be occurred.

**HTNE (Hawks based Temporal Network Embedding): method used for solving problem**

1. Solving dynamic change between node connections.
2. Hawks process well capture the exciting effects between sequential events, influence the history of current events. Then adopt it for modeling the neighborhood formation process.
3. Solving large scale networks by optimizing the likelihood of neighborhood formation sequences.

Most network embedding approaches are based solely on the static neighborhood in representing a node because static network is only accumulation of historical formation and represent the structure at one particular time.

Dynamic network attempts to capture evolving network structure based on predefined time windows, the goal is representing the snapshots in different time periods but not reveal the complete temporal process for network formation.

Current events in Hawks process influenced by both last time step and historical events with time decay effect. Hawkes process can be extended to multivariate case where the conditional intensity function is designed for each event type as one dimension in order to handle different types of events. (**Modeling Neighborhood Formation Sequence via Multivariate Hawkes Process:** infer the current neighbor formation events from the conditional intensity)

**Attention for Sequence Formation**

The affinity between the history and the target node should depend on the source node.

**Experimental set up (1. Datasets choosing. 2. Baseline Methods. 3. Parameter Settings. 4.** **Tasks and Evaluation Measures)**

**Baseline methods:**

1. **LINE**: Optimizes node representations by preserving first-order or second-order proximities for a network
2. **DeepWalk**: Applies random walks to generate sequences of nodes from the network, and then uses it as input to the Skip-gram model to learn representations.
3. **Node2vec**: Extends DeepWalk by developing a biased random walk procedure to explore neighborhood of a node, strike a balance between local and global properties of a network.
4. **ComE**: Models community embedding, which can be utilized to optimize the node embeddings by introducing a community-aware high-order proximity

**Tasks and Evaluation Measures:**

**-Node Classification**

**-Link Prediction**

**-Temporal Recommendation**

**Experimental Result**

1. **Evaluation of node embedding.**

Get results for both Node Classification and Link Prediction. Better use Logistic Regression classifier.

1. **Evaluation of the conditional intensity function.**

Design temporal recommendation task on each dataset co-author network.

1. **Network visualization.**

Qualitatively evaluate node embeddings learned by different methods.

1. **Parameter sensitivity**

Get a fixed length, then compare the curves’ tendency (eg. when is rising and when is dropping)