Temporal Community Detection: A Comparative Study of NMF-based and GNN-based Approaches

Morteza Ziabakhsh
morteza24mail@protonmail.com

February 25, 2026

Abstract

This report presents a comprehensive implementation and comparative analysis of two temporal community detection approaches applied to dynamic networks. The first model, TCDA-NE (Temporal Community Detection and Analysis with Network Embeddings), is the baseline model I used. It is a novel algorithm that combines evolutionary clustering with convex non-negative matrix factorization (Convex-NMF). This implementation follows the research paper by Yuan et al. (2025) and represents the first publicly available implementation of this method, as no official implementation was provided by the original authors (I had to implement it myself). The second model, OverlappingGNN, is the result of my efforts to improve the baseline. It leverages Graph Neural Networks (GNNs) for overlapping community detection, based on the NOCD (Overlapping Community Detection with Graph Neural Networks) framework by Shchur et al. Both models were evaluated on the Enron email corpus, a real-world temporal network dataset. My evaluation employs multiple metrics including modularity, link prediction accuracy (AUC and AP), and temporal smoothness measures (consecutive NMI and F1). The results reveal an interesting trade-off: while TCDA-NE excels in temporal smoothness due to its explicit temporal regularization term, OverlappingGNN achieves superior community quality as measured by modularity and link prediction performance. This report provides detailed implementation descriptions, experimental methodology, and comprehensive analysis of the comparative results. All implementations are availabe at: https://github.com/Morteza-24/temporal-community-detection.

Contents

1 Introduction

1.1 Background and Motivation

Community detection in networks is a fundamental problem in network science with applications spanning social network analysis, biological systems, recommendation systems, and organizational studies. Traditional community detection methods treat networks as static entities, ignoring the temporal dynamics inherent in real-world systems. However, many real-world networks are inherently dynamic, with nodes and edges appearing, disappearing, and evolving over time. This temporal dimension introduces both challenges and opportunities for community detection.

Temporal community detection (TCD) aims to identify communities in dynamic networks while accounting for their evolutionary nature. The key challenges include:

- **Temporal Smoothness**: Communities should evolve smoothly over time, avoiding abrupt changes that are unlikely in real systems.
- **Community Quality**: Detected communities should be meaningful, with strong internal connections and weak external connections.
- **Overlapping Structure**: Real communities often overlap, with nodes belonging to multiple communities simultaneously.
- **Scalability**: Algorithms should handle large-scale temporal networks efficiently.

1.2 Research Objectives

This project implements and compares two distinct approaches to temporal community detection:

1. **TCDA-NE**: A matrix factorization-based approach that explicitly models temporal smoothness through evolutionary clustering. This implementation is based on the paper "Temporal Community Detection and Analysis with Network Embeddings" by Yuan et al. (2025). Notably, the original paper did not provide an official implementation, making this the first publicly available implementation of the TCDA-NE algorithm.
2. **OverlappingGNN**: A neural network-based approach using Graph Neural Networks for overlapping community detection. This approach applies the NOCD framework by Shchur et al. to each temporal snapshot independently, enabling the detection of overlapping community structures.

1.3 Contributions

The main contributions of this work include:

- First publicly available implementation of the TCDA-NE algorithm
- Adaptation of the NOCD framework for temporal community detection
- Comprehensive evaluation framework including modularity and link prediction metrics (not included in the original TCDA-NE paper)
- Fair comparison of matrix factorization and GNN-based approaches on a real-world dataset
- Analysis of the trade-offs between temporal smoothness and community quality

2 Methodology

2.1 Problem Formulation

Given a temporal network represented as a sequence of snapshots $\{A^{(1)}, A^{(2)}, \ldots, A^{(T)}\}$, where $A^{(t)} \in \mathbb{R}^{N \times N}$ is the adjacency matrix at time t, the goal is to find community assignments $\{C^{(1)}, C^{(2)}, \ldots, C^{(T)}\}$ that:

1. Maximize community quality within each snapshot
2. Maintain temporal smoothness between consecutive snapshots
3. Optionally, allow overlapping community memberships

2.2 TCDA-NE: Temporal Community Detection with Network Embeddings

2.2.1 Overview

TCDA-NE is a novel temporal community detection algorithm that combines evolutionary clustering with convex non-negative matrix factorization (Convex-NMF). The method innovatively integrates community structure into network embedding, preserving both microscopic details and community-level information in node representations while effectively capturing the evolutionary dynamics of networks.

2.2.2 Proximity Matrix Construction

A distinctive feature of TCDA-NE is its utilization of a common-neighbor similarity matrix, which significantly enhances the algorithm's ability to identify meaningful community structures in temporal networks. The proximity matrix S combines first-order and second-order proximity:

$$S = S^{(1)} + \eta \cdot S^{(2)} \tag{1}$$

where:

- $S^{(1)} = A$ is the first-order proximity (adjacency matrix)
- $S^{(2)}$ is the second-order proximity based on cosine similarity of neighbor vectors:

$$S_{ij}^{(2)} = \frac{\mathbf{n}_i \cdot \mathbf{n}_j}{\|\mathbf{n}_i\| \|\mathbf{n}_j\|} \tag{2}$$

where $\mathbf{n}_i$ is the neighbor vector of node i.

2.2.3 Convex-NMF Formulation

TCDA-NE uses Convex-NMF, which constrains the basis matrix W to be a convex combination of data points. The factorization is formulated as:

$$S \approx SWG^T \tag{3}$$

where:

- $W \in \mathbb{R}^{N \times K}$ is the weight matrix (constrained to have rows summing to 1)
- $G \in \mathbb{R}^{N \times K}$ is the community membership matrix
- K is the number of communities

2.2.4 Evolutionary Clustering Objective

The objective function for snapshot t incorporates temporal smoothness:

$$\mathcal{L}^{(t)} = \|S^{(t)} - S^{(t)}W^{(t)}(G^{(t)})^T\|_F^2 + \alpha\|S^{(t-1)} - S^{(t-1)}W^{(t-1)}(G^{(t-1)})^T\|_F^2 \tag{4}$$

where $\alpha \in [0, 1]$ controls the trade-off between:

- **Snapshot Quality**: How well the factorization reconstructs the current proximity matrix
- **Temporal Smoothness**: How similar the current factorization is to the previous snapshot

2.2.5 Multiplicative Update Rules

The optimization uses multiplicative update rules that guarantee non-negativity:

Update for W:

$$W \leftarrow W \odot \frac{S^T SG}{S^T SWG^T G} \tag{5}$$

followed by row normalization: $\sum_j W_{ij} = 1$ for all i.

Update for G:

$$G \leftarrow G \odot \frac{S^T SW + \alpha S_{prev}^T S_{prev} W_{prev}}{GW^T S^T SW + \alpha G W_{prev}^T S_{prev}^T S_{prev} W_{prev}} \tag{6}$$

where $\odot$ denotes element-wise multiplication.

2.2.6 Algorithm Summary

Algorithm 1 TCDA-NE Algorithm

Require: Snapshots $\{A^{(1)}, \ldots, A^{(T)}\}$, number of communities K, temporal weight α, proximity weight η

Ensure: Community assignments $\{C^{(1)}, \ldots, C^{(T)}\}$

```
for t = 1 to T do
    Compute proximity matrix S^(t) = A^(t) + η · S^(2)
    Initialize W^(t), G^(t) randomly
    repeat
        Update W^(t) using multiplicative rule
        Update G^(t) using multiplicative rule with temporal term (if t > 1)
    until convergence
    C_i^(t) = arg max_k G_ik^(t) for each node i
end for
```

2.2.7 Implementation Details

The implementation in `src/tcd/models/tcda_ne/model.py` follows the exact formulation from the paper:

- **Parameters**:
  - `num_communities`: Number of communities K (default: 20)
  - `alpha`: Temporal smoothness weight (default: 0.8, as recommended in the paper)
  - `eta`: First vs second-order proximity weight (default: 5.0, as recommended in the paper)

- `max_iter`: Maximum iterations per snapshot (default: 200)
- `tol`: Convergence tolerance (default: 10^{-50})

- **Initialization**: Random initialization for W and G matrices
- **Convergence**: Based on relative change in objective function

2.3 OverlappingGNN: GNN-based Overlapping Community Detection

2.3.1 Overview

The OverlappingGNN model was my idea to address two key limitations of TCDA-NE: (1) the assumption of non-overlapping communities. In real-world networks, nodes often belong to multiple communities simultaneously. (2) The lack of usage of stronger NN-based models. This model uses Graph Neural Networks to learn soft community memberships, allowing nodes to belong to multiple communities.

The implementation is based on the NOCD (Overlapping Community Detection with Graph Neural Networks) framework by Shchur et al., which uses a simple yet effective architecture combining a GCN encoder with a Bernoulli decoder.

2.3.2 Architecture

The model consists of two main components:

1. GCN Encoder: The encoder maps node features to community memberships:

$$Z = \text{ReLU}(\text{GCN}(X, A)) \tag{7}$$

where:

- X is the node feature matrix (identity matrix when no features are available)
- A is the normalized adjacency matrix
- $Z \in \mathbb{R}^{N \times K}$ is the soft community membership matrix

The GCN layer computes:

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}) \tag{8}$$

where $\tilde{A} = A + I$ is the adjacency matrix with self-loops and $\tilde{D}$ is the degree matrix of $\tilde{A}$.

2. Bernoulli Decoder (BerPo): The decoder reconstructs the adjacency matrix from community memberships:

$$P(A_{ij} = 1|Z) = 1 - \exp(-(Z_i \cdot Z_j + \epsilon)) \tag{9}$$

where $\epsilon = -\log(1 - p_{edge})$ is a correction term based on the edge probability.

2.3.3 Loss Function

The model is trained to minimize the Bernoulli Poisson (BerPo) loss:

$$\mathcal{L} = -\frac{1}{|E|}\sum_{(i,j)\in E}\log(1 - \exp(-\epsilon - Z_i \cdot Z_j)) + \frac{\lambda}{|E'|}\sum_{(i,j)\notin E} Z_i \cdot Z_j \tag{10}$$

where:

- E is the set of edges
- E' is the set of non-edges
- λ is a balancing parameter

2.3.4 Temporal Application

For temporal community detection, the model is applied independently to each snapshot:

Algorithm 2 OverlappingGNN for Temporal Networks

Require: Snapshots $\{A^{(1)}, \ldots, A^{(T)}\}$, number of communities K
Ensure: Soft memberships $\{Z^{(1)}, \ldots, Z^{(T)}\}$
1: **for** $t = 1$ to T **do**
2: Initialize GCN with random weights
3: Train on $A^{(t)}$ using BerPo loss with early stopping
4: $Z^{(t)} = \text{ReLU}(\text{GCN}(I, A^{(t)}))$
5: Normalize $Z^{(t)}$ to $[0, 1]$ range
6: **end for**

2.3.5 Implementation Details

The implementation in `src/tcd/models/overlapping_gnn/model.py` includes:

- **Parameters**:
  - `num_communities`: Number of communities K
  - `threshold`: Membership threshold for hard clustering (default: 0.5)
  - `hidden_sizes`: Hidden layer dimensions (default: [128])
  - `max_epochs`: Maximum training epochs (default: 500)
  - `dropout`: Dropout rate (default: 0.5)
  - `lr`: Learning rate (default: 0.001)
- **Training**: Stochastic gradient descent with edge sampling
- **Early Stopping**: Patience-based early stopping on validation loss

2.4 Evaluation Metrics

A comprehensive evaluation framework was developed to assess both community quality and temporal properties. The original TCDA-NE paper only reported temporal smoothness metrics (NMI and F1), but we argue that community quality metrics are equally important. Therefore, we include modularity and link prediction metrics in our evaluation.

2.4.1 Community Quality Metrics

1. Modularity: Modularity measures the quality of community structure by comparing internal edge density to expected density under a null model:

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j) \tag{11}$$

where:

- m is the total number of edges
- k_i is the degree of node i
- $\delta(c_i, c_j) = 1$ if nodes i and j are in the same community

Higher modularity indicates better community structure with dense internal connections and sparse external connections.

2. Link Prediction (AUC and AP): Link prediction measures the predictive power of community assignments. The intuition is that nodes in the same community are more likely to form edges. We use communities at time t to predict edges at time $t + 1$:

- **AUC (Area Under ROC Curve)**: Measures the probability that a randomly chosen edge has a higher prediction score than a randomly chosen non-edge.
- **AP (Average Precision)**: Summarizes the precision-recall curve and is more robust to class imbalance.

2.4.2 Temporal Smoothness Metrics

1. Consecutive NMI: Normalized Mutual Information measures the similarity between community assignments at consecutive time steps:

$$\text{NMI}(C^{(t)}, C^{(t+1)}) = \frac{2 \cdot I(C^{(t)}; C^{(t+1)})}{H(C^{(t)}) + H(C^{(t+1)})} \tag{12}$$

where I is mutual information and H is entropy.

2. Consecutive F1: F1 score with optimal matching measures community similarity using the Hungarian algorithm:

$$F1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{13}$$

where precision and recall are computed based on community overlap after optimal matching.

3 Dataset

3.1 Enron Email Corpus

The Enron email corpus is a real-world dataset widely used for temporal network analysis. It contains emails exchanged among employees of the Enron Corporation before its collapse in 2001.

3.1.1 Dataset Characteristics

- **Nodes**: 150 employees (email addresses)
- **Edges**: Email communications (sender $\rightarrow$ recipient)
- **Time Span**: Approximately 2 years of email communications
- **Temporal Resolution**: Monthly snapshots
- **Network Type**: Directed (converted to undirected for community detection)

3.1.2 Preprocessing Pipeline

The preprocessing pipeline in `scripts/preprocess_enron.py` performs the following steps:

1. **Email Parsing**: Extract sender, recipients, and timestamp from raw email files
2. **Node Mapping**: Create contiguous node IDs (0 to N-1) for all unique email addresses

3. **Edge Construction**: Build temporal edges (sender, recipient, timestamp)
4. **Snapshot Creation**: Convert continuous-time data to discrete monthly snapshots

The processed data is stored in PyTorch Geometric's `TemporalData` format for efficient loading and manipulation.

4 Experimental Setup

4.1 Implementation Details

4.1.1 Software Stack

- **Language**: Python 3.x
- **Deep Learning**: PyTorch, PyTorch Geometric
- **Scientific Computing**: NumPy, SciPy, scikit-learn
- **Graph Analysis**: NetworkX
- **Temporal Graph Benchmark**: TGB (Temporal Graph Benchmark)

4.1.2 Model Configurations

TCDA-NE Configuration:

```
TCDA_PARAMS = {
    "num_communities": 15,
    "alpha": 0.8,          # temporal smoothness
    "eta": 5.0,            # proximity weight
    "max_iter": 200,
    "random_state": 42
}
```

OverlappingGNN Configuration:

```
NOCD_PARAMS = {
    "num_communities": 15,
    "threshold": 0.5,
    "max_epochs": 300,
    "hidden_sizes": [128],
    "random_state": 42
}
```

4.2 Experimental Procedure

The comparison script (`scripts/train_and_compare.py`) performs the following:

1. Load the Enron dataset and create monthly snapshots
2. Train TCDA-NE on all snapshots
3. Train OverlappingGNN on all snapshots
4. Compute all evaluation metrics
5. Generate comparison tables

5 Results

5.1 Performance Comparison

Table 1 presents the main comparison results between TCDA-NE and OverlappingGNN on the Enron dataset.

Table 1: Final Fair Comparison Results (Higher = Better)

Model	Avg Mod.	Link Pred. AUC	Link Pred. AP	Consec. NMI	Consec. F1
TCDA-NE	0.1840	0.4882	0.5629	0.6500	0.6600 ± 0.1140
OverlappingGNN	0.4226	0.7285	0.6897	0.3951	0.3609 ± 0.1293

5.2 Analysis of Results

5.2.1 Community Quality (Modularity and Link Prediction)

The OverlappingGNN model significantly outperforms TCDA-NE in terms of community quality:

- **Modularity**: OverlappingGNN achieves 0.4226 compared to TCDA-NE's 0.1840, indicating that GNN-based communities have much stronger internal structure.
- **Link Prediction AUC**: OverlappingGNN achieves 0.7285 vs. 0.4882, showing that its communities are more predictive of future connections.
- **Link Prediction AP**: OverlappingGNN achieves 0.6897 vs. 0.5629.

These results demonstrate that the GNN-based approach learns more meaningful community structures that better reflect the underlying network topology.

5.2.2 Temporal Smoothness (Consecutive NMI and F1)

TCDA-NE excels in temporal smoothness metrics:

- **Consecutive NMI**: TCDA-NE achieves 0.6500 compared to OverlappingGNN's 0.3951.
- **Consecutive F1**: TCDA-NE achieves 0.6600 ± 0.1140 compared to OverlappingGNN's 0.3609 ± 0.1293.

This is expected because TCDA-NE has an explicit temporal smoothness term in its objective function (α parameter), which directly penalizes large changes between consecutive snapshots. In contrast, OverlappingGNN treats each snapshot independently without any temporal regularization.

5.2.3 Computational Efficiency

TCDA-NE is significantly faster:

- TCDA-NE: 4.3 seconds
- OverlappingGNN: 115.2 seconds

The matrix factorization approach is computationally efficient due to its closed-form update rules and lack of gradient computation. The GNN approach requires training a neural network for each snapshot, which is more computationally intensive.

5.3 Key Takeaways

The results reveal a fundamental trade-off:

1. **Community Quality vs. Temporal Smoothness**: There is an inherent tension between detecting high-quality communities and maintaining temporal smoothness. TCDA-NE optimizes for smoothness at the cost of community quality, while OverlappingGNN prioritizes community quality without temporal constraints.

2. **Metric Selection Matters**: The original TCDA-NE paper only reported NMI and F1 metrics, which favor their approach. Including modularity and link prediction provides a more complete picture of community detection performance.

3. **Application-Dependent Choice**: The choice between models should depend on the application:

   - For tracking community evolution over time (e.g., monitoring organizational changes), TCDA-NE's temporal smoothness is valuable.
   - For understanding community structure at a given time (e.g., identifying functional groups), OverlappingGNN's superior quality is preferable.

6 Discussion

6.1 Strengths and Limitations

6.1.1 TCDA-NE

Strengths:

- Explicit temporal smoothness through evolutionary clustering
- Fast and computationally efficient
- Interpretable factorization (W and G matrices)
- No hyperparameter tuning for neural networks

Limitations:

- Assumes non-overlapping communities
- Lower community quality as measured by modularity
- Sensitive to initialization
- May over-smooth communities, missing genuine structural changes

6.1.2 OverlappingGNN

Strengths:

- Detects overlapping communities
- Superior community quality
- Flexible architecture (can incorporate node features)
- Better link prediction performance

Limitations:

- No explicit temporal modeling
- Computationally expensive
- Requires neural network hyperparameter tuning
- May detect different community structures in consecutive snapshots

6.2 Why Modularity and Link Prediction Matter

The original TCDA-NE paper did not include modularity and link prediction metrics, focusing only on temporal smoothness (NMI and F1). We argue that these metrics are essential for a complete evaluation:

1. **Modularity** directly measures community quality by evaluating whether detected communities have dense internal and sparse external connections. High temporal smoothness with low modularity could indicate that the algorithm is simply maintaining poor community assignments over time.
2. **Link Prediction** measures the practical utility of community assignments. If communities capture meaningful structure, nodes in the same community should be more likely to form future connections. This is particularly important for applications like recommendation systems and link prediction.
3. **NMI and F1** measure temporal consistency but not community quality. A trivial solution that assigns all nodes to the same community would achieve perfect temporal smoothness but meaningless communities.

6.3 Future Directions

Based on my findings, several directions for future work emerge:

1. **Hybrid Approaches**: Combining the temporal smoothness of TCDA-NE with the representational power of GNNs could yield the best of both worlds.
2. **Using Node/Edge Features**: Many temporal graph datasets contain node/edge features and the NOCD model can handle features really well and I think they can improve community qualities significantly and possibly make the model more robust.