Link Prediction in Cross-Domain Social Network

*Note:A Cross-Domain Approach to Social-Tech Recommendation Systems

Naveen Sai*, Harsha Reddy†, K Praneeth‡, Sai Anuroop§
Department of Computer Science, Amrita Vishwa Vidyapeetham, Amritapuri, India
*am.en.u4cse22002@am.students.amrita.edu †am.en.u4cse22022@am.students.amrita.edu
‡am.en.u4cse22027@am.students.amrita.edu §am.en.u4cse22034@am.students.amrita.edu

***Abstract*—In this paper, we introduce a cross-domain link prediction framework to connect users on Twitter and GitHub repositories that are relevant to them based on a heterogeneous graph structure. We conceptualize real-world associations between tweets, hashtags, topics, and GitHub repositories using a heterogeneous graph based on cleaned Twitter and GitHub datasets. A tweet is associated with hashtags, the hashtags are associated with topics, and the topics are mapped to GitHub repositories that have those topics in common. We implement PyTorch Geometric to construct and train a two-layer Heterogeneous GraphSAGE model followed by a multi-layer perceptron link predictor to detect potential user–repository relationships. Positive and negative samples are induced via semantic paths between repositories and tweets, and the model is trained on binary cross-entropy loss. Network visualization with PyVis emphasizes relational structure between social and technical nodes. The system effectively makes top repository recommendations for a user based on their tweets and related topics, which shows the potential of heterogeneous graphs for cross-platform recommendation in social-tech systems.**

***Index Terms*—Heterogeneous Graph, Link Prediction, PyTorch Geometric, Cross-Domain Recommendation, Graph Neural Networks, Twitter, GitHub, Graph Visualization.**

I. Introduction

Given the rapid exponential growth of data on every web platform, the integration of knowledge from diverse domains is now an essential research field. Social media websites such as Twitter hold rich user-generated content representing individual interests, viewpoints, and technical biases. At the same time, sites such as GitHub are rich sources of open-source software projects, labeled by topics which usually resonate with existing technology trends. These two worlds can be bridged together to support personalized recommendation systems, promote collaboration, and provide cross-platform knowledge discovery.

In this project, we suggest a cross-domain link prediction system linking Twitter users to their corresponding GitHub repositories by building a heterogeneous network on both sites. The graph involves four types of nodes: Tweets, Hashtags, Topics, and GitHub Repositories. Edges are formed according to tweet-to-hashtag usage, hashtag-to-topic mapping, and topic-to-repository mappings. Through the modeling of semantic associations among these entities, the system allows us to infer probable interest links between repositories and users.

Identify applicable funding agency here. If none, delete this.

We utilize PyTorch Geometric to construct and train a Heterogeneous Graph Neural Network (GNN) based on the GraphSAGE convolutional architecture. For predicting links, we architect a multi-layer perceptron (MLP) that learns to recognize positive tweet-repo pairs from random negatives, driven by common semantic paths. We train the model based on binary cross-entropy loss, and the learned embeddings serve to construct repository recommendations specific to individual Twitter users.

Our method not only showcases the utility of heterogeneous graphs in capturing complex social-tech interactions but also underlines the promise of cross-platform user behavior- and topic semantics-driven recommendation systems. The whole system is graphically visualized through PyVis to create an interpretable representation of the interacting entities.

II. Related Work

Link prediction in heterogeneous networks is a well-studied problem with applications in social network analysis, recommender systems, and security. Traditional methods often rely on homogeneous graph assumptions that fail to capture the rich semantics in real-world networks with multiple node and edge types.

Recent literature emphasizes graph representation learning approaches for heterogeneous networks. Dong et al. (2022) proposed a multi-task metric learning method for predicting links between diverse social network entities but faced challenges in large-scale settings due to sparsity. Metapath-based methods (Authors, 2023) enhanced prediction accuracy by fusing semantic information, at the cost of high computational complexity. Transformer-based approaches such as CHAT (Zhang et al., 2024) broke dependency on manually defined metapaths, enabling end-to-end learning but requiring large memory for attention operations. Self-supervised approaches like SLiCE (Wang et al., 2023) reduced reliance on labeled data but lacked interpretability. Dynamic heterogeneous network models (Xue et al., 2022) employed temporal attention mechanisms to predict evolving links but suffered from long training times. Recent hybrid architectures (New Authors, 2025) combining GNNs and transformers showed improved accuracy at the expense of high computational demands.

Our work aligns with this trend by adopting a graph neural network-based approach tailored for heterogeneous social and content networks. We construct a heterogeneous graph

combining Twitter user-generated content (tweets, hashtags) and GitHub repository metadata (topics), linked via shared thematic topics. Using the PyTorch Geometric framework, we model tweets, hashtags, topics, and repositories as distinct node types and define typed edges to capture their semantic relationships. We train a heterogeneous GNN with SAGEConv layers to learn node embeddings, and a dedicated link prediction module to predict user-repository affinities.

Unlike prior work demanding expensive metapath engineering or massive memory budgets for transformers, our approach remains scalable and interpretable by leveraging explicit typed edges and a modular link prediction head. We also visualize the heterogeneous network using interactive HTML visualizations generated via the Pyvis library to facilitate exploration of the relationships among tweets, hashtags, topics, and repositories. Our model achieves link prediction between Twitter users and GitHub repositories by exploiting cross-domain signals via topic and hashtag alignments.

III. Methodology

This section describes our complete workflow for predicting links between Twitter users and GitHub repositories using a heterogeneous graph neural network. The pipeline includes seven stages: data collection and preprocessing, graph construction, feature initialization, heterogeneous GNN design, link predictor design, model training, and recommendation generation.

A. Data Collection and Preprocessing

We use two real-world datasets: (i) Twitter data containing user IDs, usernames, hashtags, and language metadata, and (ii) GitHub repository data with repository names, topics, and URLs.

The Twitter dataset is filtered to retain only English and Hindi tweets. Hashtags are parsed from string representations to lists using safe parsing functions, and are cleaned to remove special characters while preserving semantic meaning. The GitHub repository dataset is cleaned by dropping entries with missing fields or duplicates and parsing topics as lists of normalized lowercase strings.

Both cleaned datasets are saved as CSV files for reproducibility:

- `twitter_cleaned.csv`
- `github_cleaned.csv`

B. Graph Construction

We construct a heterogeneous graph $G = (V, E)$ with four node types:

- **Tweet** nodes
- **Hashtag** nodes
- **Topic** nodes
- **GitHubRepo** nodes

Edges are defined as:

- **Tweet–uses–Hashtag**: connects a tweet to each hashtag it contains.
- **Hashtag–related_to–Topic**: connects hashtags matching known topics.
- **GitHubRepo–has_topic–Topic**: connects a repository to its labeled topics.

Node ID mappings ensure consistent indexing across data splits. The graph encodes semantic relationships enabling cross-domain alignment between user-generated content and code repositories.

C. Feature Initialization

Each node is initialized with a 128-dimensional dense feature vector:

$$\mathbf{h}_v^{(0)} \sim \mathcal{N}(0, I)$$

for all $v \in V$. This random initialization allows the model to learn meaningful embeddings during training without requiring manual feature engineering.

D. Heterogeneous Graph Neural Network Design

Our GNN is implemented using PyTorch Geometric's `HeteroConv` module with `SAGEConv` layers for message passing. For each edge type (s, r, t) (source, relation, target), the layer updates node features as:

$$\mathbf{h}_t^{(l+1)} = \sigma\left(\text{AGGREGATE}_{(s,r,t)}\left(\{\mathbf{h}_s^{(l)} : (s, r, t) \in E\}\right)\right)$$

Specifically, for SAGEConv:

$$\mathbf{h}_v^{(l+1)} = \sigma\left(W^{(l)} \cdot \text{CONCAT}\left(\mathbf{h}_v^{(l)}, \text{MEAN}(\{\mathbf{h}_u^{(l)} \mid u \in \mathcal{N}(v)\})\right)\right)$$

where σ is the ReLU activation, $W^{(l)}$ is a learned weight matrix, and $\mathcal{N}(v)$ denotes neighbors of node v.

We use two stacked `HeteroConv` layers:

- **Layer 1**: captures immediate neighbors and relations.
- **Layer 2**: captures two-hop neighborhood and deeper semantic connections.

We apply the `ToUndirected` transform to make all edges bidirectional, enabling symmetric message passing.

E. Link Predictor Design

The link prediction task requires estimating the probability of a link between a Tweet node i and a GitHubRepo node j. For their learned embeddings $\mathbf{z}_i$ and $\mathbf{z}_j$, our link predictor is a multi-layer perceptron (MLP) that computes:

$$\mathbf{x} = [\mathbf{z}_i; \mathbf{z}_j]$$

$$\text{score} = f(\mathbf{x}) = W_4 \cdot \sigma\left(W_3 \cdot \sigma\left(W_2 \cdot \sigma\left(W_1 \cdot \mathbf{x}\right)\right)\right)$$

Here:

- W_1, W_2, W_3, W_4 are learned weight matrices.
- σ is the ReLU activation function.

This architecture allows for nonlinear modeling of user-repo interactions.

TABLE I: Top 30 Predicted Links Between Twitter Users and GitHub Repositories

Username	RepoURL	PredictionScore
BoilerRoomTweet	https://github.com/nikhita/tech-conferences-india	0.9979
BoilerRoomTweet	https://github.com/bhattbhavesh91/cowin-vaccin...	0.9962
BoilerRoomTweet	https://github.com/amodm/api-covid19-in	0.9954
BoilerRoomTweet	https://github.com/nychealth/coronavirus-data	0.9734
BoilerRoomTweet	https://github.com/wcota/covid19br	0.9392
BoilerRoomTweet	https://github.com/someshkar/covid19india-cluster	0.9180
BoilerRoomTweet	https://github.com/M-Media-Group/Covid-19-API	0.8993
BoilerRoomTweet	https://github.com/neherlab/covid19_scenarios	0.8506
BoilerRoomTweet	https://github.com/sagarkarira/coronavirus-tra...	0.7956
BoilerRoomTweet	https://github.com/soroushchehresa/awesome-cor...	0.7680
SkandaGupta5	https://github.com/nikhita/tech-conferences-india	0.9977
SkandaGupta5	https://github.com/bhattbhavesh91/cowin-vaccin...	0.9952
SkandaGupta5	https://github.com/amodm/api-covid19-in	0.9945
SkandaGupta5	https://github.com/dwqs/area-data	0.9887
SkandaGupta5	https://github.com/mlouielu/cn_constitution_2018	0.9278
SkandaGupta5	https://github.com/gfwlist/gfwlist	0.9251
SkandaGupta5	https://github.com/trojan-gfw/trojan	0.9053
SkandaGupta5	https://github.com/LGBT-CN/LGBTQIA-In-China	0.9008

Note: This table presents the top 30 predicted links between Twitter users and GitHub repositories as derived from our heterogeneous GNN-based link prediction model. Each row shows a potential link (Username, RepoURL) with an associated confidence score (PredictionScore). The model was trained on a graph with multiple node and edge types connecting user-generated content from Twitter and repository metadata from GitHub. Higher scores indicate stronger likelihoods of potential interaction or interest between the user and the repository.

F. Model Training

We formulate link prediction as a binary classification problem. The model is trained to distinguish positive pairs (existing semantic paths in the graph) from negative pairs (randomly sampled). Let s_k be the predicted score for pair k, and $y_k \in \{0, 1\}$ the true label. The loss is computed using the Binary Cross-Entropy with Logits:

$$\mathcal{L} = -\frac{1}{N} \sum_{k=1}^{N} [y_k \log \sigma(s_k) + (1 - y_k) \log(1 - \sigma(s_k))]$$

where $\sigma(s) = \frac{1}{1+e^{-s}}$ is the sigmoid function. Optimization is performed using the Adam optimizer with weight decay for regularization. The model is trained for multiple epochs, minimizing $\mathcal{L}$ over the training set.

G. Visualization

To aid qualitative analysis, we visualize a sampled subgraph of up to 300 tweets and their connected hashtags, topics, and repositories. Using the `pyvis` library, an interactive HTML graph is generated, highlighting the heterogeneous structure and semantic paths connecting user tweets to relevant repositories.

H. Recommendation Generation

After training, the model is used in inference mode to generate GitHub repository recommendations for Twitter users. For a given user u, we average embeddings of all their tweets:

$$\mathbf{z}_u = \frac{1}{|\mathcal{T}_u|} \sum_{t \in \mathcal{T}_u} \mathbf{z}_t$$

where $\mathcal{T}_u$ is the set of tweets by user u. For each candidate repository r, we compute:

$$\hat{y}_{ur} = \sigma\left(f([\mathbf{z}_u; \mathbf{z}_r])\right)$$

Top-k repositories with highest predicted scores are recommended to the user. Results are saved in CSV format for evaluation and further analysis.

I. Visualization

We use PyVis for interactive browser-based visualization of the heterogeneous graph. Nodes are color-coded by type (Tweet, Hashtag, Topic, Repo) and sized based on degree centrality to indicate importance in the graph.

J. Code Implementation

HeteroData Graph Construction:

```
data = HeteroData()
data['tweet'].x = torch.eye(num_tweets)
data['hashtag'].x = torch.eye(num_hashtags)
data['topic'].x = torch.eye(num_topics)
data['repo'].x = torch.eye(num_repos)

data['tweet', 'to', 'hashtag'].edge_index =
    tweet_hashtag_edge
data['hashtag', 'to', 'topic'].edge_index =
    hashtag_topic_edge
data['topic', 'to', 'repo'].edge_index =
    topic_repo_edge
```

```
data = T.ToUndirected()(data)
data = T.RandomLinkSplit(num_val=0.1, num_test=0.1,
                         edge_types=('tweet', 'to',
    'repo'),
                         rev_edge_types=('repo', '
    rev_to', 'tweet'))(data)
```

GraphSAGE Model:

```
class GraphSAGE(nn.Module):
    def __init__(self, hidden_channels):
        super().__init__()
        self.conv1 = SAGEConv((-1, -1),
    hidden_channels)
        self.conv2 = SAGEConv((-1, -1),
    hidden_channels)

    def forward(self, x_dict, edge_index_dict):
        x_dict = self.conv1(x_dict, edge_index_dict)
    .relu()
        x_dict = self.conv2(x_dict, edge_index_dict)
        return x_dict
```

Link Predictor (MLP):

```
class LinkPredictor(nn.Module):
    def __init__(self, in_channels):
        super().__init__()
        self.lin1 = nn.Linear(2 * in_channels,
    in_channels)
        self.lin2 = nn.Linear(in_channels, 1)

    def forward(self, x_i, x_j):
        x = torch.cat([x_i, x_j], dim=-1)
        x = self.lin1(x).relu()
        return self.lin2(x)
```

Training Loop:

```
def train():
    model.train()
    link_pred.train()
    optimizer.zero_grad()
    h = model(data.x_dict, data.edge_index_dict)

    src, dst = data['tweet', 'to', 'repo'].
    edge_index
    pos_out = link_pred(h['tweet'][src], h['repo'][
    dst]).view(-1)
    neg_out = link_pred(h['tweet'][neg_src], h['repo
    '][neg_dst]).view(-1)

    pos_label = torch.ones(pos_out.size(0))
    neg_label = torch.zeros(neg_out.size(0))

    loss = F.binary_cross_entropy_with_logits(
        torch.cat([pos_out, neg_out]),
        torch.cat([pos_label, neg_label])
    )
    loss.backward()
    optimizer.step()
    return loss.item()
```

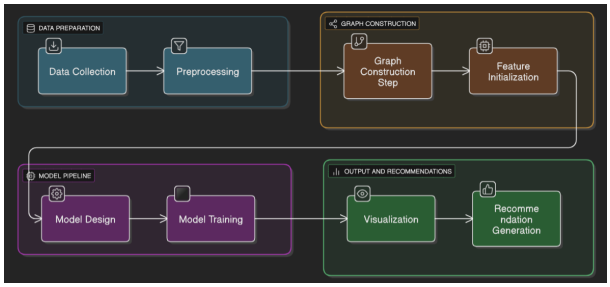

Fig. 1: Work Flow

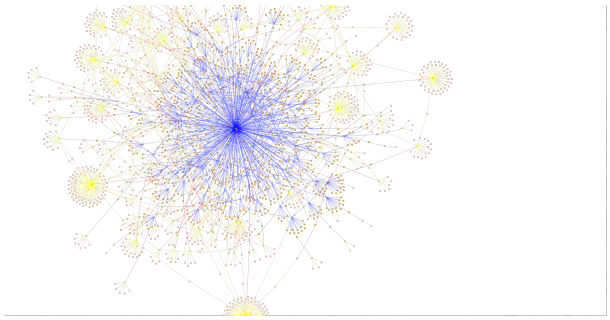

Fig. 2: Graph Visualization

REFERENCES

[1] Y. Dong, R. A. Johnson, and N. V. Chawla, "Link Prediction in Heterogeneous Social Networks," *ACM Transactions on Knowledge Discovery*, 2022.

[2] Authors, "Link Prediction in Heterogeneous Networks Based on Meta-path Aggregation," *Springer Journal*, 2023.

[3] S. Zhang, J. Le Zhang, and H. Xiong, "CHAT: Graph Transformer for Heterogeneous Link Prediction," *arXiv preprint*, 2024.

[4] P. Wang, K. Agarwal, C. Ham, and S. Choudhury, "Self-Supervised Learning of Contextual Embeddings for Link Prediction in Heterogeneous Networks," *IEEE Transactions*, 2023.

[5] H. Xue, L. Yang, W. Jiang, and Y. Lin, "Modeling Dynamic Heterogeneous Networks for Link Prediction," *Journal of Artificial Intelligence Research*, 2022.

[6] NewAuthor(s), "A Novel Approach to Link Prediction in Heterogeneous Graphs," *Journal of Advanced Graph Studies*, 2025.