Identification of Influential Users from Stack Overflow using Classical models and GNN

Group: 08

Harshavardana Reddy Kolan
Shikha Kumari

Contents:

Introduction

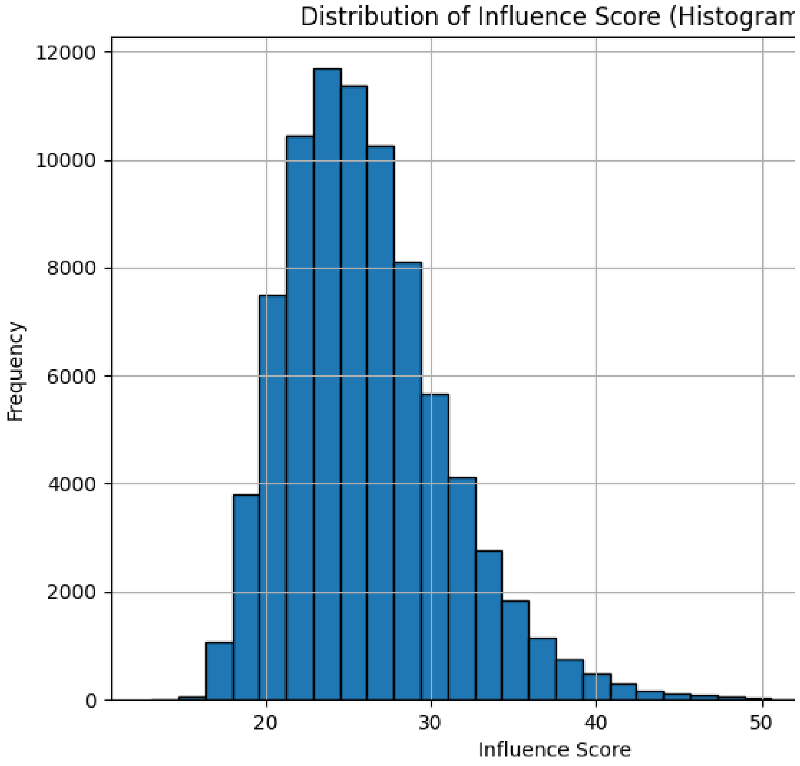
- **Problem Statement:**
  - Identifying influential users on Stack Overflow

- **Objective:**
  - Reward influential Stack Overflow experts to boost participation, engagement, and content quality.
  - Recommend them to people interested in their area of expertise.

Dataset Overview

- **What is the dataset?**
  - Stack Overflow interactions (Users, Questions, Answers)
- **How did we extract it?**
  - Collected using API
- **How did we transformed in the tabular form for classical models?**
  - Formula to define influence:

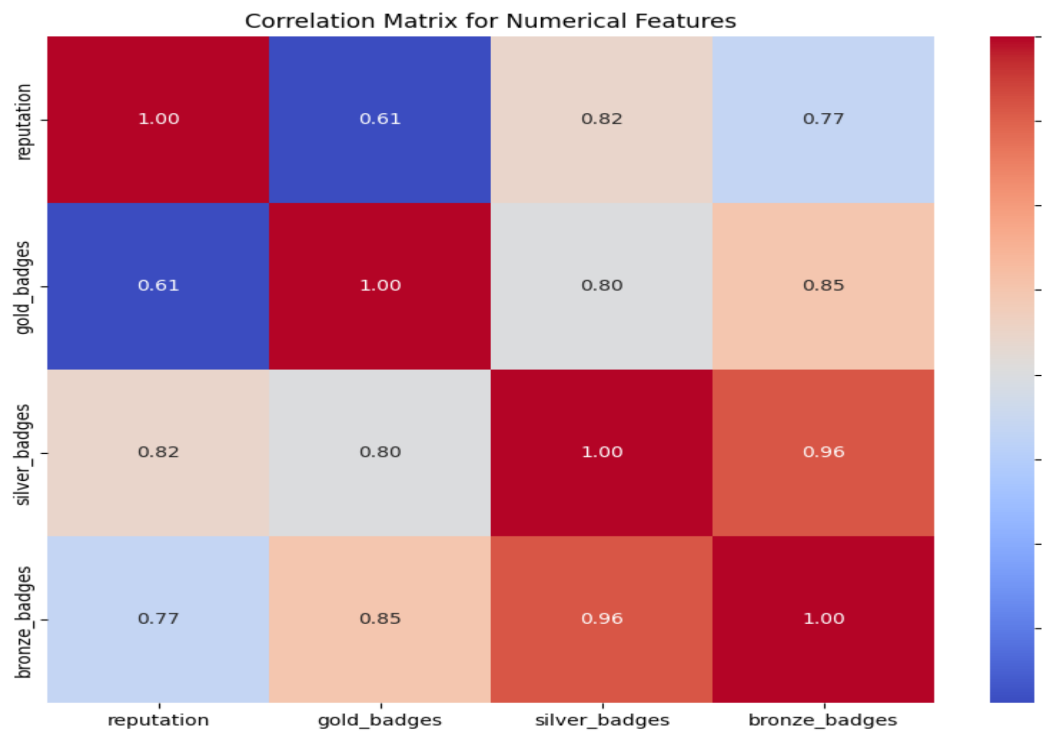
    influence_score=**reputation+3*gold_badge_count+2*silver_badge_count+ bronze_badge_count**
  - The threshold for defining influential users is the users falling in the top 10% of influence_score.
  - Aggregated questions and answers by User ID to summarize user activity (total questions, average scores, accepted answers).
  - Merged the datasets(Users, Questions, Answers) to create a single, comprehensive dataset for classical models.



Preprocessing:

Removed reputation and badge variables because:

- High correlation between features
- Used for defining target variable



Models Used:

- **Classical Models:**
  - **XGBoost** for tabular learning
- **Graph Neural Networks (GNN):**
  - Used **Heterogeneous Graph Representation** for user influence detection

Classical Model: XGBoost

- **Applied on structured tabular data**
- **Features:**
    - total_questions, avg_question_score, avg_answer_score, accepted_answers
- **Limitation:**
    - Ignores relationships between users and content (questions and answers)

Why GNNs Are a Game-Changer for This Task?

- GNNs are **essential** for this problem because **Stack Overflow interactions are inherently a networked structure**.

Graphs Model Real-World Interactions:

- Unlike tabular models, **GNNs understand relational data**.
- This allows us to **predict user influence** based on their **position in the network**.

Message Passing & Information Propagation:

- GNNs **aggregate information** from connected nodes.
- **Classical models fail to model this** interaction effect.

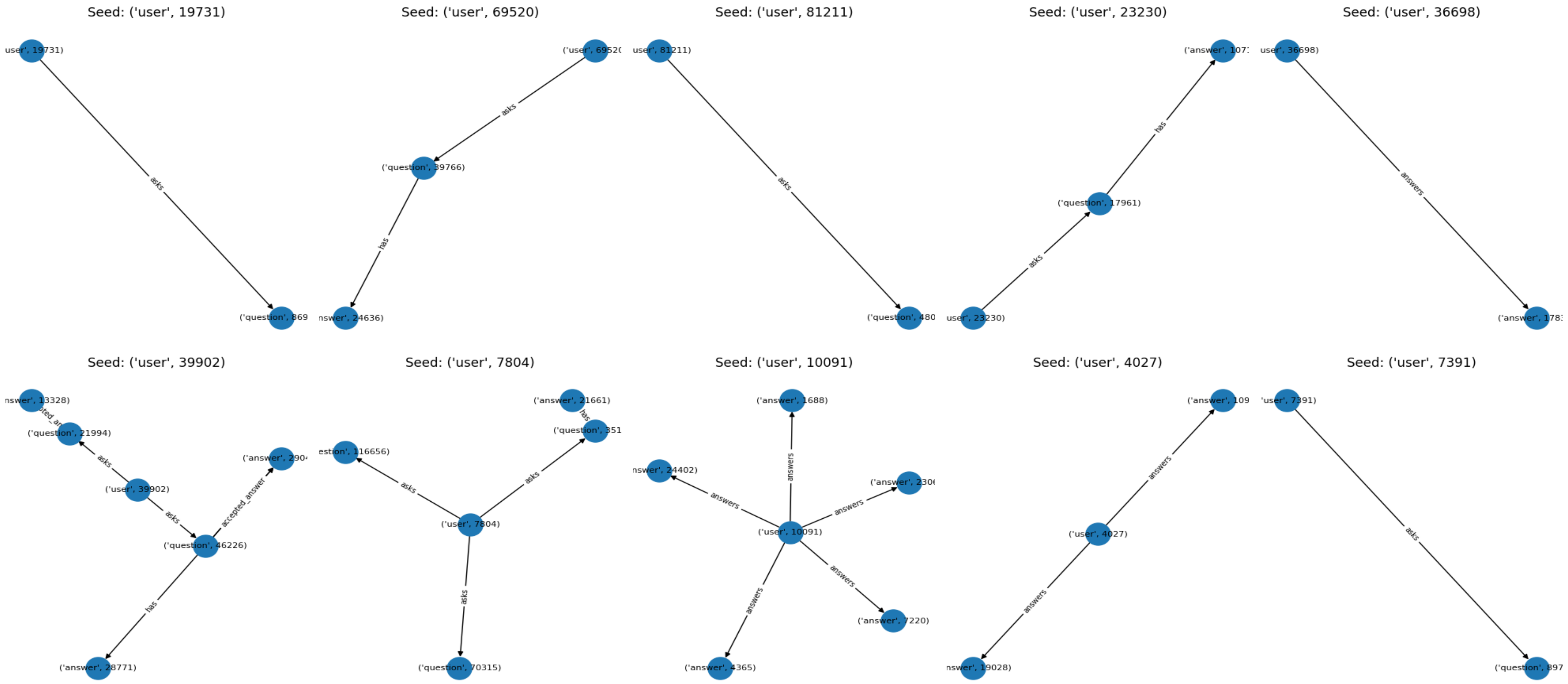
Heterogeneous GNNs Adapt to Different Node Types:

- Users, Questions, and Answers have **different roles**.
- GNNs **learn embeddings** specific to **each type of node** (e.g., an expert vs. a beginner has different engagement patterns).

GNN Structure

- **Nodes:**
  - **Users**
  - **Questions- Features(Question Score)**
  - **Answers- Features(Answer Score)**
- **Connections (Edges):**
  - **User → asks → Question**
  - **User → answers → Question**
  - **Question → has → Answer**
  - **Question → accepted_answer → Answer**
- **Reverse Edges:**
  Reverse edges capture information from the target node back to the source node, enabling bidirectional message passing and richer context for each node.
- **Self-Loops:**
  Self-loops allow each node to preserve and incorporate its own features during message passing, ensuring that unique, node-specific context isn't lost when aggregating only from neighbors.

Visualization of Graph



Heterogeneous GNN – The Why?

- Different Node and Edge Types Require Specialized Processing
  - **Multiple Node Types**
  - **Different Edge (Relationship) Types**
- User Interactions and Question-Answer Relationships Are Asymmetric
  - **Standard GNNs assume all relationships are equal**, but on Stack Overflow.
    - **Users engage in different ways**
    - **Asymmetry in Accepted Answers**
    - **Heterogeneous Edge Types Affect Message Passing**

*Standard GNNs **treat all edges the same**, which leads to **misrepresentation of influence**.*

Heterogeneous GNN – The How?

- 🛠 Training Setup for Heterogeneous GNN
- **Graph Neural Network (GNN) Model:**
  - Implemented using **PyTorch Geometric**
  - Uses **Heterogeneous Graph Convolutions** for multiple node and edge types
  - Message passing to capture complex relationships in Stack Overflow interactions
- **Hyperparameters:**
  - **Hidden Layers:** 2-layer **SAGEConv** with 64 hidden units
  - **Aggregation:** Sum pooling to aggregate neighbor information
  - **Learning Rate: 0.001**
  - **Batch Size:** Mini-batch (64) **or** Full-batch training
  - **Optimizer:** Adam(Weight Decay: **1e-4** for regularization)
- **Data Handling & Training:**
  - **Node Splitting: RandomNodeSplit** (80% train, 10% validation, 10% test)
  - **Self-loops:** Helps model **user self-engagement patterns**
  - **Mini-Batch Training: NeighborLoader** for efficiency
  - **Full-Batch Training:** When batch size is None, processing entire graph

Evaluation Metrics

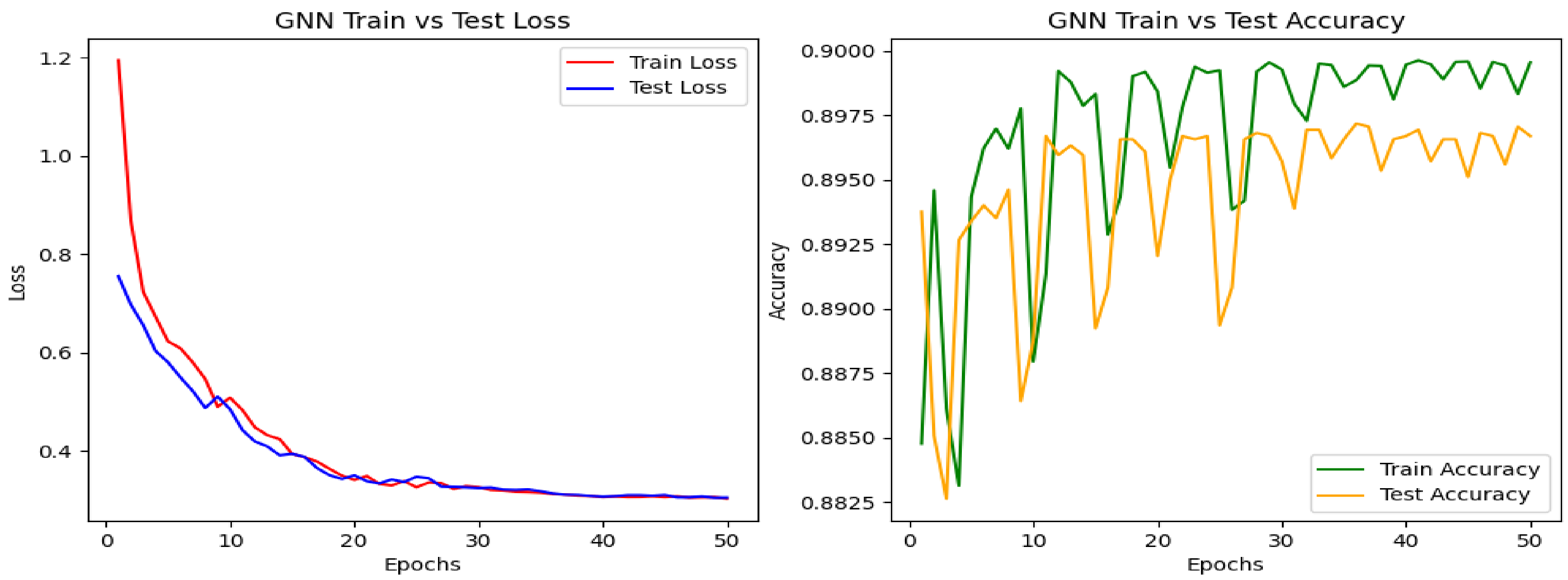
XgBoost

Metric	Value
F1-score	**0.602**
Accuracy	0.812
Precision	0.589
Recall	**0.646**
AUC	**0.646**

Heterogenous GNN

Metric	Value
F1-score	0.583
Accuracy	**0.883**
Precision	**0.623**
Recall	0.568
AUC	0.568

Train and Test (Loss and Accuracy) for GNN



Next Steps:

- Improving GNN model based on other metrics like F1-score.
- Extending to other problem statement. (In Social Network)