Heterogeneous Graph Embedding for Malicious Azure Sign-in Detection

Tad ZeMicheal, PhD | CAMLIS Oct 20, 2022

Agenda

- Problem and Motivation
- Background: Azure AD Authentication and Azure AD logs
- Authentications as a Graph
- Model Training and Results: Relational Graph Neural Network
- Key Take-Aways

Problem

1

Traditional rule-based heuristics do not flag all malicious log-ins.

2

Fully supervised ML methods require massive amounts of labelled data.

3

Other graph approaches fail to inference on previously unseen nodes.

Motivation

Heterogenous graph-based embedding allows:

Capture of structural identity and feature identify of nodes

Ability to capture evolving attacks due to connectedness of users

Minimized efforts on feature engineering tasks

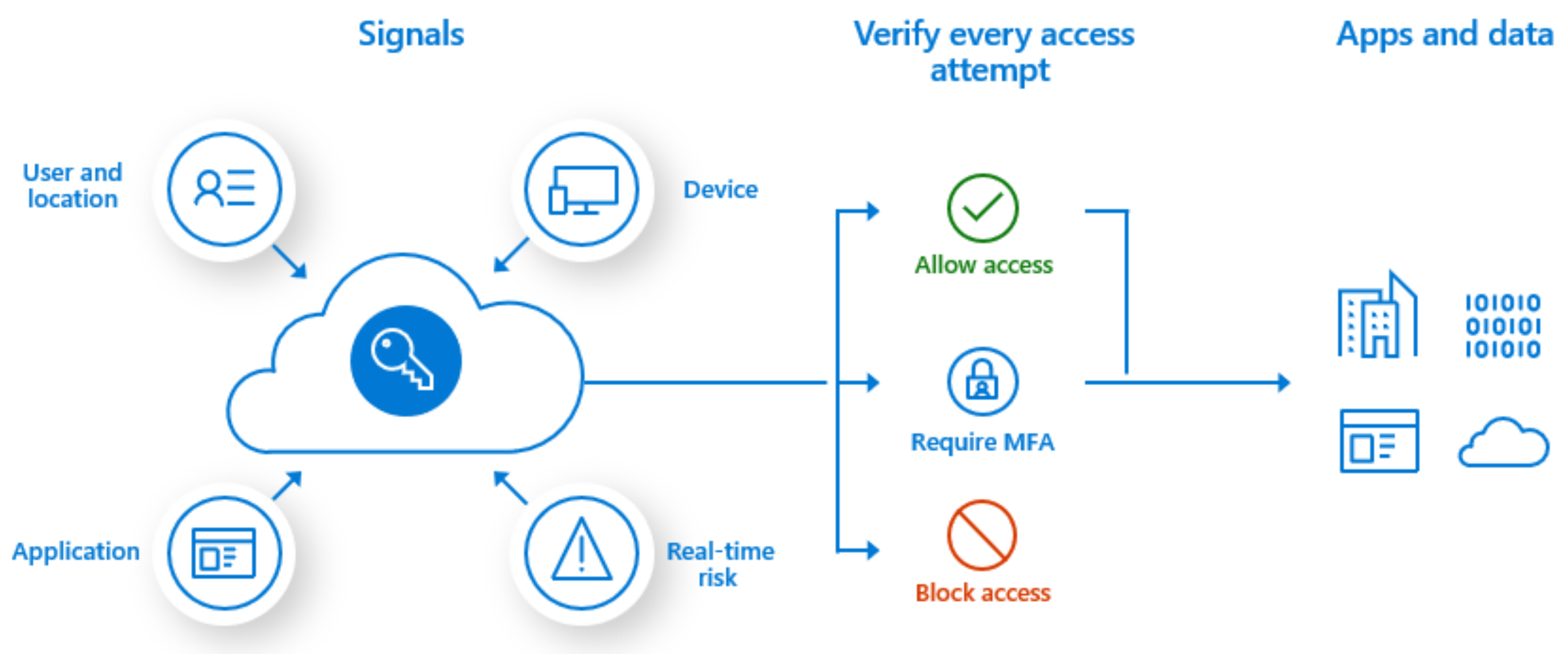
- This work follows on success of application of heterogenous GNN embedding on cyber applications such as fraud detection[1,2]
- Relation graph neural network is used to capture relation and graph structure of Azure authentication logs.

[1] Liu, Ziqi, et al. "Heterogeneous Graph Neural Networks for Malicious Account Detection.
[2] Rao, Susie Xi, et al. "xFraud: explainable fraud transaction detection." Proceedings of the VLDB Endowment 3 (2021).

Azure Authentication Process

- Registered resources are protected by Azure AD through the Azure authentication process
- Each access attempt is **verified** by Azure AD to make sure the user (through the given IP, device) has the **access right** to the application
- **Azure AD logs** provide information about sign-ins and how the resources are used by the users

  Examples of information included in Azure AD logs about each access attempt:
  - User identity (name, email, ID)
  - Application (name, ID)
  - Device information (device name, browser, OS)
  - IP address
  - Location (city, state, country)
  - Date and time
  - Authentication result (success or failure, reason)



Azure Sign-On Authentication

Sample of Azure AD Sign-in Logs

Activity Details: Sign-ins

Basic info | Location | Device info | Authentication Details | Conditional Access | Report-only | ...

Date	9/30/2021, 6:00:16 PM
Request ID	b8a2974d-21de-4ab6-b67e-6fbf21101e01
Correlation ID	386f50b3-6fe5-461c-be16-5f30c1b3527c
Authentication requirement	Single-factor authentication
Status	Success
Continuous access evaluation	No
Troubleshoot Event	Follow these steps: 1. Launch the Sign-in Diagnostic. 2. Review the diagnosis and act on suggested fixes.
Token issuer type	Azure AD

User	Isabella Simonsen
Username	isimonson@contoso.com
User ID	353564fa-8fbc-4fec-857f-ba487040f6f8
Sign-in identifier	isimonsen@contoso.com
User type	Member
Cross tenant access type	None
Client app	N/A
Client app	N/A
Application	Azure Portal
Application ID	c44b4083-3bb0-49c1-b47d-974e53cbdf3c
Resource	Windows Azure Service Management API
Resource ID	797f4846-ba00-4fd7-ba43-dac1f8f63013
Resource tenant ID	bf85dc9d-cb43-44a4-80c4-000000000000
Home tenant ID	bf85dc9d-cb43-44a4-80c4-000000000000
Home tenant name	
Client app	Browser

Details

Basic info | Location | Device info | Authentication Details | Conditional Ac

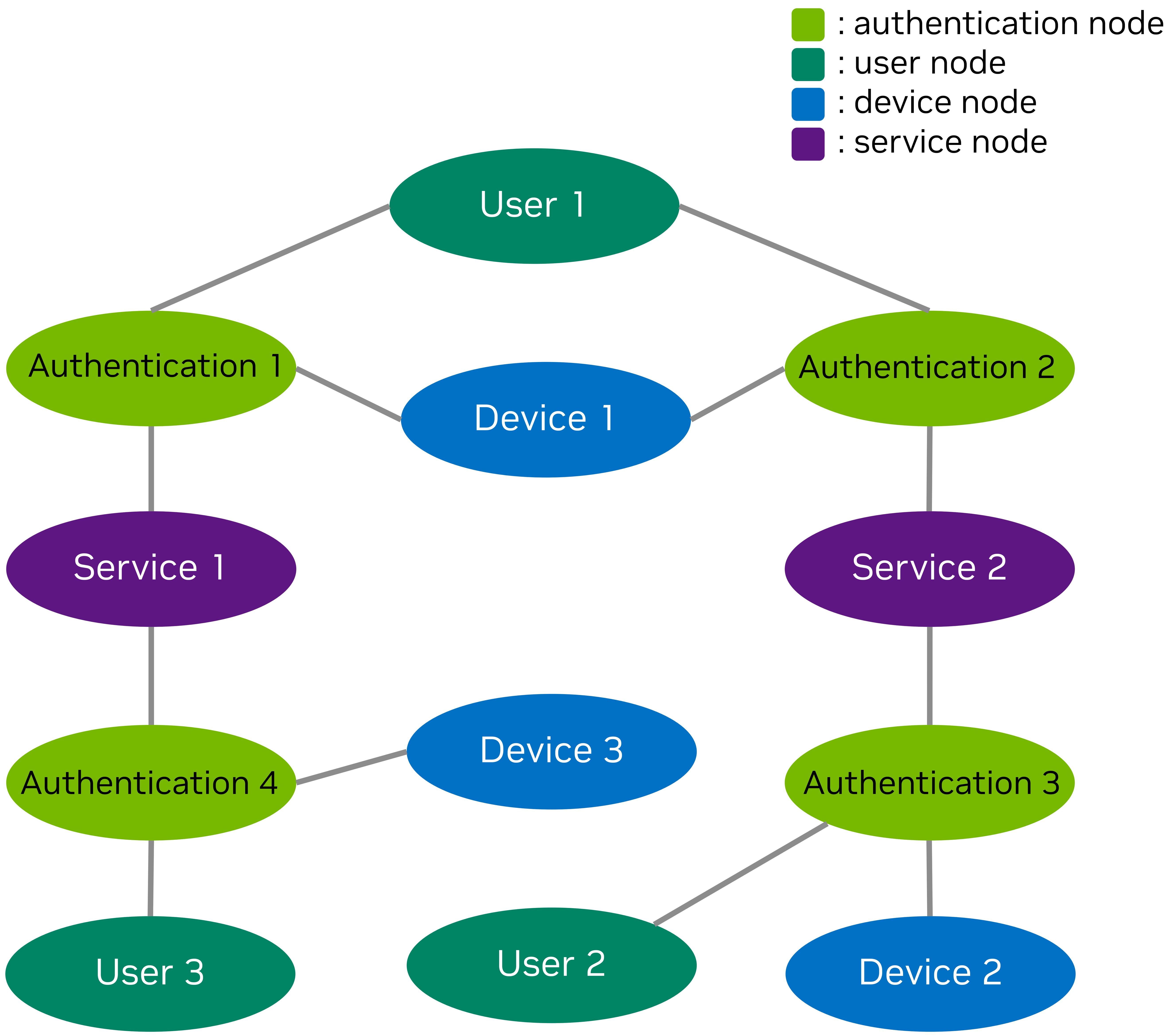
Date	4/16/2021, 6:26:30 PM
Request ID	f68a019d-372a-4763-9783-c0af80dd0100
Correlation ID	a1c3a648-db13-4716-b9bf-1423836ff217
Authentication requirement	Single-factor authentication
Status	Failure
Sign-in error code	50126
Failure reason	Error validating credentials due to invalid username or password.

Source: https://learn.microsoft.com/en-us/azure/active-directory/reports-monitoring/reference-basic-info-sign-in-logs

NVIDIA

Authentication Graph

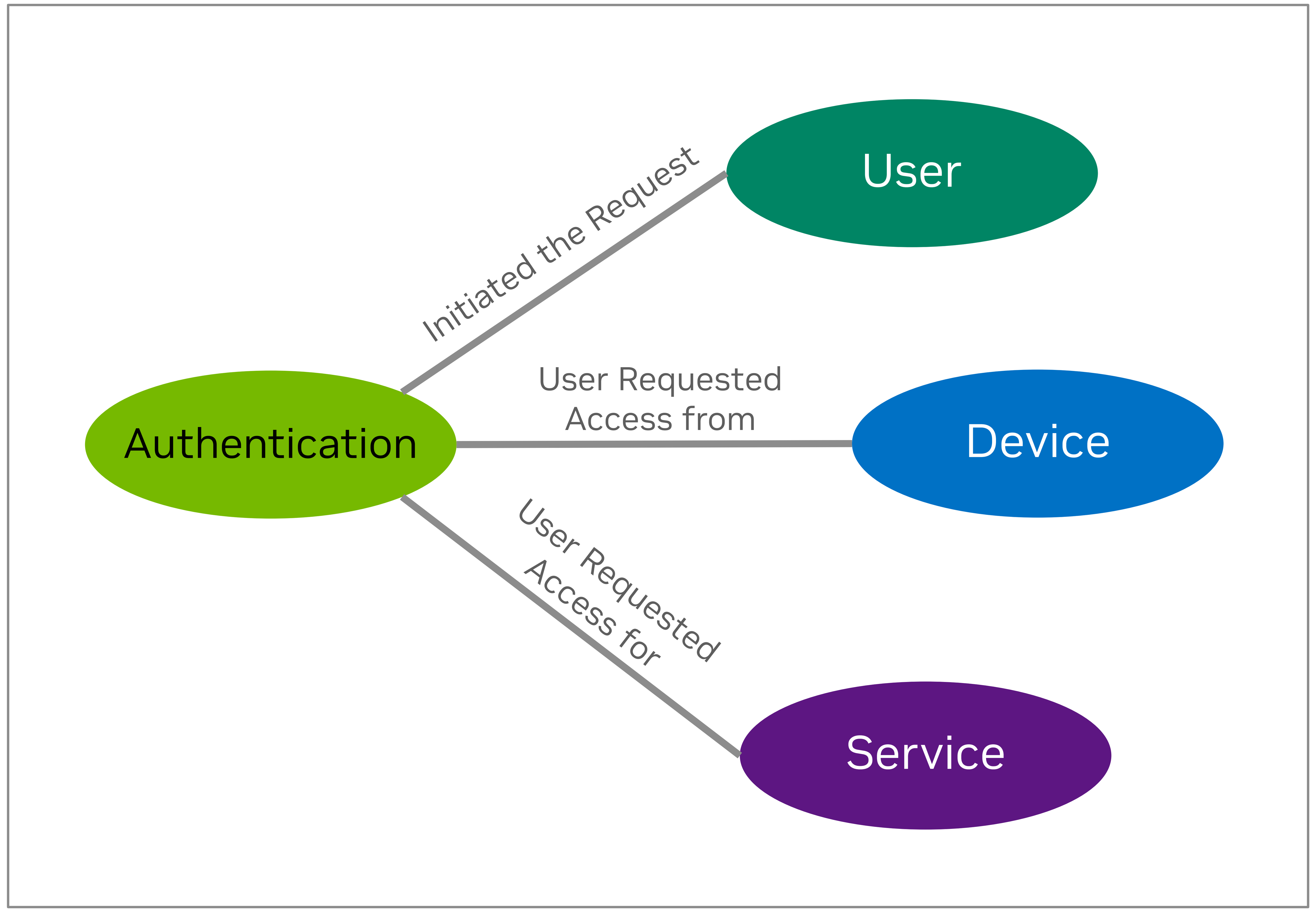
- There are 3 main entities involved in each authentication event:
  - The user who initiated the access request
  - The device used for the access request
  - The service being requested for access
- The authentication activities can be represented by a **graph**
  - The entities (user/device/service) are the nodes
  - Involvement in an access request links the user, device, and service nodes with the authentication node
  - Each authentication node will be connected to the 3 key entity nodes involved
- Each entity node can link to multiple authentication nodes
  - A user can authenticate multiple times
  - A device can be used to authenticate multiple times
  - A service can be requested access for multiple times



Graph Structure Formulation

- Input:
  - G = (V, E) , where V are nodes & E are connecting edges
  - V types: User (U) , Device (D), Service (S) and Authentication (A)
  - |V|: size of nodes
    - |V| = # unique users + # unique devices + # unique services + # authentications
  - X : feature matrix of node features
    - Authentication features (aggregated)
    - One-hot encoding (OHE) of categorical features
  - E: three types of relations connecting authentication node to user, service, and device nodes

- Output:
  - Embedding of authentication nodes



Proposed Model

Utilizing a relational graph neural network

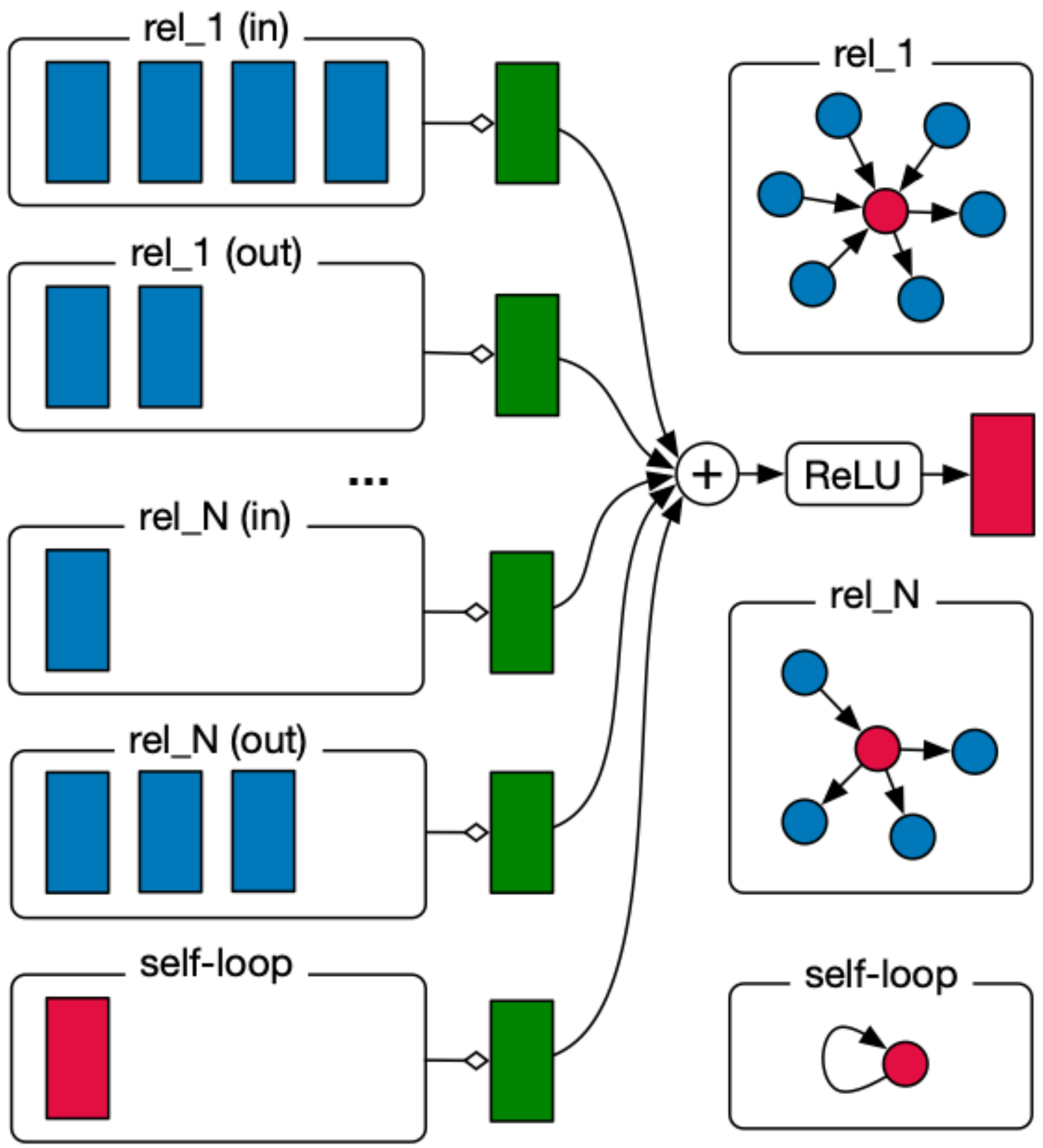
- Given graph G (V, E, X), learn an embedding of authentication
    - $Emb(A(t)) = f(U, D, S, A)$
    - Where U, D, S, A are heterogenous nodes of graph G
    - $A(t)$ is an authentication node embedding at time t
- Since $A(t)$ nodes are distinct in time t, all of time dependent dynamic features are associated as feature in $A(t)$ nodes.
- An $Emb(A(t))$ can be used for downstream task.
- For this model we tested Relational graph neural network (RGCN) as f to learn the embedding of heterogenous Azure graph.
- A semi supervised training is used to learn embedding of target nodes involved.
    - Target nodes in this case are the authentication nodes

Relational Graph Neural Network (R-GCN)

- R-GCN generalizes GCN to handle different relationship between entities.
- R-GCN uses different weights for different edge types of Heterogenous graph
  - **Relational GCN (RGCN):**

$$\mathbf{h}_v^{(l+1)} = \sigma\left(\sum_{r\in R}\sum_{u\in N_v^r}\frac{1}{c_{v,r}}\mathbf{W}_r^{(l)}\mathbf{h}_u^{(l)} + \mathbf{W}_0^{(l)}\mathbf{h}_v^{(l)}\right)$$

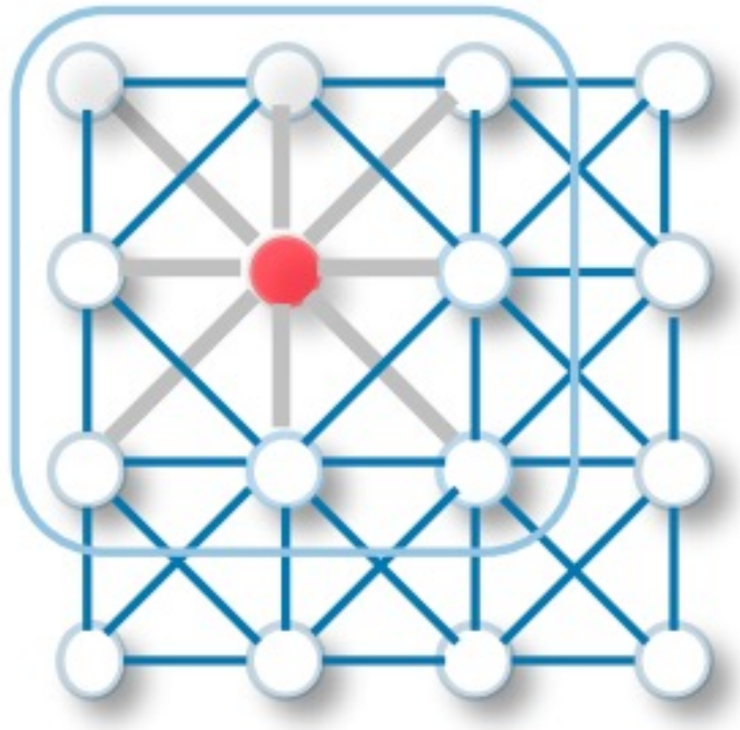
- Unlike GCN edges of same relation r are associated with same projection $\mathrm{W_r}$

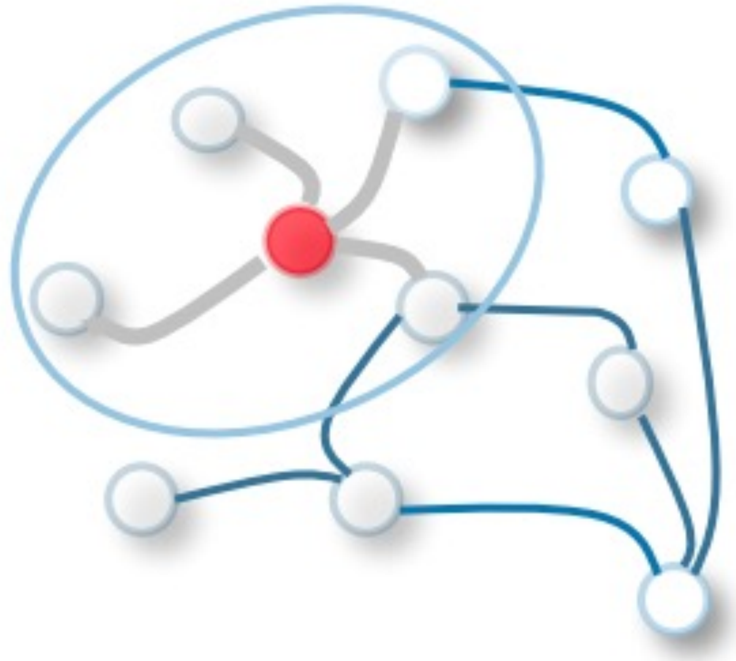


Schlichtkrull, Michael, et al. "Modeling relational data with graph convolutional networks."

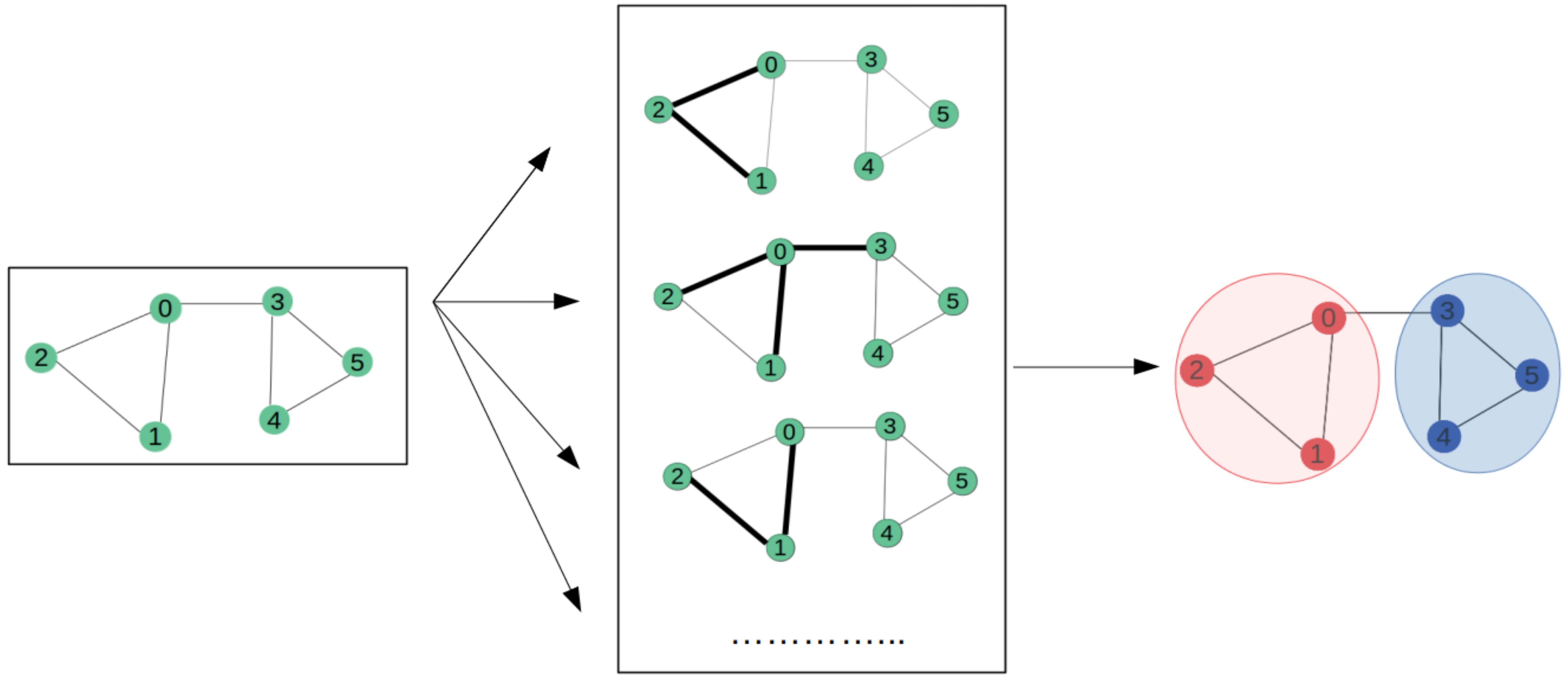
Graph Convolutional Networks

- Variant of convolutional neural network (CNN) which operate directly on graphs
  - CNN: operate on Euclidean space data
  - GCN: operate on irregular structures defined by nodes and edges





- GCN learns the features by aggregating weighted features of neighboring nodes
- Applications
  - Node/edge classification
  - Edge prediction
  - Fraud detection
  - Recommendation



[1] Kipf, Thomas N., and Max Welling. "Semi-supervised classification with graph convolutional networks." *arXiv preprint arXiv:1609.02907* (2016).
[2] Understanding Graph Convolutional Networks for Node Classification. https://towardsdatascience.com/understanding-graph-convolutional-networks-for-node-classification-a2bfdb7aba7b

Azure Log Dataset

- Azure log data of 3 months of 199 selected users with 2 compromised users for 4 days of login events.
- Number of raw authentication events: 315,234
- Raw feature size: 60
- **Training set:**
  - # Events: 45,975
  - # Features: 83
  - # Status failure (negative example) : 5,518
- **Testing set:**
  - # Events: 11,972
  - # Fraud events (negative example): 71
- **Graph:**
  - # Nodes: 62K
  - # Edges: 275,850

Feature Preprocessing

- One Hot Encoding (OHE): categorical features
  - *['riskState', 'deviceDetail.trustType', 'riskLevelDuringSignIn', 'riskLevelAggregated', 'clientAppUsed', 'deviceDetail.operatingSystem', 'date_hour']*
- Aggregation by: [“service”, “user’”, “device”, “timestamp”]
  - *sum, unique count, max*

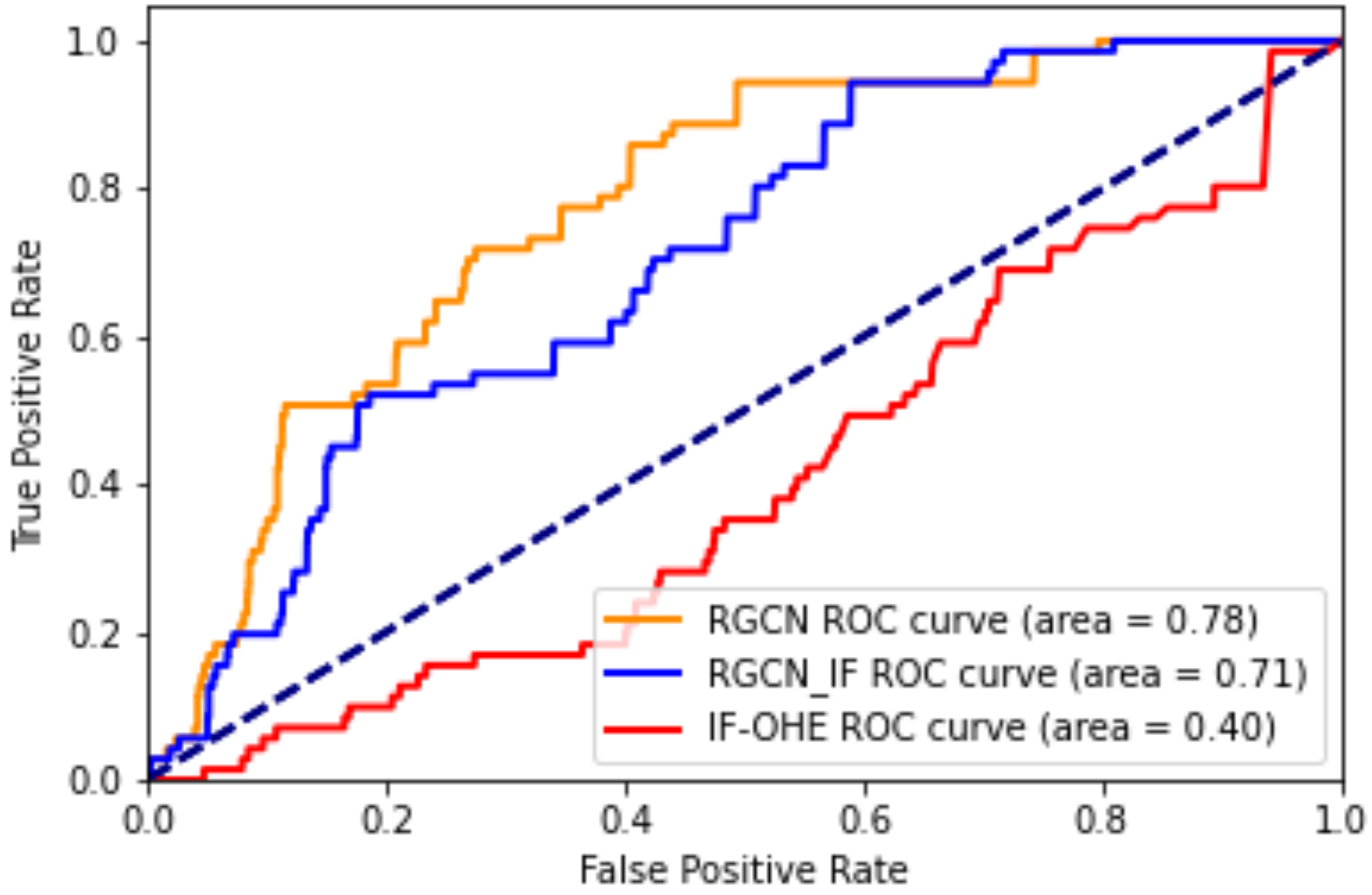
- For semi-supervised training, a binary feature of
  - **StatusFailure**: [Success/Failure]
    - Indicates whether an authentication results ”success” or “failure” due to various reasons

Experimental Result for Task 1

Task 1: Ability to identify malicious authentication requests

Averaged AUC performance of identifying malicious test events using RGCN & Isolation Forest

ROC of RGCN & RGCN_IF on authentication events

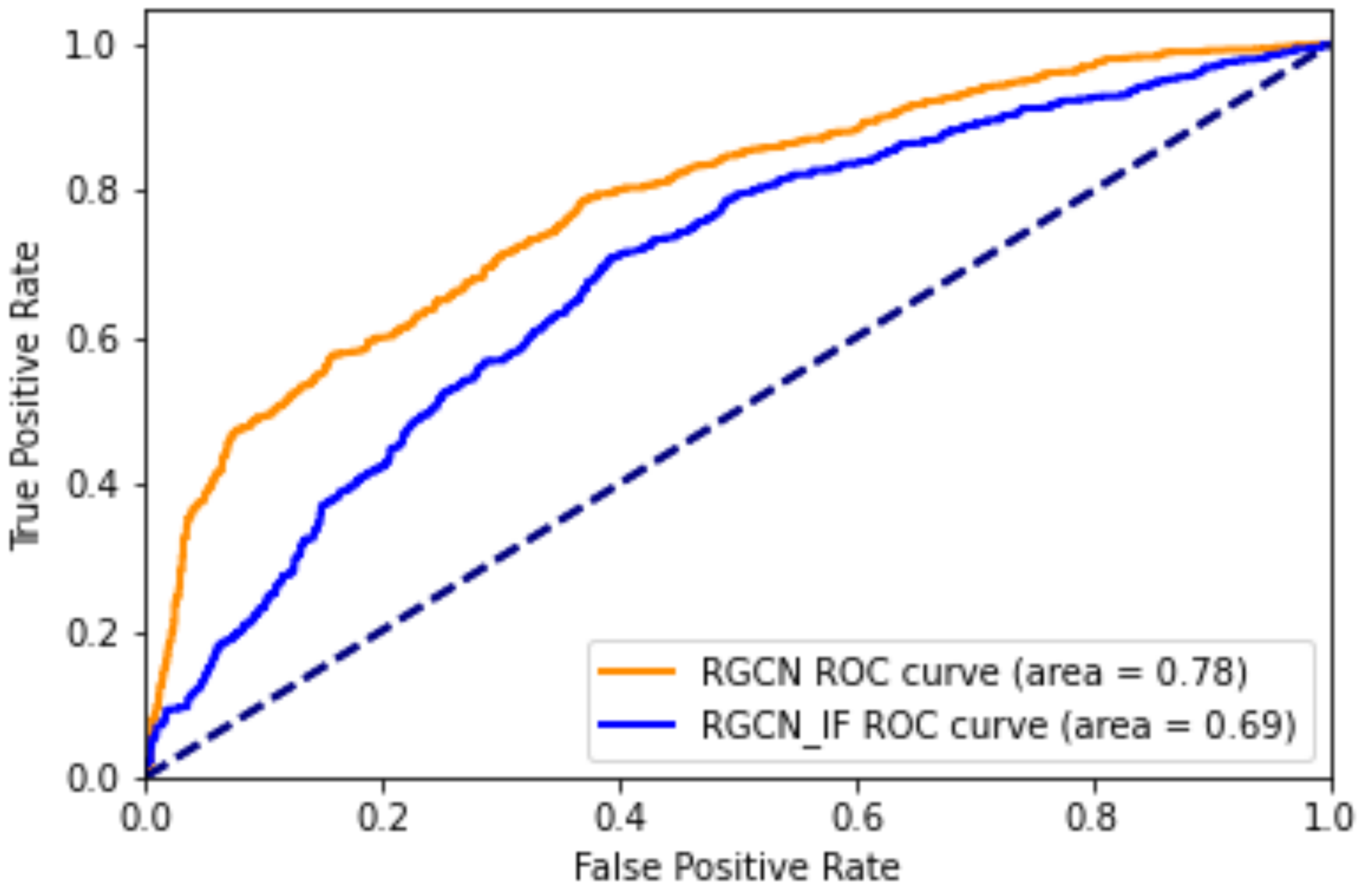


Experimental Result for Task 2

Task 2: Ability to detect malicious users on aggregated authentication requests

- Evaluated by taking median of anomaly scores across all services requested by a user per time period.
  - Detects users' activity on all application requested.
- Models' performance on aggregated authentication.

ROC of RGCN & RGCN_IF on user aggregation detection



Ranked Anomalous Based on RGCN Scores

Daily rank score of users averaged across services requested

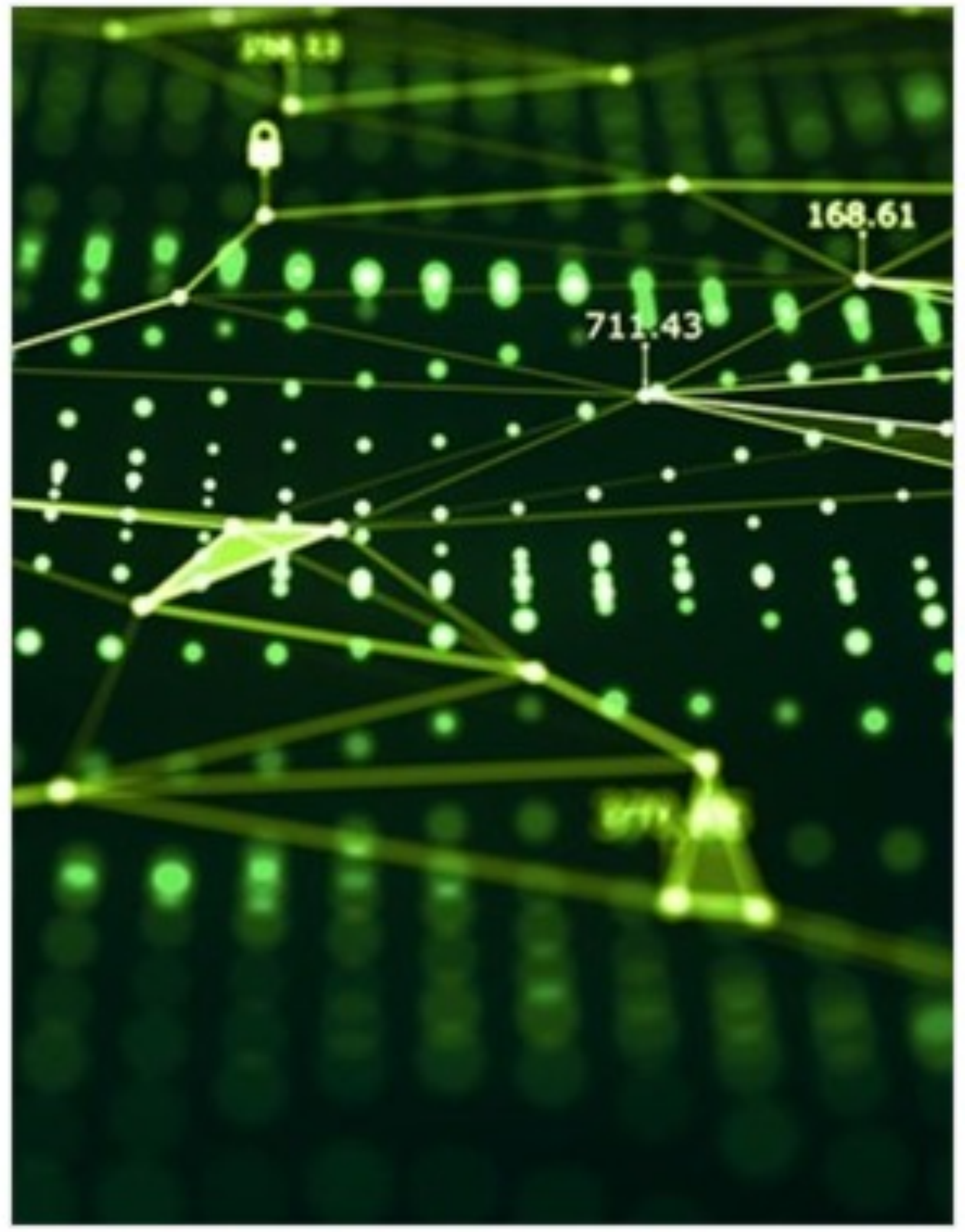
	UserId	ServiceId	FraudLabel	StatusFlag	RGCN	RGCN_IF
2136	eada5db7-3ada-45de-a971-2e985557825a	00000002-0000-0ff1-ce00-000000000000	1	1	0.998395	0.769497
12306	eada5db7-3ada-45de-a971-2e985557825a	194ffad0-4c5c-4c53-997d-eb3fdf026cc7	1	1	0.995080	0.761211
53609	6f6beb84-2600-4ee1-b035-26915aec1660	c00e9d32-3c8d-4a7d-832b-029040e7db99	0	1	0.963712	0.716407
53068	e3bde834-d0f4-4edb-8953-c84c6101002e	baaa4ff1-f56d-4f79-8a75-4c0cdf02e84a	1	1	0.936561	0.688361
5404	e3bde834-d0f4-4edb-8953-c84c6101002e	00000003-0000-0ff1-ce00-000000000000	1	0	0.906521	0.657165
12936	283c1cb1-d3cd-455f-b449-1ee8d0f52cca	1fec8e78-bce4-4aaf-ab1b-5451cc387264	0	1	0.863309	0.622619
53615	6f6beb84-2600-4ee1-b035-26915aec1660	c00e9d32-3c8d-4a7d-832b-029040e7db99	0	1	0.854788	0.606173
2132	eada5db7-3ada-45de-a971-2e985557825a	00000002-0000-0ff1-ce00-000000000000	1	1	0.848178	0.602308
18814	283c1cb1-d3cd-455f-b449-1ee8d0f52cca	29d9ed98-a469-4536-ade2-f981bc1d605e	0	0	0.831909	0.596086
12889	0df6b250-a5d8-4290-848e-29bf5fe32f9b	1fec8e78-bce4-4aaf-ab1b-5451cc387264	0	1	0.815989	0.599546

Successful login detected as fraud

Conclusion and Next Steps

- Adapting log authentication as GNN allows us to learn a richer embedding of authentication on both structural and individual entities involved without much hand-crafted feature learning
- By modeling every “authentication” as a target node, the model avoids the challenge of depending on modeling temporal historical user login information
- The inference on authentication is treated as an inductive setting on new unseen nodes
- Using RGCN semi supervised approach allows the learning to be less dependent on large amount of labelled data
- RGCN embedding on malicious authentication achieved better AUC performance compared with Isolation Forest

Try Morpheus in LaunchPad
Immediate, short-term remote access



nvidia.com/try-morpheus

Develop with Morpheus
Get Started in GitHub

https://github.com/nv-morpheus

NVIDIA