Vital nodes identification in temporal networks

Master's Thesis

Author: Martin Jolif

September 11, 2025

Internship at:
German Areospace Center (DLR) - Institut of Software Technology
April 7, 2025 - October 6, 2025

Internship Supervisors - Diaoulé Diallo & Tobias Hecking
Teacher Supervisor - Johannes Lutzeyer

Acknowledgements

I want to sincerely thank the German Aerospace Center (Deutsches Zentrum für Luft- und Raumfahrt; DLR) for giving me the chance to complete this master's thesis.

I am especially grateful to the Institute of Software Technology for providing me with the resources and facilities that made this work possible. I also want to thank the Intelligent Software Systems (ISS) Team for welcoming me into their group and creating a friendly atmosphere during my time here.

In particular, I want to thank Diaoulé Diallo and Tobias Hecking, my supervisors, for their guidance, constructive feedback, and valuable discussions throughout this project. Their expertise and advice were essential in overcoming difficulties and in improving the quality of my research.

Abstract

Temporal networks offer a powerful way to represent the dynamic behavior of many real-world systems, ranging from social interactions to communication infrastructures. Identifying vital nodes in these networks is important for a variety of applications, such as controlling epidemics, targeting marketing campaigns, or preventing cascading failures in power grids. Although many methods have been developed to find influential nodes in static networks, extending them to temporal networks remains challenging. This difficulty becomes even more pronounced when privacy restrictions limit the amount of available data.

This thesis explores Graph Neural Network (GNN)-based approaches for vital node identification (VNI) in temporal networks. I propose a method that learns temporal node embeddings from a sequence of networks and then aggregates these embeddings using an attention mechanism over time steps. This design allows the model to highlight the most relevant moments for node influence, rather than treating all time steps equally. Finally, the model predicts a vitality score for each node based on its aggregated embedding.

Experiments on several real-world datasets show that the proposed method can outperform a baseline approach in specific settings, highlighting the potential of GNNs for temporal node analysis. However, the results also reveal some important limitations. Indeed, the proposed approach depends on Suspected-Infected-Recovered (SIR) based simulations to generate ground-truth vitality scores, which are computationally demanding and sensitive to parameter choices. Additionally, the model's performance strongly relies on hyperparameter tuning and the characteristics of the dataset used for training and evaluation.

Note: there are two animated figures (2.2 & 2.3) that may not be displayed correctly in all PDF readers. For full functionality, please use a compatible reader such as Adobe Acrobat or Foxit.

Contents

1 Introduction

This master's thesis project was conducted in collaboration with the German Aerospace Center (Deutsches Zentrum für Luft- und Raumfahrt; DLR), Germany's national research and technology center for aeronautics, space, energy, transport, security, and defense.

The thesis work was carried out at DLR's Institute of Software Technology, one of the organization's 30 institutes and facilities located across Germany. The Institute focuses on enabling technologies for reliable and safety-critical software systems, artificial intelligence, high-performance and quantum computing, human-system interaction, visualization, software and systems engineering, and digital platforms.

Within the Institute, the project was hosted by the Department of Intelligent and Distributed Systems, and I was integrated into the Intelligent Software Systems (ISS) Team, a group of approximately 20 researchers and students whose work lies at the intersection of artificial intelligence, software engineering, and system science.

1.1 Motivation

In the era of interconnected systems and increasing data availability, understanding the dynamics of complex networks over time is essential for addressing real-world challenges. Applications such as power grid monitoring, epidemic containment, strategic marketing, and information propagation all benefit from the ability to identify the most influential or vital nodes within temporal networks.

Temporal networks, in which connections between nodes evolve over time, present unique challenges to model influence, spread, and systemic vulnerability. While recent advances in graph learning have improved our ability to analyze such structures, the issue of privacy-preserving node analysis remains an open problem, particularly relevant in domains involving sensitive personal or behavioral data [6].

This thesis aims to contribute to this emerging field by designing and implementing an architecture capable of identifying vital nodes in temporal networks using Graph Neural Networks. The work combines ideas from representation learning, temporal sequence modeling, and influence estimation but only briefly explores privacy aspects.

This work is part of the DLR Graduate School GANDALF project, an initiative which brings together researchers from across DLR institutes to strengthen the resilience of public transportation systems against pandemic threats. Through interdisciplinary collaboration, including information technology, microbiology, and fluid mechanics, the program explores innovative technologies for risk detection, assessment, and mitigation. In this context, understanding influence and propagation patterns in dynamic systems is a key enabler for early-warning mechanisms and targeted interventions, which is an application of the identification of vital nodes in temporal networks.

1.2 Objective

The main goal of this thesis is to design and implement a machine learning framework that can identify vital nodes in temporal networks. In particular, I aim to show that Graph Neural Network (GNN)-based methods can outperform traditional approaches in this task. The framework I propose is meant to be flexible and useful in different domains where dynamic networks play an important role. Possible applications include preventing cascading failures in power grids [5], studying how diseases spread in epidemiology [22], and understanding how information spreads online, for example, in the case of fake news on social networks.

Contact network data is typically represented as a list of triplets. Each triplet contains the IDs of two connected nodes and the timestamp indicating when the interaction occurred (Figure 1.1).

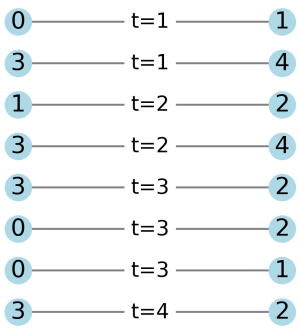


Figure 1.1: Visualization of the raw contact data underlying the temporal network

However, to apply methods from the graph learning community, particularly Graph Neural Networks (GNNs), I will convert the raw contact data into a sequence of graphs. For example, in a temporal network with 5 nodes, the data can be represented as a time series of undirected and unweighted graphs, each capturing the interactions at a given moment in time (Figure 1.2). Alternatively, it is also possible to aggregate interactions over defined time intervals, allowing for a coarser view of the network's temporal dynamics. This interval-based aggregation will be adopted in my study to avoid having too many snapshots.

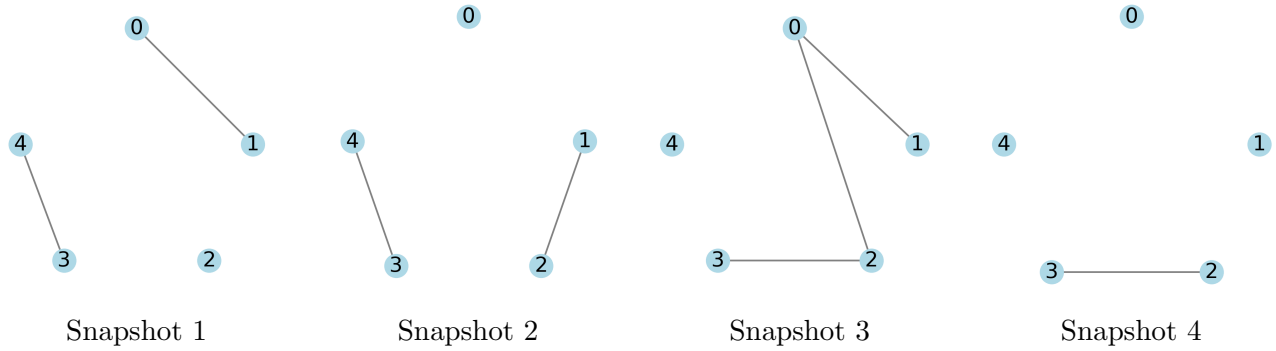


Figure 1.2: Example of a temporal network with 5 nodes associated with the contact data of Figure 1.1

The global idea of my machine learning architecture will be to process as input these sequences of graphs, also called snapshots [3, 32]. The goal would be to use Graph Neural Networks to learn temporal node feature vectors. Finally, these temporal node embeddings will be used to predict a vitality score for each node, representing the node's influence in the temporal network (Figure 1.3).

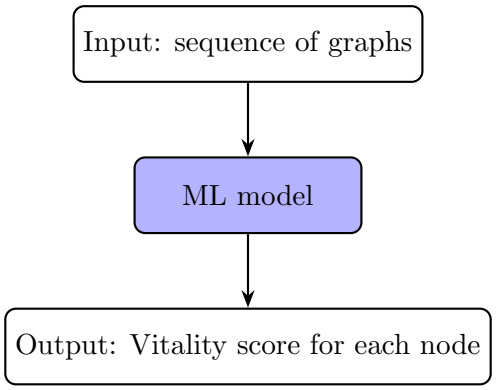


Figure 1.3: Schematic overview of the proposed architecture: learning module in blue.

For example, in the context of infectious disease spread, contact networks are often used, where nodes represent individuals and edges represent physical interactions between them. The timing and frequency of these contacts significantly influence the dynamics of disease transmission, making the identification of vital nodes in temporal networks crucial for effective containment strategies.

2 Background and related work

In conducting a literature review, I initially focused on understanding the concept of vitality and examining the existing methodologies for identifying such nodes in both static and temporal networks. While several methods have been developed for static networks, there is a noticeable gap in approaches that account for the temporal dynamics inherent in real-world systems. This gap is even more pronounced when considering privacy constraints, rendering this intersection of temporal analysis and privacy a largely unexplored research area.

Additionally, I explored methods aimed at solving related problems (section 2.5), such as the influence maximization problem, to gain insights that could be applicable to identifying vital nodes. I also analyzed various machine learning architectures designed for tasks like node classification and edge prediction in temporal graphs (section 2.4). This analysis helped uncover techniques and perspectives that could be adapted to tackle the specific challenges of vital node identification in temporal networks.

2.1 Vital nodes

We can define the vitality of a node as the number of nodes it can influence under a certain diffusion dynamic [1]. This approach is related to numerical simulations, which will be explained in the next subsection 2.1.1. Essentially, a node with a high vitality score is considered important or influential within a network.

2.1.1 Suspected-Infected-Recovered (SIR) Model

The SIR model is a classic epidemic model used in network science and epidemiology to simulate how a contagion (like a virus or information) spreads through a population. It divides the nodes (individuals) into three compartments:

- S (`Susceptible`): Not yet infected but can be.
- I (`Infected`): Currently infected and can infect others.
- R (`Recovered`): No longer infected (assumed immune and not spreading the contagion).

As we can observe in the diagram 2.1, the possible changes of states occur when infected nodes infect their susceptible neighbors with probability β and recover with probability γ:

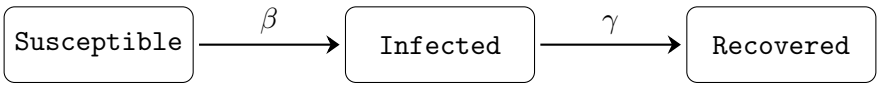


Figure 2.1: State transitions in the SIR model. Susceptible nodes become infected at rate β, while infected nodes recover at rate γ.

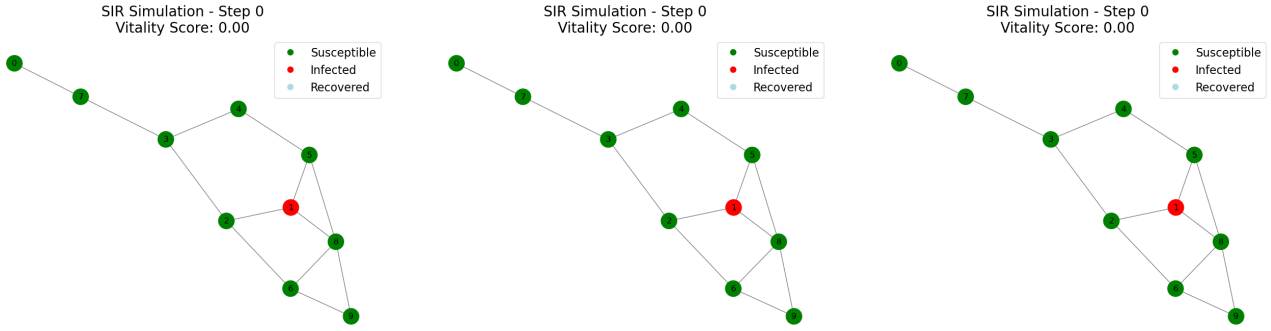
To compute the vitality score of node i, we start by having node i in the `Infected` state and all other nodes in the `Susceptible` state. Then, the simulation process is dependent on whether we process a static or a temporal network.

Static Networks

In the case of static networks, the virus spreads through the network according to the SIR model (Figure 2.1) and the simulation is stopped when there are no more nodes in the `Infected` state. The vitality score of node i is then defined as following [1]:

$$V(i) = \frac{\text{Number of nodes in the } \texttt{Recovered} \text{ state at the end}}{\text{total number of nodes in the network}} \tag{2.1}$$

As the process is stochastic, we run the process `n_simulations` times and do the average over the simulations to obtain a good statistical estimate. Here is an example for a static network (Figure 2.2):



Run 1 Run 2 Run 3

Figure 2.2: Animated SIR spreading processes, $\beta = 0.3$, $\gamma = 0.1$, and node 1 as initial infected node.

The final vitality scores of Figure 2.2 are 1, 0.8 and 0.7, so the average vitality score is equal to 0.83. In our case we would do the process `n_simulations` times to obtain a good statistical estimate of the vitality score. For example, if we do 100 runs, we obtain a vitality score of 0.86 ± 0.04.

Temporal networks

While the SIR model can be applied to static networks, many real-world processes take place in settings where interactions change over time [3, 31, 32]. In such cases, a temporal network representation is more appropriate. Temporal networks offer a more accurate way to model real-world spreading processes, as the timing and order of interactions can influence how diffusion unfolds [3]. In a static network, the contacts are considered constantly available, which can overestimate the spreading potential. Including temporal information allows us to follow the real sequence of contacts and identify time-related effects, such as bottlenecks, that can significantly impact processes like epidemics, information sharing, or influence propagation.

In the case of a temporal network, the global idea of the SIR model is the same as for a static network. However, this time, the spreading process is done through temporal paths instead of classical paths, and the simulation is stopped at the final time step of the temporal network. Moreover, another difference is that we can start the spreading process at each time step of the temporal network.

If we note $T_i = \{t_i^1, \dots, t_i^m\}$, the set of active time steps of node i (time steps where node i has at least one neighbor). Let $j \in [\![1, m]\!]$, we will look at $R_i^{t_i^j}$ which corresponds to the number of nodes in the `Infected` & `Recovered` states at the final time step, by starting the spreading process at time step t_i^j with node i as the seed node. As this spreading process is stochastic, we launch the simulation `n_simulations` times and just note $R_i^{t_i^j}$ the average over the simulation runs.

To have a unique vitality score for node i, we will look at the following three score types [3, 31]:

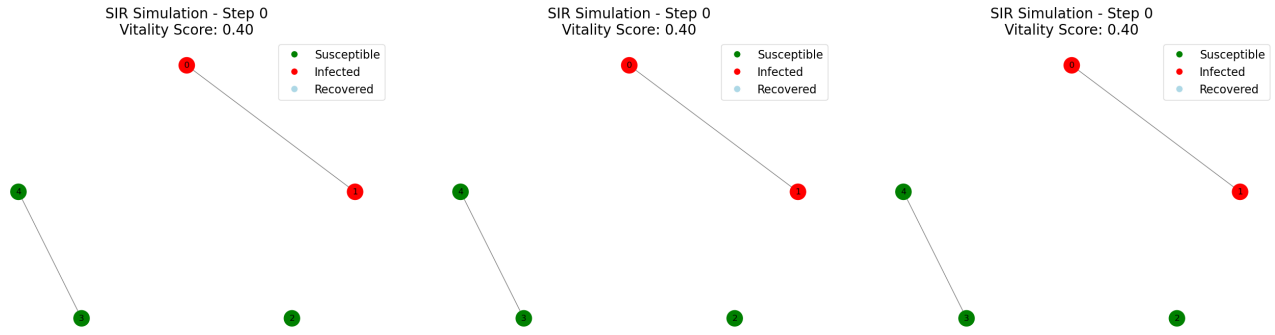
$$R_{max}(i) = \max_{t \in T_i} \frac{R_i^t}{N} \tag{2.2}$$

$$R_{mean}(i) = \frac{1}{m} \sum_{t \in T_i} \frac{R_i^t}{N} \tag{2.3}$$

$$R_{norm}(i) = \frac{1}{m} \sum_{t \in T_i} \frac{R_i^t}{T - t + 1} \tag{2.4}$$

where N is the total number of nodes in the network and T is the number of snapshots (number of time steps in the temporal network). The R_{max} and R_{mean} values correspond respectively to the maximum and average spreading capacity of node i, while the R_{norm} score corresponds to a temporally weighted spreading capacity, giving more importance to later time steps. Therefore, in the context of networks, the SIR score of a node usually refers to how influential that node is in spreading the contagion when it is selected as the initially infected node.

Here is an example for the temporal network from Figure 1.2, with $\beta = 0.3$, $\gamma = 0.1$, and node 1 as the initial infected node.



Run 1 Run 2 Run 3

Figure 2.3: Animated Temporal SIR spreading processes, $\beta = 0.3$, $\gamma = 0.1$, and node 1 as initial infected node.

The vitality scores corresponding to Figure 2.3 are: 0.4, 0.4 and 0.6. If we do 100 runs, we obtain the following vitality score: $R_{max} = 0.40$, $R_{mean} = 0.36$, $R_{norm} = 0.46$.

There are other variants of the SIR model, like the Susceptible-Exposed-Infected-Recovered (SEIR) which adds the Exposed state, which corresponds to a latency period during which individuals have been infected but are not yet infectious themselves. There is also the Susceptible-Infected-Recovered-Vaccinated (SIRV) model, which takes into account the possibility of vaccination of the susceptible population. In the literature, the SIR simulation is widely adopted for evaluating node influence [1, 29, 31, 32, 33], which motivated my decision to use it among other alternatives. I implemented both static and temporal versions of the SIR model, representing the temporal network as a sequence of `NetworkX` graphs. To efficiently run thousands of simulations, I used the `multiprocessing` library, enabling parallel execution across multiple CPU cores.

Finally, these scores can be used as ground truth for training machine learning models that predict vital nodes, as well as for evaluating the performance of ranking algorithms.

2.2 Vital nodes identification (VNI) in static networks

Vital node identification refers to the process of determining the most important or influential nodes in a network, whose activation significantly impacts the network's functionality or dynamics [20]. In the context of spreading processes, such as information diffusion or epidemic outbreaks, vital nodes are those that can influence or reach a large portion of the network. Identifying these nodes is essential for applications like epidemic control, targeted marketing, and network resilience.

While SIR simulations provide a realistic way to estimate node influence, they are computationally expensive, especially for large-scale or long temporal networks [35]. Their outcomes are also sensitive to parameters (infection rate β and recovery rate γ) and require multiple runs per node to reduce variance and obtain reliable scores. Nevertheless, they remain the closest approximation to real spreading dynamics and are widely used as ground truth for training and evaluating algorithms aimed at ranking node influence [1, 3, 6, 29, 31, 33, 36].

2.2.1 Centrality measures

To overcome the computational cost of repeated simulations, many approaches instead rely on structural properties of the network to approximate node vitality. Among these, static centrality measures provide an efficient and widely adopted set of metrics to estimate the influence or vitality of nodes based on their position and connectivity in the network. They can help determine how much a node can control the flow of information, spread information, and quickly reach all other nodes, among other capabilities. In this sense, nodes with high centrality scores are in general considered vital.

A lot of different centrality measures have been created, they go from very simple things like the degree of a node to more complex algorithms like PageRank [23]. All the static centrality measures presented here will be used later in my work (Chapter 3).

The **Degree Centrality (DC)** of node i is defined as:

$$DC(i) = \text{number of neighbors of node } i \tag{2.5}$$

This measure captures how many immediate connections a node has, indicating its direct local influence. In the context of epidemics, nodes with high DC represent individuals who can quickly spread the disease to many others.

The **Closeness Centrality (CC)** [9] of node i is defined as:

$$CC(i) = \frac{1}{\sum_{j \in V \setminus \{i\}} d_{ij}} \tag{2.6}$$

where V is the set of nodes, and d_{ij} is the shortest distance between node i and node j. This measures how close a node is to all other nodes in terms of shortest paths, reflecting how quickly it can reach the rest of the network.

The **Betweenness Centrality (BC)** [4] of node i is defined as:

$$BC(i) = \sum_{j \neq i \neq k} \frac{\sigma_{jk}(i)}{\sigma_{jk}} \tag{2.7}$$

where σ_{jk} is the total number of shortest paths from node j to node k, and $\sigma_{jk}(i)$ is the number of those paths passing through node i. Nodes with high BC values can be seen as key spreaders in an epidemic, acting as intermediaries between distant groups of individuals. The more a node lies on the critical infection pathways, the more control it has over the potential spread of the disease across the network.

The **H-index (H)** [13] of node i is defined as:

$$H(i) = \max\{h \in \mathbb{N} \mid \text{at least h neighbors of i have degree} \geq h\} \tag{2.8}$$

This measures the quality of a node's neighborhood by looking at the degrees of its neighbors. It's like evaluating a researcher's impact not just by the number of papers they have published, but also by how many citations those papers have received. In the context of epidemics, a node with a high H-index is connected to other well-connected nodes, meaning it has the potential to contribute to a wider spread of infection.

The **Local-H-index (LH)** [19] of node i is defined as:

$$LH(i) = H(i) + \sum_{j \in \mathcal{N}_i} H(j) \tag{2.9}$$

where $\mathcal{N}_i$ is the neighborhood of node i. This extends the H-index by aggregating it over a node and its immediate neighbors. This is like considering not only a researcher's impact but also the collective impact of their close collaborators.

The **Core Number (c)** [2] of node i is defined as:

$$c(i) = \max \{k \in \mathbb{N} \mid i \in \text{k-core of the graph}\} \tag{2.10}$$

where the k-core of a graph is the maximal subgraph in which every node has degree at least k within the subgraph. The Core Number measures the largest k-core to which the node belongs, indicating its position in the network's dense core structure. This means it's part of a tightly knit community of nodes that all maintain multiple connections within the same group. In the case of epidemics, a node with a high core number can act as a powerful spreader, since its infection is more likely to reach and persist within the densely connected core of the network.

The **Extended Core Number (ec)** [1] of node i is defined as:

$$ec(i) = c(i) * DC(i) + \sum_{j \in \mathcal{N}_i} c(j) * DC(j) \tag{2.11}$$

This measure considers both the node's centrality within dense communities and the connectivity and core strength of its neighbors.

The **Node Itself and Neighbor Layer information (NINL)** [36] measure of node i is defined as:

$$\text{NINL}^{(0)}(i) = DC(i) + \sum_{\substack{j \in V \setminus \{i\} \\ d_{ij} \leq L}} DC(j) \tag{2.12}$$

$$\text{NINL}^{(k)}(i) = \sum_{\substack{j \in V \setminus \{i\} \\ d_{ij} \leq 1}} \text{NINL}^{(k-1)}(j) \tag{2.13}$$

where L has been set to the average shortest path length of the graph. In my implementation, I took $k = 3$. This recursively aggregates degree information across neighborhoods up to a certain layer, reflecting multi-hop influence. Intuitively, in an epidemic process, a node with a high NINL value can act as an effective spreader, since its immediate and multi-hop neighborhoods are rich in connections, allowing the infection to propagate rapidly through successive layers of the network.

The **Improved gravity centrality (IGC)** [28] measure of node i is defined as:

$$\text{IGC}(i) = \sum_{\substack{j \in V \setminus \{i\} \\ d_{ij} \leq 3}} \frac{c(i) \cdot DC(j)}{d_{ij}^2} \tag{2.14}$$

$$\text{IGC}_+(i) = \sum_{j \in \mathcal{N}_i} IGC(j) \tag{2.15}$$

This simulates gravitational pull where nodes attract each other based on core centrality and degree, decaying with distance, capturing medium-range influence. Intuitively, in the context of epidemics, a node with a high IGC value can act as an effective spreader, since it is likely part of a dense community and close to nodes with many neighbors, enabling the virus to propagate efficiently.

The **Page Rank (PR)** [23] centrality measure of node i is defined as:

$$PR(i)^l = \sum_{j=1}^{N} \left(a_{ij} \frac{PR(j)^{l-1}}{DC(j)} \right) \tag{2.16}$$

where a_{ij} represents the connection between nodes i and j. $PR(i)^l$ is the PR value of node i at step l. The initial value is set uniformly as $PR(i)^0 = \frac{1}{N}$, where N is the number of nodes. After several iterations, the PR value gradually converges and becomes stable. I use $PR(i)$ to represent the final PR value of node i. We can imagine it as a voting system where links to a page are like votes. The more votes a page gets, especially from other important pages, the more important it becomes.

The **Extended Cluster Coefficient Ranking Measure (ECRM)** [34] of node i is defined as:

$$ECRM(i) = \sum_{j \in \mathcal{N}_i} CRM(j) \tag{2.17}$$

where $CRM(j)$ is the Cluster Ranking Measure for each neighbor j of node i, and it is computed as:

$$CRM(i) = \sum_{j \in \mathcal{N}_i} SCC(j) \tag{2.18}$$

where the Shell Clustering Coefficient (SCC) of node i is defined as:

$$SCC(i) = \sum_{j \in \mathcal{N}_i} \left((2 - C_{i,j}) + \left(2 \frac{DC(j)}{\max(DC) + 1} \right) \right) \tag{2.19}$$

where $C_{i,j}$ is the Pearson's correlation coefficient between the shell vectors of nodes i and j, and $\max(DC)$ is the maximum degree in the graph. The shell vector of node i is defined as:

$$SV_i = \left\{ \left| \mathcal{N}_i^{(1)} \right|, \left| \mathcal{N}_i^{(2)} \right|, \ldots, \left| \mathcal{N}_i^{(f)} \right| \right\}$$

where $\left| \mathcal{N}_i^{(k)} \right|$ is the number of neighbors of node i that have a core number equal to k, and f is the maximum core number in the graph. In epidemiology, a node with a high ECRM value is a person who, along with their similarly well-connected neighbors, can efficiently spread an infection through the network.

In the case of information spreading, like in epidemiology, some of these centrality measures can be criticized. For example, if we consider an epidemic spreading and suppose individual A is infected and can reach individual D. The shortest path could be $A \to B \to D$. Classical centrality measures would assume that the infection travels only through this path. However, in reality, B might not get infected (this person could be vaccinated), while the disease could still reach D via a longer path such as $A \to C \to E \to D$. In that case, individuals C and E are crucial for the propagation, but their role would be overlooked by shortest-path–based centrality measures like CC, BC, IGC (eq. 2.6, 2.7, 2.14). These measures can therefore be criticized in the case of epidemiology.

Moreover, some of these centrality measures offer advantages in terms of privacy: the level of access of the network required for their computation. In Table 2.1, I list each centrality measure along with an indication of whether it is a local or global index and specify the level of network access required for its computation.

Centrality measure	local index	global index	level of network access required
DC (eq. 2.5)	✓		1-hop
CC (eq. 2.6)		✓	full network
BC (eq. 2.7)		✓	full network
H (eq. 2.8)	✓		2-hop
LH (eq. 2.9)	✓		3-hop
c (eq. 2.10)		✓	full network
ec (eq. 2.11)		✓	full network
$NINL$ (eq. 2.13)	✓		3-hop
IGC (eq. 2.14)		✓	full network
IGC_+ (eq. 2.15)		✓	full network
PR (eq. 2.16)		✓	full network
$ECRM$ (eq. 2.17)		✓	full network

Table 2.1: Centrality measures privacy-preserving information

Therefore, the local indices, and in particular the degree centrality is the measure that preserve more of the privacy. Indeed, it requires only knowledge of the number of neighbors a node has, without access to the broader network structure.

Several of these centrality measures, such as degree, closeness, betweenness, core number, and PageRank are already implemented in the `NetworkX` library. Other measures, including H-index, NINL, ECRM, and IGC, were implemented in Python, as they build upon the existing centralities or rely on shortest path calculations, which are also supported by `NetworkX`.

2.2.2 Machine learning algorithms

Although these heuristic approaches have achieved good performance in practice, they exhibit weak adaptability, and their effectiveness is often limited to specific settings and certain dynamics [1]. To address these limitations, researchers have turned to machine learning techniques. This section will review various approaches, which have shown promise in overcoming the constraints of traditional methods.

In the work by [1], the authors propose a supervised learning framework for predicting node vitality using only a small subset of the network (0.5% of the nodes) for training, with the remaining nodes used for evaluation. Ground-truth vitality scores are obtained by simulating the SIR epidemic model. Each node is represented through hand-crafted features capturing its structural role, including connectivity, degree (eq. 2.5) and extended core number (eq. 2.11). Several machine learning models are then trained on these features, with the best results achieved by a Support Vector Regression model with a Radial Basis Function (RBF) kernel. To ensure that the training subset is structurally diverse, the authors introduce a cluster sampling method based on K-means clustering of node features, followed by uniform sampling within each cluster.

Building upon this idea of learning from node representations, the study in [29] adopts an autoencoder-based approach to generate latent features that reflect the underlying network structure. The encoder is implemented as a two-layer Graph Convolutional Network (GCN), while the decoder is a Multi-Layer Perceptron (MLP). The autoencoder is trained to reconstruct node degrees, encouraging it to capture local topological information. These latent embeddings are then used as input to another GNN-based model, which assigns vitality scores to nodes. This model is trained using SIR-based ground-truth scores and optimized with the ListMLE loss function (subsection 3.5.1), which emphasizes the correct ranking of nodes. The training and validation are performed on synthetic (Barabási-Albert) and real-world networks, respectively, with a wide variety of real-world graphs used for testing.

A related approach is proposed in [33], where the focus is on leveraging local structural patterns via feature matrices designed to encode each node's neighborhood information, including both direct connections and the degrees of adjacent nodes. These matrices are processed through a Convolutional Neural Network (CNN) architecture composed of two convolutional layers, two pooling layers, and a final fully connected layer. The model is trained using the Mean Squared Error (MSE) loss to predict vitality scores derived from SIR simulations. Similar to previous works, the authors use synthetic Barabási-Albert networks for training and evaluate performance on a range of real-world networks to demonstrate generalization, even if it performs best when the structure of the training and testing sets are similar.

Compared to classical centrality measures, these learning based approaches aim to capture more complex relationships between network structure and node influence. They can adjust to various network topologies by utilizing features that encode structural roles, latent embeddings, or local neighborhood patterns. This method helps them identify vital nodes that might not stand out using standard centrality measures. These nodes gain their influence from small but meaningful structural details, which are often missed by traditional approaches. In many cases, these learning-based methods achieve better performance in identifying vital nodes than static centrality measures. Despite these advantages, several challenges remain. Ensuring that the models generalize well to networks with very different structures is still difficult, and addressing this is key to developing reliable and scalable approaches for vital node

identification. Additionally, comparing different machine learning methods is often challenging, as evaluation strategies and datasets vary widely across studies.

2.3 Vital nodes identification (VNI) in temporal networks

Temporal networks keep track of both the timing and sequence of interactions, avoiding "false" connections that can appear when data is aggregated over time. This temporal detail is important for dynamic processes such as contagion or information spread, where cause-effect relationships and irregular activity patterns influence who can actually connect with whom. For example, in epidemiology, a person may interact with colleagues during the day at work and with family or friends in the evening. Aggregating all contacts into a static network would suggest that all connections are simultaneously active, but in reality, the timing of contacts determines the actual pathways of disease transmission. As a result, identifying vital nodes requires methods that account for temporal information rather than relying solely on static representations [3, 32].

Building on the methods used in static networks, we now extend our discussion to the identification of vital nodes in temporal networks. The study of identifying important nodes in temporal networks is considerably more challenging than in static ones. In temporal networks, a node may play different roles at different times, meaning its importance can vary over time. For instance, an individual who was highly active and influential last year may become inactive this year and thus irrelevant for information spreading initiated now. Consequently, effective identification of vital nodes in temporal settings must take into account both the structural properties of the network and the timing of interactions [3].

2.3.1 Temporal Centrality Measures

Centrality measures were first developed for static graphs, and several have since been adapted to temporal networks. Let $G = (G_1, \ldots, G_T)$ be our dynamic graph, following [15], we can extend some static centrality measures to temporal networks.

The **Temporal Degree Centrality (TDC)** [15] is defined as:

$$TDC(i) = \frac{1}{(N-1)T} \sum_{t=1}^{T} DC(i)_t \tag{2.20}$$

where $DC(i)_t$ is the degree centrality of node i in snapshot G_t. It measures how frequently a node interacts over time by averaging its number of connections across all time steps. A high TDC indicates a node that is consistently active.

The **Temporal Closeness Centrality (TCC)** [15] is defined as:

$$TCC(i) = \frac{1}{(N-1)T} \sum_{t=1}^{T} \sum_{j \in V \setminus \{i\}} \frac{1}{d_{ij}(t,T)} \tag{2.21}$$

where in this temporal setting, $d_{ij}(t,T)$ is the temporal shortest distance from node i to node j in the time interval $[t,T]$. It reflects how quickly a node can reach others over time, considering the fastest time-respecting paths. Nodes with high TCC can rapidly spread information throughout the network.

The **Temporal Betweenness Centrality (TBC)** [15] is defined as:

$$TBC(i) = \sum_{t=1}^{T} \sum_{\substack{i \neq j \neq k \in V \\ \sigma_{t,T}(j,k) > 0}} \frac{\sigma_{t,T}(j,k,i)}{\sigma_{t,T}(j,k)} \tag{2.22}$$

where $\sigma_{t,T}(j,k)$ is the number of time-respecting temporal shortest paths from node j to node k, $\sigma_{t,T}(j,k,i)$ is the number of those paths that pass through node i. It captures how often a node acts

as a bridge on time-respecting shortest paths between other nodes. A high TBC means the node plays a key role in controlling or facilitating temporal flow in the network.

These temporal centrality measures directly extend the classical definitions of degree, closeness, and betweenness from static to temporal networks by incorporating time-respecting paths and snapshot-based calculations. However, they essentially compute an average of the static centrality values across all snapshots. While this captures overall activity or influence trends, it also smooths out temporal variations and may overlook transient but critical roles a node plays at specific times. For example, a node that becomes highly central during a short but decisive time window might receive a modest score due to averaging. More refined approaches could weight centrality values by process-specific importance (e.g., peak spreading periods), incorporate temporal decay to emphasize recent activity, or model the evolution of centrality as a time series rather than a single aggregated score. Such enhancements would better capture the dynamic nature of node importance in temporal networks.

2.3.2 Baseline models

Since my focus is on identifying vital nodes in temporal networks, I looked for baseline algorithms that address this problem, with particular attention to those using the same type of ground-truth scores to enable fair comparison of results. The following two baseline models are both using the same three temporal SIR scores (eq. 2.2, 2.3 and 2.4) to evaluate (and train) their model.

Temporal Gravity (TG) model

In this part, I'll explain the Temporal Gravity (TG) model [3], which proposes a measure based on temporal shortest distances and other static centrality measures. This model is an analogy with physics and Newton's law of gravitation. The **Temporal Gravity (TG)** score is defined as:

$$TG(i) = \sum_{\substack{d_{ij} \leq R \\ i \neq j}} \frac{M_i M_j}{d_{ij}^2} \tag{2.23}$$

where M_i, M_j are node properties, d_{ij} is the temporal shortest distance between nodes i and j.

For the node property, baseline centrality measures are used: PR^s, DC^s, CC^s, BC^s, PR^m, DC^m, CC^m, BC^m, TD (eq. 2.24).
measure^m: means that the average centrality has been computed over the snapshots.
measure^s: means that the centrality has been computed on the aggregated network.

The aggregated network of a temporal network is a static graph formed by combining all temporal interactions over time. An edge exists between two nodes if they interacted at any point during the observation period (Figure 2.4).

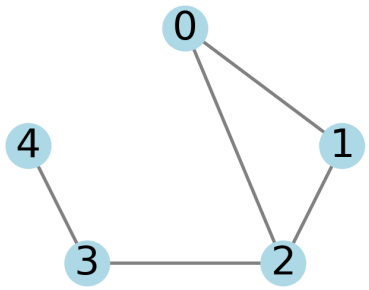


Figure 2.4: Aggregated network associated with the temporal network of Figure 1.2

They also define their own **Time degree (TD)** centrality measure:

$$TD(i) = \sum_{t=1}^{T} e^{DC(i)_t} \tag{2.24}$$

where $DC(i)_t$ is the degree centrality of node i at snapshot t. This formulation of time degree emphasizes time steps where the node has high connectivity, giving more weight to moments of high activity.

The Temporal Gravity (TG) model improves on traditional temporal centrality measures by combining temporal shortest distances with static node properties. Unlike temporal closeness or betweenness, which averages influence over time and may miss short-lived but critical roles, the TG model weights interactions by node importance, which enables better capturing dynamic influence. Its strength lies in contextualizing node relevance within the network's temporal structure. However, the model still relies on static centrality, potentially overlooking nodes that are only vital during specific moments. It also demands higher computational resources and careful parameter tuning (choice of R and node property), which can limit its scalability for large or high-frequency networks.

NFI-SGAT - Node Feature Initialization-based Streaming graph learning model with Graph Attention network

The proposed method, Node Feature Initialization-based Streaming graph learning model with Graph Attention network, NFI-SGAT [31] (Figure 2.5), starts by generating initial node features that capture the topology for the first time steps of the network. This is done using multi-scale feature vectors that include degree centrality, h-index, and k-shell metrics. Then the temporal feature vectors are weighted by the snapshot's weight, which contains the number of active nodes and the number of edges of the snapshot. This leads to a unique feature vector for each node. Once the node features are initialized, the model continuously updates node representations using a temporal streaming graph learning module. This module employs a time-aware Long Short-Term Memory (LSTM) and an attention mechanism to address the staleness problem of node memory. This mechanism aggregates information from the temporal neighbors of nodes, ensuring that the node representations remain up-to-date and relevant. The datasets are divided into three node subsets: training (80 %), validation (10 %), and testing (10 %). The training set is further partitioned into two portions based on a ratio ξ, which determines how much of the training data is used for node feature initialization versus dynamic representation learning. The authors conduct experiments on eight real-world temporal networks.

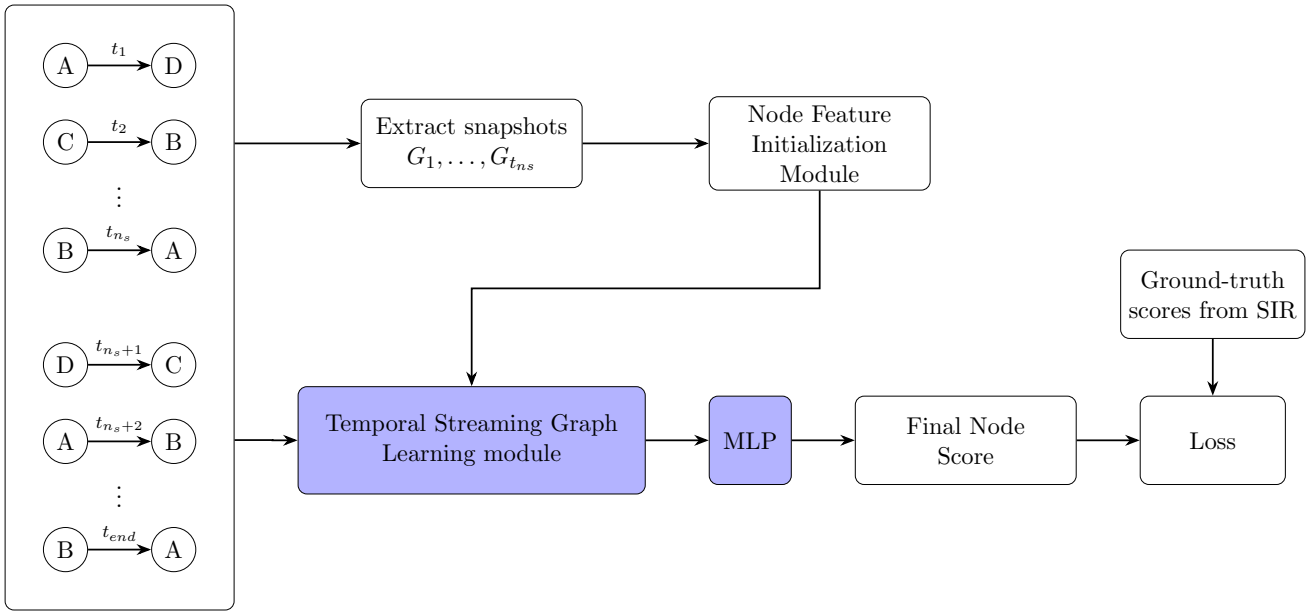


Figure 2.5: Overview of the NFI-SGAT architecture: learning module in blue.

In contrast to the Temporal Gravity (TG) model, which offers a simple and interpretable framework by combining temporal shortest distances with static node properties, NFI-SGAT prioritizes flexibility and adaptability by learning rich, time-aware node representations using LSTM and attention mechanisms, at the cost of interpretability. Both methods face broader challenges, including scalability, reducing label dependency, and improving cross-network generalization.

2.4 Temporal graph learning

Temporal graph learning aims to capture not just who's connected to whom, but when and how often these interactions occur, enabling predictions about future events (like new connections or node behavior), understanding evolving dynamics, or forecasting time-varying patterns. Models often employ structures like memory modules, time encoders, and recurrent or attention mechanisms to learn from sequences of timestamped edges, balancing both structural and temporal context.

In [30], the authors introduce the Temporal Graph Attention (TGAT) layer for representation learning on temporal graphs. This model aggregates both temporal and structural information by leveraging a self-attention mechanism that captures interactions between node features and time. To effectively encode continuous-time information, the authors use functional time encoding based on Bochner's theorem, which maps timestamps into a continuous domain. Each node is represented using its feature vector along with the encoded time of its interactions, allowing TGAT to produce time-dependent embeddings that evolve with the graph. The model is trained with a time-sensitive link prediction loss, supported by negative sampling to improve robustness. A chronological train-validation-test split (75%-15%-15%) ensures that training data reflect the temporal and structural diversity of the graph. TGAT is designed for both transductive and inductive scenarios, making it capable of generalizing to unseen nodes during inference.

Building on similar motivations, the Temporal Graph Networks (TGN) framework introduced in [24] presents another approach for modeling dynamic graphs as sequences of timestamped events. Unlike TGAT, TGN explicitly maintains a memory for each node, which stores a compressed history of its interactions. This memory is updated over time using a series of modules: a Message Function encodes interaction events, a Message Aggregator combines multiple messages, and a Memory Updater refreshes the node's memory state. An Embedding module then generates node representations based on the updated memory, helping mitigate the staleness of information. Like TGAT, TGN is trained with a chronological split and evaluated on the future edge prediction task using Binary Cross-Entropy (BCE) loss. TGN demonstrates strong performance in both transductive and inductive settings and is particularly effective at capturing long-term temporal dependencies through its memory-based architecture.

In both cases, the main goal is to obtain a temporal representation of the nodes. For vital node identification, several concepts can be reused, such as continuous-time encoding, self-attention mechanisms, and memory modules.

2.5 Related problems

In network science, several problems focus on identifying important, critical, or influential nodes, each with distinct objectives and methodologies.

One widely studied problem is the Influence Maximization problem (IM) [18, 20], a combinatorial optimization task that involves selecting a subset of nodes, known as the seed set, to maximize the spread of information, behaviors, or diseases across a network. This problem is typically modeled using diffusion models such as Independent Cascade (IC) or SIR. Influence Maximization is typically used for offensive strategies like viral marketing. For example, it helps answer the question: Which group of social media influencers should we target to promote a product and reach the largest possible audience?

Source detection [25] is a related but inverse problem. Given partial observations of infected nodes in a network, the objective is to infer the initial source(s) of the spread. This inverse problem is particularly significant in epidemiology and cybersecurity, where identifying the source of an outbreak can be crucial for understanding its spread and implementing effective control measures.

An alternative approach to Influence Maximization is the Critical Node Detection Problem (CNDP) [16]. In this problem, the goal is to identify a subset of nodes whose removal causes the greatest degradation in network connectivity according to a predefined metric, such as minimizing the number of connected node pairs. CNDP is particularly useful for defensive strategies, including network disruption, epidemic containment, and improving infrastructure resilience.

In contrast to the other problems, the Vital Node Identification (VNI) [1, 3, 29, 31, 32, 33] problem focuses on evaluating nodes individually rather than selecting an optimal subset. The goal is to assign a score to each node based on its potential impact on spreading, for example, the expected number of infections it would cause under SIR dynamics. VNI is a ranking or scoring task. These influence scores can then be used in downstream applications, such as targeted immunization or prioritizing nodes in defense strategies aimed at minimizing the risk of cascading failures.

3 Methodology

In this chapter, I introduce the datasets used in my study and describe the evaluation metrics applied to measure model performance. I then present the design of the different architectures developed, followed by the training and evaluation strategies used to validate their effectiveness.

3.1 Datasets

- Highschool (HS2011, HS2012, HS2013) [8, 21]: contacts between students in a high school in Marseille, France.
  During 4 days (Tuesday to Friday) in December 2011.
  During 9 days (Monday to the Tuesday of the following week) in November 2012.
  During 5 days in December 2013.
- Hospital-Contact (HC) [26]: contacts between patients and health-care workers (HWCs) in a hospital ward in Lyon, France, from Monday, December 6, at 1:00 pm to Friday, December 10, at 2:00 pm (2010). The study included 46 HCWs and 29 patients.
- Hypertext-2009 (HT2009) [14]: contacts between the attendees of the ACM Hypertext 2009 conference (2.5 days).
- Primaryschool (PS) [10]: contacts between children and teachers in a primary school in France, and covers two days of school activity (Thursday, October 1st and Friday, October 2nd, 2009). Here is a video to see the evolution of contacts between the children depending on their class.
- SFHH conference (SFHH) [11]: contacts between participants at the SFHH conference in Nice, France (June 4-5, 2009).
- Workplace (WP) [12]: contacts between individuals measured in an office building in France, from June 24 to July 3, 2013.
- Agent-based-models contacts (ABM) [7]: synthetic network, which have been generated through Bayesian models from the WP, PS, and two other networks.

Table 3.1 provides a summary and comparison of key characteristics across all the datasets used in this study. The networks in this study are relatively small, typically containing around a hundred nodes and time steps. They also vary considerably in terms of the number of interactions recorded, which can help to explain why some models perform better on certain datasets compared to others.

Across all datasets, there is a clear contrast between the volume of contacts during daytime and nighttime hours. Additionally, in some datasets, the impact of weekends is also evident, with a noticeable drop in interactions (Figure 3.1). These differences in interaction patterns are important to keep in mind when analyzing spreading dynamics.

Then, I used my implementation of the temporal SIR model and ran it on the different datasets with the following parameters: `n_simulations` $= 100$, $\beta = 0.1$, $\gamma = 0.01$, as in the two baseline papers studied [3, 31]. Through the simulations, I obtained the corresponding ground truth score distributions (Figures 3.2 and A.1 in the appendix). Each histogram reports the frequency of nodes for a given score, with mean and median indicated. The score ranges and distribution shapes vary considerably across datasets and score types. For example, the R_{mean} score spans roughly $[0, 0.25]$ for SFHH but $[0.5, 0.75]$ for Primaryschool, while the R_{max} score shows an almost uniform spread between 0 and 1 for SFHH but peaks sharply at 1 for Primaryschool. These distributions are important, as they serve both as ground truth for the machine learning models and for computing evaluation metrics.

As mentioned earlier, the temporal SIR model can be initiated at any given time step in the network's evolution. We observe that starting the simulation at a later time step generally leads to a lower average

Dataset name	N	T_{span}	Time resolution	T	C	E
HS2011	126	3 days	1 hour	76	28 561	1 709
HS2012	180	8.5 days	1 hour	203	45 047	2 220
HS2013	327	4 days	1 hour	101	188 508	5 818
HC	75	4 days	1 hour	97	32 424	1 139
HT2009	113	2.5 days	0.5 hour	118	20 818	2 196
PS	242	1.5 days	0.5 hour	65	125 773	8 317
SFHH	403	1.5 days	0.5 hour	64	70 261	9 565
WP	92	11 days	1 hour	275	9 827	755
ABM	2058	10 days	1 hour	240	1 975 554	67 778

Table 3.1: Datasets information. N: Number of nodes, T_{span}: time between the first and last contact, Time resolution: duration of each time interval used to generate the snapshots, T: number of snapshots, C: total number of contacts, E: number of unique contacts. These statistics highlight important differences in scale and density between datasets, which can influence how models perform.

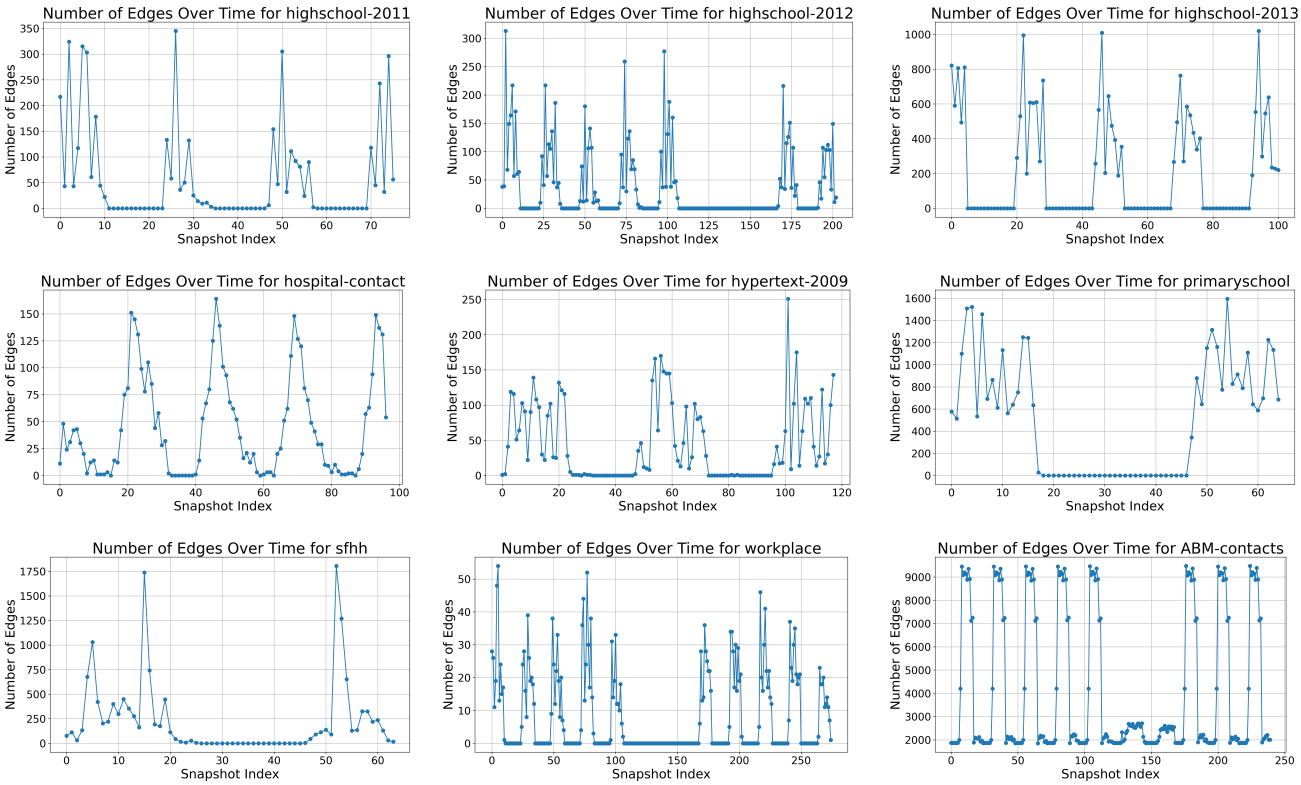


Figure 3.1: Evolution of the number of edges over the snapshots for different datasets. These temporal variations are crucial, as they reflect the underlying social rhythms that models must account for.

number of infected nodes. This outcome is expected since the process has less time to spread through the network (Figure 3.3).

The ABM network can be downloaded from the following link: https://zenodo.org/records/15076221. The specific network I used is the file named `TCN1000-medium.csv`. All other networks are publicly accessible on the SocioPatterns website: http://www.sociopatterns.org/datasets/.

3.2 Metrics

With a clear understanding of the datasets and their characteristics, we now move on to the metrics used to evaluate the performance of our models. These metrics are essential for assessing how well our models can identify vital nodes within the networks.

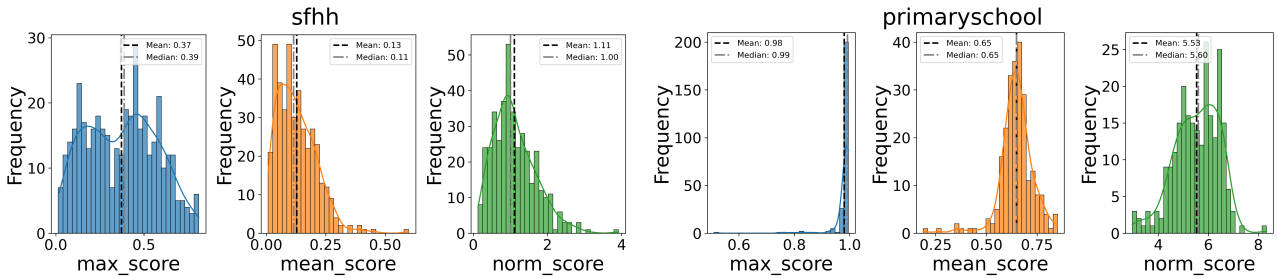


Figure 3.2: Distributions of ground-truth epidemic scores obtained from SIR simulations ($\texttt{n\_simulations} = 100,\ \beta = 0.1,\ \gamma = 0.01$). The two examples illustrate how score ranges and shapes can differ widely across datasets, which can directly affect model training and evaluation.

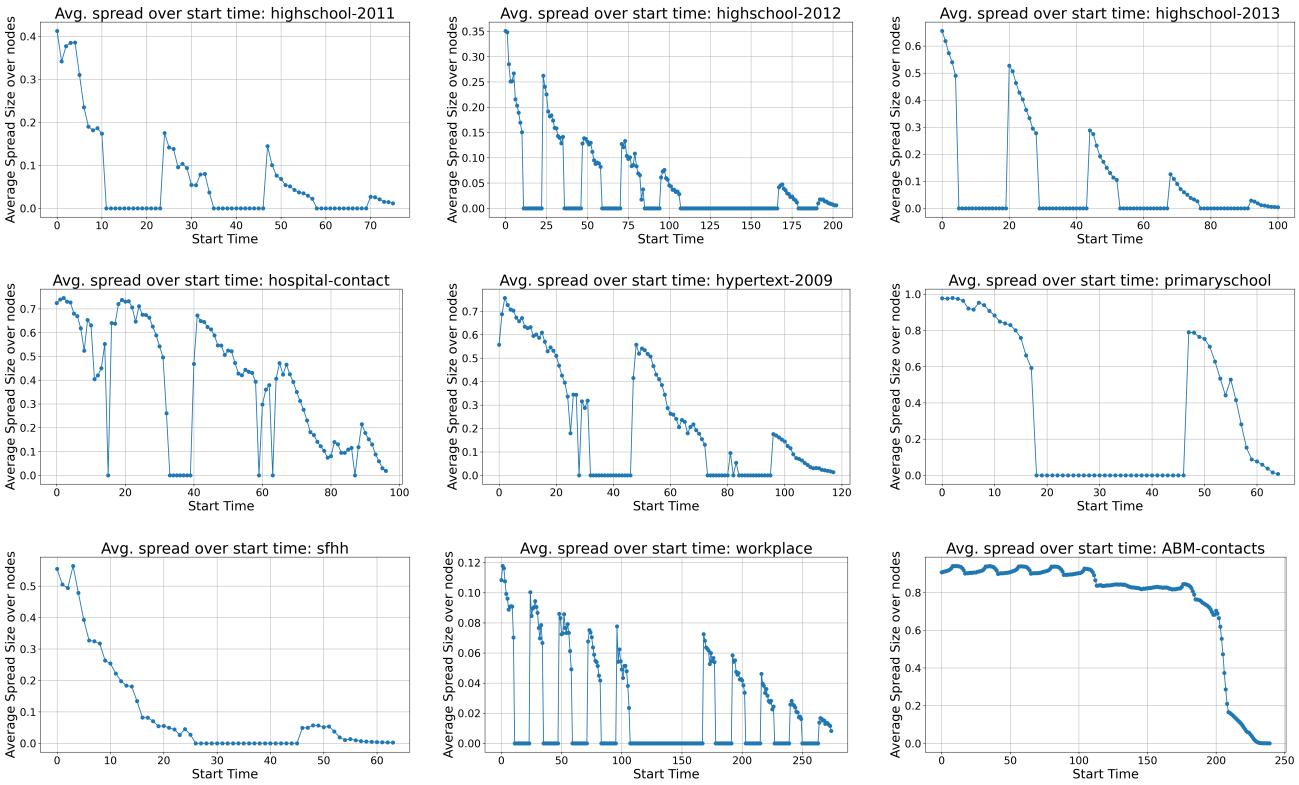


Figure 3.3: Evolution of the average proportion of nodes reached across different starting snapshots through SIR ($\texttt{n\_simulations} = 100,\ \beta = 0.1,\ \gamma = 0.01$). The general trend indicates that starting the infection later in the timeline leaves less opportunity for the epidemic to spread, resulting in smaller outbreaks.

3.2.1 Kendall's Tau

Kendall's Tau is the metric that is the most used by already existing methods to evaluate their models [1, 3, 29, 33, 31]. Let S be the ground truth scores and $\hat{S}$ the output scores of the VNI model, a pair of nodes (i, j) is said to be concordant if $s_i > s_j$ and $\hat{s}_i > \hat{s}_j$ or $s_i < s_j$ and $\hat{s}_i < \hat{s}_j$, otherwise the pair is said to be discordant (see green and red pairs in Figure 3.4).

$$\tau(S, \hat{S}) = \frac{\text{number of concordant pairs} - \text{number of discordant pairs}}{\text{total number of pairs}} \in [-1, 1] \tag{3.1}$$

$$= \frac{2}{n(n-1)} \sum_{i<j} \text{sign}(s_i - s_j)\text{sign}(\hat{s}_i - \hat{s}_j) \tag{3.2}$$

The higher the value of Kendall's tau, the closer the ranking of scores predicted by the VNI model is to

the ranking derived from ground truth scores. When $\tau = 1$, it means that the two rankings are exactly the same, and when $\tau = -1$, it means that one ranking is the reverse of the other one.
For example, if we study the following two scores:

$$S = (0.01, 0.05, 0.31, 0.32, 0.50)$$
$$\hat{S} = (0.04, 0.03, 0.32, 0.31, 0.48)$$

then $\tau(S, \hat{S}) = 0.6$. It shows that the Kendall's tau metric is highly sensible to small changes in the score values when it changes the ranking order of some node pairs (Figure 3.4).

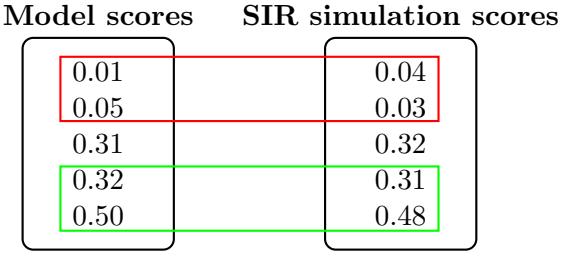


Figure 3.4: Comparison of node rankings between the model scores and ground-truth SIR simulation scores. A Kendall's Tau coefficient of 0.6 indicates a moderate correlation between the rankings.

In my implementation, I used the already available `kendalltau` function from the `SciPy` library.

3.2.2 Monotonicity index

Monotonicity index (MI) is a metric that takes a value between 0 and 1, and it measures how often a ranking method assigns the same score to different nodes, quantifying the level of tie redundancy in a ranking, higher values indicate fewer tied scores and thus a more informative (strictly ordered) ranking. It is defined as follows:

$$MI(S) = \left(1 - \frac{\sum_{s \in S} N_s \left(N_s - 1\right)}{N(N-1)}\right)^2 \tag{3.3}$$

where S denotes the score list, s denotes a value in the rank list, N_s denotes the number of values taking s in the rank list, and N denotes the number of values in the rank list.

In the two baseline papers, the authors are only reporting the value of the Kendall's tau between the VNI model and the ground truth score from the SIR simulations. This is why I also choose to report Kendall's tau value to compare the results of my models with existing methods. However, the monotonicity index is also a good indicator of ranking quality, as it reflects how well a method differentiates node importance by avoiding tied scores. A high monotonicity index suggests that a model produces a more informative and discriminative ranking, which is crucial when prioritizing nodes for interventions in temporal networks.

3.3 Baseline Methods

Having established the metrics for evaluating my models, we now turn to the baseline methods against which the proposed approaches will be compared. These baseline models have been explained in subsection 2.3.2. In this subsection I detail my work to reproduce the results presented by the authors in their papers.

Implementation of the Temporal Gravity (TG) Model

In this section, I present my reimplementation of the Temporal Gravity (TG) model [3], with the objective of reproducing the results reported by the authors and enabling a direct comparison to their findings. This ensures that the method can serve as a validated baseline for the rest of my study. (The model has already been theoretically presented in subsection 2.3.2).

I began by developing a function to compute the temporal shortest path between pairs of nodes and incorporated the previously described centrality measure implementation. To calculate the Temporal Gravity score as defined in Equation (2.23), I followed the authors' choice of setting $R = 5$, which balances performance and accuracy given the high computational cost of computing temporal shortest paths.

To assess the correctness of my implementation, I compared my results against those reported in the original paper. Table 3.2 and 3.3 show the Kendall's Tau results presented by the authors for the R_{max} score type and the difference to obtain the results from my own implementation under the same hyperparameter setting (`n_simulations` $= 100,\ \beta = 0.1,\ \gamma = 0.01$) in brackets.

Overall, the Kendall's tau correlation scores obtained with my implementation closely match those reported in the original paper across all datasets. The differences, both in terms of maximum absolute difference and mean difference, are reported in Table 3.4. The maximum absolute difference remains below 0.08 for all datasets, and the mean differences are generally very small, often near zero.

	HS2011	HS2012	HS2013	HC
$TG(TD)$	0.66 (-0.03)	0.57 (-0.04)	0.57 (-0.03)	0.61 (+0.05)
$TG(PR^m)$	0.62 (-0.03)	0.61 (-0.02)	0.58 (-0.02)	0.59 (+0.03)
$TG(PR^s)$	0.62 (-0.03)	0.54 (-0.03)	0.55 (-0.03)	0.61 (+0.01)
$TG(DC^m)$	0.70 (-0.02)	0.64 (-0.01)	0.64 (-0.01)	0.61 (=)
$TG(DC^s)$	0.65 (-0.02)	0.56 (-0.04)	0.56 (-0.02)	0.61 (+0.01)
$TG(CC^m)$	0.73 (=)	0.64 (-0.02)	0.60 (-0.01)	0.58 (-0.01)
$TG(CC^s)$	0.65 (-0.04)	0.55 (-0.05)	0.55 (-0.04)	0.61 (+0.03)
$TG(BC^m)$	0.56 (-0.02)	0.56 (-0.03)	0.44 (-0.02)	0.51 (-0.02)
$TG(BC^s)$	0.48 (-0.01)	0.43 (-0.04)	0.43 (-0.02)	0.60 (-0.01)

Table 3.2: Kendall's Tau results for the first four datasets presented in the Temporal Gravity (TG) paper [3] for the $\mathbf{R_{max}}$ score type (in brackets this is the difference to add to obtain the results from my implementation), `n_simulations` $= 100$, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed on all the nodes.

	HT2009	PS	SFHH	WP
$TG(TD)$	0.56 (=)	0.38 (-0.01)	0.48 (+0.08)	0.60 (-0.01)
$TG(PR^m)$	0.57 (-0.01)	0.43 (-0.03)	0.51 (+0.04)	0.62 (+0.04)
$TG(PR^s)$	0.54 (+0.01)	0.38 (-0.02)	0.46 (+0.02)	0.59 (=)
$TG(DC^m)$	0.61 (-0.01)	0.43 (-0.04)	0.51 (+0.02)	0.65 (+0.04)
$TG(DC^s)$	0.54 (+0.01)	0.38(-0.03)	0.46 (+0.01)	0.58 (+0.01)
$TG(CC^m)$	0.61 (=)	0.39 (-0.01)	0.52 (+0.02)	0.67 (+0.03)
$TG(CC^s)$	0.56 (=)	0.38 (-0.02)	0.48 (+0.05)	0.58 (-0.01)
$TG(BC^m)$	0.54 (-0.01)	0.34 (-0.04)	0.44 (=)	0.59 (-0.01)
$TG(BC^s)$	0.54 (=)	0.39 (-0.04)	0.45 (=)	0.53 (-0.01)

Table 3.3: Kendall's Tau results for the last four datasets presented in the Temporal Gravity (TG) paper [3] for the $\mathbf{R_{max}}$ score type (in brackets this is the difference to add to obtain the results from my implementation), `n_simulations` $= 100$, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed on all the nodes.

I also performed the same comparison for the R_{mean} and R_{norm} score variants. The corresponding difference tables are shown in Tables 3.5 and 3.6, while the full Kendall's tau results for each centrality variant are provided in the appendix (Tables A.1, A.2 & A.3, A.4). These additional comparisons further confirm that my implementation behaves consistently with the authors' original methodology. Minor differences can be attributed to factors such as randomness in the SIR simulation process, which is then used as ground truth to compute the Kendall's tau.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
max abs difference	0.04	0.05	0.04	0.05	0.01	0.04	0.08	0.04
mean difference	-0.02	-0.03	-0.02	0.01	0.00	-0.03	0.03	0.01

Table 3.4: Maximum absolute difference and mean difference between the Kendall's tau scores of the paper and from my implementation. $\mathbf{R_{max}}$ score type, `n_simulations` $= 100$, $\beta = 0.1$, and $\gamma = 0.01$. The small values confirm that the reimplementation closely reproduces the original findings.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
max abs difference	0.03	0.02	0.04	0.05	0.01	0.01	0.06	0.02
mean difference	-0.02	0.00	-0.02	0.02	0.00	0.00	0.02	0.00

Table 3.5: Maximum absolute difference and mean difference between the Kendall's tau scores of the paper and from my implementation. $\mathbf{R_{mean}}$ score type, `n_simulations` $= 100$, $\beta = 0.1$, and $\gamma = 0.01$. The small values confirm that the reimplementation closely reproduces the original findings.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
max abs difference	0.06	0.04	0.06	0.04	0.03	0.03	0.03	0.03
mean difference	0.04	0.01	-0.03	0.00	0.01	-0.02	0.00	0.00

Table 3.6: Maximum absolute difference and mean difference between the Kendall's tau scores of the paper and from my implementation. $\mathbf{R_{norm}}$ score type, `n_simulations` $= 100$, $\beta = 0.1$, and $\gamma = 0.01$. The small values confirm that the reimplementation closely reproduces the original findings.

In conclusion, the Temporal Gravity Model has been reimplemented and validated, with only minimal differences from the original results. This establishes it as a reliable baseline for further comparison in my study.

Attempted Implementation of the NFI-SGAT model

For the NFI-SGAT [31] model, I attempted to replicate the approach by implementing the initialization phase and the Time-Aware LSTM. For the Graph Attention Network (GAT) component, I used the `GATConv` class from the `PyTorch Geometric` library. However, I was unable to reproduce the results reported in the original paper. Although the authors mention using training, validation, and test sets, they do not provide details on how these splits are constructed. An omission that is significant, as performance can vary greatly depending on the data partitioning strategy. I reached out to the authors via email to request access to their implementation for a deeper understanding of their methodology, but unfortunately, I did not receive a response. Based on the reported results in the paper for the R_{norm} score type, the authors achieve a Kendall's Tau value above 0.8 for HS2011 and HS2012, 0.64 for HS2013, 0.70 for PS, and 0.90 for HC, which provides a reference point despite the lack of reproducibility.

3.4 Proposed Methods

This section presents the different model architectures that I developed for identifying vital nodes in temporal networks, aiming to capture temporal dynamics and address key limitations of existing approaches, such as the reliance on simple time-averaged centrality measures and limited adaptability of heuristic methods to diverse network structures and spreading dynamics.

The main idea is to take as input a sequence of graphs that represent the temporal evolution of the network. Graph Neural Networks (GNNs) are then employed to learn expressive node embeddings at each time step. These temporal embeddings are subsequently aggregated either through simple

strategies like averaging or more sophisticated temporal modules to produce a unique representation for each node. Finally, a regression module processes these representations to predict node influence scores, trained using regression or ranking losses with the SIR ground truth scores. This global idea is described through Figure 3.5. This idea of doing temporal node representation learning comes from several papers [24, 30, 31].

3.4.1 APool-GAT - Attention Pooling over Graph Attention Networks embeddings

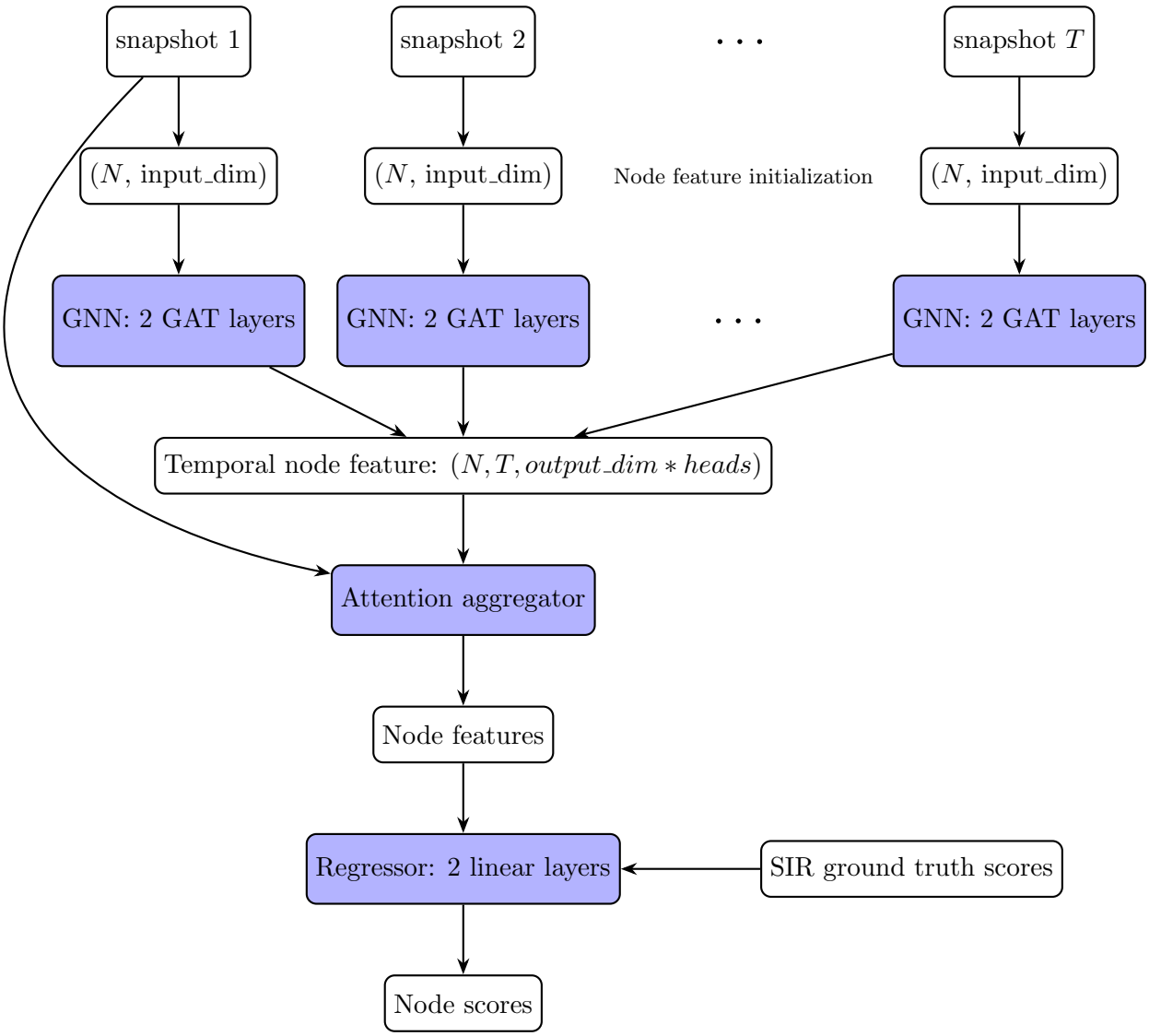


Figure 3.5: Architecture of the proposed APool-GAT model. This illustrates how structural and temporal information are integrated to identify vital nodes.

The goal of the first step is to initialize the node embedding. This is where we get features of shape: $(N, input\_dim)$. To initialize my node features, the main idea is to use static centrality measures for each snapshot. Indeed, these indices are good measures of how vital a node is, as a starting point. This kind of initialization step with static centrality measures is also done in [31].

- `centrality-neighbors`, I used: degree, betweenness, closeness, h-index, local-h-index, core-number, $NINL$, $ECRM$, IGC, IGC_+ (eq. 2.5, 2.7, 2.6, 2.8, 2.9, 2.10, 2.13, 2.17, 2.14, and 2.14). Therefore, in this case $input\_dim = 10 * (\texttt{num\_neighbors} + 1)$.
- `centrality-neighbors-7`, I used: degree, betweenness, closeness, h-index, core-number, $NINL$, IGC, IGC_+. In this case $input\_dim = 7 * (\texttt{num\_neighbors} + 1)$.

- `degree-neighbors`, I used: degree. In this case, $input\_dim = \texttt{num\_neighbors} + 1$.

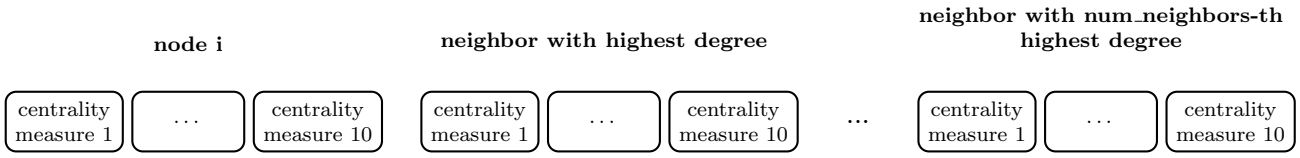


Figure 3.6: Node feature initialization via concatenation of centrality measures from node and neighbors

For even more privacy preservation at the initialization step, I also tried a fourth initialization: `degree-only`. This time I'm only computing the degree of each node as an initial embedding, I don't look anymore at the neighbors, therefore, $input\_dim = 1$.

Then, these nodes' feature vectors are passed through 2 Graph Attention Networks (GAT) layers [27] that learn a kind of weighted average of the node feature over their neighbors. We do this for all the snapshots. The GATs parameters are not shared across the time steps to take into account the fact that the snapshots could be completely different, even if it's adding more computational costs. Then, we obtain nodes' temporal embedding, which has a shape of $(N, T, output\_dim * heads)$. As said before, now the goal is to obtain a unique feature vector for each node that can later be processed by a regressor to learn the ground truth SIR score. To do so, an aggregator will be used, and its objective will be to weight the nodes' embeddings depending on their temporal evolution.

I designed several aggregators, here is one example in Figure 3.7.

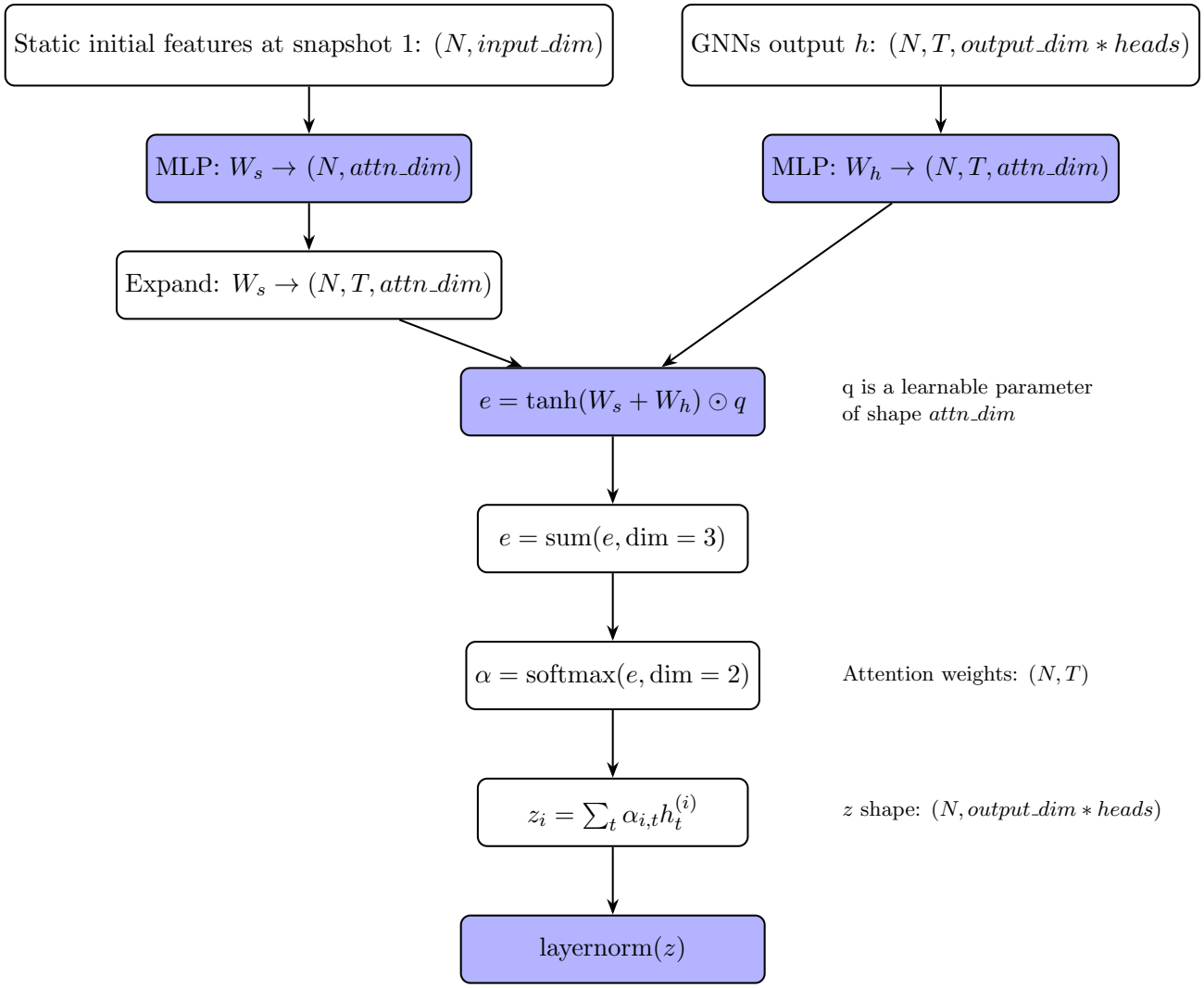


Figure 3.7: Attention aggregator architecture used in the APool-GAT model.

This aggregator is taking as input the temporal node embeddings "on the right" and also the degree of the nodes in the first snapshot "on the left". The static and temporal features are first projected into a shared attention space using two separate Multi-Layer-Perceptrons (MLPs). The static features are then broadcast across the time dimension to align with the temporal embeddings. An attention mechanism then computes importance weights for each time step, allowing the model to focus on the most relevant snapshots for each node. The final representation is obtained by taking a weighted sum of the temporal embeddings using the learned attention weights, followed by a layer normalization. This allows the model to adaptively aggregate temporal information based on both node identity and temporal dynamics.

This temporal attention aggregator overcomes the limitations of naïve pooling methods that treat all time steps equally or rely solely on the most recent snapshot. In temporal networks, a node's importance can vary sharply due to structural changes or local spreading dynamics. By learning attention weights from both static node features and evolving temporal embeddings, the aggregator highlights the most informative periods, filters out irrelevant fluctuations, and captures irregular temporal dependencies, making it more robust to noise, bursts of activity, and dormant phases that static or heuristic methods often overlook.

3.4.2 TAPool-Seq-GAT - Time-aware Attention and Sequential Pooling over Graph Attention Networks embeddings

This section introduces a second architecture, which shares the same initial steps as the previous one: feature initialization and temporal embedding extraction using GAT layers. However, in the previous architecture, we are not really processing the temporal node embeddings as a sequence, as it could be the case. Indeed, the temporal node embeddings are representing the evolution of the nodes' vitality over time, which could be modeled as a sequence.

To model the temporal dependencies between different time steps of the network, I incorporate an LSTM applied to the sequence of temporal node features. I retain the final hidden state of the LSTM for each node, which serves as its aggregated temporal representation. I also use the attention aggregator as in Figure 3.7, but this time, the "left" input is a time-weighted degree for nodes across snapshots. It applies a decay factor to give more importance to recent snapshots while still considering older ones. This reflects the intuition that recent interactions or connections are often more relevant than older ones.

Before doing the regression, I also concatenate to the LSTM and Attention aggregator outputs an average temporal embedding for each node. Averaging provides a summary of the temporal dynamics for each node, which can be useful to understand the overall temporal trend and possibly help the model to make the difference between nodes without connection and nodes with a lot of connection overall (Figure 3.8).

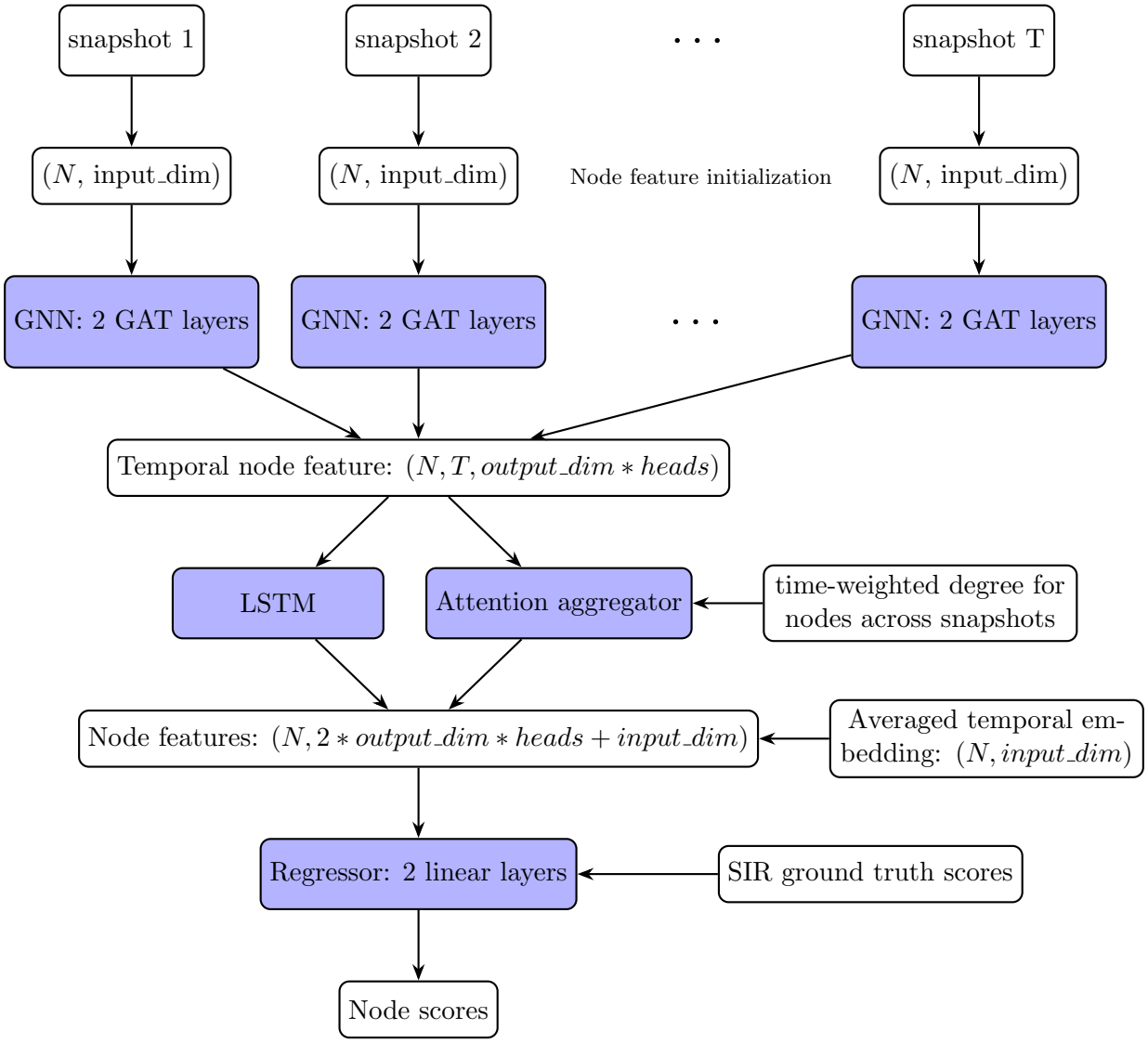


Figure 3.8: Architecture of the proposed TA-Pool-Seq-GAT model.

3.4.3 APool-Seq-GAT - Attention and Sequential Pooling over Graph Attention Networks embeddings

This section introduces a third architecture based on the last two ones. In the previous architecture, as part of the attention aggregator input, I used a time-weighted degree for nodes across snapshots. I thought that this information could not be very important. Indeed, this information should already be present in the temporal node embeddings, as they are initialized partially with their degree, and these temporal node embeddings are also an input of the temporal attention aggregator. So I decided to remove it. Moreover, to construct the final node embeddings, I also used an average temporal embedding, which doesn't help much to capture time-specific patterns or variations, and also removed it. This leads to the following architecture (Figure 3.9) that keep the more important modules from the previous approaches.

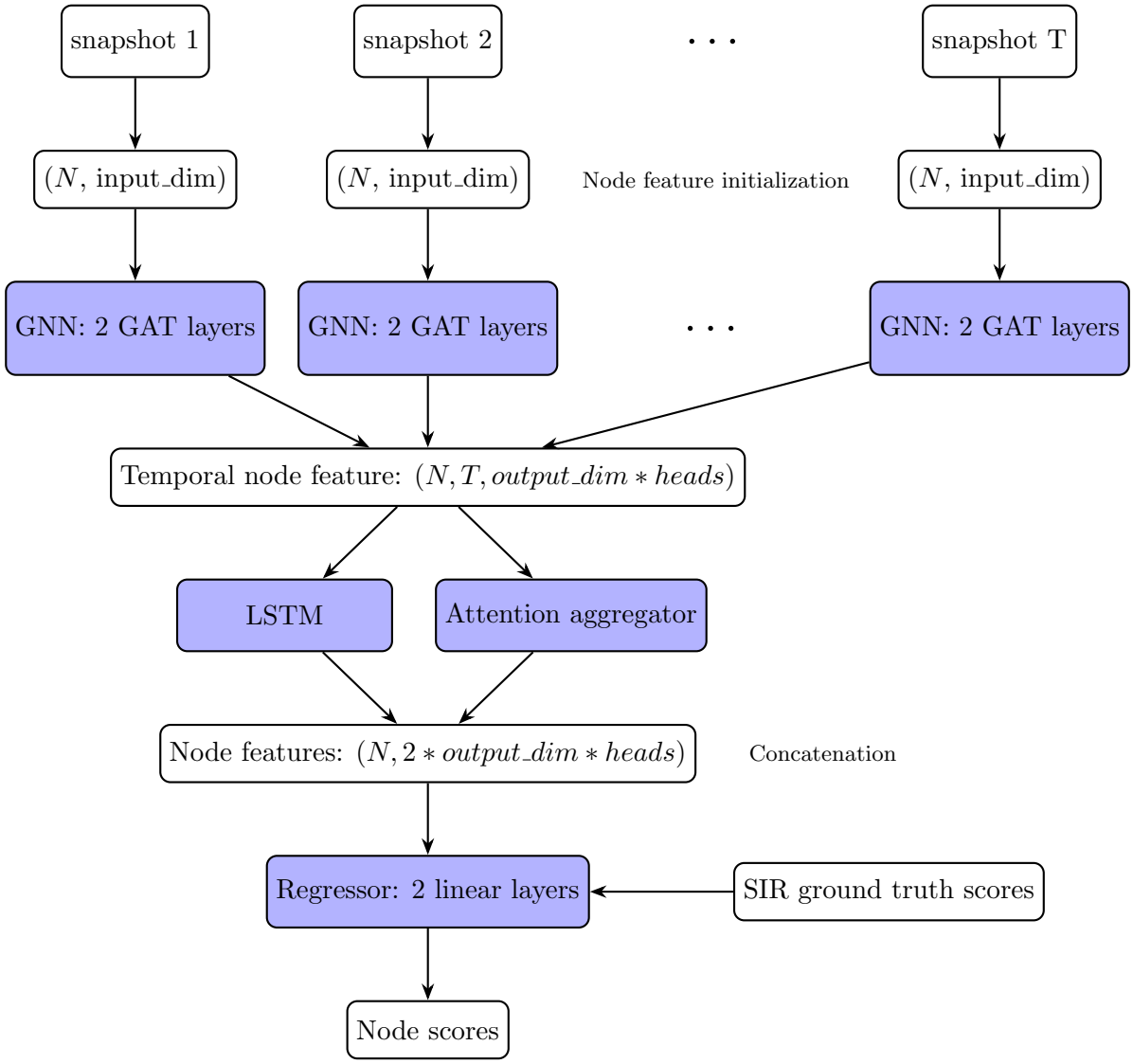


Figure 3.9: Architecture of the proposed APool-Seq-GAT model.

3.4.4 APool-T-GAT - Attention and Transformer Pooling over Graph Attention Networks embeddings

In this final variant of the architecture, based on the previous one, I replace the LSTM with a Transformer encoder to capture temporal dependencies between snapshots. As in previous models, the first two steps: node feature initialization and GAT-based per snapshot embedding remain unchanged. The resulting temporal node embeddings are then passed to the new aggregator module (Figure 3.10).

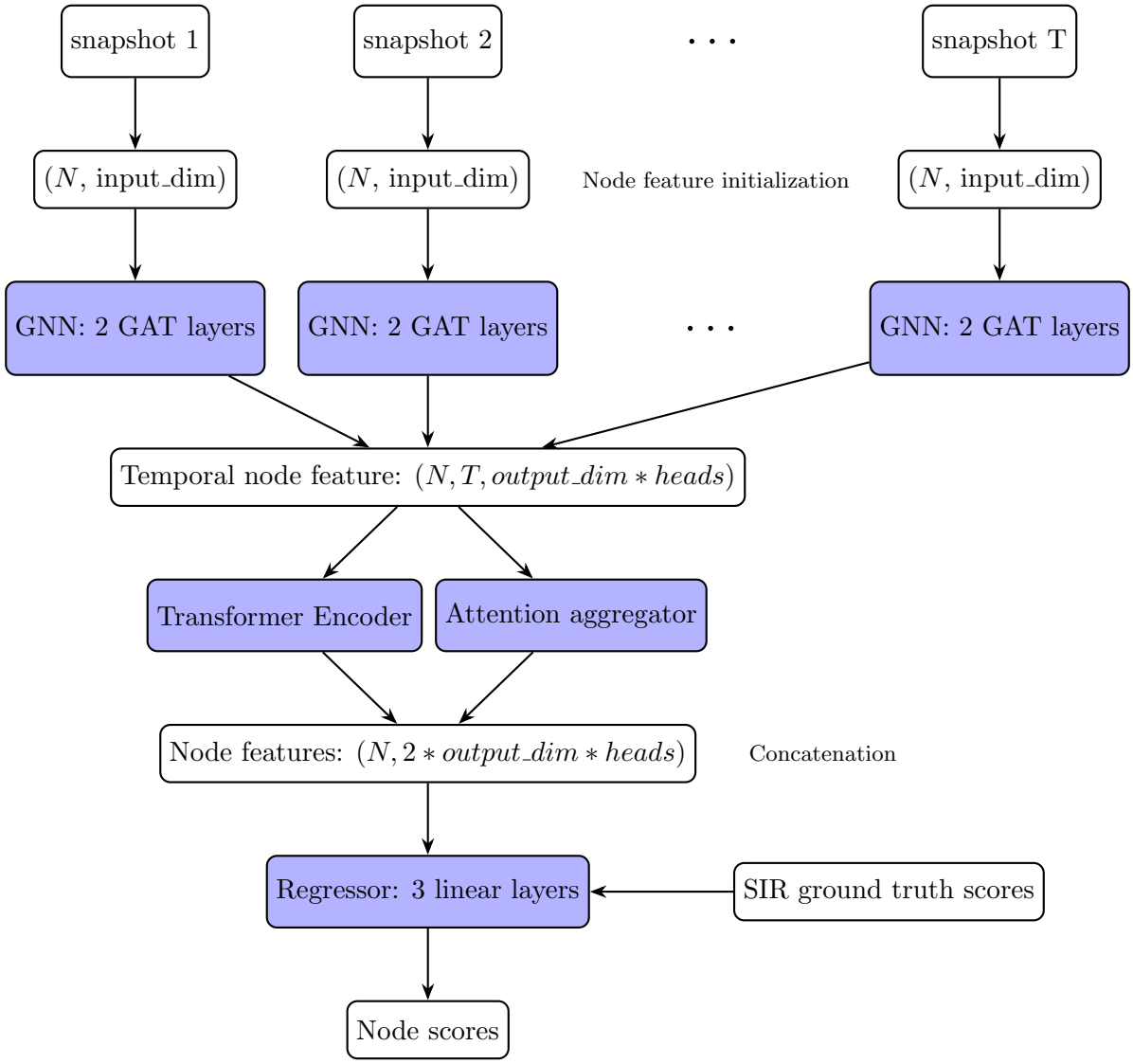


Figure 3.10: Architecture of the proposed APool-T-GAT model.

Unlike the previous aggregators, this one directly applies a Multi-Head Attention mechanism to model interactions between time steps. The key idea is to let each node attend to its own past representations and automatically learn which moments in time are most informative for predicting its importance.

To provide a notion of global temporal context, I compute the mean of each node's embeddings across all time steps and project it using a learnable Multilayer perceptron (MLP). This projection is then added back to the original temporal features to obtain an enriched input (Figure 3.11).

This enriched sequence is processed by a Multi-Head Attention layer, which outputs temporally attended embeddings for each node. To obtain a single vector representation per node, a mean pooling is applied across time, followed by layer normalization. This transformer-based design is supposed to capture long-range dependencies and contextual shifts across time more flexibly than a recurrent architecture like an LSTM, as was the case in previous architectures.

The multihead attention mechanism in APool-T-GAT provides a powerful way to model diverse temporal patterns and integrate global context, making it more expressive than single attention mechanisms.

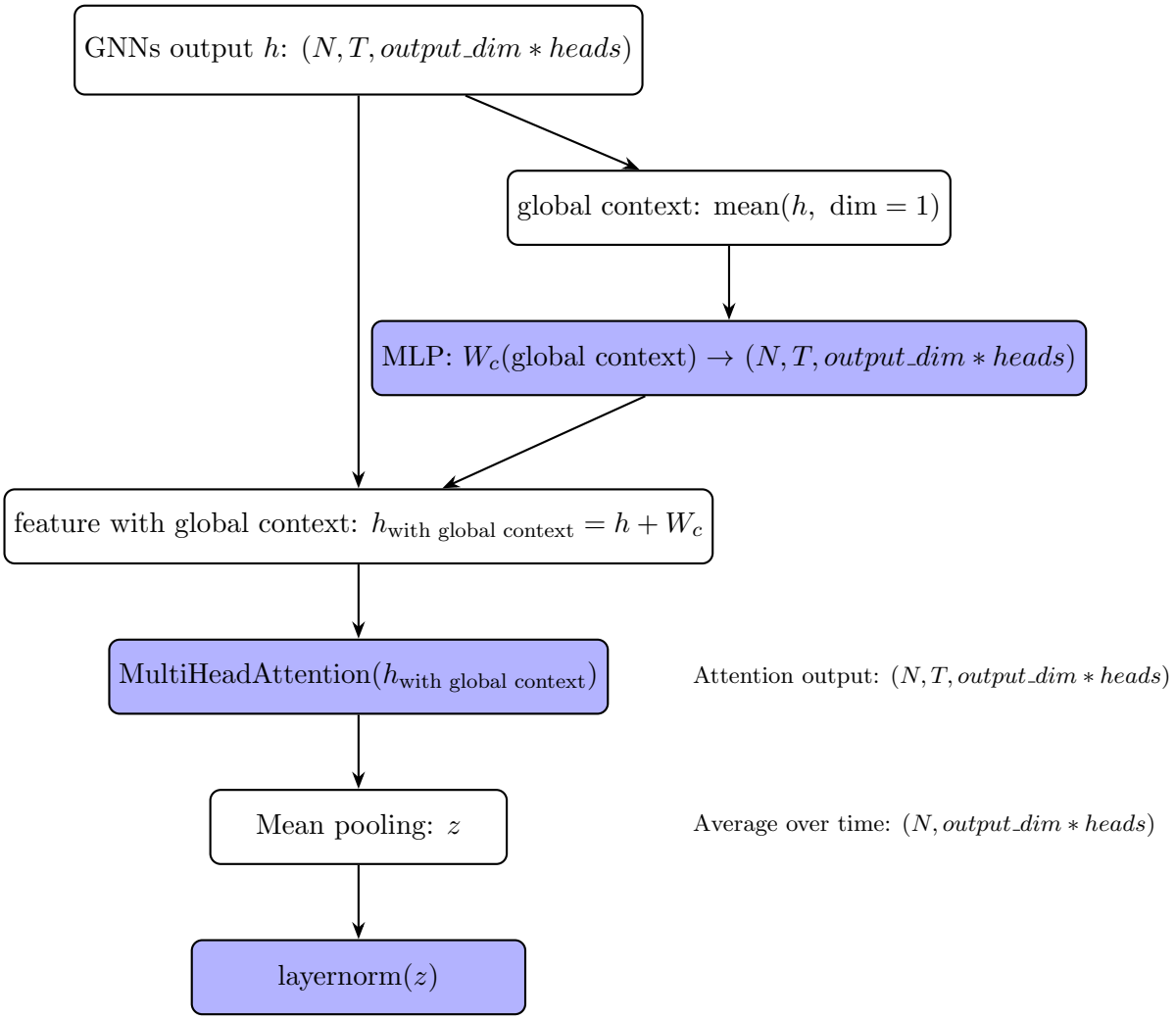


Figure 3.11: Attention aggregator architecture used in the APool-T-GAT model.

3.5 Training & Evaluation Strategy

This section will cover the different loss functions used to train the models and the evaluation strategies employed to assess their performance. Indeed, as we have seen in Chapter 2, the proposed VNI algorithms are often using different evaluation strategies or datasets, which makes the comparison between them pretty difficult.

3.5.1 Losses

This subsection will cover the different loss functions used to train the models. As the VNI problem can be seen as a ranking or regression problem, I used different types of losses.

Mean Squared Error loss

$$\mathcal{L}_{MSE}(S, \hat{S}) = \frac{1}{N} \sum_{i=1}^{N} (s_i - \hat{s}_i)^2 \tag{3.4}$$

The Mean Squared Error (MSE) loss measures how close a model's predictions are to the ground truth values. It tells, on average, how much our predictions are "off" from the truth, bigger mistakes hurt more due to squaring.

Margin Ranking loss

$$\mathcal{L}_{MR}(S, \hat{S}, \text{margin}) = \frac{1}{N^2} \sum_{1 \leq i,j \leq N} \max\left(0, \text{margin} - (\hat{s}_i - \hat{s}_j) * \text{sign}(s_i - s_j)\right) \tag{3.5}$$

If the correct item isn't ranked ahead of the other by at least the margin, the model gets penalized. It encourages separation between scores: the "better" item should not only be ranked higher but confidently so.

List Maximum Likelihood Estimation loss

$$\mathcal{L}_{ListMLE}(S, \hat{S}) = \sum_{i=1}^{N} \log \left(\sum_{j=i}^{N} e^{\hat{s}_{\pi(j)}} \right) - \hat{s}_{\pi(i)} \tag{3.6}$$

where $\pi = [\pi_1, \pi_2, \ldots, \pi_N] \in \mathcal{S}_N$ is a permutation such that $s_{\pi_1} \geq s_{\pi_2} \geq \cdots \geq s_{\pi_N}$. ListMLE is a list-wise ranking loss, it aims to rank the overall ranking results in comparison to the MSE loss, which is a point-wise ranking loss. The better the predicted scores match the correct ranking, the lower the loss.

3.5.2 Repeated Stratified K-Fold Evaluation Strategy

To ensure a reliable evaluation of my models in settings with limited data, we adopt a careful cross-validation strategy. As some networks only have around 100 nodes and we need a huge amount of them to train my models (80%), the validation/test sets (10%/10%) are sometimes quite small. To avoid applying my models only on small validation/test sets that are probably not good representations of the whole dataset, we use the `RepeatedStratifiedKFold` strategy from `scikit-learn`.

This strategy is similar to the well-known KFold cross-validation, it consists of performing KFold cross-validation multiple times with different random seeds. Moreover, the stratified part separates the nodes into 4 groups (depending on their score value) and ensures that each group is equally represented in each set (Figure 3.12).

This strategy enables us to more robustly evaluate our models by ensuring diverse and representative validation and test sets, thereby increasing the reliability and generalization of our model performance estimates. Indeed, as the Kendall's tau metric is highly sensitive to small score changes in small sets, this can lead to irregular results that further motivate the need for robust validation procedures.

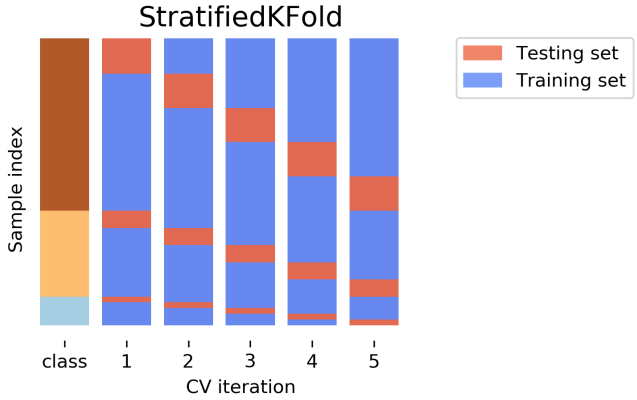


Figure 3.12: Illustration of the `StratifiedKFold` procedure with 5 folds and 3 classes. Each fold preserves the proportion of nodes from each class, ensuring balanced representation across training and validation sets.

3.6 Hyperparameter Tuning

To find the best hyperparameter (node embedding dimension, dropout, learning rate, `num_neighbors` for the initialization step...) for my models, I also used the `Weight & Biases` library to automate the hyperparameter search and visualize interactive experiment tracking. I used the Bayesian methods to search in the hyperparameter space. Bayesian search differs from other methods by optimizing parameter selection based on previous round scores. Instead of random selection, it intelligently chooses parameters, likely reaching the best set faster. This method focuses on the relevant search space and discards less promising ranges, enhancing efficiency.

Through the `Weight & Biases` library, I could track the training losses of all my experiments (Figure 3.13) and also analyze the best choices of hyperparameters depending on the different datasets and score types (Figure 3.14). Indeed, the left figure from figure 3.14 enables one to quickly see which combination of hyperparameter values leads to a high Kendall's Tau value. The figure on the right highlights the effectiveness of the Bayesian search, showing a general trend that the Kendall's Tau scores consistently improve as the search progresses.

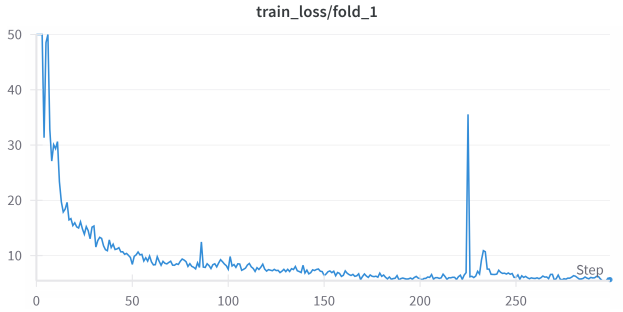


Figure 3.13: Training loss over epochs (Highschool-2011, R_{max})

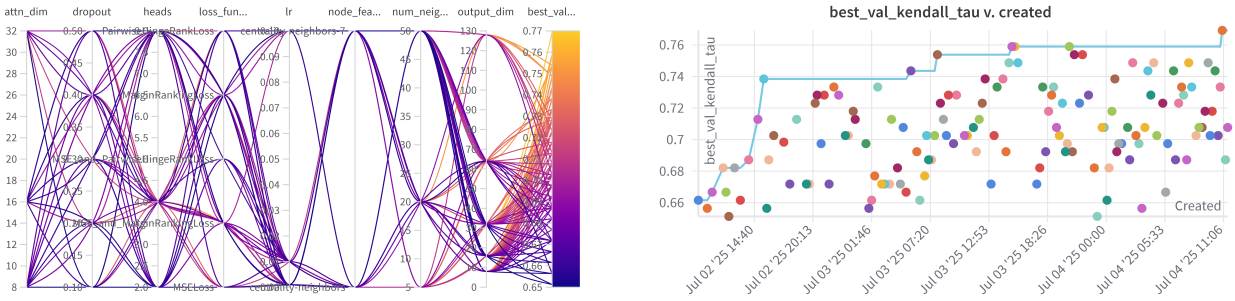


Figure 3.14: Hyperparameter tuning results (Highschool-2011, R_{max}). Left: each column shows a hyperparameter and its possible values; the last column shows the corresponding Kendall's tau on the validation set, allowing identification of effective configurations. Right: evolution of Kendall's tau over the course of the Bayesian search, showing general improvement in performance as the search progresses.

Indeed, each hyperparameter can have an impact on how well the model is learning:

- Number of GAT layers: More layers allow the model to capture higher-order graph structure but may lead to overfitting or vanishing gradients.
- Attention heads (GAT): More heads improve the model's ability to focus on different parts of the graph but increase computational cost.
- Hidden dimension (GAT): larger values enhance the model's capacity to learn richer node embeddings but may lead to overfitting and higher resource usage.

- Attention dimension for the Aggregator: Higher values improve the model's ability to capture complex temporal relationships but increase computational cost and risk of overfitting.
- Margin (MarginRankingLoss): a larger margin focuses on significant differences but risks ignoring small valid differences, while a smaller margin captures finer distinctions but may overfit to noise.
- num_neighbors: A higher number allows the model to capture more global graph structure and richer contextual information but increases computational cost and may introduce noise, while a lower number of neighbors focuses on local structure and reduces complexity but risks missing important information.
- Dropout: Higher rates reduce overfitting by preventing the model from relying too heavily on specific neurons but may harm learning if set too high, while lower dropout rates retain more information but increase the risk of overfitting.
- Learning rate: A higher rate speeds up training but risks overshooting the optimal solution and causing instability, while a lower learning rate ensures stable convergence but slows down training and may get stuck in local minima.

4 Results

In this chapter, I present the results obtained using both the baseline Temporal Gravity model and the models developed in this study. Through a comparative analysis between these models, it demonstrates both the potential and the limitations of approaches based on Graph Neural Networks for identifying vital nodes in temporal networks. All experiments involving the training of my models were conducted on an A800 GPU.

4.1 Performance of the Temporal Gravity Model

To enable a fair comparison between the Temporal Gravity (TG) model and the models developed in this study, I computed the Kendall's tau score for the TG model using the same `RepeatedStratifiedKFold` strategy applied for my models. Tables 4.1 and 4.2 respectively report, the validation scores and the test scores corresponding to the best validation performance for the R_{max} score type. Overall, we can observe that the TG model already achieves good Kendall's Tau results on all datasets.

The test scores for the R_{mean} and R_{norm} score types are respectively presented in Tables 4.3 and 4.4. The complete set of corresponding validation scores can be found in the Appendix (Tables A.5 and A.6).

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
$TG(TD)$	0.65	0.52	0.57	**0.71**	0.60	0.40	**0.54**	0.60
$TG(PR^m)$	0.58	0.59	0.60	0.64	0.60	**0.45**	0.54	0.67
$TG(PR^s)$	0.63	0.49	0.55	0.67	0.57	0.39	0.46	0.59
$TG(DC^m)$	0.70	0.63	**0.65**	0.63	0.64	0.44	0.51	**0.71**
$TG(DC^s)$	0.65	0.51	0.57	0.66	0.57	0.39	0.46	0.60
$TG(CC^m)$	**0.76**	**0.63**	0.61	0.58	**0.64**	0.42	0.54	0.70
$TG(CC^s)$	0.63	0.49	0.54	0.68	0.58	0.40	0.52	0.57
$TG(BC^m)$	0.53	0.51	0.44	0.46	0.58	0.32	0.42	0.57
$TG(BC^s)$	0.47	0.35	0.43	0.65	0.56	0.38	0.44	0.52

Table 4.1: Kendall's tau results on the **validation set** for the $\mathbf{R_{max}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
TG	0.75	0.65	0.63	0.75	0.63	0.38	0.57	0.74

Table 4.2: Best Kendall's tau results of the Temporal Gravity model on the **test set** for the $\mathbf{R_{max}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
TG	0.69	0.59	0.55	0.43	0.54	0.28	0.52	0.63

Table 4.3: Best Kendall's tau results of the Temporal Gravity model on the **test set** for the $\mathbf{R_{mean}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
TG	0.70	0.72	0.67	0.63	0.76	0.52	0.62	0.52

Table 4.4: Best Kendall's tau results of the Temporal Gravity model on the **test set** for the $\mathbf{R_{norm}}$ score type, `n_simulations` $= 100$, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

By comparing the results from Table 4.2 to those computed directly on the whole node set (Tables 3.2 and 3.3), we note that the Kendall's tau values are either slightly higher or comparable when using the `RepeatedStratifiedKFold` strategy.

From these results, we can observe that the performance of the Temporal Gravity (TG) model highly depends on the dataset. Indeed, it appears that the TG model performs well on the HS2011 dataset: Kendall's Tau is around 0.71 for the three score types, while it does not perform so well on PS: Kendall's Tau is around 0.39 over the three score types. This could be explained by the fact that the different datasets have different structural and dynamic properties. The TG model probably performs better on datasets that have, on average, a higher density score, which seems to be the case for HS2011, HC, and HT2009 (Figure 4.1), but not for PS. It remains difficult to establish a clear correlation between high performance and specific dataset properties.

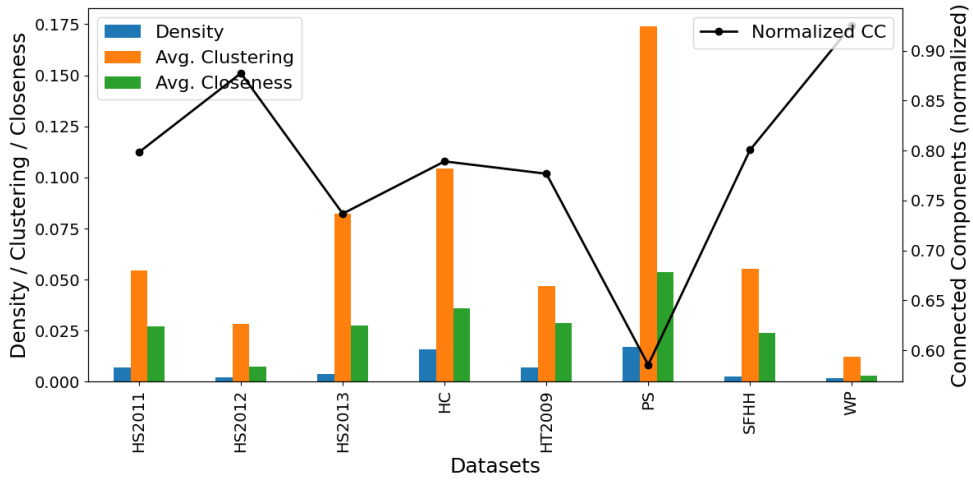


Figure 4.1: Global network statistics across datasets. Values are averaged over snapshots.

The score type used as ground truth can also be a parameter that seems to influence the performances of the TG model. Indeed, for the R_{max} score type and the HC dataset, the Kendall's Tau is 0.75, while it is only 0.43 on the R_{mean} score type. More generally, we can observe a drop in performance for all datasets between the R_{max} and R_{mean} score types.

4.2 Performance of the Proposed Models

Here are the Kendall's tau results obtained through my different architectures (Tables 4.5, 4.6 and 4.7) in comparison to the Temporal Gravity model (TG). The best corresponding results on the validation set are provided in the Appendix (Tables A.7, A.8, A.9). I focused my experimentation on the two models that seemed to be more promising than the other due to computational time constraints.

From the results presented in Tables 4.5, 4.6, and 4.7, several observations can be made. First, the APool-Seq-GAT model generally achieves the highest Kendall's tau values for both R_{max} and R_{mean} across most datasets, with particularly strong performance on HC, HT2009, and PS. This suggests that combining sequential and attention-based aggregator pooling allows the model to better capture the nodes' vitality than the other architectures tested.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT	0.67	0.53	0.57	0.77	0.72	0.52	0.54	0.61
TAPool-Seq-GAT	0.62	0.60	-	-	-	0.54	0.56	0.71
APool-Seq-GAT	**0.76**	0.50	0.57	**0.85**	**0.74**	**0.55**	**0.59**	0.53
APool-T-GAT	0.64	0.58	-	-	-	**0.55**	0.55	0.57
TG	0.75	**0.65**	**0.63**	0.75	0.63	0.38	0.57	**0.74**

Table 4.5: Corresponding Kendall's tau results on the **test set** for the $\mathbf{R_{max}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT	0.69	0.53	0.53	0.77	0.66	0.63	0.57	0.63
TAPool-Seq-GAT	0.61	-	-	-	-	**0.68**	0.53	0.58
APool-Seq-GAT	**0.75**	0.53	**0.55**	**0.83**	**0.69**	0.62	**0.61**	**0.69**
APool-T-GAT	0.58	0.51	-	-	-	0.66	0.51	-
TG	0.69	**0.59**	**0.55**	0.43	0.54	0.28	0.52	0.63

Table 4.6: Corresponding Kendall's tau results on the **test set** for the $\mathbf{R_{mean}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT	**0.75**	0.68	0.55	**0.81**	0.74	**0.71**	0.50	0.51
TAPool-Seq-GAT	0.69	-	-	-	-	0.62	0.46	-
APool-Seq-GAT	0.69	0.52	0.53	0.66	0.72	0.63	0.51	**0.56**
APool-T-GAT	0.65	-	-	-	-	0.59	0.49	-
TG	0.70	**0.72**	**0.67**	0.63	**0.76**	0.52	**0.62**	0.52

Table 4.7: Corresponding Kendall's tau results on the **test set** for the $\mathbf{R_{norm}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

In contrast, the Temporal Gravity (TG) model outperforms my architectures on HS2012 and HS2013 for R_{max} and R_{mean}, indicating that for these datasets, simpler temporal-distance-based heuristics may be more suitable than learned representations. However, identifying a clear correlation between network structural properties and model performance is challenging. Instead, the temporal dynamics of the networks appear to play a more decisive role in determining which nodes are most vital.

Regarding the R_{norm} score type, APool-GAT demonstrates strong performance across multiple datasets by surpassing APool-Seq-GAT in most cases (e.g., HS2011 and HC). Overall, while APool-Seq-GAT appears to be the most consistently strong model for predicting node vitality across different score types, the results also underline that no single model dominates across all datasets. Dataset structural and dynamic properties both clearly influence the relative performance of each approach.

4.3 Privacy Aspects

Identifying vital nodes in networks requires sophisticated methods, but privacy preservation is also crucial, especially with sensitive data like Bluetooth contact networks. This section discusses privacy-aware techniques that balance the identification of vital nodes with confidentiality.

As said before, an application of vital nodes identification in temporal networks could be epidemiology: the goal could be to identify key individuals who play a central role in the spread of a virus. During the coronavirus pandemic, several contact-tracing apps have been developed like, "TousAntiCovid" in France or "Corona-Warn-App" in Germany, to alert users if they had been in close contact with someone who tested positive for COVID-19 in the past few days, using Bluetooth connections on their smartphones. In this context, privacy is ensured by decentralizing the data on each individual's phone. Mathematically, this means that we only have access to a small local neighborhood around each person, rather than the full network.

Recent work has explored how to identify vital nodes in complex networks while preserving user privacy. In [6], the authors examine the effectiveness of centrality-based node ranking when access to the full network is restricted. They show that using only local 2-hop ego-network information can often approximate full-graph centrality measures. Leveraging this insight, they implement a machine learning pipeline that uses features from top-performing, privacy-aware centrality measures. The model is trained on a small (20%) node sample and successfully predicts influence scores for the rest. Their results demonstrate that even with limited data, machine learning models can approximate global influence rankings, offering a promising direction for privacy-preserving node evaluation.

Here are three Tables (4.8, 4.9, 4.10) comparing the results between different models and initialization methods in brackets. The corresponding results on the validation set are in Tables A.10, A.11, and A.12 in the Appendix.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT (centrality-neighbors)	0.67	0.53	0.57	0.77	0.72	0.52	0.54	0.61
APool-GAT (degree-neighbors)	0.72	0.57	0.54	0.79	0.74	0.57	**0.59**	0.57
APool-GAT (degree-only)	0.54	0.58	0.49	0.68	**0.78**	0.56	0.57	0.70
APool-Seq-GAT (centrality-neighbors)	**0.76**	0.50	0.57	0.85	0.74	0.55	**0.59**	0.53
APool-Seq-GAT (degree-neighbors)	0.63	0.48	0.52	**0.89**	0.73	**0.60**	0.55	0.57
APool-Seq-GAT (degree-only)	0.60	0.54	0.45	0.82	0.65	0.54	0.58	0.61
TG	0.75	**0.65**	**0.63**	0.75	0.63	0.38	0.57	**0.74**

Table 4.8: Corresponding results on the **test set** for the $\mathbf{R_{max}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT (centrality-neighbors)	0.69	0.53	0.53	0.77	0.66	0.63	0.57	0.63
APool-GAT (degree-neighbors)	0.69	0.49	0.54	0.71	**0.75**	0.62	0.52	0.53
APool-GAT (degree-only)	0.59	0.56	0.49	0.69	0.71	0.65	0.53	0.64
APool-Seq-GAT (centrality-neighbors)	**0.75**	0.53	**0.55**	**0.83**	0.69	0.62	**0.61**	**0.69**
APool-Seq-GAT (degree-neighbors)	0.64	0.49	0.50	0.75	0.69	**0.69**	0.52	0.52
APool-Seq-GAT (degree-only)	0.68	0.54	0.41	0.70	0.70	0.65	0.53	0.53
TG	0.69	**0.59**	**0.55**	0.43	0.54	0.28	0.52	0.63

Table 4.9: Corresponding results on the **test set** for the $\mathbf{R_{mean}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

The results in Tables 4.8, 4.9, and 4.10 show that stronger privacy constraints during initialization generally reduce model performance, reflecting the limited access to global network information. Surprisingly, in some cases, performance remains comparable or even improves slightly, suggesting that carefully chosen local information can partially compensate for missing global data. This highlights an important trade-off between privacy preservation and predictive accuracy, indicating that even with privacy constraints, it is possible to identify key nodes effectively.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT (centrality-neighbors)	**0.75**	0.68	0.55	0.81	0.74	**0.71**	0.50	0.51
APool-GAT (degree-neighbors)	0.51	0.60	0.50	**0.83**	0.67	0.64	0.50	0.61
APool-GAT (degree-only)	0.60	0.61	0.35	0.68	0.62	0.55	0.50	**0.63**
APool-Seq-GAT (centrality-neighbors)	0.69	0.52	0.53	0.66	0.72	0.63	0.51	0.56
APool-Seq-GAT (degree-neighbors)	0.54	0.58	0.51	**0.83**	0.69	0.61	0.48	0.61
APool-Seq-GAT (degree-only)	0.57	0.60	0.51	0.73	0.66	0.60	0.45	0.49
TG	0.70	**0.72**	**0.67**	0.63	**0.76**	0.52	**0.62**	0.52

Table 4.10: Corresponding results on the **test set** for the $\mathbf{R_{norm}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

If we assume that we have access to privacy-aware information like the 2-hop neighborhood of a node. In my APool-GAT & APool-Seq-GAT approaches, I use two GAT layers to learn the node feature vectors and to predict the final vitality score of a node (the aggregator part is done independently on each node). So if we assume that for each time step we have access to privacy-aware information like the 3-hop neighborhood of a node, then we can compute the node vitality.

In contrast, the Temporal Gravity model needs access to the neighbors of nodes in all future snapshots reachable from the source node to compute the temporal shortest distance. Moreover, if the node property used is closeness or betweenness, then we need full network access. This comparison highlights that GNN-based approaches can operate effectively with privacy-aware initialization, while Temporal Gravity and similar centrality-based models typically require broader network visibility to achieve comparable predictions.

4.4 Interpretability: Attention Aggregator Weights

The goal of the temporal attention weights is to measure how useful a time step's embedding is for predicting the target, relative to other steps. This means that attention is not an absolute indicator of node activity but a way of highlighting the most informative snapshots compared to the rest.

In Figure 4.2, we observe that when nodes have more connections, their attention weights often differ significantly from time steps where they are isolated. This is expected: timesteps with active connectivity are usually more informative, since they capture the dynamics that define vital nodes.

However, we also observe cases where attention weights are lower even though the node has many connections. This can be explained by the fact that GAT embeddings depend not only on the number of neighbors but also on their features. If the neighbors do not provide relevant or distinctive information, the resulting embedding will not be useful for prediction, and the attention mechanism correctly assigns it a lower weight.

Conversely, certain low-degree snapshots may still receive relatively high attention if they capture a critical structural change, such as forming a bridge to an important region of the network. In such cases, the attention mechanism emphasizes informativeness rather than sheer connectivity.

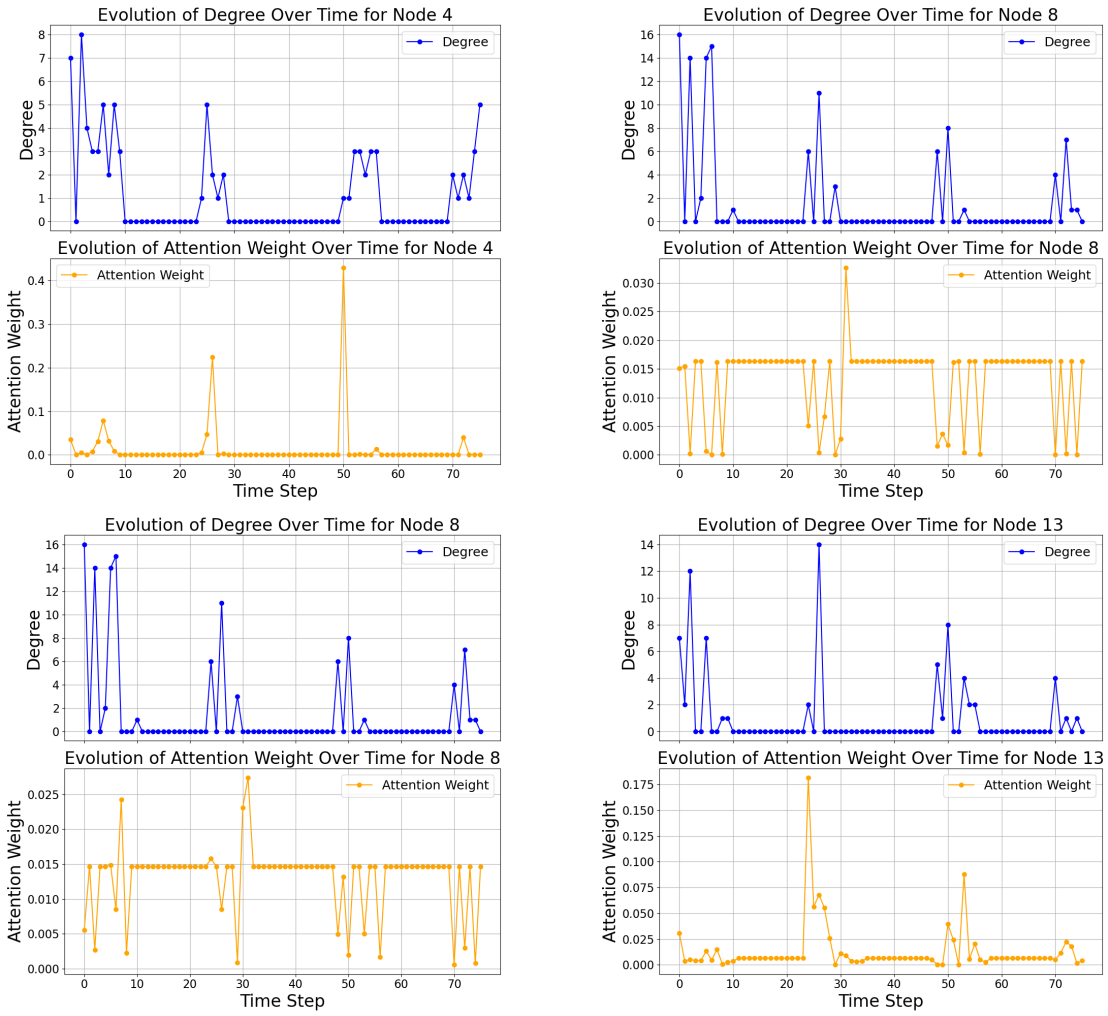


Figure 4.2: Evolution of attention weights over time for different nodes in comparison to the nodes' degree evolution (Highschool-2011, R_{max})

4.5 Limitations

When transforming raw contact data into a sequence of snapshots (Figures 1.1 and 1.2), some temporal information is lost: contacts that occur at different times may now appear at the same time step. One possible solution could be to process continuous-time temporal networks instead of discrete snapshots, as done in [24].

The models developed in this study have several limitations and drawbacks when compared to the Temporal Gravity model [3]. Indeed, the Temporal Gravity model relies on a simple formula and does not require any training data, whereas this is not the case for my architectures.

Additionally, the results are highly sensitive to the choice of the β and γ parameters in the SIR model. For instance, decreasing the value of β can prevent the diffusion dynamics from propagating through the network, resulting in all nodes having a vitality score of $\frac{1}{N}$. In such cases, it becomes impossible to accurately assess the relative importance of nodes within the network. In static networks, it is possible to define a threshold value for β below which the spreading dynamics die out, allowing us to distinguish between nodes that affect a large or small proportion of the network. Unfortunately, for temporal networks, no such clear threshold currently exists, although some studies have begun to explore this issue [17].

Moreover, the generalizability of these models is limited. Both my models and the Temporal Gravity model perform well only on certain types of network structures and may not transfer effectively to all network configurations.

Another limitation is the high sensitivity of my models to hyperparameter selection. Identifying the optimal hyperparameters requires repeated training and evaluation, which is computationally expensive.

Despite these limitations, the developed models still offer significant advantages. They are capable of learning rich temporal node feature representations and, through the use of an attention-based aggregator, can identify the most influential time steps in the network dynamics. Moreover, by using a privacy-aware feature for the initialization step, once the model is trained, we can compute the node vitality with limited access to the network and still achieve good performance.

5 Conclusion

In conclusion, many real-world systems are dynamic, and in order to capture their temporal characteristics, we have to represent them as temporal networks. Many approaches have been proposed to identify vital nodes in static networks, for example, through centrality measures or machine learning approaches. However, when it comes to temporal networks, this task is more challenging and becomes even more complex under privacy constraints.

In this thesis, I proposed a Graph Neural Network (GNN)-based approach for identifying vital nodes in temporal networks. The model first learns temporal node embeddings from sequences of networks and then aggregates them using an attention mechanism. This design allows the model to highlight the most relevant time steps when estimating node influence, rather than treating them all equally. By capturing these dynamic variations, the approach goes beyond simple averaging and is better suited to the complex, evolving nature of temporal networks.

To assess the effectiveness of this method, I compared it with the Temporal Gravity model, a widely used baseline that combines temporal shortest paths with static centrality measures. Although the Temporal Gravity model provides a strong and interpretable benchmark, the experiments show that the GNN-based approach can outperform it on certain datasets. This indicates that learning-based architectures are capable of capturing richer temporal patterns and adapting more flexibly to dynamic contexts than hand-crafted measures. Still, further work is needed to consistently surpass the Temporal Gravity model across all types of networks.

Despite these encouraging results, several limitations remain. The method depends on SIR-based simulations to produce ground-truth vitality scores, which are computationally intensive and sensitive to parameter choices. In addition, the model's performance strongly depends on hyperparameter tuning and varies between datasets, raising questions about its generalization and scalability.

Privacy-preserving aspects, although briefly considered, remain an open challenge. Many classical centrality measures require global access to the network, which is often unrealistic for sensitive data such as contact networks or social interactions. While local measures like degree centrality provide a more privacy-friendly alternative, integrating privacy-aware strategies directly into machine learning models for temporal networks is still largely unexplored and represents an important direction for future research.

Bibliography

[1] Ahmad Asgharian Rezaei, Justin Munoz, Mahdi Jalili, and Hamid Khayyam. A machine learning-based approach for vital node identification in complex networks. *Expert Systems with Applications*, 214:119086, 2023.

[2] V. Batagelj and M. Zaversnik. An O(m) Algorithm for Cores Decomposition of Networks, 2003.

[3] Jialin Bi, Ji Jin, Cunquan Qu, Xiuxiu Zhan, and Guanghui Wang. Temporal Gravity Model for Important Nodes Identification in Temporal Networks. *Chaos, Solitons & Fractals*, 147:110934, 2021.

[4] Ulrik Brandes. A faster algorithm for betweenness centrality. *The Journal of Mathematical Sociology*, 25(2):163–177, 2001. Routledge.

[5] Zhenhao Chen, Jiajing Wu, Yongxiang Xia, and Xi Zhang. Robustness of interdependent power grids and communication networks: A complex network perspective. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 65(1):115–119, 2018.

[6] Diaoulé Diallo and Tobias Hecking. Privacy-Preserving Vital Node Identification in Complex Networks Using Machine Learning. In *Complex Networks & Their Applications XIII*, pages 49–61, Cham, 2025. Springer Nature Switzerland.

[7] Diaoulé Diallo, Jurij Schoenfeld, René Schmieding, Sascha Korf, Martin J. Kühn, and Tobias Hecking. Integrating Human Mobility Models with Epidemic Modeling: A Framework for Generating Synthetic Temporal Contact Networks. *Entropy*, 27(5):507, 2025. Multidisciplinary Digital Publishing Institute.

[8] Julie Fournet and Alain Barrat. Contact Patterns among High School Students. *PLOS One*, 9(9):e107878, 2014.

[9] Linton C. Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215–239, 1978.

[10] Valerio Gemmetto, Alain Barrat, and Ciro Cattuto. Mitigation of infectious disease at school: Targeted class closure vs school closure. *BMC Infectious Diseases*, 14(1):695, 2014.

[11] Mathieu Génois and Alain Barrat. Can co-location be used as a proxy for face-to-face contacts? *EPJ Data Science*, 7(1):1–18, 2018. SpringerOpen.

[12] Mathieu Génois, Christian L. Vestergaard, Julie fournet, André Panisson, Isabelle Bonmarin, and Alain Barrat. Data on face-to-face contacts in an office building suggest a low-cost vaccination strategy based on community linkers. *Network Science*, 3(3):326–347, 2015. Cambridge University Press (CUP).

[13] J. E. Hirsch. An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences of the United States of America*, 102(46):16569–16572, 2005.

[14] Lorenzo Isella, Juliette Stehlé, Alain Barrat, Ciro Cattuto, Jean-François Pinton, and Wouter Van den Broeck. What's in a crowd? Analysis of face-to-face behavioral networks. *Journal of Theoretical Biology*, 271(1):166–180, 2011.

[15] Hyoungshick Kim and Ross Anderson. Temporal node centrality in complex networks. *Physical Review E*, 85(2):026107, 2012. American Physical Society.

[16] Mohammed Lalou, Mohammed Amin Tahraoui, and Hamamache Kheddouci. The Critical Node Detection Problem in networks: A survey. *Computer Science Review*, 28:92–117, 2018.

[17] Jack Leitch, Kathleen A. Alexander, and Srijan Sengupta. Toward epidemic thresholds on temporal networks: A review and open questions. *Applied Network Science*, 4(1):1–21, 2019. SpringerOpen.

[18] Yuchen Li, Ju Fan, Yanhao Wang, and Kian-Lee Tan. Influence maximization on social graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 30(10):1852–1872, 2018.

[19] Qiang Liu, Yu-Xiao Zhu, Yan Jia, Lu Deng, Bin Zhou, Jun-Xing Zhu, and Peng Zou. Leveraging local h-index to identify and rank influential spreaders in networks. *Physica A: Statistical Mechanics and its Applications*, 512:379–391, 2018.

[20] Linyuan Lü, Duanbing Chen, Xiao-Long Ren, Qian-Ming Zhang, Yi-Cheng Zhang, and Tao Zhou. Vital nodes identification in complex networks. *Physics Reports*, 650:1–63, 2016.

[21] Rossana Mastrandrea, Julie Fournet, and Alain Barrat. Contact Patterns in a High School: A Comparison between Data Collected Using Wearable Sensors, Contact Diaries and Friendship Surveys. *PLOS One*, 10(9):e0136497, 2015. Public Library of Science.

[22] M. E. J. Newman. Spread of epidemic disease on networks. *Physical Review E*, 66:016128, 2002. American Physical Society.

[23] Lawrence Page, Sergey Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking : Bringing Order to the Web. In *The Web Conference*, 1999.

[24] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal Graph Networks for Deep Learning on Dynamic Graphs. In *ICML Workshop on Graph Representation Learning and Beyond*, 2020.

[25] Sushila Shelke and Vahida Attar. Source detection of rumor in social network – A review. *Online Social Networks and Media*, 9:30–42, 2019.

[26] Philippe Vanhems, Alain Barrat, Ciro Cattuto, Jean-François Pinton, Nagham Khanafer, Corinne Régis, Byeul-a Kim, Brigitte Comte, and Nicolas Voirin. Estimating Potential Infection Transmission Routes in Hospital Wards Using Wearable Proximity Sensors. *PLOS One*, 8(9):e73970, 2013. Public Library of Science.

[27] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, 2018.

[28] Juan Wang, Chao Li, and Chengyi Xia. Improved centrality indicators to characterize the nodal spreading capability in complex networks. *Applied Mathematics and Computation*, 334:388–400, 2018.

[29] You Xiong, Zheng Hu, Chang Su, Shi-Min Cai, and Tao Zhou. Vital node identification in complex networks based on autoencoder and graph neural network. *Applied Soft Computing*, 163:111895, 2024.

[30] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, 2020.

[31] Mingxuan Yan, Yuexing Han, and Bing Wang. NFI-SGAT: Node feature initialization-based streaming graph learning model with graph attention network for critical node identification in temporal networks. *Neurocomputing*, 630:129679, 2025.

[32] En-Yu Yu, Yan Fu, Xiao Chen, Mei Xie, and Duan-Bing Chen. Identifying critical nodes in temporal networks by network embedding. *Scientific Reports*, 10(1):12494, 2020. Nature Publishing Group.

[33] En-Yu Yu, Yue-Ping Wang, Yan Fu, Duan-Bing Chen, and Mei Xie. Identifying critical nodes in complex networks via graph convolutional networks. *Knowledge-Based Systems*, 198:105893, 2020.

[34] Ahmad Zareie, Amir Sheikhahmadi, Mahdi Jalili, and Mohammad Sajjad Khaksar Fasaei. Finding influential nodes in social networks based on neighborhood correlation coefficient. *Knowledge-Based Systems*, 194:105580, 2020.

[35] Min Zhang, Xiaojuan Wang, Lei Jin, Mei Song, and Ziyang Li. A new approach for evaluating node importance in complex networks via deep learning methods. *Neurocomputing*, 497:13–27, August 2022.

[36] Jingcheng Zhu and Lunwen Wang. Identifying Influential Nodes in Complex Networks Based on Node Itself and Neighbor Layer Information. *Symmetry*, 13(9):1570, 2021. Multidisciplinary Digital Publishing Institute.

A Appendix

A.1 Temporal Gravity Model: Published vs. Implemented Results

	HS2011	HS2012	HS2013	HC
$TG(TD)$	0.64 (-0.03)	0.51 (=)	0.48 (-0.03)	0.39 (+0.05)
$TG(PR^m)$	0.58 (-0.03)	0.54 (+0.01)	0.47 (-0.02)	0.38 (+0.03)
$TG(PR^s)$	0.60 (-0.02)	0.49 (-0.01)	0.46 (-0.02)	0.40 (+0.01)
$TG(DC^m)$	0.65 (-0.01)	0.58 (+0.01)	0.54 (=)	0.41 (=)
$TG(DC^s)$	0.63 (-0.02)	0.51 (-0.01)	0.48 (-0.02)	0.40 (=)
$TG(CC^m)$	0.69 (-0.02)	0.57 (+0.01)	0.49 (-0.01)	0.37 (+0.01)
$TG(CC^s)$	0.63 (-0.03)	0.49 (-0.01)	0.46 (-0.04)	0.39 (+0.04)
$TG(BC^m)$	0.52 (-0.02)	0.52 (=)	0.36 (-0.02)	0.32 (+0.01)
$TG(BC^s)$	0.46 (-0.01)	0.37 (-0.02)	0.35 (-0.02)	0.39 (=)

Table A.1: Kendall's Tau results for the first four datasets presented in the Temporal Gravity paper [3] for the $\mathbf{R_{mean}}$ score type (in brackets this is the difference to add to obtain the results from my implementation), `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed on all the nodes.

	HT2009	PS	SFHH	WP
$TG(TD)$	0.50 (=)	0.26 (+0.01)	0.45 (+0.06)	0.50 (-0.02)
$TG(PR^m)$	0.51 (-0.01)	0.29 (=)	0.47 (+0.02)	0.56 (+0.02)
$TG(PR^s)$	0.49 (=)	0.27 (=)	0.43 (+0.01)	0.49 (-0.01)
$TG(DC^m)$	0.52 (+0.01)	0.28 (=)	0.47 (+0.01)	0.63 (+0.01)
$TG(DC^s)$	0.49 (=)	0.26 (=)	0.43 (=)	0.48 (=)
$TG(CC^m)$	0.53 (=)	0.25 (=)	0.47 (+0.01)	0.63 (+0.01)
$TG(CC^s)$	0.49 (=)	0.26 (=)	0.45 (+0.04)	0.48 (-0.02)
$TG(BC^m)$	0.48 (=)	0.21 (-0.01)	0.41 (=)	0.51 (=)
$TG(BC^s)$	0.47 (=)	0.29 (=)	0.41 (=)	0.42 (=)

Table A.2: Kendall's Tau results for the last four datasets presented in the Temporal Gravity paper [3] for the $\mathbf{R_{mean}}$ score type (in brackets this is the difference to add to obtain the results from my implementation), `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed on all the nodes.

	HS2011	HS2012	HS2013	HC
$TG(TD)$	0.59 (+0.03)	0.58 (-0.01)	0.60 (-0.06)	0.56 (-0.04)
$TG(PR^m)$	0.60 (+0.01)	0.62 (+0.02)	0.60 (-0.04)	0.55 (-0.02)
$TG(PR^s)$	0.57 (+0.05)	0.57 (=)	0.59 (-0.04)	0.56 (-0.01)
$TG(DC^m)$	0.66 (+0.05)	0.66 (+0.04)	0.69 (-0.01)	0.59 (+0.02)
$TG(DC^s)$	0.58 (+0.06)	0.60 (=)	0.61 (-0.03)	0.55 (=)
$TG(CC^m)$	0.63 (+0.06)	0.64 (+0.04)	0.60 (-0.03)	0.51 (+0.02)
$TG(CC^s)$	0.58 (+0.03)	0.57 (-0.01)	0.58 (-0.06)	0.55 (-0.02)
$TG(BC^m)$	0.51 (+0.03)	0.58 (+0.01)	0.47 (-0.03)	0.51 (+0.01)
$TG(BC^s)$	0.46 (+0.03)	0.44 (-0.01)	0.43 (-0.01)	0.53 (+0.01)

Table A.3: Kendall's Tau results for the first four datasets presented in the Temporal Gravity paper [3] for the $\mathbf{R_{norm}}$ score type (in brackets this is the difference to add to obtain the results from my implementation), `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed on all the nodes.

	HT2009	PS	SFHH	WP
$TG(TD)$	0.68 (-0.01)	0.47 (-0.03)	0.60 (+0.03)	0.32 (-0.03)
$TG(PR^m)$	0.71 (-0.01)	0.48 (-0.02)	0.62 (-0.01)	0.39 (+0.01)
$TG(PR^s)$	0.67 (+0.01)	0.47 (-0.02)	0.59 (-0.02)	0.31 (-0.01)
$TG(DC^m)$	0.74 (+0.01)	0.52 (-0.02)	0.63 (-0.01)	0.43 (+0.03)
$TG(DC^s)$	0.68 (=)	0.47 (-0.02)	0.58 (-0.01)	0.30 (=)
$TG(CC^m)$	0.68 (+0.01)	0.46 (-0.01)	0.59 (-0.01)	0.42 (+0.02)
$TG(CC^s)$	0.68 (=)	0.47 (-0.03)	0.59 (+0.03)	0.31 (-0.02)
$TG(BC^m)$	0.68 (+0.03)	0.40 (-0.01)	0.55 (-0.02)	0.33 (+0.02)
$TG(BC^s)$	0.65 (+0.01)	0.43 (=)	0.52 (+0.02)	0.30 (=)

Table A.4: Kendall's Tau results for the last four datasets presented in the Temporal Gravity paper [3] for the $\mathbf{R_{norm}}$ score type (in brackets this is the difference to add to obtain the results from my implementation), `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed on all the nodes.

A.2 Hyperparameter Selection and Comparisons

A.2.1 Temporal Gravity Model: Mass properties selection

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
$TG(TD)$	0.64	0.56	0.47	**0.47**	0.49	0.28	**0.52**	0.50
$TG(PR^m)$	0.58	0.60	0.47	0.43	0.51	0.31	0.51	0.62
$TG(PR^s)$	0.60	0.52	0.46	0.43	0.48	0.27	0.44	0.50
$TG(DC^m)$	0.68	**0.64**	**0.55**	0.42	**0.54**	0.30	0.49	0.68
$TG(DC^s)$	0.64	0.54	0.48	0.42	0.49	0.27	0.43	0.51
$TG(CC^m)$	**0.71**	0.64	0.50	0.37	0.53	0.26	0.49	**0.68**
$TG(CC^s)$	0.63	0.52	0.44	0.43	0.49	0.27	0.50	0.46
$TG(BC^m)$	0.57	0.58	0.36	0.38	0.54	0.24	0.40	0.53
$TG(BC^s)$	0.49	0.37	0.35	0.42	0.47	**0.31**	0.41	0.43

Table A.5: Best results on the **validation set** for the $\mathbf{R_{mean}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
$TG(TD)$	0.67	0.58	0.59	0.56	0.70	0.45	0.63	0.31
$TG(PR^m)$	0.65	0.66	0.60	0.60	0.72	0.47	0.63	0.41
$TG(PR^s)$	0.67	0.56	0.59	0.60	0.70	0.46	0.57	0.33
$TG(DC^m)$	**0.75**	**0.73**	**0.71**	**0.66**	**0.78**	**0.50**	**0.63**	0.46
$TG(DC^s)$	0.69	0.60	0.62	0.60	0.71	0.46	0.57	0.32
$TG(CC^m)$	0.73	0.71	0.60	0.60	0.72	0.44	0.58	**0.46**
$TG(CC^s)$	0.66	0.57	0.56	0.57	0.69	0.45	0.61	0.30
$TG(BC^m)$	0.60	0.60	0.48	0.57	0.74	0.39	0.52	0.39
$TG(BC^s)$	0.55	0.43	0.46	0.57	0.67	0.44	0.53	0.29

Table A.6: Best results on the **validation set** for the $\mathbf{R_{norm}}$ score type, `n_simulations` $= 100$, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

A.2.2 Proposed Approaches: Hyperparameter selection

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT	0.79	0.69	0.58	0.87	0.71	0.68	0.65	0.67
TAPool-Seq-GAT	0.79	0.71	-	-	-	0.68	0.68	0.73
APool-Seq-GAT	0.81	0.70	0.58	0.81	0.76	0.67	0.64	0.70
APool-T-GAT	0.70	0.65	-	-	-	0.66	0.64	0.7

Table A.7: Best Kendall's tau results on the **validation set** for the $\mathbf{R_{max}}$ score type, `n_simulations` $=$ 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT	0.78	0.66	0.58	0.87	0.71	0.70	0.60	0.78
TAPool-Seq-GAT	0.81	-	-	-	-	0.71	0.63	0.72
APool-Seq-GAT	0.78	0.64	0.59	0.83	0.74	0.68	0.63	0.71
APool-T-GAT	0.71	0.61	-	-	-	0.68	0.55	-

Table A.8: Best results on the **validation set** for the $\mathbf{R_{mean}}$ score type, `n_simulations` $= 100$, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT	0.75	0.65	0.58	1	0.73	0.72	0.56	0.69
TAPool-Seq-GAT	0.74	-	-	-	-	0.72	0.59	-
APool-Seq-GAT	0.76	0.66	0.57	0.91	0.73	0.70	0.60	0.66
APool-T-GAT	0.63	-	-	-	-	0.70	0.55	-

Table A.9: Best results on the **validation set** for the $\mathbf{R_{norm}}$ score type, `n_simulations` $= 100$, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

A.3 Privacy-preserving Initializations

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT (degree-neighbors)	0.74	0.67	0.53	0.84	0.72	0.69	0.64	0.70
APool-GAT (degree-only)	0.60	0.65	0.47	0.86	0.76	0.64	0.65	0.65
APool-Seq-GAT (degree-neighbors)	0.77	0.65	0.53	0.87	0.78	0.65	0.60	0.66
APool-Seq-GAT (degree-only)	0.64	0.64	0.47	0.89	0.74	0.65	0.66	0.73

Table A.10: Best results on the **validation set** for the $\mathbf{R_{max}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT (degree-neighbors)	0.74	0.64	0.54	0.84	0.66	0.69	0.56	0.71
APool-GAT (degree-only)	0.63	0.65	0.52	0.83	0.73	0.66	0.56	0.79
APool-Seq-GAT (degree-neighbors)	0.71	0.67	0.50	0.81	0.64	0.70	0.53	0.70
APool-Seq-GAT (degree-only)	0.66	0.65	0.52	0.89	0.68	0.67	0.58	0.76

Table A.11: Best results on the **validation set** for the $\mathbf{R_{mean}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

	HS2011	HS2012	HS2013	HC	HT2009	PS	SFHH	WP
APool-GAT (degree-neighbors)	0.69	0.62	0.55	0.91	0.70	0.68	0.53	0.66
APool-GAT (degree-only)	0.55	0.64	0.42	0.91	0.69	0.66	0.51	0.66
APool-Seq-GAT (degree-neighbors)	0.66	0.64	0.53	0.89	0.63	0.68	0.49	0.69
APool-Seq-GAT (degree-only)	0.60	0.65	0.55	0.90	0.64	0.70	0.55	0.63

Table A.12: Best results on the **validation set** for the $\mathbf{R_{norm}}$ score type, `n_simulations` = 100, $\beta = 0.1$, and $\gamma = 0.01$. Kendall's tau has been computed following the `RepeatedStratifiedKFold` strategy.

A.4 SIR Ground truth

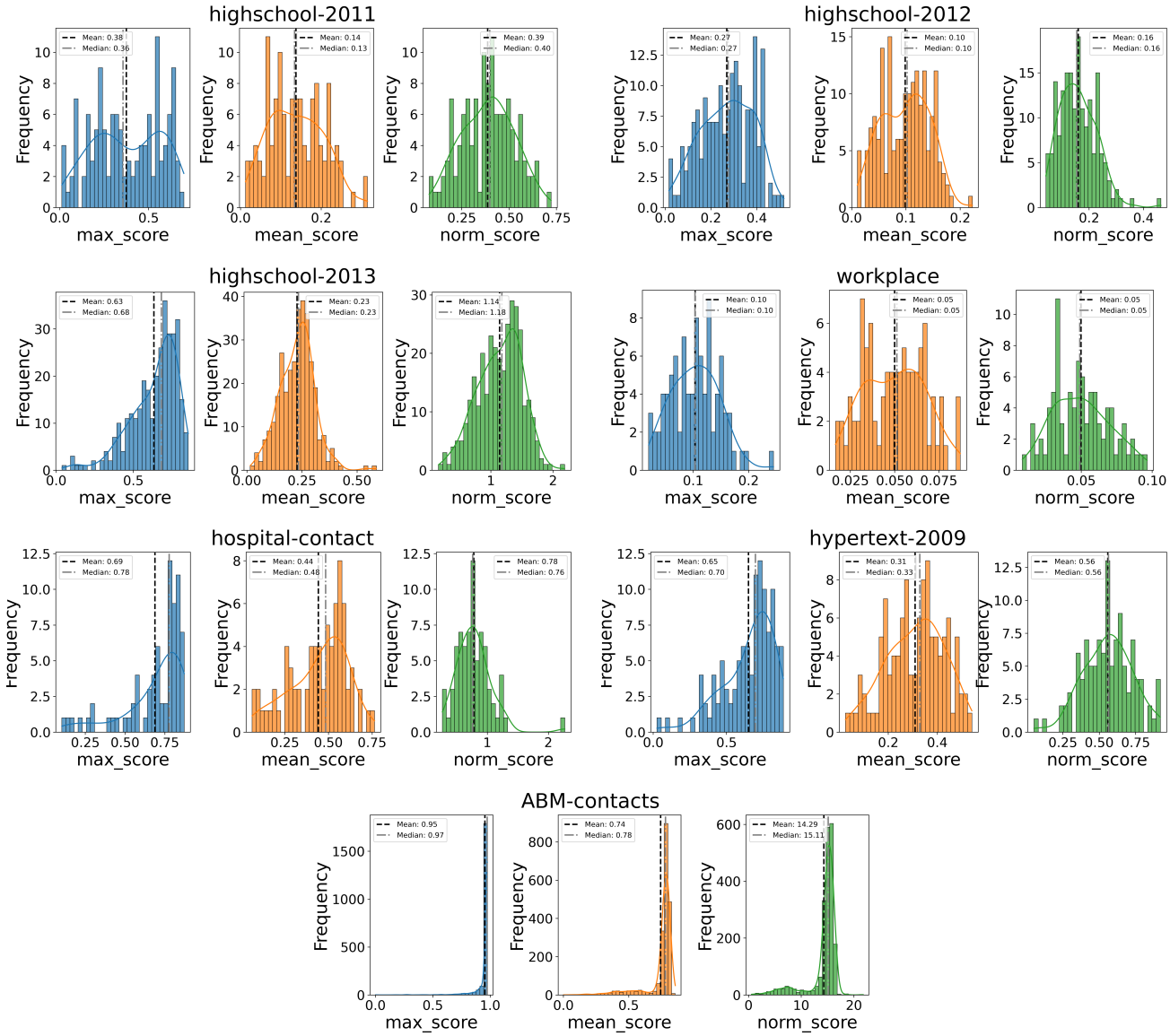


Figure A.1: SIR ground truth score distribution