Software Engineering Department Braude

Academic College

Capstone Project Phase A – 61998

Analyzing article's citations using - anomaly detection in dynamic graphs via transformer

24-2-R-2

Project Members:

Elroei Seadia, Dvir Bublil

Dvir.Bublil@e.braude.ac.il

Elroei.Seadia@e.braude.ac.il

Supervisor

Prof. Zeev Volkovich

Advisor

Dr. Renata Avros

Project's Git Repository

Table of Contents

Abstract:

This research deals with the struggle of identifying citations anomalies in academic research. Traditional citation evaluation and detection strategies frequently miss planned manipulations and sincere mistakes. To overcome these limitations, we propose a new technique using TADDY (Transformer-Based Anomaly Detection for Dynamic Graphs). After adapting TADDY to our model, it will treat citation networks as dynamic graphs and allowing analyze complex patterns over time to identify intricate citation behaviors. The research emphasizes the importance of citations anomaly detection, the limitations of existing strategies, and the way dynamic graphs and transformers provide an improved solution. In addition, our findings reveal that TADDY successfully detects anomalous citation patterns, providing a sturdy and efficient framework which can detect citation anomalies. By employing this method, we can trace articles, analyze their evolution over time, and gain valuable insights for the academic community.

Keywords:

Manipulated Citations, Dynamic Graph, Anomaly Detection, Research Analysis, Citation Networks.

1. Introduction

For decades, citation analysis has played a key role in assessing research and its effects. For example, by counting how often a study is cited or referenced by others, and by examining the citations it receives and those it makes, researchers, institutions, and policymakers have gotten useful insights into the importance of scholarly contributions. This measure has helped to inform funding choices, recognize academic accomplishments, and spot new research areas. Going beyond simple counts looking at citation patterns over time has offered more understanding of a research article's path how it shapes later work, and its overall influence on the field. But the growing problem of citation manipulation and the shortcomings of old-school methods like counting and studying citations have created a need for smarter ways to do this. While current techniques give some useful information, they often miss the mark when it comes to spotting sneaky

misconduct or keeping up with new trends. Some researchers are deliberately trying to make their work look more important by including fake citations or ones that aren't relevant. Others make honest mistakes when citing sources. These issues can make it hard to track and analyze papers and decide which are truly influential and which ones seem important because of manipulation or errors. There are existing ways to try to find unusual citation patterns that might indicate problems. These methods might look at sudden spikes in citations ("citation bursts"), how often papers are cited together ("co-citation patterns"), or how individual researchers cite sources. However, these approaches have limitations [1]. They might miss important new discoveries if they focus too much on bursts. They might not catch problems in single papers if they only look at connections between papers. And they can be influenced by how researchers in different fields typically cite sources or who they collaborate with. This means they might miss real problems, like attempts to manipulate citations, or mistake innocent mistakes for something more serious. Research is constantly evolving, and the way papers are cited is changing too. We need better tools to keep up with this complexity. Some researchers are looking at using machine learning to improve how we detect unusual citation patterns. This research explores a promising tool called Transformer-based Anomaly Detection for Dynamic Graphs (TADDY). We will use TADDY to analyze citation networks as constantly changing graphs in different timestamps. By doing this, it can learn complex patterns that help differentiate between real problems, like attempts to manipulate citations, and simple mistakes. By tracking the evolution of citations over time, we can construct a detailed trajectory for each article, revealing how its citation patterns change [2]. This enables us to identify articles that exhibit anomalous behavior, either consistently or over specific periods. Adapting TADDY on citation networks could lead to a more reliable and adaptable way to find unusual patterns, ultimately helping to ensure research is honest and transparent, and provide as a tool to track research behavior over time.

This research will delve into the background and related work on citation analysis, mathematical frameworks underpinning our approach, the working process from data collection to model evaluation, and the design and implementation of TADDY algorithms. Highlighting the effectiveness of TADDY in improving citation anomaly detection and as a result of that, promoting research analyze by tracking his path during analyzing his behavior in several timestamps.

2. Background and Related Work

2.1. Why Accurate Citations Matter

The cornerstone of educational research rests on the inspiration of accurate citations, ensuring the reliable trade of know-how. Both intentional manipulation (expect quotation cartels) and accidental errors (lacking citations) can infiltrate the network, distorting the real impact of studies. When researchers stuff their work with unrelated or made-up references, they twist the research scene [3]. This practice boosts citation numbers for personal gain, which shakes people's faith in research and makes it harder to spot important work. Citation manipulation, as exemplified by the "citation cartels" identified by Kojaku [4] , distorts the research landscape and erodes public trust.

Even honest slip-ups, like forgetting to cite sources or human mistakes, can cause big problems. These mistakes often occur due to human error or the fast-paced changes in research fields, paint a wrong picture of scholarly impact. For example, think about trying to see the whole picture of a topic when key pieces are missing or wrong! Besides just counting citations and finding manipulations, looking at how they connect can help us understand how research grows and changes, find key influencers, and map out knowledge networks. To protect the truth and value of citation data, we need to tackle both tampering and mistakes and spot them as odd occurrences.

2.2. The Limits of Existing Methods and a New Approach

Uncovering hidden patterns within the citations network is crucial for preserving research integrity and ensuring accurate analysis. Identifying the ones anomalies – unusual citation patterns that deviate from the norm – is crucial for upholding accept as proper with in instructional discourse.

Imagine a bustling city where streets represent research papers and intersections symbolize citations. New buildings (papers) are constantly under construction, and existing roads (citations) are built, rerouted, or even closed as new connections form. Traditional methods for anomaly detection, like counting the traffic on a single street, fail to capture the complexity of this dynamic urban landscape.

Current methods for detecting citation anomalies often rely on techniques that struggle to keep pace with the increasingly sophisticated tactics employed by those seeking to manipulate citations. In addition, traditional citation analysis methods, like counting citations and analyzing co-citations, are easy to bypass through strategic citation manipulation, as Fong and Wilhite's study found [5]. These methods often miss subtle patterns of

citation manipulation and struggle to keep up with the changing tactics researchers use to boost their citation numbers. They can't adjust to new manipulation strategies or catch the small details of unintended mistakes.

For example, one common approach relies on centrality measures like degree centrality (number of citations received) or betweenness centrality (importance of a publication in connecting others). While these metrics offer a basic understanding of citation patterns, they have significant limitations. A study [6] highlights how these measures can be easily manipulated by citation cartels artificially inflating citation counts. Additionally, they struggle to identify subtle anomalies, such as missing citations or unusual citation patterns within specific research communities.

Other processes consist of statistical analysis and key-word analysis. Statistical strategies, like studying citation frequency or co-quotation analysis (identifying papers regularly referred to collectively), can offer a simple expertise of citation styles. However, they warfare to seize the intricate relationships within quotation networks and are not adaptable to evolving tendencies. Analyzing the key phrases used inside citations can provide insights into the context of citations, however depending totally on keywords can be deceptive, as similar key phrases can seem in unrelated contexts. Additionally, this approach requires extensive information pre-processing and can be computationally expensive.

Recent advancements have explored using neural networks for anomaly detection in quotation networks. These networks are powerful tools, capable of gaining knowledge of complex, non-linear relationships inside statistics. However, neural networks also have barriers in this context. The "black box" hassle makes it tough to apprehend how they arrive at their anomaly detections, that could preclude consider and restriction the ability to refine and improve the exist version [7]. Neural networks require big amounts of fantastic training statistics to function effectively, and acquiring clean and complete citation statistics may be challenging, mainly for emerging research fields where facts might be scarce. Moreover, neural networks trained on historical records might warfare to evolve to new and evolving manipulation tactics employed by awful actors.

These limitations highlight the need for a more robust and adaptable approach to citation anomaly detection.

This is where our study comes in. We suggest a new approach using the Transformer-based Anomaly Detection for Dynamic Graphs (TADDY) framework. While we'll explain TADDY in more detail later, the main idea is this: TADDY views the citation network as a changing graph (over a timestamps) where each publication is a point, and citations are the links between them. By analyzing this dynamic graph, TADDY can uncover useful information that might indicate manipulation or errors. Our goal is

to develop a more robust and adaptable approach for discerning truth from anomalies within citation data, fostering a trustworthy and reliable academic ecosystem. This approach also provides the ability to analyze and track the evolution of research over the years.

2.3. Why Dynamic Graphs are Key to Citation Anomaly Detection

The traditional methods, which includes counting the wide variety of instances a paper is referred to and more, offer only a basic understanding and fail to capture the dynamic and evolving nature of networks (as shown in **Fig. 1**.). New publications appear each day, continuously converting the landscape of citations. Subtle anomalies, like a surprising increase in citations from a particular group of authors or lacking hyperlinks inside a studies network, can without difficulty be neglected.

This is in which dynamic graphs come into play. Dynamic graphs treat citations networks as dwelling entities that evolve and adapt over time. This approach allows researchers to:

- Uncover Hidden Knowledge Flows: Dynamic graphs move beyond static snapshots, reflecting the continuing float of citations and the development of recent connections between papers. They capture the continuous development of studies, where new findings generate fresh references and reshape the educational panorama.
- Highlight Subtle Anomalies: By closely monitoring the evolution of relationships in the community, dynamic graphs can monitor subtle adjustments in quotation styles that is probably missed via conventional analysis. For instance, an unexpected upward push in citations from a specific group of authors over a short duration may want to imply capacity manipulation.
- Stay Ahead of Manipulation: As research practices evolve, so do the techniques of folks that would possibly try and manipulate the system. The adaptable nature of dynamic graphs allows in figuring out new types of citation misconduct, acting like a vigilant observer who watches for suspicious patterns.

Using dynamic graphs, researchers can develop more effective and flexible methods for detecting citation anomalies. This leads to a more trustworthy academic environment, where the integrity of citations is maintained, and the true impact of research is accurately reflected. Dynamic graphs are crucial for ensuring that knowledge flows freely and that significant discoveries are properly recognized. In addition, provide us clearly image about the research and discover the ability to track ana analysis research according their evolving.

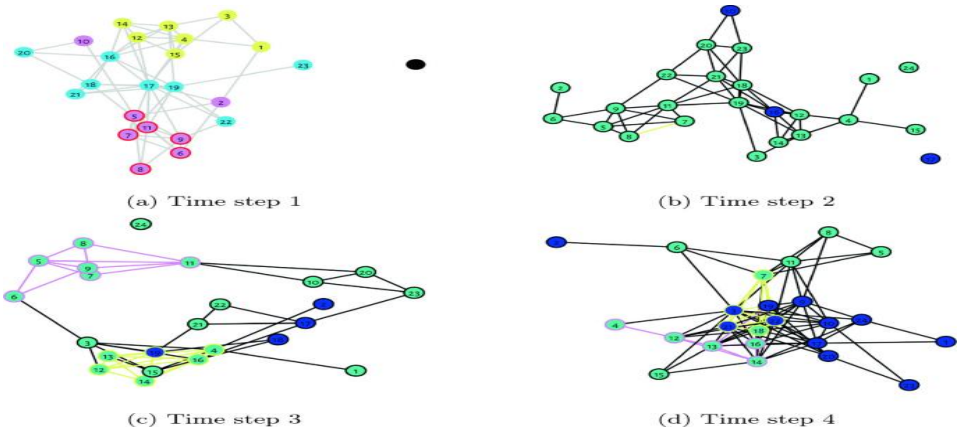

Fig. 1. An example of the evolution of a nature network with different time periods(a, b, c, d). That can be represent a dynamic graph.

2.4. Anomaly Detection in Dynamic Graphs

Detecting unusual connections in dynamic information networks is vital for researchers, especially since these networks are constantly evolving. Traditional methods, such as analyzing structural links or using deep learning techniques, can uncover hidden patterns but often have limitations. For example, approaches like GOutlier [8] and CAD focus on structural connectivity and changes in edge weights to detect anomalies, while deep learning methods like NetWalk [9] and AddGraph [10] use advanced models like Graph Convolutional Networks (GCNs) [11] and Gated Recurrent Units (GRUs) [12] to capture both spatial and temporal patterns.

The TADDY framework stands out by using a transformer network to simultaneously handle both spatial and temporal information, unlike other methods that separate these tasks. TADDY also better in analyzing unlabeled data, such as social media posts without captions, through a unique encoding process that integrates both spatial and temporal features. This makes TADDY particularly effective in identifying subtle and significant anomalies in dynamic networks, offering a more comprehensive solution compared to existing approaches in the context of anomalies detection in dynamic graphs.

2.5. Transformers

Imagine a detective board filled with evidence: photos of suspects, maps of locations, and a messy timeline of events. In the more traditional approach, a detective would carefully look at each piece of such evidence one after another to put the story together. But TADDY is the detective, and this tool can scrutinize everything instantaneously through its transformer network. This "tool" is the transformer network, a powerful technique in neural network. It is an ultra-powered brain that is good at understanding relations and context between different pieces of information. In TADDY's case, the information is the ever-changing network of connections in a dynamic graph (especially the constantly changing nodes and edges and, as a result, their context and role). The transformer is designed in such a way as to enable a look at the structure of the network (who is connected to whom) and also how the structure of that network is changing over time (new connections appearing, old ones fading). Again, this joint analysis, being driven by the attention mechanism **[fig2]**, empowers TADDY to pull out important patterns or relationships in dynamic complex networks even if these patterns are very weak or changing very fast. That is why other approaches are more like detectives with individual tools, and TADDY's transformer network makes it the super sleuth. It can analyze the whole "crime scene" at once, which thereby leads to understanding the evolving information landscape. Be able to effectively analyze the dynamic network of citations and their interactions to extract relevant information where traditional methods of mitigation are falling short. The attention mechanism permits the transformer network to focus on certain nodes and edges in the dynamic graph by similarity calculation and weighting clues proportional to their relevance, which sets out the importance of that clue for comprehension of overall network architecture and changes.

- Core mathematical equations behind the attention mechanism: $\boldsymbol{H^{(l)} = attenion\left(H^{(l-1)}\right) = softmax\left(\frac{Q^{(l)}K^{(l)T}}{\sqrt{d_{emb}}}\right)V^{(l)}}$ The output hidden state of a layer is calculated by applying a SoftMax function to the dot product of the query and key matrices, scaled by the square root of the embedding dimension, and then multiplying the result by the value matrix.

$\boldsymbol{Q^{(l)} = H^{(l-1)} * W_Q^{(l)}}$ The query matrix equal to the multiplying of the previous layer's hidden state with ta learnable weight matrix.
$\boldsymbol{K^{(l)} = H^{(l-1)} * W_K^{(l)}}$ The key matrix equal to the multiplying of the previous layer's hidden state with ta learnable weight matrix.
$\boldsymbol{V^{(l)} = H^{(l-1)} * W_V^{(l)}}$ The value matrix equal to the multiplying of the previous layer's hidden state with ta learnable weight matrix.

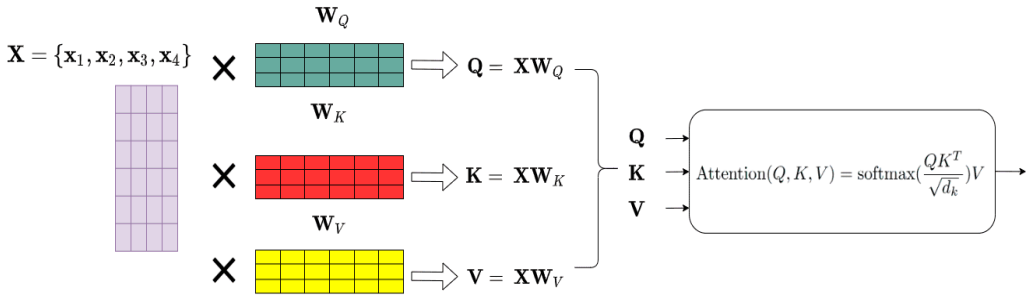

Fig. 2. Example of transformer, uses 3 different representations: the Queries, Keys and Values of the embedding matrix. This can easily be done by multiplying our input $X \epsilon R^{N*d_{model}}$ with 3 different weight matrices $W_Q, W_k, W v \epsilon R^{N*d_{model}}$. In essence, it's just a matrix multiplication in the original node embeddings. The resulted dimension will be smaller: $d_k < d_{model}$

2.6. Snapshots

Snapshots refer to temporary captures of the dynamic graph at specific moments. Think of the dynamic graph as a constantly changing web of connections. Snapshots are like grabbing still frames of this ever-evolving network, capturing its state at specific points in time. These snapshots are crucial for TADDY's analysis, but unlike relying solely on individual pictures, TADDY's strength lies in its ability to analyze both the snapshots themselves and how the connections within them evolve over time.

2.7. Timestamps

Timestamps are like labels on those snapshots of the dynamic graph and determine the exact moment to capture the state of the network. Imagine the dynamic graph as a constantly changing network of connections, and snapshots as pictures capturing its state at specific moments. Timestamps act like labels on these pictures, telling you exactly when each snapshot was taken. This allows TADDY to not only analyze the connections within each snapshot, but also track how they change over time. It's like comparing photos taken at different points to see how the network evolves.

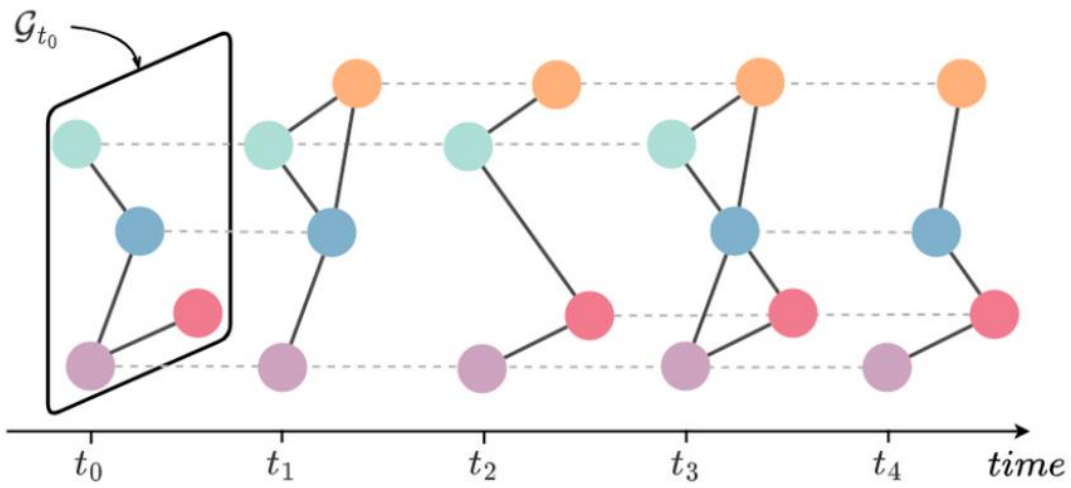

Fig. 3. A Discrete-Time Dynamic Graph defined over five timestamps (in each timestamp we took a snapshot of the evolving graph). The evolution of a Continuous-Time Dynamic Graph through the stream of events until the timestamp t_4

2.8. Binary adjacency matrix

Binary adjacency matrix acts like a map of connections, telling you who's connected to whom at a specific moment.

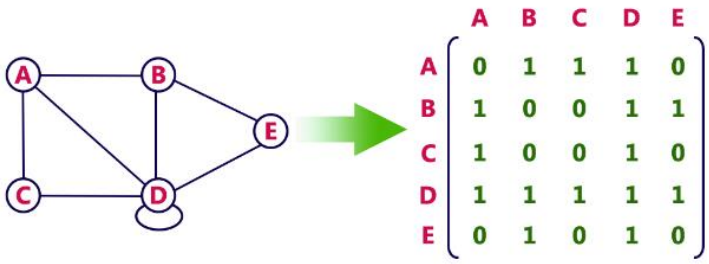

Fig. 4. An example of binary adjacency matrix for given graph, one indicating there is connection between two points, otherwise zero

2.9. Anomaly edge detection to ranking problem

Our approach to detecting unusual connections and anomalies in dynamic networks is rooted in ranking dynamics. We conceptualize the network as a complex system where nodes are ranked based on their interaction levels and their context (structural and temporal). Edges represent the connections between these ranked entities. Our model focuses on identifying disruptions in this ranking order. A sudden surge in interactions for a previously low-ranking node is considered an anomaly, potentially signaling a hidden connection or suspicious activity. By prioritizing these ranking shifts, we efficiently uncover unusual patterns that deviate from the network's normal behavior.

2.10. Anomaly score

An anomaly score is a numerical value that represents how much an element deviates from what is considered normal or expected within a dataset. The anomaly score helps identify patterns that deviate from the norm. It's like having a spotlight that shines on potentially suspicious connections, allowing to take a closer look and understand why they might be unusual.

2.11. Target Edge

Each edge in dynamic graphs is viewed as the center of the sampled substructure and is denoted as a target edge.

2.12. Target Nodes

The source node and destination node of the target edge denote as target nodes.

2.13. Contextual Nodes

Neighboring nodes of target node in the sampled substructure, are denote as contextual nodes.

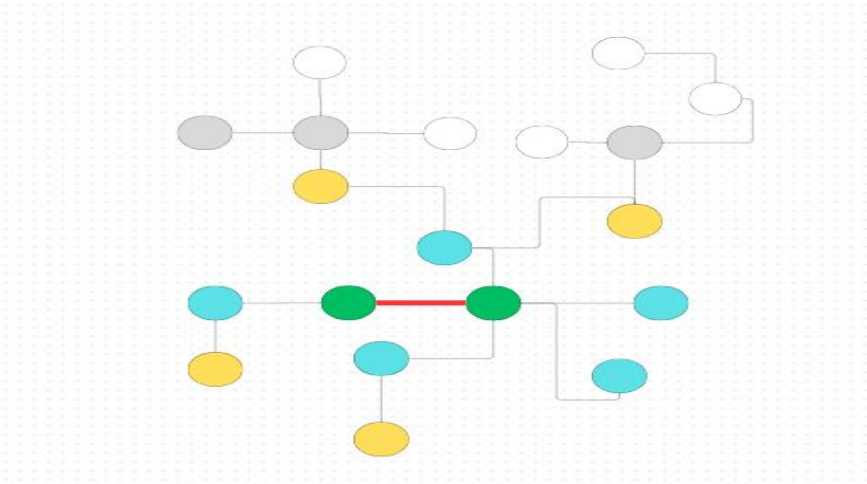

Fig. 5. Illustrate a graph where the target edge is highlighted with a red line, connecting two target nodes shown in green. Surrounding these are contextual nodes: blue for nodes at distance 1 from the target nodes, yellow for distance 2, and gray for distance 3 or more. Here, the contextual nodes are determined based on distance, but different methods can define contextual nodes differently, which will be described in more detail in the next sections.

2.14. Spatial Local Information

This refers to details about the immediate surroundings of a particular point

or element within a network. Imagine a map – spatial local information for a city might include the names of its closest neighboring towns, the types of businesses located nearby, or the presence of a park across the street.

2.15. Spatial Global Information

This focuses on the bigger picture within a network, considering the entire structure and how different elements relate to each other across space. Going back to the map analogy, spatial global information would be understanding the city's location within the country, its connection to major highways, or its position relative to other large cities.

2.16. Temporal Information

This refers to information related to time. In a network context, it could be when connections were formed, how they change over time, or the order in which events occur within the network. Imagine a dynamic traffic map – temporal information would tell you the current traffic flow, how it changes throughout the day, or how congestion builds and dissipates over time.

3. Mathematical Background

3.1 Notations

$\boldsymbol{A} \in \boldsymbol{R^{n_x n}}$ **–** The Binary adjacency matrix, each entry in the matrix corresponds to the connection between two specific nodes (i, j). 0 indicates that there is no direct connection between node i and node j. 1 indicates that there is direct connection between node i and node j.

$\boldsymbol{A^t}$ **-** The binary adjacency matrix at timestamp t.

$\boldsymbol{n^t}$ – The number of nodes at timestamp t.

$\boldsymbol{m^t}$ – The number of edges at timestamp t.

$\boldsymbol{G} = \{\boldsymbol{g^t}\}_{\boldsymbol{t=1}}^{\boldsymbol{T}}$ – A graph steam with a maximum timestamp of T.

$\boldsymbol{g^t} = (\boldsymbol{V^t}, \boldsymbol{\varepsilon^t})$ – The snapshot graph at timestamp t.

$\boldsymbol{V^t}$ – The node set at timestamp t.

$\boldsymbol{\varepsilon^t}$ – The edge set at timestamp t.

$\boldsymbol{v_i^t} \in \boldsymbol{V^t}$ – A node with index i at the timestamp t.

$\boldsymbol{\varepsilon_{i,j}^t} = (\boldsymbol{v_i^t}, \boldsymbol{v_j^t}) \in \boldsymbol{\varepsilon^t}$ **–** An edge between $\boldsymbol{v_i^t}$ and $\boldsymbol{v_j^t}$ at the timestamp t.

$\boldsymbol{f}(\cdot)$ – Anomaly score function.

$\boldsymbol{G_\tau^t} = \{\boldsymbol{g^{t-\tau+1}}, \dots, \boldsymbol{g^t}\}$ – The sequence of graphs with timestamp t as the end.

$\boldsymbol{S}(\boldsymbol{e_{tgt}^t})$ – The substructure node set of target edge $\boldsymbol{e_{tgt}^t}$.

$x_{diff}(v_j^i)$ – The diffusion-based spatial encoding of node v_j^i.

$x_{dist}(v_j^i)$ – The distance-based spatial encoding of node v_j^i.

$x_{temp}(v_j^i)$ – The relative temporal encoding of node v_j^i.

$x(v_j^i)$ – The fused encoding of node v_j^i.

$X(e_{tgt}^t)$ – The encoding matrix of target edge e_{tgt}^t.

$H^{(l)}$ – The output embedding of the l-th layers of Transformer.

$Q^{(l)}$ – The query matrix of the l-th layers of Transformer.

$K^{(l)}$ – The key matrix of the l-th layers of Transformer.

$V^{(l)}$ – The value matrix of the l-th layers of Transformer.

$z(e_{tgt}^t)$ – The embedding of target edge e_{tgt}^t.

$W_Q^{(l)}, W_K^{(l)}, W_V^{(l)}$ – The learnable parameters of Transformer.

$e_{pos,i} \in \varepsilon^t$ – The i-th positive edge from ε^t.

$e_{neg,i} \in \varepsilon_n^t \sim P_n(\varepsilon^t)$ – The i-th negative edge by negative sampling $P_n(\varepsilon^t)$.

w_S, b_S – The learnable weights and parameters of the scoring module.

k – The number of contextual nodes.

d_{enc} – The dimension of encoding.

d_{emb} – The dimension of embedding.

L – The learnable parameters of Anomaly Detector.

τ – The size of time window.

$n_s = \tau(k+2)$-is the number of nodes in the substructure of node set $S(e_{tgt}^t)$.

e_{pos}, i - The i -th positive (normal) edge.

e_{neg}, i – The i -th negative (pseudo-anomalous) edge.

$S \in R^{n_x n}$ – graph diffusion matrix.

$T \epsilon R^{n_x n}$ - is the generalized transition matrix derived from A.

Q_k - The weighting coefficient that determines the balance between global and local information.

Source: [13]

3.2. Graph Diffusion Matrix

The Graph Diffusion Matrix is essentially important in modeling the flow of information over a network. In our case, this helps capture structural and temporal dynamics within the graph, which is applied to anomaly detection. It is done by simulating the diffusion process using the Personalized PageRank (PPR) method over the matrix. Personalized PageRank (PPR) is a variant of the classic PageRank algorithm that considers the importance of nodes relative to some particular node or group

of nodes. PPR simulates a random walk on a graph in which, at each step, a person can either continue moving with probability α or return back to the starting node. Such an approach focuses the ranking in a local neighborhood of nodes and is hence quite useful where the relationship of a node to a specific set of nodes is more important than its global importance. The PPR formula reflects such a localized importance and is useful in identifying anomalies by pointing out deviations from the expected spreading pattern within the network.

There's a probability $\boldsymbol{\alpha}$, The formula for PPR is:

$S^{PPR} = \boldsymbol{\alpha}(\boldsymbol{I_n} - (\mathbf{1} - \boldsymbol{\alpha})\boldsymbol{D}^{-\frac{\mathbf{1}}{\mathbf{2}}}\boldsymbol{A}\boldsymbol{D}^{-\frac{\mathbf{1}}{\mathbf{2}}})^{-\mathbf{1}}$

$\boldsymbol{\alpha}$ is the teleport probability, controlling the likelihood of returning to the starting node.

$\mathbf{I_n}$ is the identity matrix.

$\boldsymbol{D}$ is the diagonal degree matrix of the graph.

3.3. Single-layer linear encoding learning function

The single-layer linear encoding learning function translates differences in node attributes into a linear encoded format. This is useful for many tasks (distances, connections, etc.). We'll be using this approach in three different contexts:

Diffusion-Based Spatial Encoding

$$\boldsymbol{x_{diff}}(\mathbf{v}_j^i) = \boldsymbol{linear}(\boldsymbol{rank}\left(\mathbf{S}_{e_{tgt}}^i\right)[\boldsymbol{idx}(\mathbf{v}_j^i)]) \in \boldsymbol{R}^{d_{enc}}$$

Distance-Based Spatial Encoding

$$\boldsymbol{x_{dist}}(\mathbf{v}_j^i) = \boldsymbol{linear}(\boldsymbol{min}(\boldsymbol{dist}(\mathbf{v}_j^i, \mathbf{v}_1^i), \boldsymbol{dist}(\mathbf{v}_j^i, \mathbf{v}_2^i))) \in \boldsymbol{R}^{d_{enc}}$$

Temporal Encoding

$$\boldsymbol{x_{temp}}(\mathbf{v}_j^i) = \boldsymbol{linear}(||\boldsymbol{t} - \boldsymbol{i}||) \in \boldsymbol{R}^{d_{enc}}$$

3.4. Encoding fusion

Encoding fusion, in general, refers to the process of combining information extracted from different encoding methods.

$$\boldsymbol{x}(\mathbf{v}_j^i) = \boldsymbol{x_{diff}}(\mathbf{v}_j^i) + \boldsymbol{x_{dist}}(\mathbf{v}_j^i) + \boldsymbol{x_{temp}}(\mathbf{v}_j^i) \in \boldsymbol{R}^{d_{enc}}$$

In our case, this fused encoding represents each node in the substructure with a comprehensive blend of spatial and temporal information, allowing for a more detailed and nuanced understanding of the node's role within the graph.

3.5. Encoding Matrix

For a given target edge e_{tgt} , the encoding of each node in its substructure node set is calculated and stacked into an encoding matrix $\boldsymbol{X}(\boldsymbol{e_{tgt}})$

$$\boldsymbol{X}(\boldsymbol{e}_{\boldsymbol{tgt}}^{\boldsymbol{t}}) = \oplus_{\mathbf{v}_j^i \in S(e_{tgt}^t)} [\boldsymbol{x}(\mathbf{v}_j^i)]^T \in \boldsymbol{R}^{(\tau(k+2)) \times d_{enc}}$$

3.6. Scoring Module

A scoring module, typically a fully connected neural network layer with a sigmoid activation function, computes the anomaly scores for each edge embedding:

$$\boldsymbol{f}(\boldsymbol{e}) = \boldsymbol{Sigmoid}(\boldsymbol{z}(\boldsymbol{e})\boldsymbol{w_S} + \boldsymbol{b_S})$$

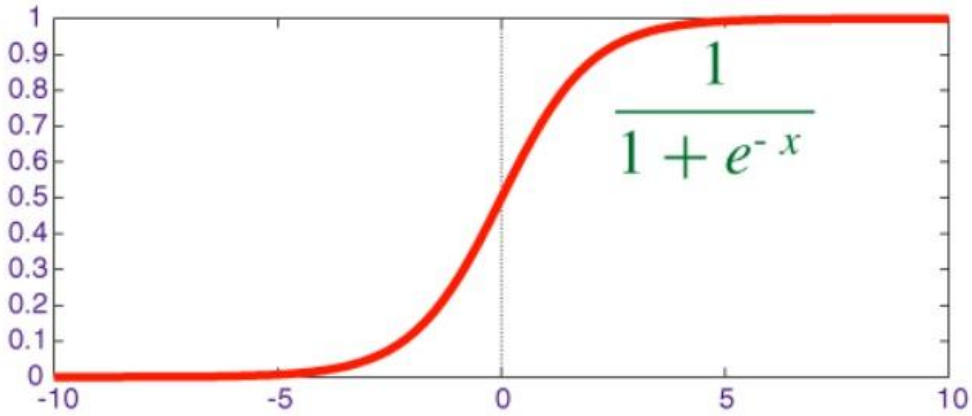

Fig. 6. The sigmoid activation function maps input values to a range between 0 and 1, making it useful for binary classification tasks by converting linear outputs into probabilities

3.7. Loss Function

A loss function is how the progress of a model is rated. Cross-entropy loss, used for classification problems, gives feedback when the model's guesses are off. By reducing this loss, we help the model get better at making accurate predictions. In our case the model is trained by using a binary cross-entropy loss function

$$\boldsymbol{L} = -\sum_{\boldsymbol{i=1}}^{\boldsymbol{m^t}} \mathbf{log}\,(\mathbf{1} - \boldsymbol{f}(\boldsymbol{e_{pos}}, \boldsymbol{i})) + \mathbf{log}\,(\boldsymbol{f}(\boldsymbol{e_{neg}}, \boldsymbol{i}))$$

Which helps differentiate between normal and pseudo-anomalous edges.

3.8. Hyperparameters

- **Number of Contextual Nodes (**k**)**: Determines the number of neighboring nodes in edge-based substructure sampling. Larger k

improves performance but increases computational cost. K = 5 balances efficiency and accuracy.

- **Time Window Size:** This hyperparameter $\boldsymbol{\tau}$ determines the length of the graph sequence the algorithm analyzes along the timeline. For instance, with $\boldsymbol{\tau}$ =3, we utilize 4 snapshots spanning t-3 to t (t,t-3,t-2,t-1). This time window enables capturing the dynamic evolution of node connections over time.
- **Encoding/Embedding Dimension (d):** Specifies the vector dimensionality for node encoding. Smaller d may miss information, while larger d can introduce noise. d = 16 is generally effective.
- **Number of Transformer Layers (L):** Controls the number of layers in the dynamic graph transformer. More layers enhance node interactions, but too many slow down the model. $L = 2$ is typically optimal.
- **Training Ratio:** Refers to the proportion of data used for training. Higher ratios improve performance but increase the risk of overfitting.

3.9. Edge-based substructure sampling

In the context of anomaly edge detection within dynamic graphs, capturing the spatial-temporal context of a target edge is crucial for effective identification. This involves pinpointing not only the target nodes directly connected to the potentially anomalous edge, but also the context nodes surrounding them at various points in time. Prior research suggests that anomalies often reside within local substructures of the graph, prompting our focus on these smaller, highly connected areas.

To efficiently select the most informative context nodes, particularly in large, intricate networks, we leverage a diffusion-based sampling technique. This method contrasts with simpler approaches like H-hop sampling, which selects neighbors based solely on their distance from the target nodes. Diffusion-based sampling, in contrast, grants a global perspective of the graph, allowing us to assess the relative importance of each potential context node in relation to the target edge. This enables a more nuanced understanding of the context surrounding the potential anomaly.

The dynamicity of the graph demands to be refined by one step further. We decompose the dynamic graph in discrete time slices and treat each slice as a static graph. On these static representations, the algorithm relies on a diffusion-based sampling technique to capture temporal evolution of context nodes surrounding the target edge. The process of substructure generation is such that the target edge's spatial-temporal context is well

captured through informative nodes collected from different time slices. This set of nodes will compose the final substructure for the target edge, and its context will be elaborated. This will facilitate the analysis of its potential anomalies**.**

Formally:

Based on the adjacency matrix, **S** is defined as the diffusion matrix graph by summing an infinite series $\boldsymbol{S} = \sum_{\boldsymbol{k=0}}^{\infty} \boldsymbol{Q_k T^k}$.

The value $\boldsymbol{S_{i,j}}$ reflects the relative importance of node **j** to node **I** . A higher value indicates a stronger connection between them, while a lower value indicates a weaker connection. Unlike the adjacency matrix, the values in S are continuous, not just 0 or 1, which helps in understanding the relative importance of neighbors and provides insights into the strength of their relationships.

To ensure the convergence of the infinite series, let's consider some stringent conditions, the sum of $\sum_{\boldsymbol{k=0}}^{\infty} \boldsymbol{Q_k}$ should approach 1, and Q_k should be [0,1]. Additionally, the eigenvalues of **T -** $\boldsymbol{\lambda_i}$ bounded [0,1].

By making specific adjustments to the definitions of **T** and **Q**, we can obtain different versions of graph diffusion that emphasize various types of information. For example, a higher eigenvalue λ will focus on the local neighborhood of the target node.

Given a diffusion matrix **S**, row $\boldsymbol{S_i}$ shows the connection of node i to all other nodes from a global perspective rather than a local one. Each cell $\boldsymbol{S_{i,j}}$ represents the degree of connectivity between **i** and **j** with a continuous value between 0 and 1, allowing for a more precise distinction between nodes compared to other approaches like h-hop method as mentioned before.

By leveraging this approach, the algorithm can select a fixed number of the most relevant and important context nodes for a given target node. For each target edge we can compute the context vector for this edge by summing the context vectors computed for the two endpoint nodes of this edge. Then sorting the context vector and choose the **K** highest nodes, indicating the most significant nodes, to form the set of context nodes $\boldsymbol{U}(\boldsymbol{e_{tgt}})$. The target nodes are not chosen when selecting the K context nodes but are added to the final sample set of the substructure. This results - $\boldsymbol{S}(\boldsymbol{e_{tgt}}) = \{\boldsymbol{v_1}, \boldsymbol{v_2}, \boldsymbol{U}(\boldsymbol{e_{tgt}})\}$ is the final group of sampled nodes in the substructure for the target edge $\boldsymbol{e_{tgt}}$.

Moving from static graphs to dynamic ones, a mechanism is introduced to handle the evolving nature of connections over time. Given a target edge within a time period t, denote a sequence of graphs (snapshots) $G_\tau^t = \{g^{t-\tau+1}, \dots, g^t\}$ of length τ.

For each graph $g^i \in G_\tau^t$, we compute the diffusion matrix S^i and extract its

connectivity vector $s^{i}_{e^{t}_{tgt}}$, selecting the top k nodes and adding the target nodes. Finally, the result of multiple time series t is getting the substructure node set $s(e^{t}_{tgt}) = U^{t}_{i=t-\tau+1}(s^{i}(e^{t}_{tgt}))$.

3.10. Spatial-temporal Node Encoding

Spatial-temporal node encoding is a technique used to represent nodes in dynamic graphs by incorporating both spatial (structural) and temporal (time-based) information. This encoding is crucial for understanding the roles and interactions of nodes in evolving networks, such as social networks, e-commerce platforms, and cybersecurity systems.

Unlike static graphs, which have attributes and data similar to other entities (such as images) with raw features for each element, dynamic graphs often lack inherent attributes. This means it is challenging to find suitable natural data inputs for neural network models. Therefore, the critical question arises: how do we construct informative encodings as inputs for neural networks from dynamic graphs? While there are currently several solutions, they are not the most efficient or effective. For example, a commonly used method is node identity encoding, which works similarly to one-hot encoding in NLP, representing each node with a unique vector. However, this approach has limitations:

- Limited Information: This encoding cannot simultaneously capture spatial/structural and temporal information, representing only the node identity and struggling to reflect its structural role and temporal state in the graph.
- Unsuitable for Large, Dynamic Graphs: Nodes are frequently created and removed, making this method impractical.

The new approach is designed to build a new spatial-temporal node encoding for dynamic graphs. This encoding consists of three components, which are eventually combined into a single vector that serves as the node encoding input containing comprehensive information. These components are**:**

Diffusion-Based Spatial Encoding, Distance-Based Spatial Encoding, Relative Temporal Encoding

The first two encodings represent the structural role of each node from both a global and local perspective, respectively. The temporal encoding provides information about the timing of each element within the substructure. A crucial note: we will use a linear learning function instead of the COS/SIN functions used in the original Transformer, as these are more flexible for learning relationships between different time sequences or locations in the graph.

Now, let's delve into the three encodings:

Diffusion-Based Spatial Encoding

Diffusion provides a global perspective, capturing the global role of a node and understanding its structural function. This information is crucial for the neural network to comprehend the relative importance of each node in relation to the target edge, effectively understanding the strength of the connection between the target edge and each contextual node. To avoid identical encodings for nodes with the same diffusion values, we will use rank-based encoding. For each node in the substructure sampled at a single time point for the target edge, a ranked list based on their diffusion values is created, and each node's rank is used as the basis for its encoding. The diffusion-based spatial encoding is computed using a single-layer linear transformation function which is similar to the positional encoding in Transformers.

Distance-Based Spatial Encoding

The goal is to capture the local role of a node concerning the target edge. Unlike diffusion, which focuses on the global role, local information helps the neural network learn the immediate significance of a node relative to the target edge. For each node in the substructure at a single time point in the target edge's node group, its distance to the target edge serves as the basis for encoding. The distance to the target edge is defined as the minimum of the relative distances to the two nodes comprising the edge, with the distance to the target nodes themselves being zero.

Relative Temporal Encoding

The goal is to capture the temporal information of each node relative to the target edge in the substructure at each time point. Unlike the absolute time encoding in the original Transformer, we propose a relative encoding for dynamic graphs, comparing different time points of a node. The time difference is essential for identifying normal or anomalous target edges in a dynamic graph. For each node in the substructure of the target edge at the current timestamp, the source of the relative temporal encoding is defined as the difference between the time when the target edge occurred (T) and the current timestamp (I).

Fusion the Encodings

The final step is merging the three encodings to obtain a comprehensive and informative picture for each node in the substructure of the target edge's node group. The merged encoding serves as the next input stage for the Transformer. For computational efficiency, the Fusion combine the components using summation instead of concatenation, resulting in a simpler and more efficient calculation.

Ultimately, for each target edge, we compute the encoding of each node in

its substructure's node group and assemble these encodings into a single encoding matrix. This matrix represents the "attributes" of the target edge and serves as the input to the Transformer.

3.11. Dynamic graph transformer

Analyzing dynamic graphs, where connections evolve over time, poses a challenge for neural networks. Capturing both spatial (network structure) and temporal (time-varying changes) information is crucial for accurate anomaly detection. Our proposed approach introduces a novel single transformer encoder architecture to effectively capture both spatial and temporal information simultaneously. This eliminates the need for complex hybrid systems or multiple separate models. The encoder takes as input the nodes embedding vectors for multiple time steps, obtained from the previous stage. It then undergoes two key processes:

- **Transformer Model:** This generates meaningful encoding vectors for each node based on the input, utilizing multiple attention/encoding layers. These layers enable nodes to "communicate" and exchange information, allowing each node to learn to focus on other relevant nodes in the context of the target edge. This captures complex relationships between nodes in both time and space.
  The last layer in the transformer module $\boldsymbol{H^{(l)}}$ acts like a final filter, refining its understanding of each node in the network. The resulting output, called the "output node embedding matrix" ($\mathbf{Z}$), is essentially a collection of "node profiles." Each row in this matrix represents a single node, with the corresponding embedding vector capturing its key characteristics and connections within the network.

- **Pooling Model:** The goal of this section is to transform the encoding matrix Z into a single vector for the target edge $\boldsymbol{z(e_{tgt}^t)}$ **.** This vector represents the overall significant features of the target edge, capturing the information learned from the connected nodes. We employ a pooling method previously used in [14], which involves simply averaging the enhanced encoding vectors of the relevant nodes for the target edge. Mathematically, this is represented as the following function:
  $\boldsymbol{z(e_{tgt}^t) = pooling(Z) = \sum_{k=1}^{n_s} \frac{(Z)_k}{n_s}}$
  This pooling method effectively combines the information from the connected nodes into a single vector, providing a comprehensive representation of the target edge for anomaly detection, the single vector for given edge is $\boldsymbol{z(e_{tgt}^t)}$.

3.12. Discriminative Anomaly Detector

Finding anomalies in dynamic graphs without any pre-labeled examples can be extremely confusing. Here is how this is achieved in our model. First, pseudo-negative edges are created by randomly selecting two nodes from the training data that are not yet connected. If the pair of nodes selected already have a connection, they are discarded, and another pair is selected until an unconnected pair is found. For each of these pseudo-negative edges, the surrounding nodes are examined, and their spatial and temporal information is encoded. These encodings are then fed into a transformer model specifically designed for dynamic graphs, producing an overall encoding vector for each pseudo-negative edge. This vector is run through a neural network with a sigmoid function, resulting in an anomaly score ranging from 0 to 1, where 0 signifies normality and 1 indicates a highly anomalous edge. The loss function (binary cross-entropy loss) treats normal edges as 0 and pseudo-negative edges as 1. The network minimizes the difference between the desired and actual outputs during training to better distinguish normal from anomalous edges. In this manner, anomalies can be detected even without pre-labeled examples. Generating pseudo-negative edges exposes a variety of potential anomalies to the network.

4. Working Process

4.1. TADDY Overview

The TADDY framework for detecting anomalies in dynamic graphs works in a few clear steps using the methods that we explained and show in the previous sections. First, it looks at the local areas around each edge it wants to examine and picks out the most relevant nearby nodes based on how they connect within the graph. Then, it gathers detailed information about each node, including how it's connected to others and how it changes over time. This information is turned into a feature vector for each node. These vectors go into a transformer model that learns to understand patterns in both space (how nodes are connected) and time (how these connections change). The model then pools these learned patterns to create a representation for each edge. Finally, it scores each edge to see how likely it is to be an anomaly. To train the model, both real data and artificially created anomalies are used, so the model learns to tell the difference between normal and abnormal edges effectively.

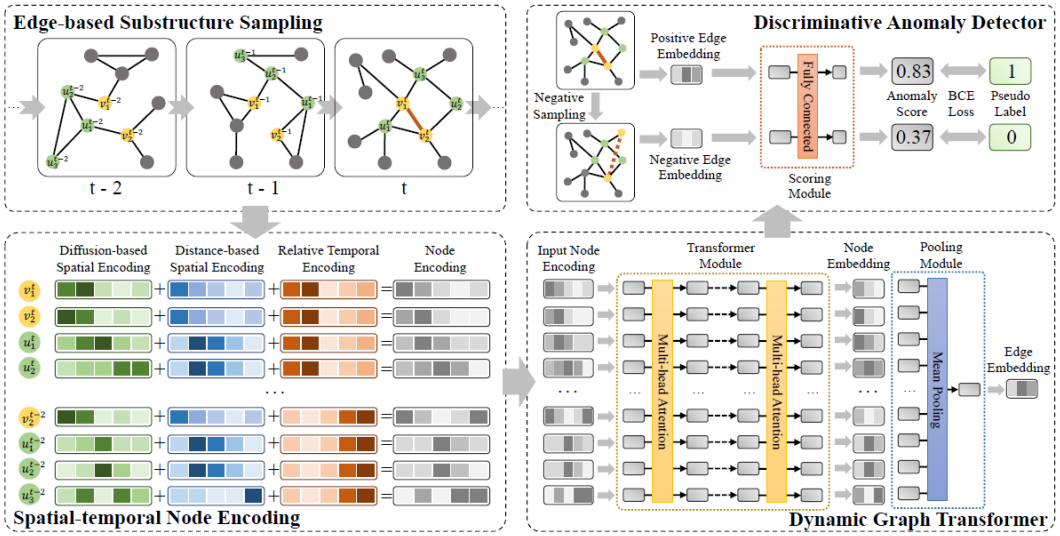

Fig. 7 . . The overall framework of TADDY. The framework is composed of four components: edge-based substructure sampling, spatial-temporal node encoding, dynamic graph transformer, and discriminative anomaly detector.

4.2. Model

To build the citation networks and use the TADDY framework to detect anomalies, first comprehensive dataset of citations is gathered. This dataset is crucial as it forms the basis for constructing the citation networks. The data is preprocessed to extract relevant citation relationships between publications, turning these relationships into nodes (publications) and edges (citations) within a graph structure.

Constructing the Dynamic Graph, Once the static citation network is established, it is converted into a dynamic graph by taking snapshots over a specified time window, this time window as mentioned before calling timestamps, for example timestamps can be chosen as a years, months, days according to the data set. This process will help to captures the temporal evolution of citation patterns. Each snapshot represents the state of the citation network at a specific point in time. By slicing the data into time intervals, a series of static graphs is created, which helps observe changes in citation patterns over time.

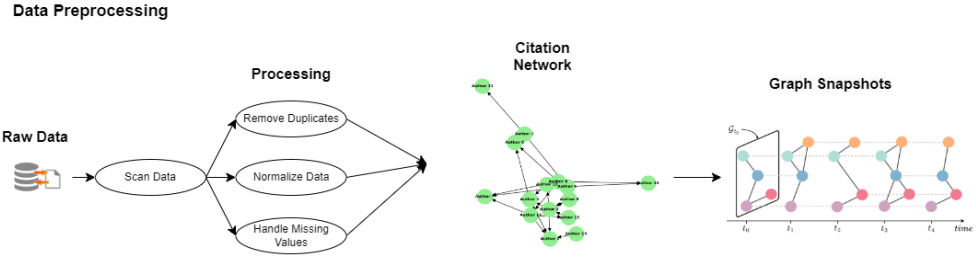

Fig. 7 The preprocessing of the dataset, handle it from the raw data to the suitable data input to TADDY model (original illustration, the last photo source already mention above).

4.3. Edge-Based Substructure Sampling

In the dynamic graph, the focus is on the local neighborhoods around each edge under examination to extract contextual nodes for subsequent analysis. For each citation (target edge), edge-based substructure sampling is used, treating it as the center of the graph, to identify relevant nearby nodes. Relevance is determined using a diffusion method, and detailed information about each relevant node, including its connections, is gathered. The output of this method is a list of contextual nodes specific to the target edge. These nodes will be crucial in the next stage for extracting important information, understanding their level of impact on the target edge, and ranking them accordingly.

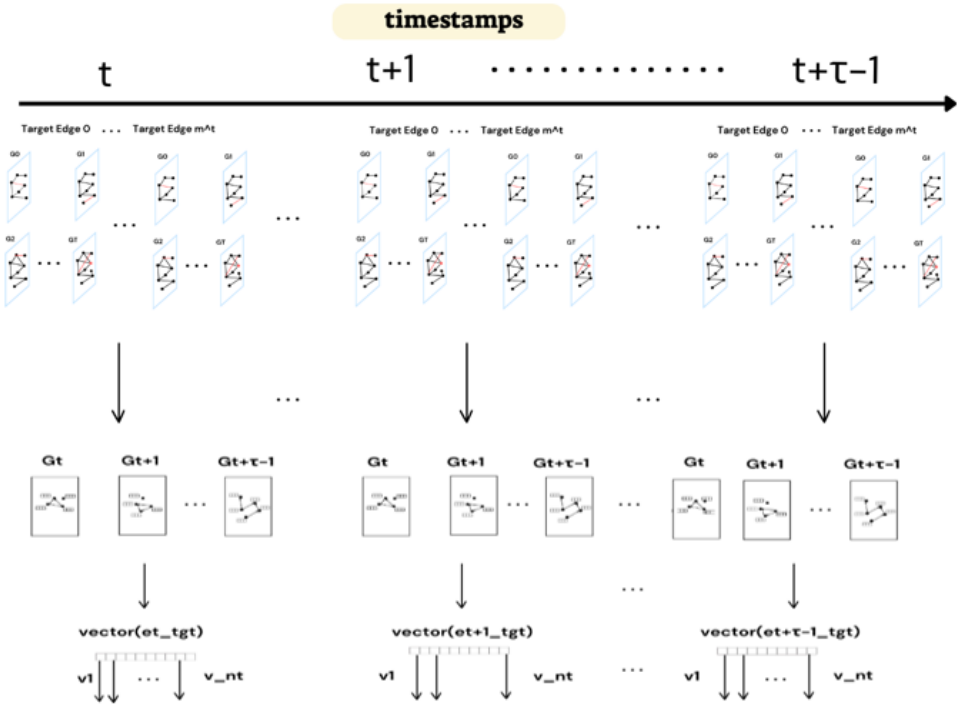

Fig. 8. The figure first shows choosing the target edge for each snapshot in each timestamp, then illustrates a methodology for extracting and encoding contextual nodes for target edges in a dynamic graph across multiple timestamps $G_t, G_{t+1}, \ldots, G_{t+\tau-1}$ For each snapshot, contextual nodes surrounding the target edge are extracted using diffusion-based technique. These nodes are then converted into vector representations where each vector element represents features of the contextual nodes. (original illustration)

4.4. Spatial-Temporal Node Encoding

Next, spatial-temporal node encodings are created by converting node features into feature vectors that capture both spatial (citation relationships) and temporal (evolution of these relationships) characteristics. This process considers both static attributes of the nodes, such as citation counts and publication year, as well as dynamic attributes, like changes in citation counts over time. These features are extracted from the nodes identified in

the previous stage of edge-based substructure sampling. This step ensures that the feature vectors accurately reflect the evolving nature of the nodes' relationships within the citation network, provide to the next step the ability to detection both normal and abnormal behaviors.

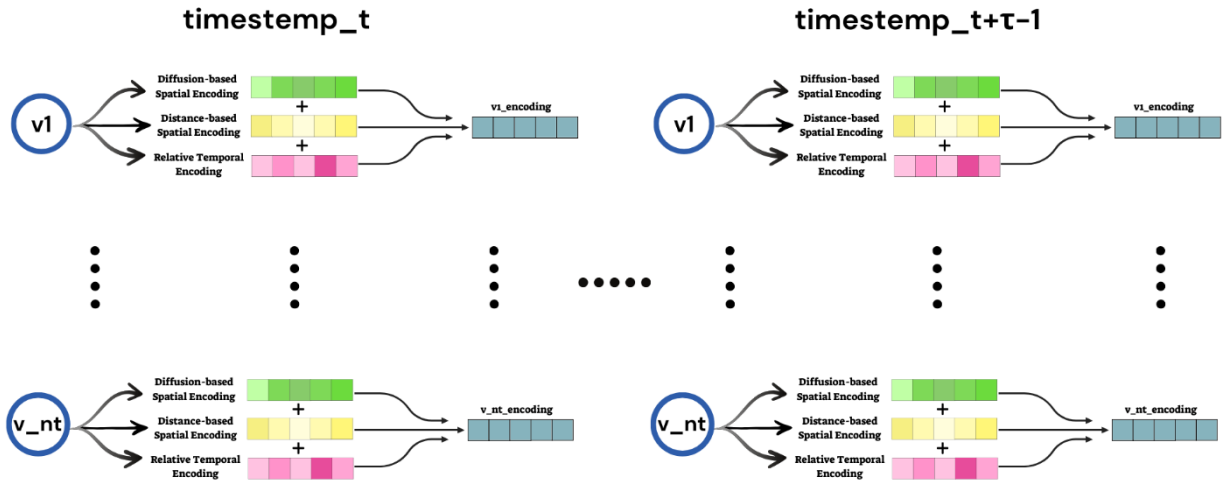

Fig. 9. The process for encoding contextual nodes within the target edge set in a dynamic graph across multiple timestamps For each node in the target edge's contextual node set, encoding is performed using diffusion-based spatial encoding, distance-based spatial encoding, and relative temporal encoding. These encoded features are then fused to create a comprehensive node encoding. This process is repeated for each node at each timestamp (original illustration)

4.5. Dynamic Graph Transformer

Then fed these feature vectors into a dynamic graph transformer, which excels at learning complex, non-linear patterns in the data. The transformer processes the spatial and temporal information together, learning intricate patterns of connectivity and evolution within the dynamic citation graph. Using attention mechanisms, the model assigns different weights to different parts of the input features, allowing it to focus on the most relevant aspects of the citation data. This way, the transformer creates a comprehensive representation for each citation by pooling the learned spatial and temporal features, that let us the power to provide, based on the inputs we received, a particularly concise and informative target edge vector embedding which provides valuable information and can help in predicting and identifying anomalies in the citation network.

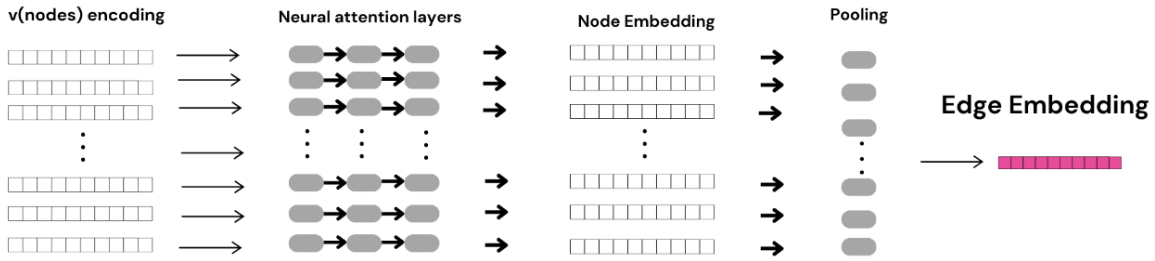

Fig. 10. Capturing cross-domain context over multiple timestamps and aggregating it into a significant vector. These encoded representations are passed through neural attention layers, which highlight the most relevant features. The resulting node embeddings are combined through a pooling mechanism to produce a cohesive summary. The final step generates an edge embedding that integrates the spatial and temporal information from the contextual nodes, providing a comprehensive edge vector representation. (original illustration)

4.6. Discriminative Anomaly Detector

Finally, the model uses a discriminative anomaly detector to assign an anomaly score to each citation, indicating the likelihood of it being an anomaly. This detector evaluates the representations created by the transformer model, comparing them against patterns of normal and anomalous behavior learned during training. Anomalies might include unusual citation patterns that could indicate manipulation or significant deviations from typical citation behaviors.

4.7. Training the Model

Since the dataset is not labeled, a method to train the model to distinguish between normal and abnormal citations is needed. To tackle this, negative sampling is used, where negative edges (citations that do not exist in the original network) are artificially created to serve as anomalies. This method allows the model to learn the difference between valid (positive) citations and anomalous (negative) citations. Both real citation data and these artificially generated negative samples are used in the training process, enhancing the model's ability to identify subtle manipulations and unintentional citation errors within dynamic citation networks.

By combining all stages of the TADDY framework, we aim to develop a robust solution that promotes the integrity and transparency of academic

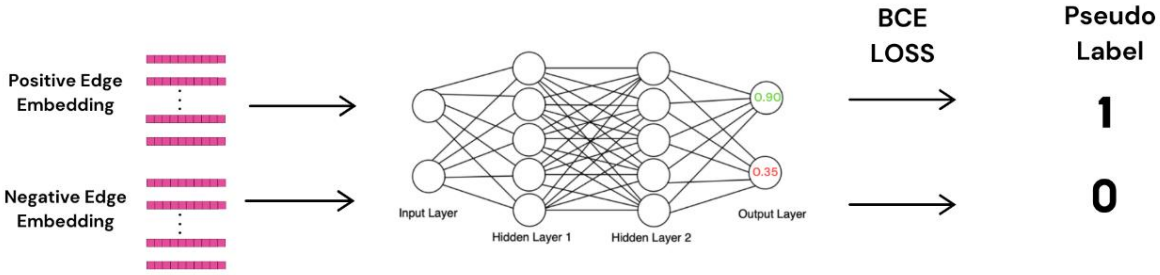

Fig. 11. This figure shows the scoring process for edge embeddings using a neural fully connected network. Positive and negative edge embeddings are input into the network, which consists of two hidden layers and an output layer that assigns scores. These scores are compared against pseudo labels (1 for positive and 0 for negative) using binary cross-entropy (BCE) loss to train the model, helping to identify anomalous edges in the dynamic graph.

research. This comprehensive approach ensures that our model can adapt to the evolving nature of citation networks, effectively distinguishing between genuine anomalies and honest mistakes.

4.8. Summary of Citation Network Anomaly Detection

To detect anomalies in citation networks, by using the TADDY framework employs a multi-step process. A comprehensive dataset of citations is collected and preprocessed to create a static graph representation for each edge in each timestamp, where publications are nodes and citations are edges. This graph is then transformed into a dynamic graph by capturing snapshots at specified intervals (all the timestamps), allowing for the analysis of citation patterns over time.

Edge-based substructure sampling focuses on local neighborhoods around each citation, gathering detailed information about neighboring publications and their temporal changes. These features are converted into spatial-temporal node encodings, capturing both the structure of the citation network and its evolution.

A dynamic graph transformer processes these encodings, learning complex patterns through attention mechanisms to identify crucial information. The resulting representations are evaluated by a discriminative anomaly detector to assign scores to potential anomalies. To enhance model performance, negative sampling is employed during training, generating artificial anomalies to improve the model's ability to distinguish between normal and abnormal citation patterns.

By identifying anomalous citation patterns, we enable to track the evolution of research over time. This includes determining when a research area becomes stable, experiences rapid growth, or undergoes significant shifts in direction. By understanding these dynamics, researchers can gain valuable insights into the trajectory of specific research topics and identify emerging trends.

5. Evaluation and Verification

The evaluation of the TADDY framework emphasizes its capacity to identify unusual connections within dynamic graphs. By leveraging six benchmark datasets, the study introduces anomalies at different rates to thoroughly assess TADDY's performance. The primary evaluation metric

employed is the ROC-AUC score, which serves as an effective measure of the framework's ability to differentiate between normal and anomalous edges. To provide a comprehensive analysis of TADDY's effectiveness, the research involves a comparative study against six established methods: DeepWalk [15], Node2vec [16], Spectral Clustering [17], NetWalk [18], AddGraph [19] , and StrGNN [20]. This benchmarking process is crucial for situating TADDY within the larger landscape of anomaly detection techniques, highlighting its strengths and weaknesses compared to these existing alternatives. Verification of TADDY's performance is conducted by assessing its accuracy in identifying the anomalies that were intentionally introduced. This step not only quantifies the efficacy of the method but also aligns its outcomes with the insights provided by domain experts. By comparing TADDY's results with expert expectations, the study aims to validate the framework's practical applicability in real-world settings. We will try to use this process of evaluation and refinement to contribute enhancing TADDY's performance, ensuring that it meets the demands of accurate anomaly detection in dynamic graph structures for citation networks. In summary, the study employs a thorough methodology that combines quantitative performance metrics, comparative analysis, and domain expert validation to evaluate TADDY's effectiveness.

6. Testing Plan

The testing plan for TADDY aims to evaluate its performance under a range of conditions. The datasets are split into training and testing sets, with anomalies introduced into the test set at varying levels. This approach helps assess how well TADDY performs in different scenarios.

The plan also includes testing TADDY's computational efficiency and scalability to ensure it can handle diverse datasets and anomaly rates effectively. By comparing TADDY's performance with baseline methods across these different conditions, the testing provides a thorough understanding of the framework's strengths and limitations.

We'll demonstrate the unit test cases designed to guarantee the reliability of our model's core functionalities. By leveraging the unit test framework.

Test ID	Description	Function Tested	Expected Result
1.1	Test dataset Loading	Load_dataset	Function successfully loads the dataset.
1.2	Failed Loading the Dataset	Load_dataset	Function failed to loads the dataset.
1.3	Handling dataset	Handle_dataset	Returned preprocessing data that

			matches the expected format.
2.1	Citation Network (graph) Generation	Gen_citation_network	Creating edges and nodes according to the loaded dataset.
2.2	Dynamic Graph Processing	Snap_graph	Taking snapshots of the graph according to the timestamp that was chosen, that will demonstrate a dynamic graph
3.1	Each edge in the snapshots consider as the center of the graph (will call as target edge) and extract relevant nearby nodes	edge-based substructure sampling	Extracting context nodes of each target edge.
3.2.1	Understand the structural role of each node from global perspective.	Diffusion-based Spatial Encoding	Extracting relevant features.
3.2.2	Understand the structural role of each node from local perspective	Distance- based Spatial Encoding	Extracting relevant features.
3.2.3	Understanding the temporal importance of the node structure at specific time.	Relative Temporal Encoding	Provides and extract relevant features and information that depended on time.
3.2.4	Converting node features into feature vectors that capture both spatial and temporal characteristics.	Spatial-Temporal Node Encoding	Successful create node encoding by fussing 3.2.1-3.2.3.
4.1	Getting as input the node encoding and running transformer module and pooling module.	Dynamic graph transformer	Successful create edge embedding vector, for each target edge.
4.2.1	Generate a random anomaly that will help with the training of the model	Negative_Sampling	Successful create random anomalies in the graph (citation network)
4.2.2	Failed to generate a random anomaly	Negative_Sampling	Will try to choose different nodes to create an anomaly edge.
4.3	Testing the module by using the Negative edges and Positive edge, and process the training module	Discriminative anomaly detector	Successfully giving anomaly score and pseudo label them.

7. References

[1] Nur Aiza, W. S., Shuib, L., Idris, N., & Normadhi, N. B. A. (2023). Features, techniques and evaluation in predicting articles' citations: A review from years 2010–2023 Scientometrics. Akadémiai Kiadó, Budapest, Hungary.

[2] Bai, X., Xia, F., Lee, I., Zhang, J., & Ning, Z. (2016). Identifying anomalous citations for objective evaluation of scholarly article impact. *PLOS ONE, 11*(9), e0162364.

[3] Fong EA, Wilhite AW (2017) Authorship and citation manipulation in academic research. PLoS ONE

12(12): e0187394. https://doi.org/10.1371/journal.pone.0187394

[4] Kojaku, S., Livan, G., & Masuda, N. (2021). Detecting anomalous citation groups in journal networks. Scientific Reports, 11(1), 14524.

[5] Zhang, G., Ding, Y., & Milojević, S. (2012). Citation content analysis (CCA): A framework for syntactic and semantic analysis of citation content. arXiv:1211.6321 [cs.DL]

[6] Joshi, P.B., Pandey, M. (2024). Deception Through Manipulated Citations and References as a Growing Problem in Scientific Publishing. In: Joshi, P.B., Churi, P.P., Pandey, M. (eds) Scientific Publishing Ecosystem. Springer, Singapore. https://doi.org/10.1007/978-981-97-4060-4_17

[7] Petch, J., Di, S., & Nelson, W. (2022). Opening the black box: The promise and limitations of explainable machine learning in cardiology. *Journal Name*.

[8] *C. C. Aggarwal, Y. Zhao, and S. Y. Philip, "Outlier detection in graph streams," in ICDE. IEEE, 2011, pp. 399–409.*

[9] W. Yu, W. Cheng, C. C. Aggarwal, K. Zhang, H. Chen, and W. Wang, "NetWalk: A flexible deep embedding approach for anomaly detection in dynamic networks," in SIGKDD, 2018, pp. 2672–2681.

[10] L. Zheng, Z. Li, J. Li, Z. Li, and J. Gao, "AddGraph: Anomaly detection in dynamic graph using attention-based temporal gcn." in IJCAI, 2019, pp. 4419–4425.

[11] Zhang, S., Tong, H., Xu, J. *et al.* Graph convolutional networks: a comprehensive review. *Comput Soc Netw* **6**, 11 (2019). https://doi.org/10.1186/s40649-019-0069-y

[12] Dey, R., & Salem, F. M. (2017). Gate-variants of gated recurrent unit (GRU) neural networks. Circuits, Systems, and Neural Networks (CSANN) LAB, Michigan State University.

[13] Y. Liu et al., "Anomaly Detection in Dynamic Graphs via Transformer," in IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 12, pp. 12081-12094, 1 Dec. 2023, doi: 10.1109/TKDE.2021.3124061.

[14] Y. Liu, Z. Li, S. Pan, C. Gong, C. Zhou, and G. Karypis, "Anomaly detection on attributed networks via contrastive self-supervised learning," IEEE TNNLS, 2021.

[15] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in SIGKDD, 2014, pp. 701–710.

[16] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in SIGKDD, 2016, pp. 855–864.

[17] U. Von Luxburg, "A tutorial on spectral clustering," Statistics and computing, vol. 17, no. 4, pp. 395–416, 2007.

[18] W. Yu, W. Cheng, C. C. Aggarwal, K. Zhang, H. Chen, and W. Wang, "NetWalk: A flexible deep embedding approach for anomaly detection in dynamic networks," in SIGKDD, 2018, pp. 2672–2681.

[19] L. Zheng, Z. Li, J. Li, Z. Li, and J. Gao, "AddGraph: Anomaly detection in dynamic graph using attention-based temporal gcn." in IJCAI, 2019, pp. 4419–4425.

[20] L. Cai, Z. Chen, C. Luo, J. Gui, J. Ni, D. Li, and H. Chen, "Structural temporal graph neural networks for anomaly detection in dynamic graphs," arXiv preprint arXiv:2005.07427, 2020.