תמונה שמכילה גופן, גרפיקה, טקסט, עיצוב גרפי

התיאור נוצר באופן אוטומטי

Software Engineering Department

Braude College of Engineering

Capstone Project Phase A

**25-1-R-3**

**Detecting Citation Anomalies in Hyperbolic Space with the Poincaré Ball Model**

Yotam Gilad- 315819599

Tomer Ben Lolo-318318987

Supervisor: Dr. Dvora Toledano Kitai

<https://github.com/YotamG12/1.-Analyzing-Dynamic-neural-network-graph-within-Hyperbolic-Space.git>

Table Of Contents

[1.Introduction 1](#_Toc189337938)

[2.Theoretical Background 3](#_Toc189337939)

[2.1 Embeddings in Static Networks 3](#_Toc189337940)

[2.1.1 Embedding in Simple Static Networks 3](#_Toc189337941)

[2.1.2 Euclidean Embedding in Complex Static Networks 4](#_Toc189337942)

[2.1.3 Hyperbolic Space 5](#_Toc189337943)

[2.1.4 Hyperbolic Embedding in Complex Static Networks 5](#_Toc189337944)

[2.2 Dynamic Networks 6](#_Toc189337945)

[2.2.1 Embedding in Dynamic Networks 6](#_Toc189337946)

[2.2.2 Dynamic Networks in hyperbolic space 7](#_Toc189337947)

[2.3 Anomaly Detection in Citation Networks 7](#_Toc189337948)

[2.3.1 Isolation Forest 8](#_Toc189337949)

[2.4 Attention Mechanism 9](#_Toc189337950)

[3. Preliminary 9](#_Toc189337951)

[3.1 Problem Formulation 9](#_Toc189337952)

[3.2 Hyperbolic Geometry 10](#_Toc189337953)

[3.3 Feature Maps 12](#_Toc189337954)

[3.4 Sleeping Beauty- 14](#_Toc189337955)

[3.5 Falling Star- 14](#_Toc189337956)

[4. Proposed Model 15](#_Toc189337957)

[4.1 Hyperbolic Structural Attention (HSA) 16](#_Toc189337958)

[4.1.1 Feature Vector 16](#_Toc189337959)

[4.1.2 Embedding Feature Vectors in hyperbolic space 17](#_Toc189337960)

[4.1.3 Transformation into Higher Representation 18](#_Toc189337961)

[4.1.4 Attention Score 20](#_Toc189337962)

[4.1.5 Data Aggregation 21](#_Toc189337963)

[4.1.6 Transition to Euclidean Space 22](#_Toc189337964)

[4.2 Position Embedding 22](#_Toc189337965)

[4.3 Anomaly Detection (IF) 23](#_Toc189337966)

[4.3.1 Preparing the Input 23](#_Toc189337967)

[4.3.2 Model Training and Anomaly score 24](#_Toc189337968)

[4.3.3 Result Analysis 25](#_Toc189337969)

[5.Expected challenges 27](#_Toc189337970)

[6. Evaluation Plan 27](#_Toc189337971)

[7. Expected Results 28](#_Toc189337972)

**Abstract.** The project aims to enhance the integrity and reliability of scientific communication by detecting citation anomalies through hyperbolic geometry, specifically leveraging the Poincaré ball model. Citation anomalies such as misrepresented or irrelevant references undermine the credibility of scholarly work. Traditional citation analysis methods are labor-intensive and susceptible to biases, highlighting the need for automated solutions essential.

To address this challenge, the project introduces a novel approach that utilizes hyperbolic embeddings to capture the hierarchical structures of citation networks. These embeddings effectively model the exponential growth of relationships in scale-free networks, such as scholarly citations. The proposed model, inspired by the DynHAT framework, comprises three key components: Hyperbolic Structural Attention (HSA), Position Embedding, and Anomaly Detection using Isolation Forest. HSA extracts node features and embeds them in hyperbolic space, employing attention mechanisms to emphasize meaningful connections. Position embeddings incorporate temporal context, enabling the model to track shifts in citation behavior over time. Finally, anomaly detection identifies papers exhibiting irregular citation patterns, such as abrupt increases or declines in influence.

The model's performance will be assessed by introducing controlled perturbations into citation networks to evaluate its anomaly detection capabilities. Expected outcomes include the identification of at least 80% of anomalies, and the demonstration of the framework's effectiveness. By integrating hyperbolic geometry with dynamic graph analysis, this research provides a scalable and accurate approach to identifying citation misuse, thereby improving the evaluation of scholarly networks.

# 1.Introduction

Within the interconnected network of scientific literature, citations serve as the foundation of scholarly communication, enabling the validation of ideas, the contextualizing findings, and the tracing of knowledge development. However, citations are not universally reliable. Some may be misrepresented, taken out of context, or rendered inaccurate. These citation anomalies, whether intentional or inadvertent, distort the scientific narrative and undermine the credibility of the academic ecosystem.

Detecting and addressing citation anomalies is a challenging task, particularly given the exponential growth in the volume of published research. Traditional methods for assessing citation accuracy rely primarily on manual review, a resource-intensive process that is also vulnerable to subjective biases. Recent advancements in computational models, particularly in the domain of natural language processing (NLP) and machine learning, present a promising pathway for automating the detection of citation anomalies improving both precision and scalability.

This work introduces an innovative approach for detecting citation anomalies using embeddings in hyperbolic space, specifically utilizing the **Poincaré ball model**. Unlike traditional Euclidean embeddings, hyperbolic ones are particularly effective for modeling hierarchical and relational data, such as citation networks. By embedding scientific articles, their citations, and associated textual content into hyperbolic space, we aim to capture subtle relationships and identify inconsistencies that may indicate citation misuse or misrepresentation.

Through this innovative application of hyperbolic geometry, our approach aims to address the following objectives:

1. To model the hierarchical structure of citation networks efficiently using hyperbolic space.
2. To detect semantic and contextual discrepancies in cited text by comparing with their usage in citing articles.
3. To develop a robust algorithm capable of scaling large datasets while maintaining high accuracy in anomaly detection.

By using the framework of the Poincaré ball model, we aim to advance the field of citation analysis and contribute to the broader objective of enhancing the integrity and reliability of scientific communication. This project outlines the theoretical foundation of our approach, provides a detailed description of the methodology, and discusses the implications of detecting citation anomalies using hyperbolic space representations.

The integration of hyperbolic geometry with citation networks offers a promising approach to uncovering latent hierarchical relationships, detecting anomalies, and deepening the understanding of complex interconnections within scientific literature. Recent studies have explored this intersection, addressing critical challenges in network analysis.

The study [[3]](#kix.xn8bbq7dbxvu) Introduces a dynamic graph embedding model designed to capture both structural and temporal dynamics of evolving graphs, such as social or communication networks. The proposed approach leverages the Lorentz space for dynamic graph embedding combining a structural layer that maps graph topology into Lorentz space with a temporal layer that employs self-attention mechanisms to model time-evolving dependencies. A key advantage of this method lies in its ability to effectively balance structural proximity with temporal dynamics.

The study [[1]](#kix.g0flb11izhsl) investigates the potential of hyperbolic geometry to enhance the optimization and performance of Binary Neural Networks (BNNs). The authors introduce the Hyperbolic Binary Neural Network (HBNN), which leverages the Riemannian exponential map to reformulate the constrained optimization problem in hyperbolic space as an unconstrained problem in Euclidean space. Furthermore, they propose the Exponential Parametrization Cluster (EPC) method, which employs a diffeomorphism to contract the segment domain, thereby increasing the likelihood of weight flips and maximizing gain in BNNs. The hyperbolic framework significantly improves the network's generalization capabilities, particularly for tasks involving hierarchical or complex relationships.

# 2.Theoretical Background

The study of citation anomalies and the structural analysis of citation networks intersect with a range of fields, including graph theory, network science, and computational linguistics. To properly contextualize the use of the Poincaré ball model in hyperbolic space for detecting citation anomalies, it is crucial to understand the evolution of network analysis methodologies and their relevance to this area of study.

## 2.1 Embeddings in Static Networks

### 2.1.1 Embedding in Simple Static Networks

Simple static networks represent relationships in a fixed, unchanging structure, often modeled using graph representations as adjacency matrices or edge lists. These networks are fundamental for understanding pairwise connections, such as those present in co-authorship graphs or direct citation links. Early methods in citation analysis relied primarily on simple static network models, utilizing metrics such as degree centrality, clustering coefficients, and shortest paths to infer influence or detect anomalies.

**Euclidean space** embedding has played a foundational role in network analysis, offering a robust framework for projecting high-dimensional relationships into lower-dimensional spaces. This capability is critical for tasks such as visualization and understanding the structural properties of networks. Notable techniques including node2vec, DeepWalk, and Graph Convolutional Networks (GCNs) utilize Euclidean embeddings to perform various tasks, such as analyzing citation networks, detecting anomalies, classifying nodes, and predicting missing links. A key characteristic of Euclidean spaces is their linear expansion as the distance from a given point increases.

#### 2.1.1.1 DeepWalk-

DeepWalk is an unsupervised learning algorithm designed to generate low-dimensional vector representations (embeddings) for nodes in a graph. By combining local and global structural information, DeepWalk employs truncated random walks to produce sequences of nodes, treating these sequences analogously to sentences in natural language processing. This enables the algorithm to capture both neighborhood-based and broader graph properties effectively.

#### 2.1.1.2 Node2vec-

Node2vec [[7]](#kix.duq3pja7sfl) builds upon the principles of DeepWalk, enhancing the random walk strategy through a tunable trade-off between breadth-first and depth-first search. This flexibility allows Node2Vec to capture a broader range of structural patterns in the graph. Like DeepWalk, it generates low-dimensional embeddings that preserve the relational properties of nodes, making it particularly effective for downstream tasks such as classification, clustering, and link prediction.

While simple static networks are computationally efficient and relatively straightforward to analyze, they fall short in representing dynamic or hierarchical relationships that are prevalent in real-world systems, such as citation networks. These limitations become particularly pronounced when analyzing large-scale datasets, where complex interactions and temporal dynamics play a critical role in shaping the network structure.

### 2.1.2 Euclidean Embedding in Complex Static Networks

To overcome the limitations of simple static networks, researchers have turned to complex static networks that incorporate additional layers of information, such as weighted edges, multi-modal nodes, and hierarchical structures. In citation networks, these models facilitate more nuanced representations, such as assigning edge weights based on citation frequency or differentiating between self-citations and external references. These enhancements allow for a more detailed analysis of relationships and influence within the network, providing greater insight into the structure and dynamics of scholarly communication.

#### 2.1.2.1- Graph Attention Networks

Graph Attention Networks (GAT) [[10]](#kix.sxk5fx5c3tkt) utilize masked self-attentional layers to overcome the limitations of earlier approaches based on graph convolutions or their approximations. By employing a self-attention mechanism, GAT dynamically assigns weights to neighboring nodes, enabling the model to prioritize relevant connections and effectively capture complex relational patterns within the graph.

The self-attention mechanism is a sophisticated approach that our model effectively leverages to enhance its performance and representation capabilities.

#### 2.1.2.2 Graph Convolutional Networks

Graph Convolutional Networks (GCNs) [[11]](#kix.gvq4nc5iq7vp) are a class of neural networks specifically designed to learn from graph-structured data, extending the concept of convolution to non-Euclidean domains. GCNs operate by aggregating information from the local neighborhood of each node, allowing nodes to iteratively update their representation based on their own features and those of their neighbors. This iterative process, known as message passing, is applied across layers, enabling the network to capture both local and global structural patterns in the graph. GCNs are highly effective for a variety of tasks such as node classification, link prediction, and graph-level classification, as they preserve the relationships and attributes inherent to the graph. By leveraging adjacency matrices and node feature vectors, GCNs efficiently encode the structures and attributes of graphs into low-dimensional representations, making them well-suited for downstream tasks that require a nuanced understanding of graph relationships and properties.

These methods assume that the representation space is Euclidean, which limits their ability to accurately model structures with hierarchical complexity or large-scale patterns, as is often the case in social networks, and many dynamic graphs.

Hyperbolic space offers a promising alternative for graph modeling, particularly in complex graphs with hierarchical structures. In recent years, an increasing number of studies have attempted to generalize graph convolution to hyperbolic space.

### 2.1.3 Hyperbolic Space

To effectively model structures that exhibit natural expansion, hyperbolic space, characterized by its constant negative curvature, provides a unique and robust geometric framework. This space is particularly well-suited for representing hierarchical and scale-free structures frequently observed in complex datasets such as citation networks, taxonomies, and social networks. The strength of hyperbolic space lies in its intrinsic property of exponential growth, which aligns closely with the branching patterns of hierarchical systems and networks. In such structures, relationships typically emanate from central, highly influential nodes and extend outward to peripheral, less connected ones.

Unlike Euclidean space, which faces challenges in efficiently representing growth patterns due to its linear and flat geometry, hyperbolic space offers a compact, efficient, and semantically meaningful framework for capturing these intricate relationships. This geometric advantage allows hyperbolic embedding to more effectively preserve distances, proximities, and inherent hierarchical structures within the data. Consequently, hyperbolic embedding enables more accurate modeling of subtle and complex relationships, making them highly suitable for analyzing hierarchical and scale-free systems.

### 2.1.4 Hyperbolic Embedding in Complex Static Networks

#### 2.1.4.1 HGCN

HGCN [[2]](#kix.okhzymkafktx) is a static embedding method that combines the expressive power of Graph Convolutional Networks (GCNs) with the unique properties of hyperbolic geometry to learn node representations for hierarchical and scale-free graphs. This approach leverages the advantages of both GCNs and hyperbolic space, enabling more effective modeling of complex structures inherent in such networks.

#### 2.1.4.2 EvolvedGCN

EvolveGCN [[2]](#kix.okhzymkafktx) is a temporal model that extends the GCN framework by computing a separate GCN model for each time step. The model is dynamically updated with each new input, utilizing a recurrent neural network (RNN), such as a Gated Recurrent Unit (GRU), to capture temporal dependencies.

The notable advantage that hyperbolic space offers for static graphs provides a strong incentive to further explore its application to dynamic graphs, where the structure evolves over time.

## 2.2 Dynamic Networks

Dynamic networks offer a framework for capturing the temporal evolution of relationships by modeling how connections form, dissolve, or evolve over time. In the context of citation analysis, dynamic networks are essential for tracking research trends, identifying emerging fields, and detecting citation anomalies that occur within specific timeframes.

Advanced techniques such as temporal graph networks, dynamic stochastic block models (DSBM), and time-aware node embeddings have been applied to dynamic citation networks. These methods enable the analysis of shifting influence patterns and the detection of inconsistencies. However, their reliance on Euclidean or other static spaces imposes limitations, as these spaces struggle to accurately represent complex hierarchies and hyperbolic-like structures that are intrinsic to citation data.

### 2.2.1 Embedding in Dynamic Networks

Embedding in dynamic networks involves learning low-dimensional representations of nodes, edges, or subgraphs in networks that evolve over time. Unlike static networks, dynamic networks account for changes in topology, node attributes, or edge weights, necessitating embeddings that adapt while preserving temporal consistency. Techniques for dynamic network embedding typically extend static methods by incorporating temporal information through recurrent neural networks (RNNs), temporal attention mechanisms, or snapshot-based models that process sequential graph states. These methods aim to capture both structural and temporal dependencies, facilitating applications such as real-time node classification, link prediction, and anomaly detection. By modeling the temporal evolution of the graph, dynamic embeddings offer a more nuanced and more flexible understanding of network behavior over time.

#### 2.2.1.1 DynamicTriad -

DynamicTriad [[2]](#kix.okhzymkafktx) focuses on the specific structure of triads to model how close triads are formed from open triads in dynamic networks, capturing the evolution of relationships between nodes over time.

#### 2.2.1.2 DySAT -

DySAT [[2]](#kix.okhzymkafktx) utilizes Graph Attention Networks (GAT) as a static layer combined with a self-attention mechanism to capture the temporal evolution of graphs. This approach allows the model to adaptively focus on relevant node interactions and track changes in the network structure over time.

### 2.2.2 Dynamic Networks in hyperbolic space

Dynamic networks in hyperbolic space enhance this capability by integrating temporal dynamics into the embeddings. Recent advancements, such as Hyperbolic Dynamic Graph Neural Networks (HDGNNs) and time-aware hyperbolic embeddings, have demonstrated potential in capturing the interplay between structural hierarchy and temporal evolution. By embedding nodes into hyperbolic space, these methods maintain the integrity of hierarchical relationships while adapting to changes over time, thus offering a robust framework for anomaly detection in citation networks.

**Poincaré embeddings** are well-suited for modeling the tree-like hierarchy in citation networks, where influential papers occupy central nodes, and less-cited works radiate outward.

In contrast to static embeddings, which assume a fixed graph structure, dynamic Poincaré embeddings are designed to adjust to temporal changes in node connections, edge weights, and the network's topology. This process often involves modeling network snapshots at different time intervals or utilizing continuous-time frameworks to capture evolving structural patterns.

## 2.3 Anomaly Detection in Citation Networks

Anomaly detection in citation networks entails identifying irregular patterns or unexpected behaviors, such as papers exhibiting unusual citation trends or outlier connections. Given that Citation networks are hierarchical and scale-free, they are particularly well-suited for analysis in hyperbolic space, where the exponential growth of relationships can be efficiently modeled. Anomalies in these networks may include papers experiencing sudden spikes, potentially signaling breakthrough works, or those with abnormally low citations counts, suggesting overlooked or underappreciated research. Furthermore, connections that significantly deviate from established citation patterns, such as self-citations or citations between unrelated domains, may indicate anomalous behavior. Machine learning methods, including graph neural networks (GNNs) and hyperbolic embedding such as Poincaré embeddings, are well-suited for this task as they effectively capture both the structural and temporal properties of the network. By employing these methods, researchers can uncover emerging trends, detect fraudulent citations, and identify valuable contributions that may have been overlooked, thereby enhancing the understanding and integrity of citation ecosystems.

Common techniques for anomaly detection in graphs include the Local Outlier Factor (LOF), which evaluates anomaly scores by analyzing a node’s local density relative to its neighbors, and Isolation Forest, which isolates anomalies within the feature space.

These methods consider both the attributes of individual nodes and their local context, making them particularly well-suited for heterogeneous networks.

The underlying rationale for these approaches is based on the observation that nodes in a well-structured graph generally conform to expected patterns, whether in terms of their features, relationships with neighboring nodes, or their role within the broader network structure. Anomalies, on the other hand, may appear as nodes with atypical attributes, unexpected connections, or behaviors that deviate significantly from their local or global context.

### 2.3.1 Isolation Forest

Isolation Forest [[6]](#kix.fbvdhqsh3y88) is a widely used anomaly detection algorithm designed to identify outliers by isolating them in the feature space. It operates on the premise that anomalies are data points that are both rare and distinct, making them easier to separate from the rest of the data compared to normal points.

The algorithm constructs a collection of random binary decision trees by recursively splitting data based on randomly chosen features and split values. Due to their distinctive nature, anomalies require fewer splits to be isolated, resulting in shorter path lengths in the decision trees. The anomaly score for each data point is computed as the average path length across all trees, where shorter paths indicate a higher likelihood of being an anomaly.

Isolation Forest is computationally efficient, scales effectively to large datasets, and does not rely on prior assumptions about the underlying data distribution. For graph-based anomaly detection, Isolation Forest can be adapted by applying it to node embeddings or local neighborhood features extracted from the graph, leveraging the graph’s structural properties while maintaining the algorithm’s efficiency.

In hyperbolic space, anomalies can be identified by analyzing points with unexpected radial distances, angular deviations, or atypical hierarchical positioning, assigning them higher anomaly scores to capture their divergence from expected patterns.

## 2.4 Attention Mechanism

Attention mechanisms [[9]](#kix.eb5r20nhmb3x), particularly self-attention, have transformed deep learning by empowering models to selectively focus on the most relevant parts of the input data.

The core of the attention mechanism lies in computing the relevance of different input elements, assigning weights that determine their contribution to the output.

For a given a query , key , and value , the attention score is computed as

where represents the dimensionality of the key vectors.

The Softmax function [[5]](#kix.d70swrpjf8er) ensures that the attention weights are normalized, allowing the model to prioritize elements that are most relevant to the query. This mechanism serves as the foundation for many modern architectures, enabling them to dynamically capture relationships within the data.

Self-attention generalizes the attention mechanism by using the same input to derive the query , key , and value representations through learned projections. This allows the model to effectively capture intra-sequence relationships, making it particularly useful for graph-based and hierarchical data.

# 3. Preliminary

## 3.1 Problem Formulation

In this work, we formally define the problem of dynamic graph representation learning. A dynamic graph is defined as a series of observed static graph snapshots, = where is the number of steps in time. Each snapshot = (, ) is a weighted and directed network snapshot recorded at time ,where is the set of vertices and is the corresponding adjacency matrix at time step . Unlike some previous methods that assume links can only be added in dynamic graphs, we also support removal of links over time. Dynamic graph representation learning aims to learn a mapping function that obtains a low-dimensional representation for each node at time steps . Each node embedding preserves both local graph structures centered at and its temporal evolutionary behaviors such as link connection and removal up to time step .

A notable application of this approach is the **Poincaré Embedding**, which maps hierarchical data—such as academic taxonomies, organizational charts, or citation networks—into a hyperbolic space. This method excels in preserving intrinsic relationships and latent structural patterns inherent to such datasets.

Poincaré embeddings are particularly effective in modeling research domains, where central nodes correspond to highly influential and widely cited papers, while peripheral nodes represent emerging, niche, or specialized works. By maintaining the contextual hierarchy in a natural and interpretable manner, this approach provides valuable insights into the organization and evolution of knowledge within complex networks.

## 3.2 Hyperbolic Geometry

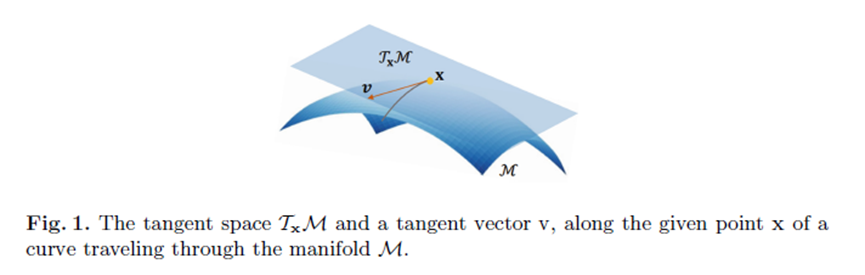
Hyperbolic geometry is a branch of non-Euclidean geometry characterized by constant negative curvature which distinguishes it from the flat nature of Euclidean geometry and the positive curvature of spherical geometry. This unique curvature results in several properties that are fundamentally different from those of Euclidean such as exponential growth of distances. In hyperbolic space, distances between points grow exponentially as they move outward from the origin.

This property makes hyperbolic geometry especially suitable for representing hierarchical, tree, or scale-free structures commonly found in networks, such as citation networks. By leveraging its exponential growth property, hyperbolic space efficiently encodes multi-scale relationships while preserving both local and global structures.

A Riemannian manifold is a generalization of the concept of a curved surface to higher dimensions, providing a mathematical framework for analyzing spaces that may not adhere to the flat structure of Euclidean geometry. This space extends the concept of a two-dimensional surface to a higher dimension. At each point on the Riemannian manifold , there exists a **tangent space**which is a Euclidean space of the same dimensionality as intuitively represents all possible tangential directions passing through point (see Fig. 1).

The tangent space provides a local linear approximation of the manifold, making it possible to apply familiar Euclidean operations such as vector addition and scalar multiplication in a localized context. This is particularly useful for computations, as it simplifies working with the non-linear structure of the manifold.

In the context of hyperbolic geometry, tangent space plays a crucial role in embedding and optimization tasks, such as projecting points from the hyperbolic space to its tangent space for computations and then mapping them back to the manifold.



#### In hyperbolic space, the Riemannian manifold framework enables:

#### Exponential Growth Representation: the metric of hyperbolic manifolds inherently models the exponential expansion observed in hierarchical and tree-like structures.

#### Embedding Techniques: nodes or data points from hierarchical datasets, such as citation networks or taxonomies, can be embedded into hyperbolic manifolds, preserving both local neighborhoods and global hierarchical relationships.

#### Hyperbolic space can be described using several models, such as the **Poincaré** ball model, the hyperboloid model, and the Klein model, all of which are Riemannian manifolds. These models provide different coordinate systems and interpretations for hyperbolic geometry but share the same underlying structure and properties.

#### In this project, we will adopt the Poincaré ball model.

#### **Poincaré Ball Model**

The Poincaré model avoids the distortions associated with representing hierarchical relationships in Euclidean space. By accurately preserving the exponential growth of relationships, it allows for a more faithful representation of citation hierarchies and other scale-free structures. This capability is critical for tasks like anomaly detection, where subtle deviations in structure or position must be identified and analyzed. One of the many applications of the Poincaré ball model is the **Poincaré embedding**, which maps hierarchical data—such as academic taxonomies, organizational charts, or citation networks—into a hyperbolic space. Poincaré embeddings are particularly effective in modeling research domains, where central nodes correspond to highly influential and widely cited papers, while peripheral nodes represent emerging, niche, or specialized works.

**Space Definition:**

The *n*-dimensional Poincaré ball is defined as:

where is the curvature parameter and is the Euclidean norm of the vector .

The factor ​ defines the radius of the ball.

If , it degrades to Euclidean space, i.e.,

**Distance Metric**:

The hyperbolic distance between two points∈, is:

)

: Curvature of the hyperbolic space.

: Euclidean norm

*arcosh* is the inverse hyperbolic cosine function.

This formula captures the hyperbolic nature of space, where distances are distorted relative to Euclidean geometry and grow exponentially as points approach the boundary of the ball .

The curvature parameter scales the geometry, with larger negative values of deepening the hyperbolic space and accentuating its non-Euclidean properties. This exponential growth in distances allows the metric to effectively represent hierarchical and scale-free relationships, making it particularly well-suited for structures with complex or layered connectivity.

## 3.3 Feature Maps

A **feature map** is a transformation process that converts raw input data into a higher-dimensional representation, enhancing its utility for tasks such as classification, regression, clustering or other predictive analyses.

By emphasizing specific patterns or properties, feature maps enable algorithms to extract and leverage meaningful insights that might remain hidden in the original data space.

The primary goal of a feature map is to project data into a more expressive feature space where relationships and patterns become more apparent. This transformation:

1. Highlights significant attributes or interactions within the data.
2. Simplifies the task of identifying correlations, clusters, or separable classes.
3. Enhances the performance of downstream algorithms by providing a representation better suited for the task at hand.

In machine learning, feature maps play a crucial role in various contexts. For example, in Convolutional Neural Networks (CNNs), feature maps represent the output of convolutional layers. Filters within these layers detect and emphasize specific patterns in the input, such as edges, textures, or shapes in images.

Feature maps are critical in graph-based learning frameworks, where they encode graph elements (nodes, edges, or subgraphs) into low-dimensional vector spaces while preserving structural and attribute information. Examples include node2vec, DeepWalk that utilize random walks to generate embeddings that capture node connectivity and positional relationships, and Graph Neural Networks (GNN’s) that employ feature maps to aggregate node attributes and structural information via message-passing mechanisms, enabling tasks like node classification or link prediction.

Feature maps serve as a bridge between raw data and the task-relevant feature space, unlocking the potential for more effective learning and analysis.

**Hyperbolic Embeddings and Feature Maps**

Hyperbolic embeddings utilize feature maps to transform hierarchical or scale-free data into hyperbolic space. This approach provides a more efficient and compact representation of such structures compared to Euclidean space, due to the intrinsic properties of hyperbolic geometry, such as its ability to model exponential growth and hierarchical relationships.

Feature maps in hyperbolic embeddings are designed to emphasize key structural properties, such as hierarchy and scale-free patterns, which are prevalent in many complex datasets. By leveraging the unique properties of hyperbolic space, these feature maps facilitate the encoding of relationships in a manner that aligns with the intrinsic geometry of the data. This alignment enables hyperbolic embeddings to effectively capture and represent hierarchical structures (e.g., tree-like relationships) and networks with power-law distributions (e.g., scale-free graphs). Consequently, hyperbolic embeddings are particularly well-suited for tasks such as hierarchy representation, link prediction, and clustering.

By incorporating feature maps, hyperbolic embeddings achieve a powerful synthesis of structural fidelity and computational efficiency, offering a robust tool for analyzing complex data geometries.

**Exponential and Logarithmic Maps**

To facilitate computations and transformations between hyperbolic space and tangent space, hyperbolic embeddings rely on the exponential map and logarithmic map:

Exponential Map :

* Projects a vector from the tangent space at a reference point back into hyperbolic space.
* This operation is crucial for transferring learned updates or features from the Euclidean tangent space into the hyperbolic manifold.

logarithmic Map :

* Inverse of the exponential map, projecting points in hyperbolic space back onto the tangent space at .
* Enables operations like gradient descent in the Euclidean tangent space, which is computationally more straightforward.

Feature maps in hyperbolic embeddings allow models to alternate seamlessly between hyperbolic and Euclidean spaces. This enables leveraging Euclidean efficiency for computation while preserving the structural integrity of hyperbolic geometry. In citation networks, this approach models the hierarchical importance of papers, positioning influential works near the origin and peripheral ones further out, reflecting their relative significance in the network.

Formally,

Exponential **map definition:**

whereis conformal factor and is Mobius addition,

defined for any as:

![A black rectangular object with blue border

Description automatically generated](data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAAqkAAABbAQMAAABqLY6/AAAABlBMVEX///9Oldnc/2ebAAAAAXRSTlMNPlek2wAAAFxJREFUeF7t2aENwDAQxVCnk7ebdbNGMgxOQCUbfvB0/MbHgZ5rXTb1rsOGxn3o2lhiLZZYiyXWYom1WGItlliLJdZiibVYYi2WWIsl1mKJtVhiLZZYO8SOX72PJ7ZRBPSM992JAAAAAElFTkSuQmCC)logarithmic **map definition:**

## 3.4 Sleeping Beauty-

A “Sleeping Beauty” [[8]](#kix.pc2te8fe6myv) refers to a node in a dynamic network that remains unnoticed or insignificant for an extended period but suddenly becomes highly influential.

This phenomenon often occurs when new discoveries or changing circumstances render the node relevant.

For example, an old research paper might attract significant attention years later if it becomes pertinent to a newly emerging field.

## 3.5 Falling Star-

A “Falling Star” [[4]](#kix.5u4k16njytnv) refers to a node in a dynamic network that begins as highly influential but gradually loses its prominence over time.

This decline can occur due to network evolution, the emergence of more important nodes, or a reduction in the activity or relevance of the original node. For instance, in citation networks, a paper may initially receive significant attention but eventually be overshadowed by newer research.

# 4. Proposed Model

Our model draws inspiration from the DynHAT framework which was originally published in 2022 [[2]](#kix.okhzymkafktx). Building on the principles of DynHAT, we propose an adapted approach tailored for detecting anomalies in article citation networks.

We begin by utilizing a dynamic graph series with

and , based on an input dataset. In this graph series, the nodes represent papers, while the directed edges capture the citations on time *t*.

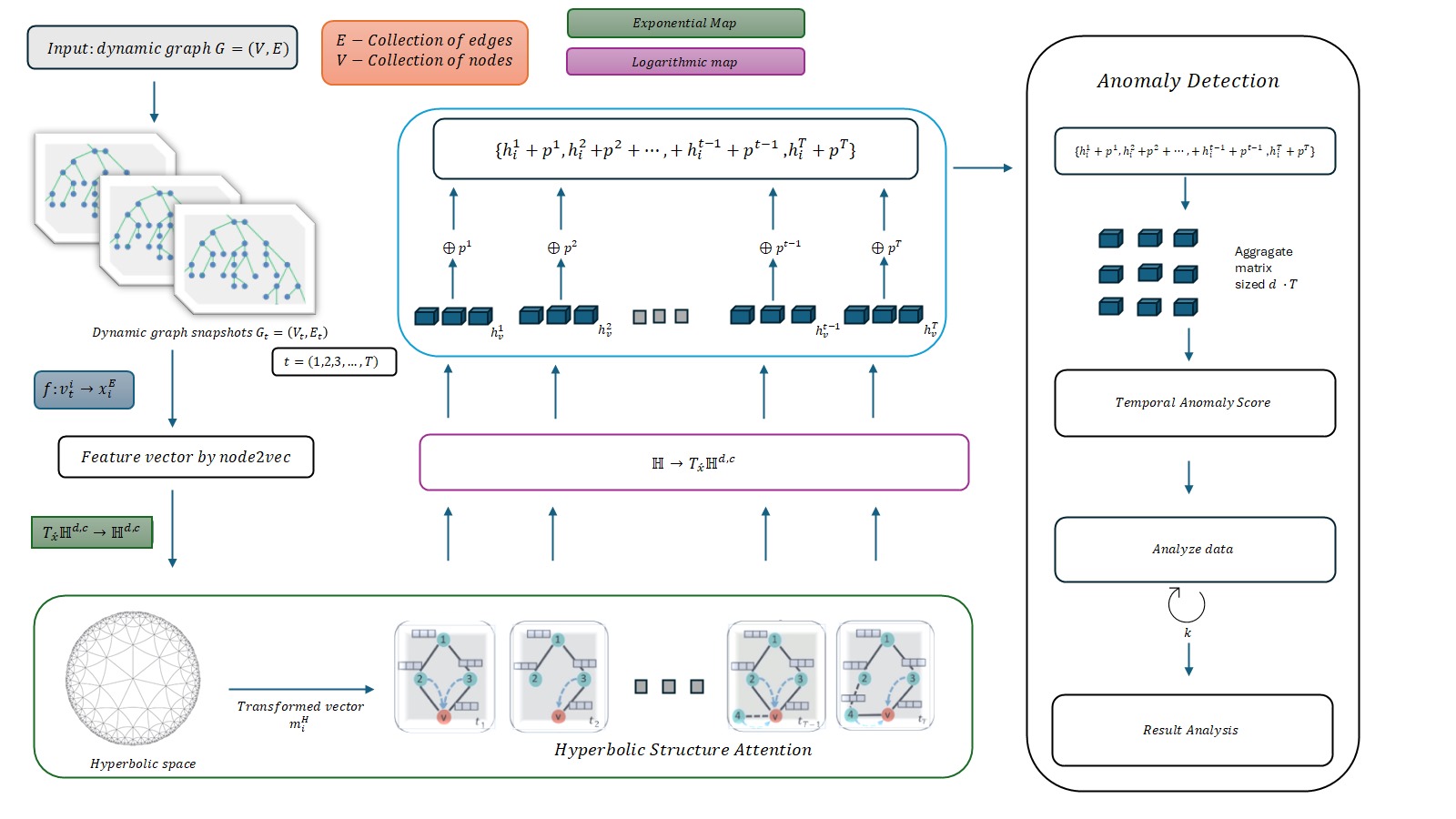
The proposed model is divided into three phases: two modular components followed by an anomaly detection phase.

The first module, Hyperbolic Structural Attention (HSA), processes the dynamic graph series to generate feature vectors for each node. These feature vectors are then projected into hyperbolic space, where node embeddings are aggregated using a self-attention mechanism based on hyperbolic proximity.

The second module, Position Embedding, takes the sequences of node representation from the HSA module as input. Leveraging the advantages of self- attention, this module fuses the final embedding by assessing the significance of each temporal graph snapshot. The output of this module is the set of final dynamic node representations.

In the final phase, the model is trained to detect the anomalies in nodes. This phase identifies papers with significant changes in citation patterns, such as those experiencing a decline in citations or a sudden increase after a period of inactivity.

Fig.2



The details of each phase are elaborated in the following paragraphs.

## 4.1 Hyperbolic Structural Attention (HSA)

The input of this module is a dynamic graph composed of a series of snapshots

### 4.1.1 Feature Vector

To construct a feature vector from the dynamic graph using Node2Vec [[7]](#kix.duq3pja7sfl), each snapshot ​, is treated as a static graph. Node2Vec is then applied individually to each snapshot to generate node embeddings. For each ​, the input includes the graph structure (nodes and edges), the number of random walks per node, the walk length, and the hyperparameters (return parameter) and (in-out parameter) to control the bias of the random walks.

Node2Vec produces an embedding for each node, forming a feature vector for the snapshot​​. To capture the temporal dynamics of the graph, these feature vectors ​​ can be aggregated across snapshots using techniques such as concatenation, averaging, or a recurrent neural network. This methodology effectively integrates both structural and temporal information from the dynamic graph into the resulting feature vector.

To construct the feature vector, we employ the Node2Vec algorithm.

The procedure for applying Node2Vec to dynamic graphs is detailed below:

**Algorithm 1: Creating a Feature Vector**

Input:

A dynamic graph

Node2Vec hyperparameters

Output: A feature vector ;

1: For each snapshot , perform biased random walks using (return parameter to control backtracking) and (in-out parameter for exploration vs. exploitation).

2: Train a skip-gram model on the generated sequences to learn node embeddings.

3: Compute difference features capturing changes between and.

4: Use embeddings from earlier snapshots to initialize later ones.

5: Aggregate embeddings across time using concatenation, averaging, or recurrent models.

6: Return the final feature vector capturing structural and temporal dynamics.

### 4.1.2 Embedding Feature Vectors in hyperbolic space

To map the feature vector to hyperbolic space via exponential map, we consider the case where the reference point is .

In this case, the formula for the exponential map is given by



The feature vector is projected to hyperbolic space via exponential map .

This feature mapping is essential for preserving the information from the Euclidean space and transferring it to hyperbolic space, where the subsequent aggregation and processing can take place.

To represent the vectors more accurately and to capture their relations with other nodes, we transform these vectors into space that reflects each node's role and significance within the network.

**Algorithm 2: Mapping Feature Vectors from Euclidean to Hyperbolic Space**

Input: Feature vector for each node, curvature constant.

Output: Hyperbolic feature vector.

1: For each feature vector :

2: Compute Euclidean norm .

3: If |:

4: Project to hyperbolic space: =;

5: Return Hyperbolic feature vector for each node.

### 4.1.3 Transformation into Higher Representation

The vector undergoes a transformation to enhance the information regarding relationships and the structure of the graph. This transformation is represented by:



where is the weighted matrix, represents the multiplication operation in hyperbolic space, represents the sum operation in hyperbolic space called mobius addition and *b* is a bias vector in the tangent space, providing flexibility to the model.

This transformation consists of two main operations:

1. **Multiplication in Hyperbolic Space:**

The operation is carried out using an exponential map in hyperbolic space:



=

1. **Mobius Addition:**

The second operation, Mobius addition, is given by:

The multiplication in hyperbolic space consists of three steps:

1. **Projection into the Tangent Space**:

First, the vector is projected into the tangent space via the logarithmic map: ([Fig.3](#kix.8hu8edjvz5r2)) where



(10)

This projection preserves the hyperbolic distance and captures the local structure of the node.

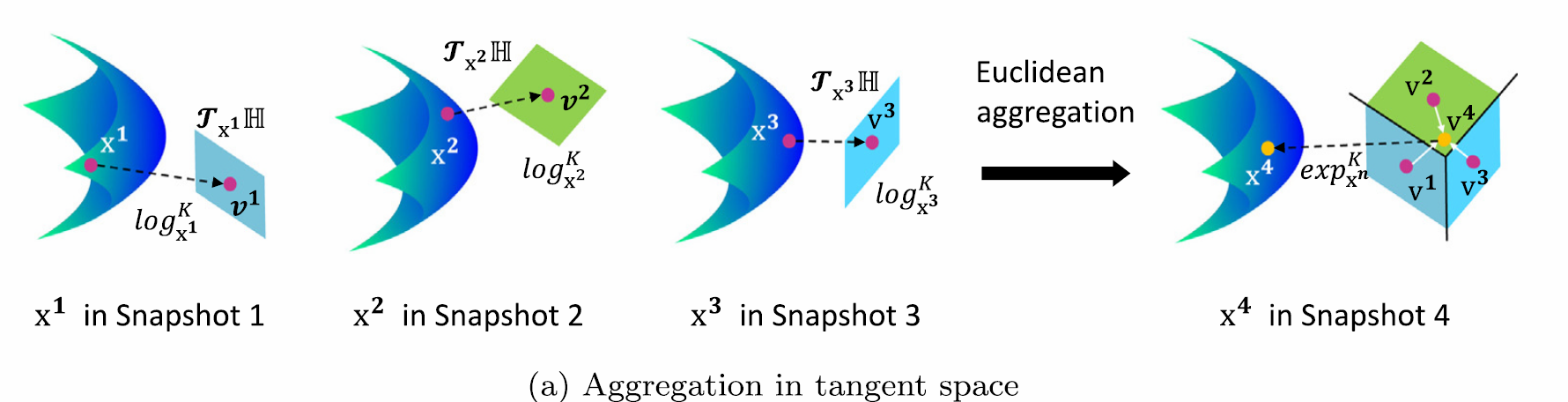


Fig.3

1. **Linear Transformation**:

Next, a linear transformation [[11]](#kix.gvq4nc5iq7vp) by multiplying the weight matrix with

The linear transformation maps one vector space to another while preserving vector addition and scalar multiplication, thereby enhancing the structural features of each node.

1. **Projection Back into Hyperbolic Space**:

Finally, the improved vector is mapped back into hyperbolic space ([Fig.4](#kix.u7gs18ofexa3)) using the exponential mappreserving the hyperbolic structure:**(***W***)**

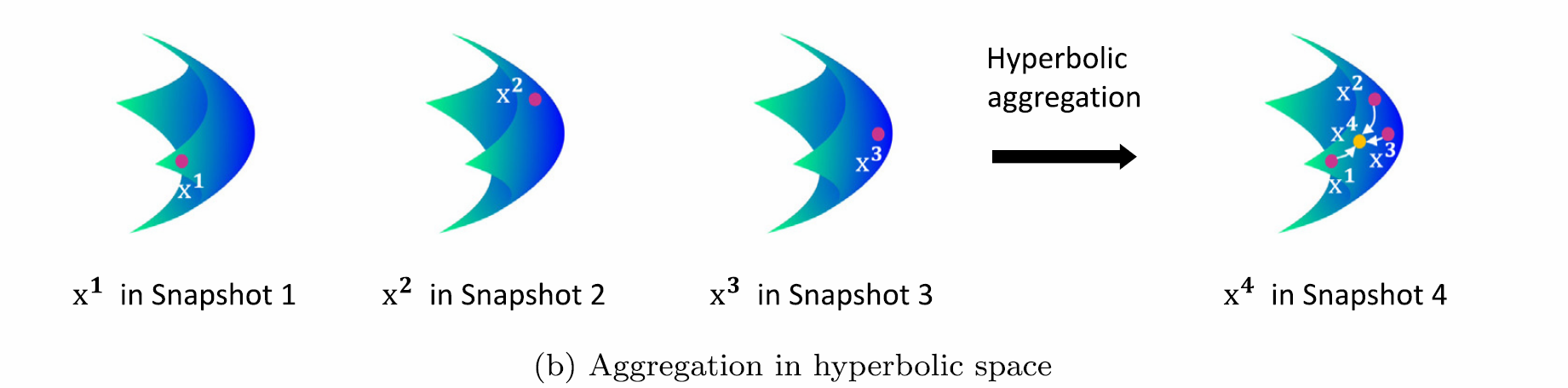


Fig.4

The multiplication in hyperbolic space results in an enhanced vector that retains its hyperbolic properties allowing it to integrate seamlessly into the next layers of the model which depends on the hyperbolic structure of the neural network.

Due to the addition rules in hyperbolic geometry, the mobius addition operation requires the use of the matched weight matrix *W*. This operation involves multiplying the weight matrix *W* with the vector , and then applying the Mobius addition to integrate the bias vector *b* in the tangent space. This ensures the transformation respects the geometry of the hyperbolic space while adjusting for the structural features of the node.

To evaluate the relationships between two nodes, we use a hyperbolic self- attention mechanism. In this work, self-attention serves several key purposes:

1. Model Relationships in Citation Networks: Self-attention enables the model to focus on neighboring nodes and their embeddings, effectively capturing the complex structural relationships within the network.
2. Incorporate textual content: By integrating the textual content of papers, self-attention helps identify semantic anomalies in citations, such as unexpected references or thematic inconsistencies.
3. Enhance hyperbolic embedding: The self-attention mechanism selectively attends to nodes that are structurally and semantically relevant, thereby improving the expressiveness of hyperbolic embeddings, which are particularly suited for representing hierarchical and scale-free structures.

This approach ensures that both the structural and semantic properties of the citation network are effectively modeled, facilitating robust analysis and anomaly detection.

**Algorithm 3: Transformation into Higher Hyperbolic Representation and Self-Attention**

Input: Hyperbolic feature vector , weighted matrix , bias vector , curvature constant .

Output: Higher-level hyperbolic representation.

1: For each :

2: Compute projection into the tangent space:

3: ) = ensuring .

4: Apply linear transformation in the tangent space:

5: = .

6: Project back into hyperbolic space using the exponential map:

7: = **()**.

8: Return Higher-level hyperbolic representation .

### 4.1.4 Attention Score

Given hyperbolic transformed vectors ,, we employ a weighing mechanism to compute the relationship between two entities and .

The relationship score is calculated using a sigmoid function,

which ensures that the attention scores are lie within a range suitable for comparison:



where is a learnable factor that computes a scalar score from the concatenated transformed vectors ||.

The self-attention function is computed as:



where is set of neighboring nodes of node , ensuring that the attention weights sum to 1. This normalization allows the model to emphasize the most relevant neighbors while maintaining global consistency within the network.

**Algorithm 4: Computing Attention Scores in Hyperbolic Space**

Input: Transformed hyperbolic vectors, for nodes , learnable weight vector , curvature constant , neighborhood set , sigmoid function.

Output: Attention scores and normalized attention weights .

1: For each pair of nodes where :

2: Project vectors into tangent space:

3: ,.

4: Concatenate tangent space representations:

5: .

6: Compute relationship score:

7: .

8: For each node :

9: Compute SoftMax-normalized attention weights:

10: .

11: Ensure normalization condition:

12: .

13: Return Attention scores and normalized attention weights ;

### 4.1.5 Data Aggregation

To effectively capture both the local and hierarchical relationships among entities the following steps are taken:

1. The hyperbolic transformed vector is first computed in the tangent space using the function.
2. The result is then transformed back to hyperbolic space using the function.

Next, by multiplying the computed attention weights , we obtain the aggregated representation which represents the combined information of node *i* and its neighbors in hyperbolic space. This aggregation respects the underlying geometry of the data, effectively incorporating both local and hierarchical relationships.

### 4.1.6 Transition to Euclidean Space

Transitioning to Euclidean space simplifies computations and ensures compatibility with downstream algorithms. This transformation is achieved using the logarithmic map:



By Incorporating the aggregated representation at time step ,the model gains the ability to encode dynamic changes in the network or data over time.

The resulting representations are then fed into the ETA Model 2 for further processing, enabling the dynamic analysis of the dynamic network's structure over time.

**Algorithm 5: Transitioning Aggregated Hyperbolic Representations to Euclidean Space**

Input: Aggregated hyperbolic representation for each node at time step , curvature constant

Output: Euclidean representation and processed output from Position Embedding Model 2.

1: For each node at time step :

2: Apply the logarithmic map to project into the tangent (Euclidean) space:

3: .

4: Encode temporal dynamics by incorporating time step :

5: = ;

6: Pass the Euclidean representation to Position Embedding Model 2:

7: = ().

8: Return Euclidean representation and processed output from Position

Embedding Model 2.

## 4.2 Position Embedding

Position Embedding addresses the challenge of the model’s inherent lack of awareness regarding the sequential order of nodes. By encoding positional information into node representations, this technique enables the model to distinguish between different positions within a sequence.

In the proposed model, we take the Euclidean representation and add the corresponding position embedding to each node at time *t*, resulting in:

.

The temporal ordering enables the model to understand the position of each node throughout the timeline.

The outcome is a set of temporal representations that combine local structure in Euclidean space with the dynamic changes over time.

**Algorithm 6: Position Embedding in Temporal Modeling**

Input: Euclidean representations for each node at time steps , position embedding vector for each *t* ∈}.

Output: Temporal sequences with position embeddings for each node.

1: For each node :

2: For each time step :

3: Add position embedding:

4: = + ;

5: Form temporal sequence for each node:

6: = ;

7: Return temporal sequences with position embeddings ;

## 4.3 Anomaly Detection (IF)

### 4.3.1 Preparing the Input

Given the resulted positioned output , those representations are aggregated into a matrix of size where is the number of nodes in the graph, is the dimensionality of the embedding vector , and is the number of time steps.

The matrix is then flattened along the temporal dimension to construct a comprehensive feature vector ​ for each node in the size of :



​

To ensure the model's robustness and comparability of features, a normalization technique such as Min-Max Scaling is applied, making the data suitable for anomaly detection tasks such as Isolation Forests.

**Algorithm 7: Preparing Input for Anomaly Detection using Isolation Forests (IF)**

Input: Temporal position-enhanced representations

for all nodes *v*,

Dimensionality of embedding vector ,

number of time steps , number of nodes .

Output: Normalized feature vectors ∈ for all nodes , ready for anomaly detection.

1: Construct an aggregated matrix of size :

2: = , where }.

3: For each node :

4: Flatten the temporal dimension of the representation matrix:

5: , where .

6: Normalize feature vectors using Min-Max Scaling:

7: .

8: Return Normalized feature vectors ∈ for all nodes , ready for Isolation Forests.

### 4.3.2 Model Training and Anomaly score

Selecting an appropriate model training and anomaly detection framework depends on the dataset’s characteristics, including its size, dimensionality, and computational constraints. This selection is critical to ensure both efficiency and accuracy in the analysis. Below, we outline three widely used tools for anomaly detection, highlighting their features and suitability for different scenarios:

#### 4.3.2.1 Scikit-Learn

Scikit-Learn is a versatile and user-friendly machine learning library in Python. It provides a comprehensive suite of tools for supervised and unsupervised learning, data preprocessing, model selection, and evaluation. Notably, it implements algorithms such as Isolation Forest, One-Class SVM, and various clustering methods, making it highly effective for small to medium-sized datasets and rapid prototyping. Scikit-Learn’s popularity stems from its simplicity, ease of use and seamless integration with Python’s data science stack, including libraries like NumPy and Pandas.

#### 4.3.2.2 H2O.ai

H2O.ai is an open-source platform designed for scalable and distributed machine learning. It supports a wide range of algorithms, including Isolation Forest, Autoencoders, and Deep Learning models, catering to both data scientists and business users. Its ability to efficiently process large datasets makes it a robust choice for anomaly detection in high-volume contexts. Additionally, H2O.ai offers AutoML capabilities, enabling automated model training and selection. The platform integrates seamlessly with Python, R, and enterprise environments, further enhancing its flexibility and usability.

#### 4.3.2.3 PySpark

PySpark is the Python API for Apache Spark, a distributed computing framework tailored for large-scale data processing. It is particularly effective for handling big data and offers libraries such as MLlib for machine learning, alongside support for streaming and SQL operations. While PySpark does not natively implementIsolation Forest, third-party libraries like SynapseML can be employed for this purpose. PySpark is an excellent choice for scalable anomaly detection and machine learning tasks involving massive datasets.

### 4.3.3 Result Analysis

#### 4.3.3.1 Representation of Anomaly Scores

To facilitate the analysis of the anomaly scores for Autonomous System (AS) nodes over time, a flexible data structure such as a Pandas DataFrame, will be utilized. Each row in the DataFrame corresponds to a single data point and includes the following columns:

* Node ID (AS): A unique identifier assigned to each node.
* Timestamp: The precise time at which the anomaly score was recorded.
* Anomaly Score: The calculated anomaly score for the node at the given timestamp.

This structured approach enables efficient data manipulation, filtering, and visualization, providing a robust foundation for comprehensive analysis.

#### 4.3.3.2 Basic Statistical Analysis

To identify patterns and potential anomalies, we will compute fundamental statistical measures for each node. The Key metrics are as follows:

* Average Anomaly Score per Node: The mean anomaly score for each node across all temporal snapshots.
* Variance of Anomaly Scores: The extent of fluctuation in anomaly scores for each node over time. Nodes exhibiting high variance will be flagged for further examination as they might indicate irregular behavior.
* Comparison with Network-Wide Averages: Each node's average anomaly score will be compared against the overall network-wide average anomaly score to detect outliers.

Nodes that consistently deviate from the network average or demonstrate abnormally high variance will be considered as potential candidates for further in-depth analysis.

#### 4.3.3.3 Normalization

To ensure consistency and comparability across varying scales of anomaly scores, a normalization process can be applied. Specifically, min-max normalization can be used to rescale scores to a standardized range of 0 to 1.

This normalization is essential for enabling meaningful comparisons and between nodes and facilitating clear and effective visualizations.

#### 4.3.3.4 Visualization of Anomaly Scores

To derive insight into the temporal behavior of nodes, several visualization techniques can be employed:

* Heatmaps: Anomaly scores of nodes over time will be represented, with nodes as rows and timestamps as columns. This approach provides a comprehensive overview of patterns and potential outliers.
* Scatter Plots: Anomaly scores will be plotted against time for individual nodes or selected node groups, enabling a detailed examination of specific temporal trends.
* Line Graphs: The mean anomaly score for the entire network, as well as for nodes, will be displayed over time, to highlight deviations and emerging trends.

These visualization techniques can facilitate an intuitive and systematic identification of nodes exhibiting anomalous behavior and significant temporal trends.

#### 4.3.3.5 Temporal Change Analysis

To understand how anomaly scores evolved over time, the following analyses were conducted:

* Snapshot-to-Snapshot Changes:
  + The change in anomaly scores between consecutive snapshots for each node will be calculated.
  + Nodes with significant changes will be identified, potentially indicating potential incidents or noteworthy events.
* Aggregate Metrics:
  + The average and maximum changes in anomaly scores for each node across all snapshots will be computed.
  + Nodes exhibiting consistently high changes will be flagged for further investigation.

#### 4.3.3.6 Identification of Extreme Cases

Nodes will be identified based on the following criteria:

* Highest Anomaly Scores: Nodes that consistently scored significantly above the network average will be flagged.
* Largest Temporal Changes: Nodes that exhibited the most significant score fluctuations between consecutive snapshots will be highlighted.

# 5.Expected challenges

Several challenges are anticipated in the implementation of this project, , each of which plays a crucial role in ensuring the accuracy and efficiency of the proposed model.

1. **Dataset Selection:** Identifying the most suitable dataset for the model is a fundamental challenge. The dataset must be representative of real-world citation networks while containing enough variability to allow for meaningful anomaly detection. Additionally, ensuring data quality and handling missing or incomplete citations pose further complexities.
2. **Computational Constraints:** Given the potentially large size of citation networks, selecting appropriate computational tools and frameworks is essential. Efficiently processing high-dimensional hyperbolic embeddings and large-scale temporal graphs requires significant computational resources, making scalability a key concern.
3. **Parameter Optimization:** Determining optimal hyperparameters, such as the length of time steps, the structure of attention mechanisms, and the threshold for anomaly detection, is critical for maximizing accuracy. This process requires extensive experimentation and validation to balance sensitivity and specificity in detecting citation anomalies.

Through an iterative process of trial and error, we aim to refine and adjust these configurations to ensure the development of a robust and reliable model capable of effectively detecting citation anomalies.

# 6. Evaluation Plan

The proposed model’s validity and reliability will be assessed by introducing artificial noise into the initial network, creating a controlled environment to evaluate sensitivity to structural changes. This validation strategy involves injecting noise by adding random nodes, either isolated or connected to existing nodes. Such alterations may disrupt the network's inherent structure, generating unexpected anomalies.

The validation process evaluates the model's ability to detect and interpret these changes by comparing anomaly scores before and after the noise injection. A specific focus will be on assessing how the model identifies significant deviations, particularly in highly connected or previously stable nodes that transition into anomalous states post-intervention.

The hypothesis underlying this approach assumes that the model will effectively detect and score anomalies arising from structural disruptions, thereby demonstrating its robustness in identifying impactful changes within dynamic networks. This methodology provides a quantitative measure of the model's effectiveness, offering insights into its ability to capture and interpret unexpected patterns caused by noise.

# 7. Expected Results

The proposed model aims to achieve robust and precise anomaly detection in dynamic citation networks through the integration of hyperbolic geometry, self-attention mechanisms, and temporal embedding.

The expected results should highlight the model's capacity to effectively analyze and interpret dynamic citation networks for anomaly detection, demonstrating its contribution to advancing graph-based anomaly detection methodologies.

The proposed model is expected to achieve at least 80% detection rate for anomalies within dynamic citation graphs, effectively demonstrating its capability to identify and capture significant deviations within the network’s evolving structure over time.

**Bibliography**

**[1]** Chen, J., Xiang, J., Huang, T., Zhao, X., & Liu, Y. (2025, January 7). Hyperbolic binary neural network. arXiv.org.<https://arxiv.org/abs/2501.03471>

**[2]** Duan, D., Zha, D., Yang, X., Mu, N., Shen, J. (2022). Dynamic Network Embedding in Hyperbolic Space via Self-attention. In: Di Noia, T., Ko, IY., Schedl, M., Ardito, C. (eds) Web Engineering. ICWE 2022. Lecture Notes in Computer Science, vol 13362. Springer, Cham.<https://doi.org/10.1007/978-3-031-09917-5_13>

**[3]** Duan, D., Zha, D., Liu, Z., & Chen, Y. (2024). Dynamic Graph Embedding via Self-Attention in the Lorentz Space. Proceedings of the 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Tianjin, China, pp. 199–204.

**[4]** Fire, M., & Guestrin, C. (2018, October 13). *The rise and fall of network stars: Analyzing 2.5 million graphs to reveal how high-degree vertices emerge over time*. arXiv.org. <https://arxiv.org/abs/1706.06690>

**[5]** GeeksforGeeks. (2024, November 19). *Softmax activation function in neural networks*. <https://www.geeksforgeeks.org/the-role-of-softmax-in-neural-networks-detailed-explanation-and-applications/>

**[6]** GeeksforGeeks. (2024, July 15). What is Isolation Forest? GeeksforGeeks.<https://www.geeksforgeeks.org/what-is-isolation-forest/>

**[7]** Grover, A., & Leskovec, J. (2016). Node2vec. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 855–864.<https://arxiv.org/pdf/1607.00653>

**[8]** Ke, Q., Ferrara, E., Radicchi, F., & Flammini, A. (2015). Defining and identifying Sleeping Beauties in science. Proceedings of the National Academy of Sciences of the United States of America, 112(24), 7426–7431.<https://www.jstor.org/stable/26463784>

**[9]** Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023, August 2). Attention is all you need. arXiv.org.<https://arxiv.org/abs/1706.03762>

**[10]** Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018, February 4). *Graph attention networks*. arXiv.org. <https://arxiv.org/abs/1710.10903>

**[11]** Zhang, R. (2023, December 17). A journey into linear algebra: Exploring linear transformations. Medium.<https://rendazhang.medium.com/a-journey-into-linear-algebra-exploring-linear-transformations-5d8b00f36be8>