ISSN: 0067-2904

Construction of a Graph of Similar Trajectories Based on the Minimum Spanning Tree (MST) and Node Similarity Function (NSF)

Alyaa T. Muhi*, Tawfiq A. Al-Assadi
Department of Computer Science, College of Computer Science & Information Technology, University of Babylon, Iraq

Received: 2/12/2023 Accepted: 2/10/2024 Published: 30/10/2025

Abstract

In recent years, the graph-based approach has become increasingly popular for modeling real-world data because graphs represent a wide variety of data, such as social networks and trajectory mining. In this paper, we propose a Weighted Adjacency Matrix (WAM) for constructing a graph based on edges from the features of each trajectory and a vertex from the frame number. In trajectory mining, finding objects that have similar patterns of motion is a common data analysis; thus, selecting effective algorithms is necessary for grouping information and showing trajectories as graphs. A new algorithm is proposed to calculate Graph Trajectory Similarity (GTS) between two trajectories of graphs, the Nodes Similarity Function (NSF). The NSF is introduced in our model for comparing similarity between two trajectories of graphs. A minimum spanning tree (MST) is a new schema used to prune edges and nodes of a graph.

Keywords: Computer Vision (CV), Video tracking (VT), Trajectories, Graph Trajectory similarity (GTS), Nodes similarity function (NSF), Weight adjacency matrix (WAM).

إنشاء رسم بياني لمسارات مماثلة بناءً على الحد الأدنى من الشجرة الممتدة (MST) ودالة تشابه العقد (NSF)

علياء طالب رحيم* , توفيق عبدالخالق الاسدي
قسم علوم الحاسبات ، كلية علوم الحاسبات وتكنولوجيا المعلومات ، جامعة بابل ، العراق

الخلاصة

في السنوات الأخيرة، أصبح النهج القائم على الرسم البياني أكثر شعبية كوسيلة لنمذجة البيانات الخاصة بالعالم الحقيقي. يمكن استعمال هذه الرسوم البيانية لتمثيل مجموعة واسعة من البيانات، مثلا الشبكات الاجتماعية والمستندات النصية وشبكات تنظيم البروتين أو الجينات. يمكن تمثيل وإنشاء الرسوم البيانية التي تعتمد على المسارات بطرق مختلفة وذلك من خلال الاعتماد على التكنولوجيا المستعملة للحصول على البيانات، مثل نظام تحديد المواقع العالمي (GPS) والكاميرات ....الخ. عملية العثور على الكائنات (الاهداف) ذات أنماط حركة مماثلة مهمة تحليل البيانات الشائعة في تعدين المسار؛ وبالتالي، فإن وجود خوارزمية فعالة للمسارات أمر ضروري لتجميع المعلومات وعرضها كرسوم بيانية. في هذا البحث، تم اقتراح طريقة جديدة لبناء رسم بياني يعتمد على كلا الحافتين من ميزات كل مسار ورأس من رقم الإطار. لمقارنة رسمين بيانيين للمسارات، نقدم دالة

*Email: alyaatr.sw.phd@student.uobabylon.edu.iq

تشابه العقدة (NSF). تم تقديم مخطط جديد لتشذيب حواف ورؤوس الرسم البياني وخوارزمية جديدة لحساب التشابه بين رسمين بيانيين لإنشاء تشابه مسار الرسم البياني (GTS).

1. Introduction

With the evolution of surveillance systems and tracking operations, moving objects' motion dynamics can generate an important part of their data trajectories. In computer vision, one necessary and important process is the detection and tracking approach to objects in scene video. Traditionally, detection of a moving target can be detected by extracting the region of variation in the sense of video from a background of images. Recently, the detection process has employed deep learning techniques, examples of which are YOLO, CNN, and RCNN [1].

The object-tracking process is significant for computer vision monitoring and surveillance applications. Extracting and selecting features to build the model is one challenge in object-tracking. It is appropriate for distinguishing and tracking the object. The morphology operation describes trajectories for selecting different human behaviors and splitting video frames based on this operation [2]. This process assists operators in various videos, including situations where individuals are loitering, sprinting, or engaging in abnormal behavior such as crimes, thefts, or walking in reverse directions [3].

Recently, computer science and different science fields have widely used graph theory. Many tasks in the computer vision (CV) and image processing fields use graphs, including object detection, tracking, and image segmentation [4]. Graphs can represent the relationships between objects as a set of block diagrams. Graphs are represented as a set of edges and vertices, where the vertices indicate the objects in the video and the edges indicate the relationship between nodes [5]. Graph Trajectory Similarity (GTS) is a method used to calculate the similarity between trajectories in a graph. It is significant in many applications, such as route planning [6], trajectory clustering [7], and transportation optimizations [8]. A trajectory depicts the movement of objects in space over time and is commonly represented as a series of discrete locations [9]. Similarity determines if two objects are similar in behavior by using a variety of methods, including those based on edit distance [10], spectral similarity [11], optimum transport [12], and behavioral equivalency [13].

Graph similarity is calculated by determining the degree of similarity between two graphs (a value between 0 and 1). Since we already know which nodes correspond to one another, it stands to reason that if a given node's neighbors in both graphs are similar, then that node's connectedness (in terms of edge weights) to its neighbors will also be similar. The concept of graph similarity has garnered significant interest in various domains, including computer vision [14] and network science [15]. However, the exploration of graph similarity from a systematic perspective by theoretical computer scientists has been limited. Instead, they have primarily focused on specific instances of similarity, such as isomorphism [16] and bisimilarity [17], which have been extensively studied. Nonetheless, it is deemed valuable to formulate a theoretical framework for evaluating graph similarity, which involves comparing diverse similarity metrics, analyzing their algorithmic and semantic characteristics, and thus enhancing our comprehension of their appropriateness for different types of applications. This research study examines the computational difficulty associated with a specific group of similarity measures that extend the similarity generated from the similarity of nodes in graphs. In a broad sense, similarity is often understood as the measure of closeness concerning a given metric.
Our proposed method for the construction of trajectory graphs based on the Weighted Adjacency Matrix (WAM). This paper proposes a novel approach that constructs the graph

using both edges from each trajectory's features and vertices from frame numbers. For comparing two graphs of trajectories, we introduce a Node Similarity Function (NSF). A new schema for pruning edges and vertices of the graph and a new algorithm for calculating the similarity between two graphs to generate Graph Trajectory Similarity (GTS) are also presented. Overall, our contributions are as follows:

- A WAM method for constructing a graph.
- A new schema for pruning a graph by using MST.
- A new algorithm for comparing two graphs and calculating their similarity.

The remainder of the paper is structured as follows: Section 2 outlines related works. Section 3 describes the proposed method in detail. The results and discussion of the experiment are presented in Section 4, while the conclusion is presented in Section 5.

2. Related Works

This part talks about the studies that have already been done on Graph Trajectory Similarity (GTS) from three different points of view: (1) computing trajectory similarity; (2) Minimum Spanning Tree (MST); and (3) Nodes Similarity Function (NSF). Several recent studies have introduced innovative approaches to pedestrian and multi-object tracking, showcasing significant advancements in accuracy and robustness across challenging datasets such as PETS2009-S2L1 and Oxford Town Centre. In crowded environments, one notable approach [18] integrates probability fields with a movement feature space to improve pedestrian tracking accuracy. This method effectively models the likelihood of pedestrian presence and improves trajectory characterization, demonstrating superior performance in handling occlusions and maintaining trajectory consistency. In a different study [19], a multi-object tracking framework is created by combining a network flow model with ORB descriptors. This makes it easier to link detections across frames and makes tracking more reliable. This framework achieves better tracking accuracy and fewer identity switches compared to traditional methods by using efficient feature extraction and matching. This has been shown in a number of datasets, such as PETS2009-S2L1 and Oxford Town Centre. Another way to improve multi-pedestrian tracking is to use a hybrid approach [20] that combines particle filters with network flow models. This method handles uncertainties in pedestrian movements in a probabilistic way and improves detection association over time. This method proves effective in scenarios with occlusions and complex interactions among pedestrians, contributing to reliable and consistent trajectory tracking. Finally, an innovative approach [21] integrates local-to-global trajectory models for multi-target tracking, leveraging both local movement patterns and global trajectory information to enhance tracking accuracy and maintain trajectory consistency in challenging environments. Collectively, these studies show big steps forward in making tracking work better in a wide range of difficult tracking situations found in datasets such as PETS2009-S2L1 and Oxford Town Centre.

2.1 Trajectory similarity computation

The current approaches for computing trajectory similarity can be roughly classified into two groups: learning-based and non-learning-based. Most approaches that do not rely on learning treat trajectories as spatial curves and utilize computational geometry to simplify calculations [18, 19, and 20]. Often, these methods rely on untested ad-hoc heuristic principles, potentially leading to suboptimal results. Furthermore, the development of these methods for specific distances makes their application to other similarity metrics challenging. Recent advances in artificial intelligence (AI) [21] have led to the development of several learning-based approaches [22, 23], which embed the spatial-temporal properties of trajectories into vectors to determine their similarity. These strategies typically employ a recurrent neural network (RNN) for sequence modeling and backpropagation through time (BPTT) to fine-tune

the parameters. While these techniques perform well on short trajectory datasets, their effectiveness declines with the length of the trajectory datasets.

2.2 Minimum Spanning Tree (MST)

For selecting video frames without segmenting the video into shots or scenes, the authors employ a graph-theoretic divisive clustering approach based on the creation of a Minimum Spanning Tree (MST). This enables a quick and straightforward calculation of a video summary for a certain number of key frames. In a similar way, our study looks at the problem of summarizing videos by turning it into a graph-based clustering problem. In this case, a cluster, which is a connected part of the graph calculated from a graph partition, is a group of similar frames. The proposed method, the Hierarchical Summarizing Approach (HSUMM) [24], uses an MST to bypass the pre-processing stages. Instead of directly calculating the graph partition over the MST, the proposed method uses a hierarchical clustering technique to derive a weight from the MST, allowing for easier cluster inference. Furthermore, HSUMM is capable of applying a similarity measure across clusters (i.e., a set of frames) during graph partitioning because it uses a graph-based hierarchical clustering algorithm rather than only evaluating the similarity between individual frames.

2.3 Nodes Similarity Function (NSF)

There are many similarity-measure methods based on unsigned social networks. Traditional approaches to measuring the similarity of nodes mainly focus on their feature values, such as the Jaccard coefficient, cosine similarity, Euclidean distance, etc. However, these similarity measures primarily concentrate on the node's features and do not consider the link structures among objects. Link-based techniques are often used to assess the similarity between nodes in network structures. These approaches utilize various algorithms, including the shortest path algorithm, Random Walk with Restart (RWR), SimRank, and personalized PageRank [25]. According to research conducted by Liben and Kleinberg [26], it has been suggested that using the shortest path between two nodes in a network can be effective for friend recommendations. In this section, a fundamental node similarity metric is introduced to assess the closeness between two nodes within a graph-based framework. The function $sim(v_i,v_j)$ is defined to represent the similarity between node v_i and node v_j [27].

3. Proposed system

The following explanation applies to the proposed model, as shown in Figure 1:

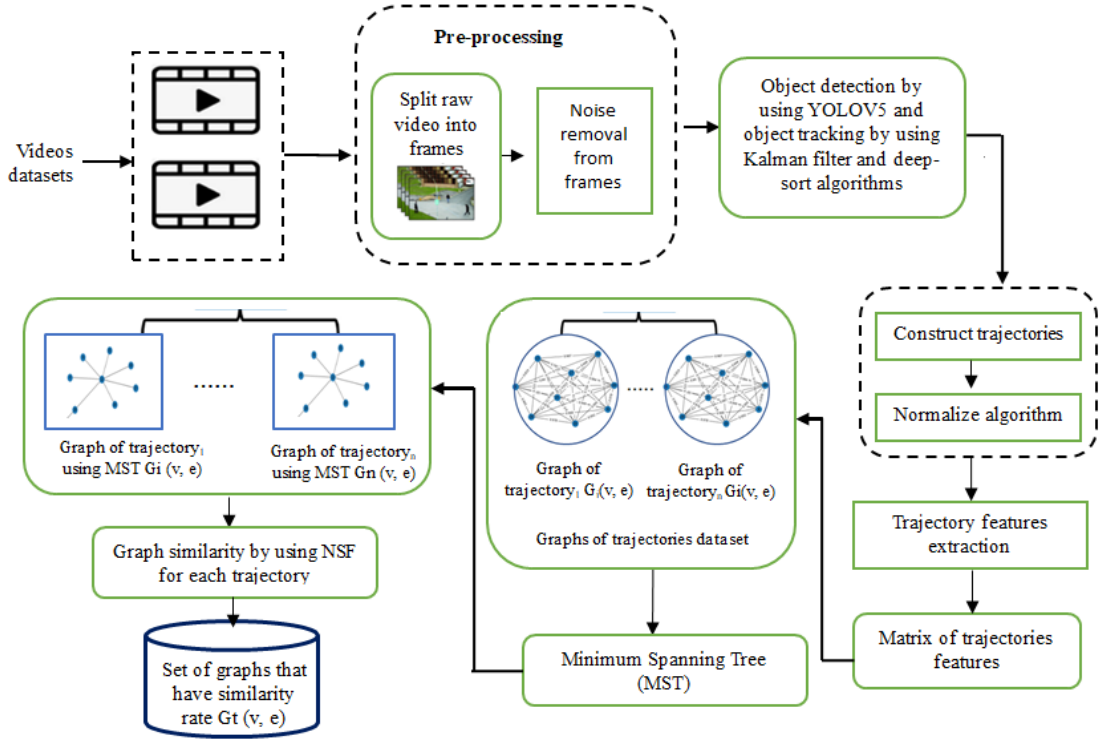
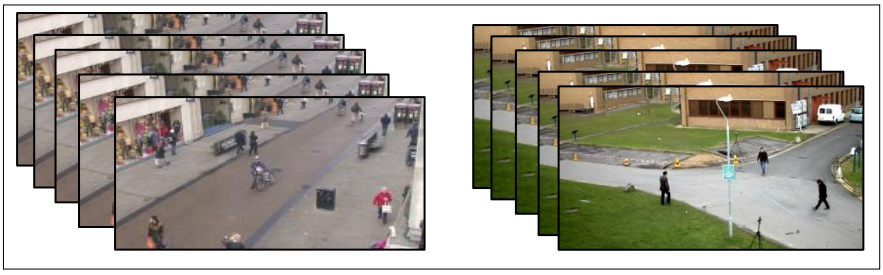


Figure 1: Block diagram of proposed system model

3.1 Video reading and Data pre-processing

This work utilizes the datasets PEST2009 SL2 and Oxford Town Centre, as illustrated in Figure 2. The pre-processing stage begins after splitting the video into frames, during which some images may require operations to be ready for detection and tracking, as some frames may exhibit noise or blurring. Another issue that may arise are outdoor changes, such as foggy weather and variations in ambient illumination, which are referred to as appearance variation. These conditions necessitate techniques to enhance and filter the images. The resulting frames should be free from noise and aberration.



(a) (b)

Figure 2: Benchmark datasets (a) oxford town center (b) PEST2009 SL2

3.2 Trajectory construction

Geometrically, an object occupies a specific location on the image plane, and its location is updated in each consecutive frame based on its moving distance. The line constructed by connecting multiple sequences of points is called a trajectory. Indeed, a trajectory represents a

moving object's path through space over time. During the trajectory construction stage, the trajectory of motion for each target is illustrated. According to equation (1), there is a center point (Cx, Cy) inside each bounding box for a moving item in each frame. From these points, we can calculate the center of the bounding box and draw a trajectory of moving objects in a video. The tracker connects the current point with the next point and continues across all frames to construct the unique trajectory for each object in the scene. Many points on the trajectory may exhibit a minor change in the movement of a person during the video. The system uses normalization to remove these points, depending on the sum of changes in the angle of direction when a target moves in a scene. After normalization, the remaining points of the trajectory provide valuable information about changes in the direction and pose of a target. These features are utilized for constructing graphs.

$$Trajectory\ (Tr) = T(x_{\text{center}}, y_{\text{center}}) \tag{1}$$

3.3 Extraction matrix of the trajectory feature

The shape of the trajectory shows the behavior of the moving of an object in the frame of a video and can be represented in a variety of shapes, including a straight line, a curved line, or a zigzag path depending on movement shape and orientation. It is constructed from suitable center points of a bounding box across the sequence of frames. Several features could be extracted from the trajectory of an object, including position, direction, velocity, and aspect ratio. These features indicate the behavior of the moving object in the scene. These features depict an object's motion within the scene.

- ***Distance feature:*** (distance length) refers to the difference between two positions of the target, if (x1, y1) is the first position in the first frame, (x2, y2) the second position for the same object in the next frame, utilize Euclidean distance for calculating the distance as equation (2) [28].

$$Distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2}$$

- ***Velocity and acceleration feature:*** object velocity represents the rate of change of object location over time. Velocity is repressed as an object's motion between frames can be calculated. The motion of an object between frames can be calculated based on the distance and time between two frames. The velocity of a moving object is shown in equation (3,4) [29].

$$\text{velocity} = \frac{\text{distance between two frames}}{\text{time (frame rate1)}} \tag{3}$$

$$\text{Frame rate} = \frac{1}{\text{frame per second (fps)}} \tag{4}$$

- ***Orientation (direction) feature****:* the angle among two points in the line space of the trajectory segment can estimate a target at each location by using the formula in equation (5):

$$\theta = \tan^{-1}\left(\frac{oppsite\ lne}{Ajust\ line}\right) = \tan^{-1}\frac{y2-y1}{x2-x1} \tag{5}$$

When (Θ) is the angle of direction among two points that represents the previous position and current position for the target in the scene [30].

3.4 Construction graph for each trajectory

A graph is constructed for each object's trajectory by utilizing the similarity between two nodes. A trajectory denotes the position of an object in each frame, with each node representing a point of the trajectory object in a specific frame. Cosine similarity (CS) is applied between every two distinct points of all trajectory nodes. The value of cosine similarity indicates the strength of the relationship between nodes, acting as weights that represent the weighted

similarity between every pair of nodes. The similarity function compares nodes based on direction, distance, and velocity features. Consequently, a graph is built that represents the behavior and connections among the trajectory object's nodes within the video frame. After applying the cosine similarity measure, a weighted adjacency matrix (WAM) is obtained. This WAM enables the construction of an undirected weighted graph, which delineates the object trajectory pattern behavior. The WAM contains symmetric data, necessitating data reduction to either its upper triangle or lower triangle to streamline the representation.

3.5 Mining graphs based on Minimum Spanning Tree (MST)

One definition of a spanning tree of a connected undirected graph G = (V, E) is a tree T that contains all of the vertices in G. Each edge in a disconnected graph G will have a spanning tree Ti, where (i) is the number of the edge, and the set of all these spanning trees together will be called the spanning forest of the graph G. The spanning trees can have different weights if we attach a weight (wi) against each edge of the graph. The minimal weighted spanning tree, often known as the Minimal Spanning Tree (MST), is one of the most popular and useful of these. It finds use in several scientific and technological domains [31]. Many algorithms have been designed to reduce the cost of trees and make them asymptotically efficient, such as Kruskal's minimum scanning tree algorithm and Prim's minimum scanning tree algorithm. Those algorithms proposed the idea of removing the cycles from the graphs to compose a tree [32]. In this paper, Prim's Minimum Spanning Tree algorithm is used to find the minimum spanning tree of a connected, undirected graph with weighted edges. This algorithm starts with a single vertex and then repeatedly grows the tree by adding the edge of minimum weight that connects a vertex in the tree to a vertex outside the tree until all vertices are included in the tree.

In the context of video frames, you could think of each vertex as representing a single frame, and the weighted edges between vertices represent the relationships or differences between frames. Just as Prim's algorithm seeks to connect all vertices while minimizing the total weight of edges, in video processing, you might want to analyze a sequence of frames to identify the most important or representative frames while minimizing redundancy or data duplication. Finally, the steps of the proposed MST algorithm are shown in figure.

3.6 Graph similarity by using nodes similarity function (NSF)

To predict the similarity score of two graphs, a graph similarity computation by node similarity function (GSimNSF) was proposed. Calculating multiple graph distance/similarity metrics is NP-hard, and the central operation of graph similarity search is a difficult issue. The Node Similarity Function method is used to evaluate a group of two sets of nodes to determine a similarity between them. If two nodes share many neighbors, they are similar. The Jaccard metric (also known as the Jaccard Similarity Score) and the Overlap coefficient (also known as the Szymkiewicz-Simpson coefficient) are used based on node similarity to calculate pair-wise similarities. In this research, we calculate the Jaccard similarity using the following equation (6).

$$J(A,B) = \frac{\lfloor A \cap B \rfloor}{\lfloor A \rfloor + \lfloor B \rfloor - \lfloor A \cap B \rfloor} \quad (6)$$

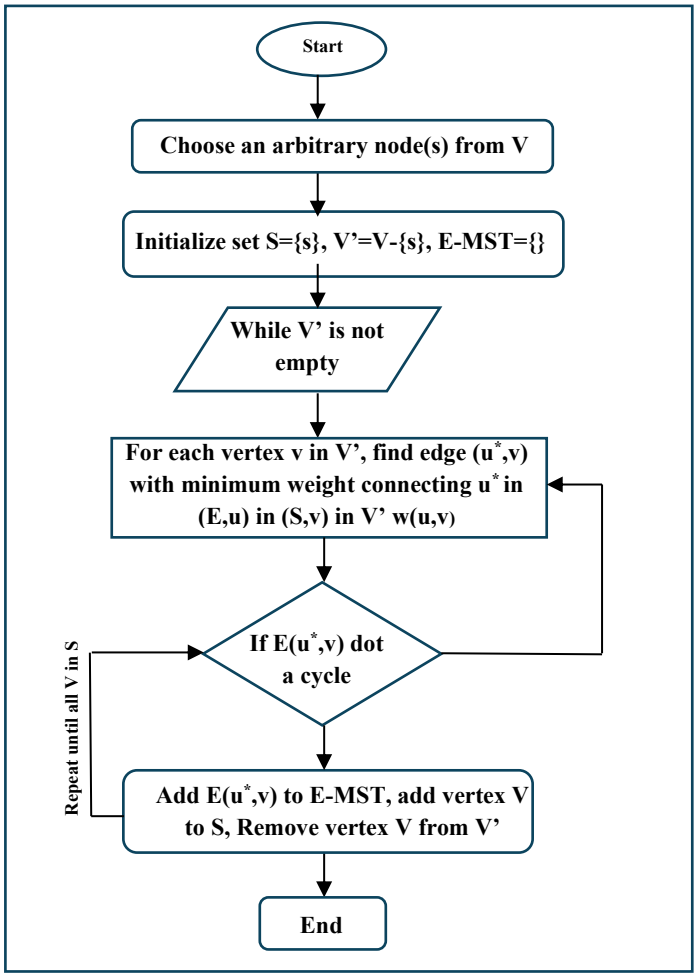


Figure 2: Flowchart of Mining graph based on Minimum Spanning Tree **(**MST**)** algorithm

Algorithm 2: Graph Similarity Computation based on Node similarity function (GSimNSF)

Input: *Two graphs, Graph1, and Graph2*
Output: *Return the similarity score as the result of the Node Similarity algorithm for the given graphs.*
Begin

Step 1: *Extract the sets of nodes from both Graph1 and Graph2:*
nodeSetGraph1 = set (graph1.nodes)
nodeSetGraph2 = set (graph2.nodes
Step 2*: Find the intersection of the node sets:*

intersectionNodes=intersectionnodeSetGraph1.intersection(nodeSetGraph2)
Step 3**:** *Initialize a set V'=V-{s} to keep track of unselected vertices (remaining vertices expect s)*
Step 4: *Calculate the union of the node sets. The union contains all the unique nodes from both graphs.*
UnionNodes = nodeSetGraph1.union(nodeSetGraph2)
Step 5: *Determine a similarity measure or metric to assess the similarity between the graphs based on their nodes*
SimilarityMeasure = len(intersectionNodes) / len(unionNodes)
Step 6: *Obtain a similarity score between graph (1) and graph (2) based on their nodes.*
SimilarityScore = SimilarityMeasure
End

4. Experiment results and discussion

This section displays the results of the proposed system steps that have been discussed in Section (3).

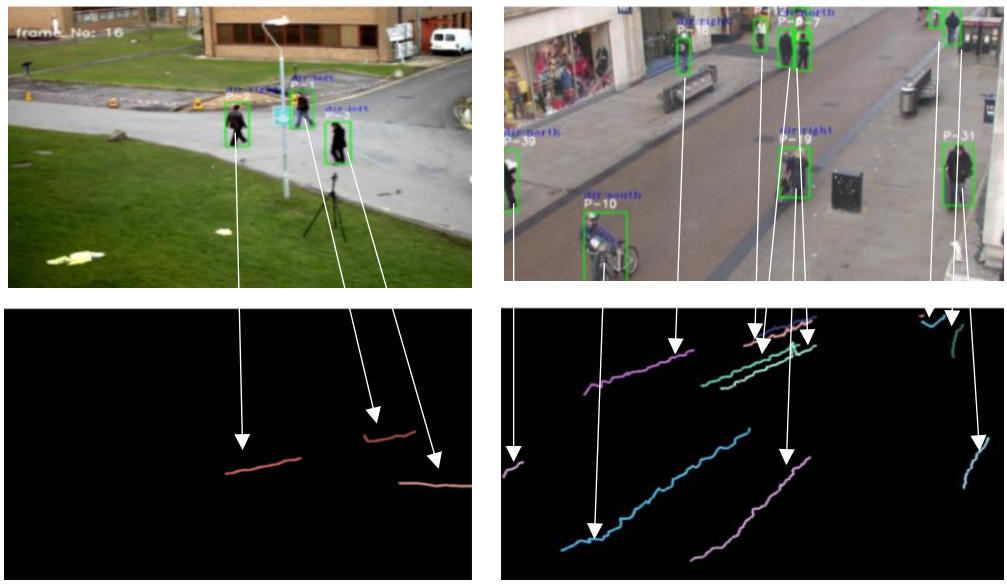
4.1 datasets

In our experiment, we have used two benchmark datasets, PETS 2009 SL2 and Oxford Town Centre datasets. The PETS 2009 dataset is divided into three parts, each of which features a multi-view sequence of pedestrians moving through an outdoor setting. The gold standard is an 875-image sequence called "the person counting sequence" [33]. The Oxford Town Centre dataset is comprised of a 5-minute video with 7500 annotated frames, 6500 of these frames are used as training data, and 1000 are used for testing pedestrian recognition models. Oxford's surveillance camera captured the information for use in developing activity and facial recognition technologies [34].

4.2 Trajectory construction result

To evaluate the performance of the proposed methods, experiments were carried out on video sequences from two datasets, such as PETS 2009 SL2 and Oxford Town Centre Dataset. In this study, YOLOv5 and Deep-SORT algorithms are used to detect and track objects. To obtain the trajectory for each object with a unique ID and frame number, our detection and

tracking model employs a centroid tracker in addition to using both the Kalman filter and Hungarian algorithms after the pre-processing step. The line connecting the centers of objects in each frame constitutes the trajectory. Figure 2 demonstrates the correlation between the trajectories of all tracked objects in the scene and the frame number. The lengths of trajectories vary from one object to another. In addition, the lengths of trajectories are added to distinguish the trajectories with similar shapes.



(a) (b)

Figure 2: Sample of trajectory objects by (a) PETS 2009 SL2 dataset (b) Oxford Town

4.3 Feature extraction result

The trajectory shape illustrates the behavior of a target's motion, leading to the extraction of various features from the target's trajectory, such as position, distance (D), direction (Θ), and velocity (v), as demonstrated in Tables 1, 2, and 3 for three objects from the PETS 2009 SL2 dataset. These extracted features are stored in a CSV file as an actual dataset to represent the moving behavior of the target in a scene. The position feature indicates the differences between two people's positions. This positioning feature can be utilized to calculate the amount of change exclusively in x-coordinates, y-coordinates, or both coordinates when the target is moving. The distance feature is calculated based on the Euclidean distance measure as the initial feature. The second feature, the angle of direction, is determined for a target at each point when the angle between two points refers to the previous and current locations of the target in the frame. The resultant value of the angle of direction (Θ), used with the change in coordinates for x and y values, helps to detect the orientation of the moving target in the video, which may be one of four directions: east, west, north, or south. The block ID for each object is determined by dividing the current block position by the block size (16, 16) and adding 1, which yields the row and column index of the block. Calculating the block ID is useful for finding similarities between objects or within the same object, thus identifying similar behavior among objects. The velocity feature is calculated by dividing the distance value by time (1 frame per second).

Table 1: The trajectory features extraction for object ID_1 based on PETS 2009 SL2 dataset

Object ID	X	Y	Frame No	Direction	Distance	velocity	Aspect ratio	Orientation	Block ID
2	394	341	6	8	7	10.45	0.38	Right	B2522
2	533	298	35	63	7	13.73	0.42	North	B3419
2	540	276	59	69	9	17.65	0.36	South	B3418
2	518	297	26	27	4	7.41	0.42	Right	B3319
2	509	303	25	34	11	21.15	0.42	Right	B3219
2	526	288	57	40	17	33.33	0.37	Right	B3318
2	482	319	18	24	10	17.86	0.4	Right	B3120
2	468	320	16	7	8	14.29	0.4	Right	B3020
2	463	323	15	31	6	10	0.39	Right	B2921
2	440	331	12	16	7	12.07	0.39	Right	B2821
2	420	336	10	18	6	10	0.38	Right	B2721
2	401	340	7	0	6	8.45	0.38	Right	B2622
2	371	348	4	9	12	16	0.38	Right	B2422
2	498	312	20	14	4	7.27	0.41	Right	B3220
2	547	291	55	6	20	39.22	0.38	Left	B3519

Table 2: Weight Adjacency Matrix (WAM) of features for object ID_2 based on IPETS 2009 SL2 dataset

Object ID	X	Y	Frame No	Direction	Distance	velocity	Aspect ratio	Orientation	Block ID
1	607	279	17	8	7	11.86	0.46	Left	B3818
1	618	277	15	18	6	10	0.45	Left	B3918
1	633	274	13	11	5	8.06	0.45	Left	B4018
1	653	268	10	0	8	13.33	0.45	Left	B4117
1	670	262	6	30	8	11.94	0.45	Left	B4217
1	676	259	4	27	4	5.33	0.46	Left	B4317
1	638	272	12	22	5	8.62	0.47	Left	B4017
1	594	267	19	63	7	12.07	0.48	North	B3817
1	591	261	20	9	6	10.91	0.48	Left	B3717

Table 3: The trajectory features extraction for object ID_3 based on PETS 2009 SL2 dataset

Object ID	X	Y	Frame No	Direction	Distance	velocity	Aspect ratio	Orientation	Block ID
3	845	373	5	4	15	18.75	0.5	Left	B5324
3	809	376	8	11	16	25.81	0.49	Left	B5124
3	693	370	16	8	14	25	0.45	Left	B4424
3	569	339	26	66	17	31.48	0.42	North	B3622
3	623	365	21	4	15	28.85	0.44	Left	B3923
3	636	366	20	4	13	23.64	0.44	Left	B4023
3	651	368	19	8	15	25.86	0.45	Left	B4123
3	668	368	18	0	17	30.36	0.46	Left	B4223
3	679	368	17	0	11	18.64	0.45	Left	B4323
3	868	375	4	5	23	30.67	0.49	Left	B5524
3	710	374	15	13	17	28.33	0.46	Left	B4524
3	562	323	27	24	12	23.08	0.42	Left	B3621
3	793	373	9	5	12	17.91	0.48	Left	B5024
3	588	361	24	10	11	20.37	0.43	Left	B3723

4.4 Graph construction result

After the trajectory has been constructed for each object, a matrix will be constructed containing the trajectory of an object with the number of frames in which the trajectory is present. A graph is built by converting the Weight Adjacency Matrix (WAM) to nodes and edges because a graph is better for representing huge data. Moreover, the graph reflects patterns frequently for each object. A graph is constructed for each object's trajectory by using methods of similarity between every two nodes, as shown in Table 4. This graph will be stored in the graph dataset to prepare it to be input for the graph mining algorithm to mine the frequent pattern. Graphs can represent huge data with important information that reflects the conduct of the object. Figure 3 shows an example of converting each trajectory into a weighted, undirected graph based on nodes' features.

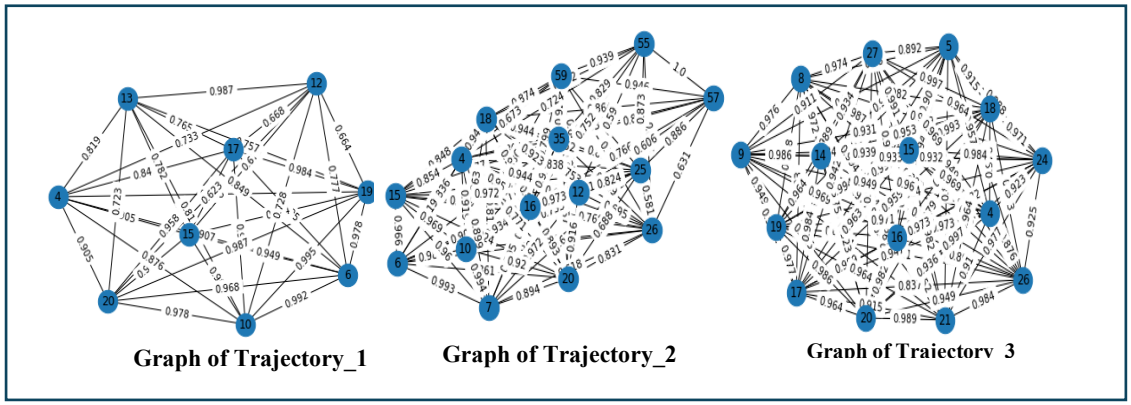


Figure 3: the graphs of trajectory features extraction for three objects based on PETS 2009 SL2 dataset

4.5 Graph based on (MST) and similarity score result

The procedure starts by adding to the tree the edges that have the smallest weight incident to each vertex in the graph. Then it continues building the forest by adding edges from each tree using the same method of determining the minimum-weight edge from each tree to a different tree. With each iteration, the number of trees in each linked component of the graph is cut in half; therefore, the process terminates after a logarithmic number of iterations. When this happens, the collection of edges that emerges is known as the least spanning tree, and Figure 5 displays the MST result. The node similarity function (NSF) is used to determine the degree of similarity between the two graphs, and the results are shown in Table 4 and Figure 4.

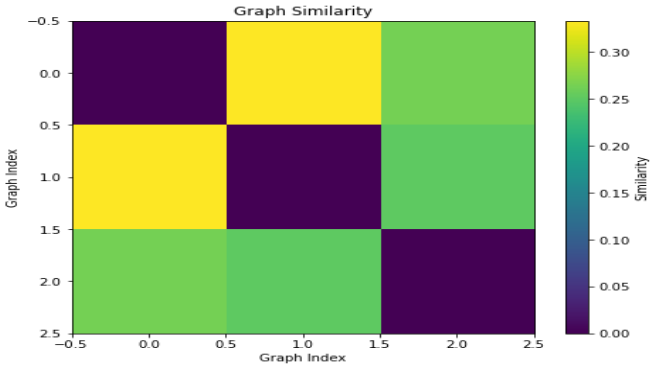


Figure 4: Graphs similarity scores based on Node similarity function (NSF)

Table 4:Graph matching score based on Node similarity function (NSF)

Object ID		Similarity Rate
1	**2**	0.3333
1	**3**	0.2631
2	**3**	0.25

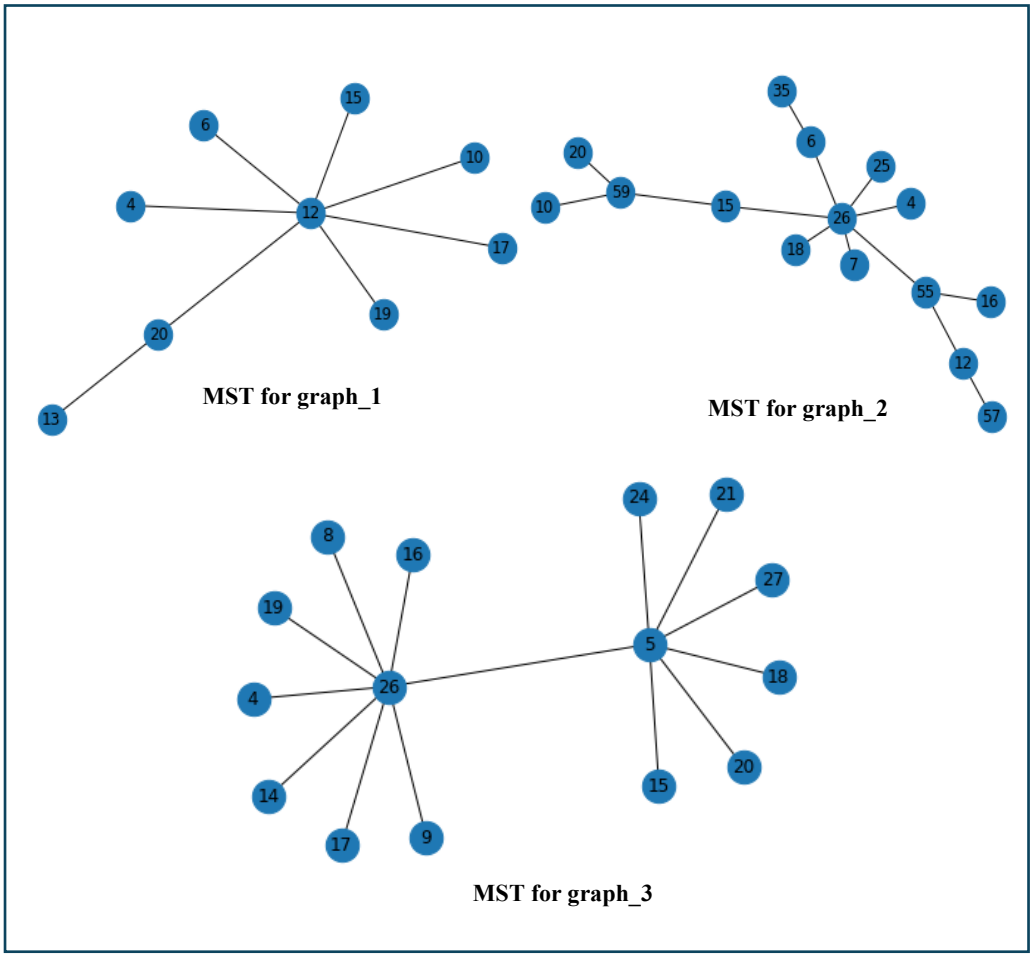


Figure 5: Graphs based on MST

5. Evaluation Metric

Evaluating a model is an essential part of any system. When a model is evaluated using one metric, it may produce a good skill score and satisfy reasonable results, but when evaluated against other metrics, the results may be poor. Therefore, to evaluate the results of system stages such as WAM, MST, and NSF, we must use suitable metrics. In this step, we use several metrics to assess the results of moving targets. These metrics based on equations (7-11) include accuracy, confusion matrix, precision, recall, and F-score. Table (6) shows the results of the evaluation stage according to performance metrics based on three videos of the PEST2009 SL2 and Oxford Town Centre datasets.

1. **Mean Squared Error (MSE):** Compute the MSE between the WAM of your model and the ground truth WAM.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(W_{ij} - W'_{ij}) \quad (7)$$

Here, W_{ij} *and* W'_{ij} are the elements of the WAMs of your model and the ground truth, respectively.

2. **Tree Similarity (TS):** Measure the similarity between the MSTs produced by your model and the ground truth MST using metrics like Jaccard index.

$$Jaccard\ Index = \frac{|E_{model} \cap E_{gt}|}{|E_{model} \cup E_{gt}|} \quad (8)$$

Here, E_{model} and E_{gt} are the sets of edges in the MSTs of your model and the ground truth, respectively.

3. **Total Weight Comparison (TWC):** Compare the total weights of the MSTs from your model and the ground truth.

$$Total\ Weight\ Difference = |\sum_{e \in MST_{model}} w_e - \sum_{e \in MST_{gt}} w_e \quad (9)$$

Here, w_e is the weight of edge e.

4. **Precision and Recall:** Calculate the precision and recall for the identified similarity frames.

$$Precision = \frac{|F_{model} \cap F_{gt}|}{|F_{model}|} \quad (10)$$

$$Recall = \frac{|F_{model} \cap F_{gt}|}{|gt|} \quad (11)$$

Table 6: demonstrates the performance and strengths of your graph-based trajectory clustering method across multiple evaluation criteria for both datasets. The high values across all metrics indicate the effectiveness of your approach in achieving accurate and reliable results.

Table 6: performance Metrics for model based on PEST2009 SL2 and Oxford Town Centre Datasets

Dataset	MSE	TS	TWC	Precision	Recall	F1 Score
PET2009 SL2	0.302	0.132	8.0679	92.34%	90.56%	91.44%
Oxford Town Center	0.221	0.012	7.8945	91.12%	93.45%	92.27%

5. Conclusion

In this study, we offer a graph-based model for representing trajectories, and we use edge and vertex similarity metrics to quantify how well two trajectories' graph representations match up with one another. Vertex-based similarity provides superior grouping over edge-based similarity, as evidenced by the correlation value. Two separate datasets are used to test the efficacy of the graph-based trajectory representation for trajectory clustering. Three performance measures—WAM, MST, and NSF—are calculated by varying the trajectory length. The trajectories are transformed into graph data by employing variables such as direction, distance, velocity, and aspect ratio to create Weight Adjacency Matrix (WAM) representations that may be fed into a graph mining method or graph convolution neural network (GCN). High precision in aggregating objects with similar behaviors and features is achieved by modeling their paths as graphs so that the outcomes are ideal. We anticipate employing graph mining or graph convolution neural networks (GCNN) in the future to identify candidate frames that accurately represent each object's behavior and then gather these frames to produce the summary video. The proposed model's evaluation results for PEST2009 SL2 and Oxford Town Centre datasets range from 90 to 95%.

5. Acknowledgments

This work was supported by the College of Information Technology, University of Babylon, Hillah-Babil, Iraq. The authors who presented the study have no conflict of interest. We hereby confirm that all the figures and tables in the manuscript are ours.

References

[1] Ali, M. Najm and Y. H., "Automatic Vehicles Detection, Classification and Counting Techniques / Survey," *Iraqi Journal of Science,* vol. 61, no. 7, p. 1811–1822, Jul 2020.

[2] S. J. Shahbaz, A. A. D. Al-Zuky, and F. E. M. Al-Obaidi, "Real-Night-time Road Sign Detection by the Use of Cascade Object Detector," *Iraqi Journal of Science,* vol. 64, no. 6, pp. 3164-3175, Jun 2023.

[3] Y. Yang, J. Cai, H. Yang, J. Zhang, and X. Zhao, "TAD: A trajectory clustering algorithm based on spatial-temporal density analysis," *Expert Systems with Applications,* vol. 139, p. 112846, Jan 2020.

[4] W. M. Brich and I. H. Ali, "Predicting Motion Direction and Person re- identification across surveillance Cameras Network using (LSTM)," in *In 2022 3rd Information Technology To Enhance e-learning and Other Application (IT-ELA)*, Baghdad, Iraq,, 2022.

[5] S. A. Bhavsar, V. H. Patil, and A. H. Patil, "Graph partitioning and visualization in graph mining: a survey," *Multimedia Tools and Applications,* vol. 81, no. 30, p. 43315–43356, May 2022.

[6] P. Han, J. Wang, D. Yao, S. Shang, and X. Zhang, "A Graph-based Approach for Trajectory Similarity Computation in Spatial Networks," in *In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, New York, 2021.

[7] S. Zhou, P. Han, D. Yao, L. Chen, and X. Zhang, "Spatial-temporal fusion graph framework for trajectory similarity computation," *World Wide Web,* vol. 26, no. 4, p. 1501–1523, Sep 2022.

[8] S. Wang, Z. Bao, J. S. Culpepper, and G. Cong, "A Survey on Trajectory Data Management, Analytics, and Learning," *ACM Computing Surveys,* vol. 54, no. 2, pp. 1-36, Mar 2021.

[9] A. Hamdi, K. Shaban, A. Erradi, A. Mohamed, S. K. Rumi, and F. D. Salim, "Spatio-temporal data mining: a survey on challenges and open problems," *Artificial Intelligence Review,* vol. 55, no. 2, p. 1441–1488, Apr 2021.

[10] A. Dawar and K. Khan, "Constructing Hard Examples for Graph Isomorphism," *Journal of Graph Algorithms and Applications,* vol. 23, no. 2, p. 293–316, 2019.

[11] Stefan Klus and Tuhin Sahain, "A spectral assignment approach for the graph isomorphism problem," *Information and Inference: A Journal of the IMA,* vol. 7, no. 4, p. 689–706, Feb 2018. https://doi.org/10.1093/imaiai/iay001

[12] M. Juhlin, F. Elvander, and A. Jakobsson, "Defining Graph Signal Distances Using an Optimal Mass Transport Framework," in *In 2019 27th European Signal Processing Conference (EUSIPCO)*, A Coruña, Spain, 2019.

[13] M. Li, J. Yu, H. Xu, and C. Meng, "Efficient Approximation of Gromov-Wasserstein Distance Using Importance Sparsification," *Journal of Computational and Graphical Statistics,* vol. 32, no. 4, p. 1512–1523, Feb 2023.

[14] S. Bouhenni, S. Yahiaoui, N. Nouali-Taboudjemat, and H. Kheddouci, "A Survey on Distributed Graph Pattern Matching in Massive Graphs," *ACM Computing Surveys,* vol. 54, no. 2, p. 1–35, Feb 2021.

[15] H. T. Trung, T. Van Vinh, N. T. Tam, H. Yin, M. Weidlich, and N. Q. Viet Hung, "Adaptive Network Alignment with Unsupervised and Multi-order Convolutional Networks," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, 2020.

[16] M. Grohe and P. Schweitzer, "The graph isomorphism problem," *Communications of the ACM,* vol. 63, no. 11, p. 128–134, 2020.

[17] Y. Zakowski, P. He, C.-K. Hur, and S. Zdancewic, "An equational theory for weak bisimulation via generalized parameterized coinduction," in *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, New Orleans, Louisiana, United States, 2020.

[18] P. N. a. D. Garayalde, "Pedestrian tracking using probability fields and a movement feature space," *DYNA,* vol. 84, no. 200, p. 217–227, Jan 2017.

[19] Z. X. J. L. a. J. J. J. Chen, "Multi-object tracking based on network flow model and ORB feature," *Applied Intelligence,* vol. 52, no. 11, p. 12282–12300, Feb 2022.

[20] J. Z. Z. H. a. J. H. Y. Cui, "Multiple pedestrian tracking by combining particle filter and network flow model," *Neurocomputing,* vol. 351, p. 217–227, July 2019.

[21] J. W. Z. W. Y. G. a. Y. L. S. Zhang, "Multi-target tracking by learning local-to-global trajectory models," *Pattern Recognition,* vol. 48, no. 2, p. 580–590, 2015.

[22] T. Kociumaka, A. Mukherjee, and B. Saha, "Approximating Edit Distance in the Fully Dynamic Model," in *In 2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, Santa Cruz, CA, USA, 2023.

[23] D. Yao, G. Cong, C. Zhang, and J. Bi, "Computing Trajectory Similarity in Linear Time: A Generic Seed-Guided Neural Metric Learning Approach," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, Macau SAR, 2022.

[24] D. Yao, H. Hu, L. Du, G. Cong, S. Han, and J. Bi, "TrajGATGraph-based Long-term Dependency Modeling Approach for Trajectory Similarity Computation," in *In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Washington DC USA, 2022.

[25] Subharun Pal, "Intersection of Artificial Intelligence and," *International Journal of Innovative Research in Engineering & Multidisciplinary Physical Sciences,* vol. 11, no. 3, pp. 1-7, may 2023.

[26] S. Zhou, P. Han, D. Yao, L. Chen, and X. Zhan, "Spatial-temporal fusion graph framework for trajectory similarity computation," . *World Wide Web,* vol. 26, no. 4, pp. 1501-1523, July 2022.

[27] Y. Zhang, A. Liu, G. Liu, Z. Li, and Q. Li, "Deep Representation Learning of Activity Trajectory Similarity Computation," in *In 2019 IEEE International Conference on Web Services (ICWS)*, Chicago, IL, USA, 2019.

[28] Y. Sun, P. Li, Y. Liu, and Z. Jiang, "Feature Extraction and Clustering for Static Video Summarization," *Research Square Platform LLC,* vol. 10, no. 6, pp. 1-14, Mar 2021.

[29] T. Zhang, Y. Gao, B. Zheng, L. Chen, S. Wen, and W. Guo, "Towards distributed node similarity search on graphs," *World Wide Web,* vol. 23, no. 6, p. 3025–3053, Jun 2020.

[30] A. Kumar, S. S. Singh, K. Singh, and B. Biswas, "Link prediction techniques, applications, and performance: A survey," *Physica A: Statistical Mechanics and its Applications,* vol. 553, pp. 124-289, Sep 2020.

[31] M. Xu, "Understanding Graph Embedding Methods and Their Applications," *SIAM Review,* vol. 63, no. 4, p. 825–853, Jan 2021.

[32] Ghazal, T. M, "Performances of K-Means Clustering Algorithm with Different Distance Metrics," *Intelligent Automation & Soft Computing,* vol. 29, no. 3, pp. 735-742, 2021.

[33] P. Ayegba, J. Ayoola, E. Asani, and A. Okeyinka, "A Comparative Study Of Minimal Spanning Tree Algorithms," in *2020 International Conference on Innovations in Intelligent Systems and Applications (INISTA)*, Novi Sad, Serbia, 2020.

[34] P. Ayegba, J. Ayoola, E. Asani and A. Okeyinka, "A Comparative Study Of Minimal Spanning Tree Algorithms," *2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS)*, Ayobo, Nigeria, 2020, pp. 1-4, doi: 10.1109/ICMCECS47690.2020.240900 .