Heuristics Based Isomap

Project Group

Student : Mengoli Emanuele
Student : Florido Poka Samir Fernando

Professor Liberti Leo
Course: INF580

March 22, 2024

Overview

Introduction

Project Background

- Investigates the role of the Isomap algorithm in dimensionality reduction.
- Focus: Solving Euclidean Distance Geometry Problems (EDGP).
- Builds upon the foundational work by Tenenbaum et al. [1] and Liberti & D'Ambrosio [2].

Research Objectives

- Aim: To propose six completion algorithms for Isomap applications to EDGP.
- Goal: Enable the transformation of high-dimensional data into insightful, lower-dimensional forms.

Application and Innovation

- Practical use case: Design of a neural network model, ISOPredictor.
- Purpose: To understand and predict the dynamics of moving sensor networks.
- Method: By inferring Isomap instances.

Methodology

Procedure

- Generation of connected graphs
- Apply our 6 heuristics of Isomap on those graphs

Algorithm ISOMAP Pseudocode

Require: Simple connected weighted graph $G = (V, E, d)$
1: Complete graph G
2: Compute the distance matrix $\tilde{D} \in \mathbb{R}^{n \times n}$, where $\tilde{D}_{ij} = d_{ij}$, denoting the weight from node i to node j
3: Find the approximate Gram matrix $\tilde{B}$ of $\tilde{D}$
4: Conduct Principal Component Analysis (PCA)
5: **return** $X \in \mathbb{R}^{n \times K}$

Floyd-Warshall Completion algorithm

Algorithm Floyd Warshall Algorithm

Require: Simple connected weighted graph $G = (V \in \mathbb{R}^n, E, d)$

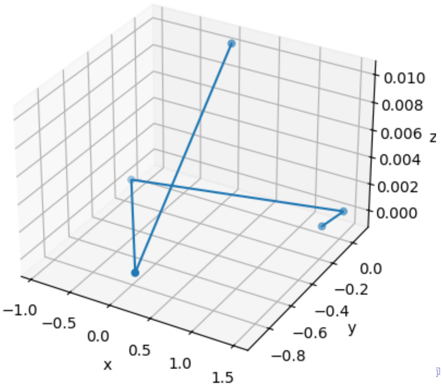
convert G to matrix D
for each $i \in V$ **do**
    for each $j \in V$ **do**
        for each $k \in V$ **do**
            if $i \neq k$ and $j \neq k$ and $i \neq j$ **then**
                if $D[i][j] > D[i][k] + D[k][j]$ **then**
                    $D[i][j] \leftarrow D[i][k] + D[k][j]$
                end if
            end if
        end for
    end for
end for
return D

Push and Pull

Another approach to compute $\tilde{D}$ is by solving the following Semi-Definite Program (SDP):

$$\max_{P} \sum_{\{u,v\}\in E} \|x_u - x_v\|^2 \quad \text{subject to} \quad \forall \{u,v\} \in E \quad \|x_u - x_v\|^2 \leq d_{uv}^2$$

Figure: Isomap obtained with Push and Pull SDP with an inital graph of 5 vertices



Breadth first search

Algorithm Modified Breadth-First Search, realization in $\mathbb{R}^K$

Require: $G(V, E)$ with weights on edges $w : E \to \mathbb{R}^+$
Ensure: Positions of nodes $P : V \to \mathbb{R}^K$, distance matrix $D \in \mathbb{R}^{n \times n}$, Z explored set

```
r ← argmax_{v∈V} deg(v)                                ▷ Node with the highest degree
P[r] ← 0                                               ▷ Position r at origin in K-space
Z ← {r}, Q ← FIFO queue initialized with r
while Q not empty do
    v ← Q.dequeue(), x_v ← P[v]
    for each w ∈ neighbors of v do
        if w ∉ Z then
            Z ← Z ∪ {w}, d_vw ← w(v, w)                ▷ Weight of edge (v, w)
            x_w ← sample from S^{K-1}(x_v, d_vw), P[w] ← x_w, Q.enqueue(w)
        end if
    end for
end while
for each pair (u, v) ∈ V × V do
    if (u, v) ∉ E and u ≠ v then
        d_uv ← ||P[u] − P[v]||_2                       ▷ Euclidean distance
        Add edge (u, v) to E with weight d_uv
    end if
end for
D ← Euclidian distances on P
return D
```

Slack/Surplus Variables (SSV)

Another technique for optimizing the objective function involves introducing slack/surplus variables. This is achieved by minimizing the expression:

$$\min_{P} \sum_{\{u,v\} \in E} s_{uv}^2 \quad \text{subject to} \quad \forall \{u,v\} \in E \quad \|x_u - x_v\|^2 = d_{uv}^2 + s_{uv}$$

K-means completion algorithm

Algorithm K-means

Require: Simple connected weighted graph $G = (V \in \mathbb{R}^n, E, d)$

Convert graph G to distance matrix D
Initialize *vertices_attained* as an empty list ▷ List to track vertices attained
Initialize *centroids_list* as an empty list ▷ List to store centroids
while $|\text{vertices\_attained}| < n$ **do** ▷ Until all vertices are attained
    $r \leftarrow \operatorname{argmax}_{v \in V} \deg(v) \setminus \text{centroids\_list}$ ▷ Node with highest degree not in centroids list
    centroids_list ← append r ▷ Add centroid to centroids list
    for neighbour $\in$ neighbours(r) $\setminus$ vertices_attained **do** ▷ For unattained neighbours of centroid
        vertices_attained ← neighbour ▷ Mark neighbour as attained
    end for
end while
for (a, b) in V **do**
    $D_{a,b} \leftarrow D(a, \text{centroid}(a)) + D(\text{centroid}(a), \text{centroid}(b)) + D(\text{centroid}(b), b)$
end for
return D

Random Path

Algorithm Random Path

```
Require: Simple connected weighted graph G = (V, E, d)
Convert graph G to distance matrix D
Choose a random vertex u as the starting point
Initialize seen as an empty list                          ▷ List to track visited vertices
seen.append(u)                                            ▷ Add starting vertex to seen list
while len(seen) < size of D do
    v ← 0, weight ← 0, Q ← List of all vertices
    Q.remove(u)                                           ▷ Remove current vertex from Q
    for each vertex k in Q do
        if D[u][k] ≠ 0 then
            Q.remove(k), v ← k, weight ← D[u][v]          ▷ Update v and weight
        end if
    end for
    while len(Q) > 0 do
        for each vertex k in Q do
            if D[u][k] == 0 and D[v][k] ≠ 0 and u ≠ k then
                Q.remove(k), v ← k
                D[u][k] ← weight + D[v][k], D[k][u] ← weight + D[v][k]
                weight ← D[u][k]                          ▷ Update weight
            end if
        end for
    end while
    u ← random vertex not in seen                         ▷ Choose a random unvisited vertex
    seen.append(u)                                        ▷ Add the chosen vertex to seen list
end while
return D
```

Real Case Application: ISOPredictor

- Designed to predict changes in the geometric configuration of a graph (e.g., a moving sensor network).
- Utilizes sequences of graph configurations represented by Isomaps as input.
- Simulates movement of network graphs with stochastic translation.
- Outputs a predicted next state with preserved isomap dimensionality.
- Trained using MSE loss comparing predicted and actual next states.

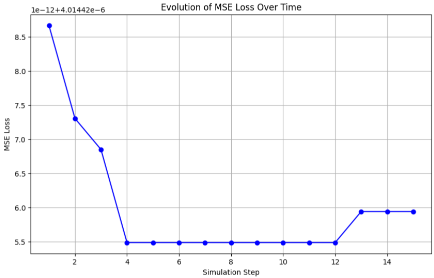
$$\boxed{\text{Input Layer } (K)} \longrightarrow \boxed{\text{Hidden Layer } (64)} \longrightarrow \boxed{\text{Leaky ReLU}} \longrightarrow \boxed{\text{Output Layer } (K)}$$

Input: Isomaps

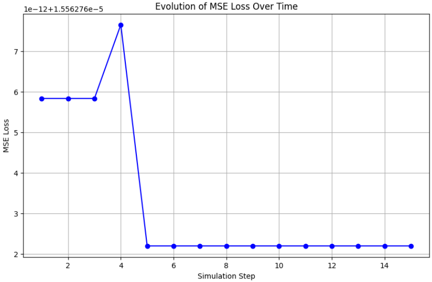
Output: Predicted State

Figure: Neural Network Architecture of the ISOPredictor

Experimental Results



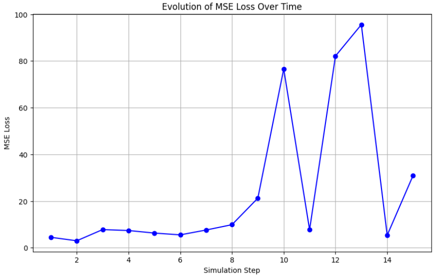
(a) MSE with constant uniform movement $\mathcal{U}(-5, 5)$ per step for all the network



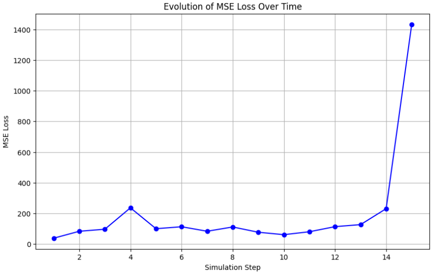
(b) MSE with constant normal movement $\mathcal{N}(2, 2.5)$ per step for all the network

Figure: Comparison of MSE with constant movements

Experimental Results



(a) MSE with random uniform movement $\mathcal{U}(-5, 5)$ per step for each node



(b) MSE with random normal movement $\mathcal{N}(2, 2.5)$ per step for each node

Figure: Comparison of MSE with random movements

ISOPredictor - heuristics comparison

Algorithm	MSE: Constant Uniform Movement Per Step	MSE: Constant Normal Movement Per Step	MSE: Uniform Movement Movement Per node	MSE Score with Normal Random Movement Per node
Floyd-Warshall	10^{-5}	10^{-6}	10^{1}	10^{3}
K-means	10^{-4}	10^{-5}	10^{2}	10^{3}
Random	10^{-6}	10^{-6}	10^{1}	10^{3}
BFS	10^{-1}	10^{-1}	10^{2}	10^{3}
Push and pull	10^{-6}	10^{-6}	10^{3}	10^{1}
SSV	10^{-6}	10^{-6}	10^{3}	10^{1}

Table: MSE Scores for Different Algorithms per order of magnitude

Conclusion

Project Summary

- Developed six Isomap-based completion algorithms to address the Euclidean Distance Geometry Problem (EDGP).
- Introduced a neural network predictor, ISOPredictor, for understanding moving network dynamics through Isomap instances.

Achievements

- The ISOPredictor achieved robust performances in learning dynamics of moving networks.
- Enabled robust inference of changes over time.

Future Directions

- Explore incorporating directional constraints into the model.
- Simulate node failures to enhance predictor flexibility.
- Apply the model to real-world logistics challenges, such as truck coordination in large-scale moving graph networks.

Q&A

References

J. B. Tenenbaum, V. d. Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

L. Liberti and C. d’Ambrosio, “The isomap algorithm in distance geometry,” in *16th International Symposium on Experimental Algorithms (SEA 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.