# Exploring substantial statistical significances in graph similarity measures

Ramraj Thirupathyraj, Prabhakar Ramachandra Gupta, and Sivakumar Palanisamy

***Abstract*—Our study aims to provide a quantized similarity score for comparison between two graphs based on various meaningful graph properties. For the purpose of such computation, graph topological structure, individual vertex influence, vertex closeness and their linkage to other vertices, and finally, various paths between vertices are leveraged to bring out crucial features that would convey the extent of resemblence between any two graphs. By utilizing these graph properties, our method calculates three quantized values with range (0,1] and fuse them together to present a similarity score *sim* between 0 and 1. The idea behind this study is based on such measures that can discriminate graphs under various conditions. Our experiments revealed the potentiality of one of these measures to identify differences in two graphs when the other two measures failed to differentiate them. By providing a graph similarity score, we strongly emphasize the strength of our method to distinguish between non-isomorphic, near-isomorphic and perfect isomorphic graphs. The empirical study conducted on the benchmark Graph Kernel dataset showed promising results to categorize a graph pair through an isormorphic score of 0.95 - 1, 0.90 - 0.95, and more.**

***Index Terms*—Graph Isomorphism, Graph Similarity, Krylov Subspace, JS Divergence, Alpha Centrality**

## I. Introduction

SEVERAL algorithms are proposed in the past years viz., VF2 [1] and SPath [2] which enhanced the traditional tree-based algorithm Ullman algorithm [3] for performance improvisation of Graph Isomorphism (GI) problem. Table 1 provides a broad classification of representative methods that solves Exact Graph Matching whereas Table 2, presents a heuristic comparison of various approaches. In Table 3, a representative list of algorithms which employ various exact graph matching methods can be found. Table 4 and Table 5 provide a list of Frequent Subgraph Mining (FSM) algorithms which employ the state-ofthe-art methods and Topological Indexing methods employed in Chemical Graph Theory respectively. However, Bipartite graph matching can be solved efficiently (in Linear time) by Linear Programming [4]. The matching problem becomes NP-Hard when considering pair-wise constraints [5].

In [6], Laszlo Babai proposed a theoretical proof that claims GI problem can be solved in quasi-polynomial time $O(exp(\log V)^{O(1)})$ whereas the previous upper bound was $O(exp(\sqrt{V \log V})$. Still the problem remains computationally expensive for discriminating very high symmetric structures [7], [8]. A variety of methods similar to [9], [10] can be seen in the literature to check GI using Maximum Common Subgraph (MCS). After finding the MCS, a cost function is associated which is used as a metric (distance) yielding a time complexity of O((V + 1)!). Alternate methods like computing Graph Edit Distance (GED) for isomorphism checking are found in the literatures [11], [5]. For unlabelled graphs, GED problem is equivalent to MCS problem [11]. There are always significant graph pairs for which GI can be resolved, unlike Knot Theory [12].

Providing a metric for GI problem to match graph similarities by quantizing the result provides more meaningful inferences [13]. Research works based on finding meaningful inferences in Graph quantization is found scarce which has promising wide spread uses viz., Drug Design, Gene Matching, finding Neurological Disorder, Protein-Protein matching, Active-Moiety findings, Tensor Image analysis, Chemo Informatics, Social Network Analysis, Machine Learning and so on [14][15][16][17][18][19] and [20].

## II. Related Work

Voluminous work on GI is dedicated to the search for an exact isomorphism between two graphs or subgraphs. But it is too restrictive of comparing graphs as it outputs TRUE for isomorphic graphs, FALSE otherwise. Hence, it is rarely used in practice, because few graphs completely match in real-world applications. GI problem has been studied from various points of view [21]. Moderately exponential Complexity algorithms [6], [16] for GI and subgraph isomorphism is $O(exp(V^{\frac{1}{2}+O(1)})$ with $V^K < m(V) < c^V$ with m(V) representing the computational measure on a problem of size V and $k > 0, c > 1$. Formally, m(V) has exponential growth if for all $k > 0$, m(V) = $\Omega(V^k)$ and for all $\varepsilon > 0, m(V) = O((1+\varepsilon)^V)$. Alternate kernel methods viz., Random walk kernels [22], [23] reduced time complexity from $O(V^6)$ to $O(V^3)$ using Sylvester equation. Several Complex Networks and Chemoinformatics applications are based on other kernel based variants viz., Cyclic Pattern Kernel [24], Edit distance based kernel [25], subtree-pattern Kernel [26], weighted decomposition kernel [27] and Linear-Time Graph Kernels [28] are scalable for smaller graphs and have scope for improvisation [29].

However, for highly topologically similar structures the problem still remains expensive to compute. Most approaches

Ramraj Thirupathyraj is with the Department of Artificial Intelligence and Data Science, Coimbatore Institute of Technology, Coimbatore, INDIA-641014. E-mail: ramraj@cit.edu.in

Prabhakar Ramachandra Gupta is with Coimbatore Institute of Technology, Coimbatore, INDIA-641014. E-mail: rpcitpr@gmail.com.

Sivakumar Palanisamy is a research project associate at Coimbatore Institute of Technology. E-mail: sivakumar.dpalanisamy@gmail.com.

have shown to be efficient for specific purposes and the provided information is limited [14]. The measures employed are confined to topological structures and needs structural differences analyzation. It is cardinal that a similarity score captures and quantifies topological similarity meaningfully, in the sense, if the similarity score between two graphs g ands is one (i.e sim(g,s) = 1), $G_g \cong G_s$, whereas if $sim(g,s) > 0.9$, it can be inferred that there exists two subgraph $G_{sub_g} \subseteq G_g$ and $G_{sub_s} \subseteq G_s \mid G_{sub_g} \cong G_{sub_s}$.

The objective of this work is to propose a computationally efficient metric to quantify graph similarities. We define a similarity metric *sim* able to identify and quantify structural differences. The *sim* associates to each topological structure a set of probability distribution functions (PDFs) representing all nodes connectivity distances, and compare them, by standard information-theoric metrics and alpha-centrality measures which goes in-line with the framework [30],[13]. It is formulated by a few distance-based PDF vectors in a three-term function in which each term gives a fraction of the similarity score. The first term ($\delta$) targets on topological-level discrimination, second term ($\Delta$) focusses on node-linkage level discrimination whereas the last term ($\omega$) emphasizes on path-level discrimination. The first term is achieved by mapping graphs in kylov subspace. The second term is formulated by comparing graphs through their network distance-distribution and capturing global topological differences. To distiguish between graphs, this measure utilizes the connectivity of each node and the way each element is connected throughout the network by looking at the nodes' distance closeness distributions. The last term employs alpha centrality measure to analyse differences in the way such connectivity occurs.

The resultant *sim* score allows us to compare graphs efficiently with higher precision. We empirically proved that our proposed method for computing graph similarity is effective as extensive computational experiments showed that *sim* do not present any counterexample when recognizing non-isomorphic structures.

## III. Methodology

### A. Mapping Graphs in Krylov Subspace

Graphs are mapped in kylov subspace and similarity between them is computed.It is expected that if $G_g \cong G_s$, they point to the same position in the subspace with zero distance between them.

Motivation Studies [31], [32] and [33] revealed that the truncated power iterative method of z-order represents a matrix as a span in z-order Krylov subspace for some appropriately chosen z. An undirected graph G = (V, E) of size $|V|$ is defined by a finite vertex set $V = V_1, V_2, ..., V_{|V|}$ and an edge set $E \subseteq (V \times V)$ with $A \in \mathbb{R}^{(V \times V)}$ being the adjacency matrix with $|.|$ denoting the cardinality of the specified graph.

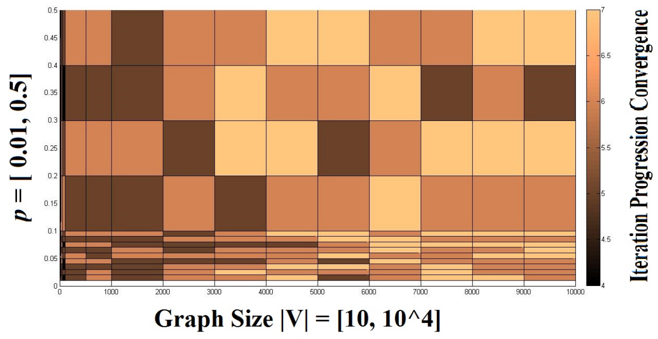


Fig. 1: Iterative Progresive Convergence of z

Attenuating A with an arbitrary vector $\omega_0 = 1 \in 1^{|V| \times 1}$ produces a z-order Krylov subspace of the form

$$\phi_G = [\phi_G^1, \phi_G^2, \phi_G^3, ..., \phi_G^z] \in \mathbb{R}^{|V| \times z} \tag{1}$$

With $\phi_G^0 = \omega_0$ and at iteration i, $\phi_G^i \in \mathbb{R}^{|V| \times 1}$ is generated as stated here:

$$\phi_G^i = \frac{A \times \phi_G^{i-1}}{(\frac{\sum A \times \phi_G^{i-1}}{|V|})} \tag{2}$$

This value represents individual vertex influence with respect to overall vertex influences after i iterations in power iterative method. Successive iterations agglutinates $\phi_G^i$ with $\phi_G$ and the process terminates after z iteration. To confine, choosing the right value of z becomes cardinal.

**Choice of z**: If z is very small (for instance, z = [2, 4]), the size of $\phi_G$ will be very small which adversely affects the embedded features of the graph representation and for $Z > |V|$, the computational complexity for generating becomes high. Hence, an iterative convergence approach is adopted for choosing the right value of $z$. As $\phi_G^i \in \mathbb{R}^{|V| \times 1}$ is appended, the size of $\phi_G$ changes from $\mathbb{R}^{|V| \times (i-1)}$ to $\mathbb{R}^{|V| \times (i)}$. For faster convergence, we set the stopping criteria in this work as $\sum(\phi_G^i - \phi_G^{i-1}) \leq \Delta$ with $\Delta = 0.01$.

For analyzing the convergence in this computational setup, a set of graphs of sizes varying within the set $|V|$ = 10, 25, 50, 100, 200, 500, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000 with wiring index (p) ranging between p = [0.01,0.5] are chosen. From this experimental study, we found that $z$ value ranges between [4,8]. 1 shows that the convergence is very fast with z=[4,8], irrespective of the size of $|V|$ and the iteration progression convergence is closely related to ratio of $\lambda_1$ and $\lambda_{|V|+}$, where $\lambda_i$ is the $i^{th}$ dominant eigen value and $\lambda_{|V|+}$ is the set of all non-negative eigen values [33]. From application perspective (viz., brain network connectivity) [34] for compaing any two connected graphs of arbitrary sizes, a common fixed size representation is necessary to provide a direct comparison.

### B. Graphs as fixed size representations

As the interest is in modelling a set of vectors to best fit the resultant model $\phi_G$, the crux is in finding a maximum

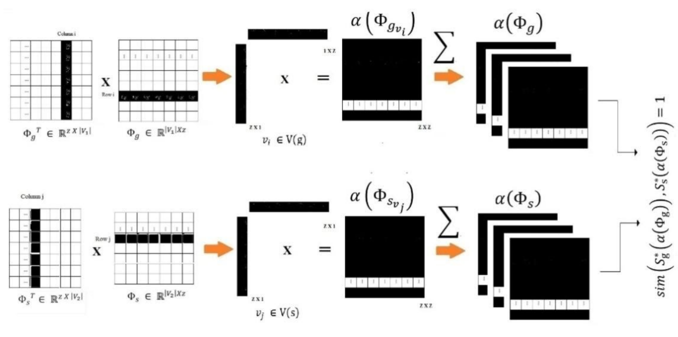

Column i
Row i
$\Phi_g^T \in \mathbb{R}^{z \times |V_1|}$
$\Phi_g \in \mathbb{R}^{|V_1| \times z}$
1XZ
ZX1
ZXZ
$v_i \in V(g)$
$\alpha(\Phi_{g_{v_i}})$
$\alpha(\Phi_g)$
Column j
Row j
$\Phi_s^T \in \mathbb{R}^{z \times |V_2|}$
$\Phi_s \in \mathbb{R}^{|V_2| \times z}$
$v_j \in V(s)$
$\alpha(\Phi_{s_{v_j}})$
$\alpha(\Phi_s)$
$sim\left(S_g^*(\alpha(\Phi_g)), S_s^*(\alpha(\Phi_s))\right) = 1$

Fig. 2: The direct comparative model for Graph Similarity check based on topological-level discrimination.

likelihood Gaussian distribution. In this brief, the distribution function $\alpha : \mathbb{R}^{|V| \times z} \rightarrow \mathbb{R}^{z \times z}$ can be given as

$$\alpha(\phi_G) = \sum_{v_i \in G} ((\phi_G(i,:)^T - \mathbb{1}) \times (\phi_G(i,:) - \mathbb{1})) \in \mathbb{R}^{z \times z} \quad (3)$$

with $\phi_G(i,:) \in \mathbb{R}^{1 \times z}$ being the ith row of $\phi_G$ as it represents the influence proliferation of vertex i towards its neighbours. As $\alpha(\phi_G)$ proliferates vertex influences, it is positive semi-definite (PSD)[1].

**Lemma:** For any given permutation matrix P with 1 as a unit vector, we have $\alpha(\phi_G) = P.\alpha(\phi_G).P^T$ is a graph invariant.

*Proof.* Considering u and v as two vectors of equal size, it is known Cov(u,v)=Cov(P.u,P.v) $\forall u, v$ $(PP^T)_{ij} = \sum_{x=1}^{n} P_{ix}.P_{xj}^T = \sum_{x=1}^{n} P_{ix}.P_{jx}^T$
However, when $P_{ix} = 0$, $P_{ix}.P_{jx} = 0$ as well. Generally, $P_{ix} \neq 0$, but then $\nexists i^z \neq i : P_{i^z x}$ is non zero i.e., i is the only row with 1 in column x and $PAP^T = PAP^{-1}$. Based on the above settings, we have $(PAP^T)^2 = ... = PA^2P^T$. Similarly for any value of z, it is possible to show that

$$(PAP^T)^Z = PA^ZP^T \Rightarrow (PAP^T)^Z \times \mathbb{1} = PA^ZP^T \times \mathbb{1} = PA^Z\mathbb{1}$$

$$\begin{aligned} \alpha(\phi_G)_{i,j}^{PAP^T} &= Cov((PAP^T)^i \times \mathbb{1}, (PAP^T)^j \times \mathbb{1}) \\ &= Cov(PA^i\mathbb{1}, PA^j\mathbb{1}) \\ &= Cov(A^i\mathbb{1}, A^j\mathbb{1}) \\ &= \alpha(\phi_G)_{i,j}^{A} \quad \forall i, j \end{aligned}$$ ■

Both Euclidean and non-Euclidean kernel methods viz., (Log Euclidean Riemannian Manifolds) LERM [35], Log Divergence [35] can be adopted as direct comparative models to compute topological level discrimination. Among them, in this work, Bhattacharya Kernel function $D_B$ is chosen as it ignores Riemannian manifolds in $\alpha(\phi_G)$ [36], [35]. We compute the distance between the two Probability Distribution Function (PDF) computed using the $\alpha(\phi_G)$ of the two graphs, where $\phi_G$ represents the z-order krylov subspace. The Bhattacharya distance $D_B$ is given as below

$$D_B(\alpha(\phi_{G_1}), \alpha(\phi_{G_2})) = \frac{1}{2} \ln \frac{det|\frac{\alpha(\phi_{G_1}) + \alpha(\phi_{G_2})}{2}|}{\sqrt{det|\alpha(\phi_{G_1})|.det|\alpha(\phi_{G_2})|}} \quad (4)$$

where $det|.|$ denotes the determinant of a matrix. The first term of Bhattacharya Distance representing Mahalanobis distance becomes 0 since the mean values of the two distributions are equal and will be an unit vector. Since, our study aims to provide a quantized value to represent the similarity between two graphs, the following equation is applied for our first term.

$$\delta = exp^{-(\frac{1}{\gamma})*(D_B(\alpha(\phi_{G_1}), \alpha(\phi_{G_2})))} \quad (5)$$

where $\gamma$ factor is introduced to lessen the effect of exponential usage since plain exponential values decrease heavily even a for a small Bhattacharya distance ($D_B$). For instance, for a distance value of 0.1, $exp^{-D_B}$ dropped to 0.90, which is undesirable since near-isomorphic graphs could be 90% to 99% similar. In our analysis, Bhattacharya distance-$D_B$ measure values were 0.139 and 0.155 after removing 2 edges and 3 edges from the reference graph with 20 nodes. For such $D_B$ values, respective $exp^{-D_B}$ were 0.87 and 0.856 which is a highly unlikely value for highly similar graphs. Therefore, we stress the importance of $\gamma$ factor in adjusting the effect of exponential on *sim* score. We used $\gamma = 0.25$ in our computations but a sensitive analysis is yet to be conducted. The score given by equation 1 ranges between (0,1] and its

**Algorithm 1:** $Vertex\_Influence\_Diff(G_1, G_2)$

```
$\phi_{G_1} = compute\_phi\_G(G_1)$; %from Eq. 1 and 2
$\phi_{G_2} = compute\_phi\_G(G_2)$;
$\alpha(\phi_{G_1}) = compute\_alpha\_phi\_G(G_1)$; %from Eq. 3
$\alpha(\phi_{G_2}) = compute\_alpha\_phi\_G(G_2)$;
$distance = D_B(\alpha(\phi_{G_1}), \alpha(\phi_{G_2})))$; %from Eq. 4
$\delta = exp(-(distance))$%from Eq. 5
return $\delta$;
```

computational complexity is $O(V_G.z^2 + E_G.z + z^{2.373})$ which still polynomial time ($P_{TIME}$) [35]. Hence, $\delta$ can be used as a direct comparative model to ascertain the topological similarity between graphs. Our Empirical study in the section 3.3 shows the effects of equation 1 on random graphs. However, it can be shown that this intuition is not entirely correct unless enhanced further, which can be ascertained by the following lemma.

**Lemma:** For $G_1 \cong G_1$, $\delta$ always yields 1, whereas the converse is not true.

*Proof.* As $\phi_G$ focusses on vertex-influence proliferation, the similarity scores between highly similar non-isomorphic graph pairs (for instance, (a, b), (c, d) and (e, f)) as shown in Fig. 3 yield 1 which can be ascertained by a primitive computational study as well. ■

[1]Some of the covariance matrices ($\alpha(\phi_G)$) were not PSD in our analysis. We added a very small value (0.000015) to the diagonal elements of $\alpha(\phi_G)$ to maintain them as PSD. Further, because of this minuscule addition, we believe that we don't alter anything in this study

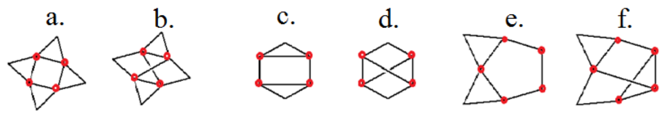


Fig. 3: Graphs Pairs with high topological similarity [(a,b), (c,d), (e.f)]

### *C. Empirical Results - The motivation to look beyond Vertex-Influence*

We conducted an experiment to check the above Lemma with n different graph clusters C=$c_1, c_2, .., c_n$ with varying graph sizes $|V|$ = [10,100]. While graphs within a cluster $c_i \in C$ are of same size, they are only near-isomorphic implying their high topologically similarity and varying edge connectivity. The inferred results presented in the below figure 4 shows many False-Positive values having a similarity score of 1 representing perfect-isomorphism while they are not. The results of this experiment validated the above Lemma and motivated us to study further to differentiate near, perfect and non isomorphic graphs. We found that false-positives are due to the fact that $\alpha(\phi_G)$ focusses on the graph topological structure of G but fails to identify how a vertex is close to another vertex within G in terms of influence proliferation. Hence an enhanced measure based on Information-Theoretic approach is formulated from $\phi_G$.

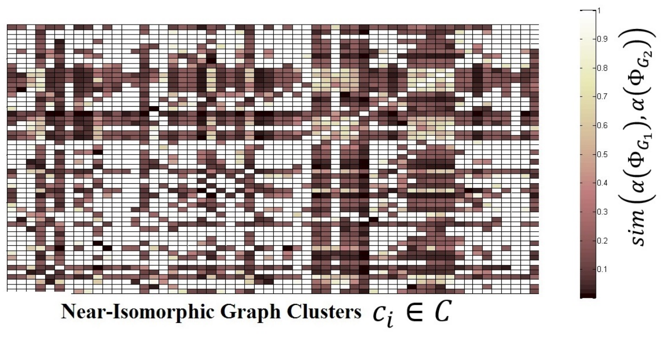


Fig. 4: Similariy Scores of Near-Isomorphic graph clusters by considering $\delta$ alone

### *D. A New Measure NNCD Network Node Closeness Distribution*

**Motivation** As $\phi_G$ reveals vertices proliferation towards their neighbours, $\phi_G^i = \phi_G^i(j), 1 \leq j \leq |V|$ denotes the proportion of influence pertaining to vertex *j* after *i* iterations. Hence, it can be considered as a PDF element similar to the studies [30], [13] where every column vector of G is used as a PDF of different weights. Particularly, $\phi_G^i(j)$ gives the fraction of total influence distribution of vertex *j* at the $i^{th}$ iteration. In brief, the z column vectors of the span $\phi_G = [\phi_G^1, \phi_G^2, ..., \phi_G^z]$ gives the vertex-influence distribution which embeds the given graphs structural information.

For two near similar graphs, $\delta$ may be equal to 1 but vary with their vertex-closeness measure. In this context, the term closeness corresponds to how one vertex is close to another in terms of its influence proliferation towards its neighbours. To measure vertex closeness, we consider the concept of divergence. Among various standard Information-Theoretic divergence measure approaches (viz., I-divergence and J-divergence), we stayed in-line with the framework [36] to utilize the Jensen-Shannon (JS) Divergence for meauring vertex-closeness since it facilitates assigning different weights (influence, in this case) to each probability distribution.

The JS-Divergence is formulated as below:

$$JSD_G(\phi_G^1, \phi_G^2, ..., \phi_G^z) = \frac{1}{|V|} \times \sum_{i,j} \phi_G^i(j) log(\frac{\phi_G^i(j)}{\mu_j}) \quad (6)$$

where $\mu_j = \frac{\sum_{i=1}^{|V|} \phi_G^i(j)}{|V|}$ represents the average of the vertex-influence distributions of the $|V|$ vertices.

We introduce a new quantized measure known as *Network Node Closeness Distribution - NNCD* by normalizing the JS-Divergence of $\phi_G$ with $log|V|$. The NNCD value of a given graph can be calculated using its $\phi_G$ as:

$$NNCD = \frac{JSD_G(\phi_G^1, \phi_G^2, ..., \phi_G^z)}{log|V|} \quad (7)$$

The $\phi_G$ for NNCD calculation is computed using power iterative method from equation 2 but with $\phi_G^0 = \omega_0 = TW(G)$, where TW(G) is a vector holding topological weights of nodes in a graph G. The topological weight TW(G) is computed through the below algorithm.

**Algorithm 2:** *compute_topological_weight*$(G)$

```
$w\_matrix$ = zeros($|V| \times |V|$); %weight matrix
for $i$ = $1$ : $max\_nodes$ do
    for $j$ = $(i+1)$ : $max\_nodes$ do
        $vec\_i$ = G(:,i); %node_i_connections
        $vec\_j$ = G(:,j); %node_j_connections

        diff = sum(XOR($vec\_i, vec\_j$));
        $w\_matrix(i,j)$ = $w\_matrix(j,i)$ = diff;
    end
end
tw = sum($w\_matrix$); %row-wise addition
return tw;
```

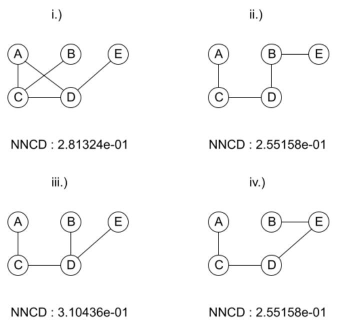


Fig. 5: Graphs with same number of nodes but different connectivity

As per the above formulation, if p value is higher, NNCD(G) will be low and signify G as strongly connected. If vertices of G are isolated (p = 0), NNCD score becomes infinity and in a completely connected graph (with p = 1), NNCD score becomes zero. An introductory example to ascertain the above statement is presented in figure 5 (i.) to (iv). The presented graphs possess equal number of nodes, however, their NNCD scores are varied depending upon their connectivity. In simple terms, the NNCD score will be low if a graph is well connected.

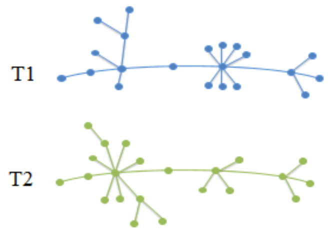


Fig. 6: Graphs with varying node linkage and vertex closeness

Additionally, in figure 6, the vertices closeness $T_2$ is much higher than $T_1$ which results in $NNCD_{T_2} < NNCD_{T_1}$. A brief study on the influence of edges and vertices removal on NNCD scores is presented in Supplementary note. Based on (2), a new distance measure can be formulated as difference between the graphs averaged NNCD values.

$$\Delta = exp^{-(\frac{|sqrt(NNCD_{G_1})-sqrt(NNCD_{G_2})|}{2})} \tag{8}$$

The complete steps involved in computing $\Delta$ is packed in the Algorithm-2 given below.

**Algorithm 3:** $Node\_Connectivity\_Diff(G_1, G_2)$

```
$TW(G_1) = compute\_topological\_weight(G1)$;
$TW(G_2) = compute\_topological\_weight(G2)$;

$\phi_{G_1} = compute\_phi\_G(G1, \omega_0 = TW(G_1))$; %from Eq.1 & 2
$\phi_{G_2} = compute\_phi\_G(G2, \omega_0 = TW(G_2))$;

$JSD(\phi_{G_1}) = JS\_Divergence(\phi_{G_1})$; %from Eq. 6
$JSD(\phi_{G_2}) = JS\_Divergence(\phi_{G_2})$;

$NNCD\_G_1 = comp\_NNCD(JSD(\phi_{G_1}))$; %from Eq. 7
$NNCD\_G_2 = comp\_NNCD(JSD(\phi_{G_2}))$;

return $\Delta(NNCD\_G_1, NNCD\_G_2)$; %from Eq. 8
```

A simple experiment conducted to emphasize the significance of NNCD score showed promising results as presented in the Table I since NNCD scores discern the discrimination between similar graphs presented in Figure 3 for which $\delta$ values are exactly 1.

| **Properties** | **Graphs** | | | | | |
|---|---|---|---|---|---|---|
| | **a** | **b** | **c** | **d** | **e** | **f** |
| NNCD | 0.1117 | 0.1098 | 0.0422 | 0.04863 | 0.1384 | 0.0657 |
| $\delta$ | 1 | | 1 | | 0.94376 | |
| $\Delta$ | 0.9986 | | 0.9924 | | 0.94379 | |

TABLE I: NNCD Discrimination

Inferences from [11], [13] indicate that distance based on a metric (similar to NNCD) for most k-regular graphs (graphs in which all nodes have degree k) as 0. The following heat map shows the discrimination of k-regular ( $K3-K20$ ) based upon Eq.2 The results indicate that the similarity score is higher towards the diagonal and categorized graphs as similar. The discretion of similarity scores based on NNCD values is due to the fact that NNCD captures the Node-Linkage Connectivity patterns alike topological structural similarity identification approach presented in Eq. 1.

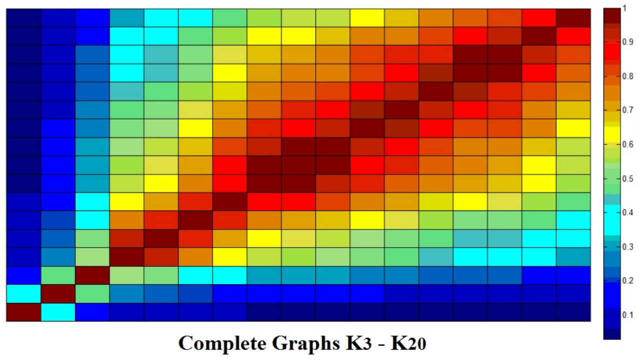


Fig. 7: Complete Graphs K3 - K20

## E. *Effects of NNCD on Erdös-Rényi (ER) Network Graphs*

Using standard network models, the properties of NNCD are illustrated with an empirical study by considering a set of Erdös Rényi (ER) networks, Lovász, L & Paul Erdös (1993), generated by randomly connecting pairs of vertices with wiring probability $p$. The objective of this study is to investigate whether NNCD scores produce any substantial statistical significance or not. Graphs with $|V| = [25, 500]$ are chosen with $p = \frac{1}{log|V|}$, so that no vertex becomes isolated (Refer Supplementary).

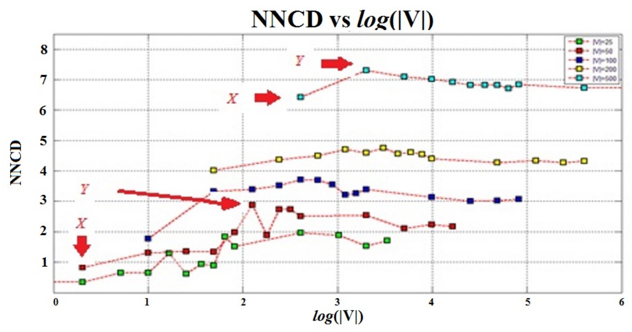


Fig. 8: NNCD vs $log|V|$

In general, for a low value of p, the graph consists of a set of small connected components and as p increases above a particular threshold value $P_T = \frac{1}{N}$, the graph collapses resulting in a large connected component. The proerties of NNCD are studied for ER network model. Figure 8 represents the average results of 50 ER network independent (directed Graphs) of various sizes against the rewiring probability ranging between (P = 0.01 to 0.50) in the logarithmic scale.

It is observed that NNCD values increase gradually in accordance with the wiring probabilities and attain the peak at a critical value, $p_c = \frac{1}{logN}$ beyond which the NNCD values become stable as all vertices of graphs are closer to each other inferring that all the nodes will collapse into a single large connected component. For $|V| = 50$, the corresponding X and Y indications represent the low and high NNCD values for the corresponding wiring probabilities p = 0.01 and 0.25(i.e. $\frac{1}{log50}$ ). Similarly, for $|V| = 500$, the peak value $Y$ is attained when p=0.16 (i.e. $\frac{1}{log500}$ ). Above $p_c$ value, the network collapses in a single large connected component and hence the NNCD becomes stable corresponding to the percolation transition. Based on the inference, NNCD scores exhibit substantial statistical significances in studying real-world graphs and in graph comparisons. When p is 0.01, the graphs possess many isolated vertices, however, individual NNCD scores vary.It is also investigated whether the scores discriminates the topologically similar graphs with in various clusters of C. X.X depicts the discrimination shown by NNCD for these graphs with in a cluster possessing $\Delta$ close to 1, but not 1.However, the scores between inter clusters are nominally high.

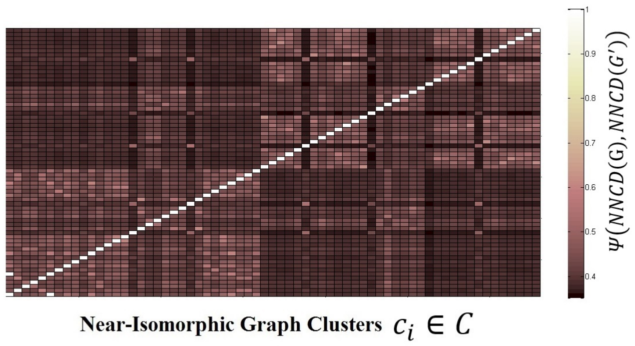


Fig. 9: Near Isomorphic Graph Clusters

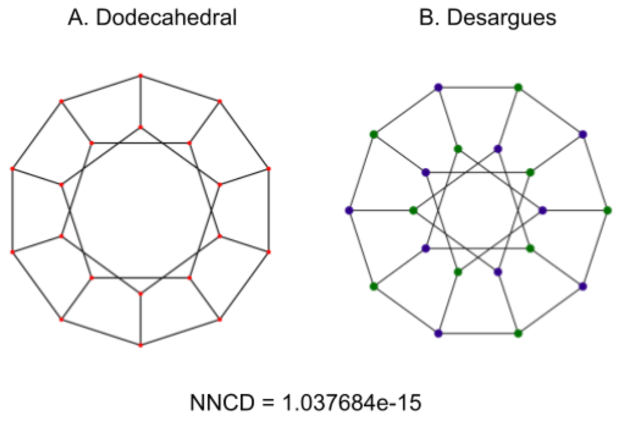


Fig. 10: Examples of pair of graphs that have the same NNCD values that depicts their same Closeness distribution

In brief, it is evident that $\Delta$ exhibits substantial statistical significance along with GI checking. Differing from closeness centrality, NNCD may be suggested as a straight forward method to assist in a few Graph Similarity Check applications. However, for special cases of graph variants (Desargues and Dodecahedral, for instance), Eq. 2 fails to discriminate as they have similar topological structures with different connectivity patterns. The cardinal reason for this indiscriminaton is the NNCD score focuses on the influences of vertices over others but unable to identify specificity of the same. Hence, it fails to discern the difference in the node-linkage distribution.

### *F. Global Path-Level Connectivity Pattern in Graphs*

The global connectivity pattern reveals the overall connectivity in the graph. Hence, the problem zeros down to considering the differences in the node-centrality values and can be computed using $\alpha$-centrality measure. In this work, the $\alpha$-centrality vertex measure is considered for path-level discrimination. Our experiments revealed the significance of this measure for graph pairs not distinguishable by the $\Delta$. Hence, we consider comparing $\alpha$-centrality values of graphs and their complements [37] along with NNCD values differences (Refer Supplementary).

The $\alpha$-centrality of a graph G is computed as below:

$$\alpha(G) = (I - alpha * A^T)^{-1} * e \tag{9}$$

where $alpha$ is the attenuation factor, A is the adjacency matrix, I is the Identity matrix and $e = \mathbb{1}^{|V|\times 1}$ is the external influence in alpha-centrality calculation. To maintain the optimal value of $alpha$ for a proper balance between external influence and node connectivity influence, the value of $alpha$ is set to 95% of $\frac{1}{\lambda_A}$, where $\lambda_A$ is the absolute value of the largest eigenvalue.

**Algorithm 4:** $Alpha\_Centrality\_Diff(G_1, G_2)$

```
$\alpha(G_1) = alpha\_centrality(G_1)$; % from Eq.5
$\alpha(G_2) = alpha\_centrality(G_2)$; % from Eq.5

$sorted\_alpha\_c1 = sort(\alpha(G_1))$;
$sorted\_alpha\_c2 = sort(\alpha(G_2))$;

$max\_nodes = max(|V|_{G_1}, |V|_{G_2})$;
i=0;
diff=0;

for $i$ = $1$ : $max\_nodes$ do
    if $(i > |V|_{G_1})$ $or$ $(i > |V|_{G_2})$ then
        diff += 1;
        continue;
    else
        if $sorted\_alpha\_c1(i) \neq sorted\_alpha\_c2$ then
            diff += $|$ $sorted\_alpha\_c1(i) - sorted\_alpha\_c2$ $|^2$
        end
    end
end
return (diff / $max\_nodes$);
```

[2]We added a minimum penalty of 0.01 even if the absolute difference is lesser than 0.01.

The $\alpha(G) \in R^{|V|\times 1}$ will provide centrality values of all nodes of a graph G. To compute alpha centrality difference between the two given graphs $G_1$ and $G_2$, we first sort the alpha centrality vectors $\alpha(G_1)$ and $\alpha(G_2)$ computed using the above formula and find the difference between the alpha centrality value of every nodes. For graphs with unequal number of nodes i.e $|V|_{G_1} \neq |V|_{G_2}$, we penalized their similarity score by increasing their alpha centrality difference by 1 for every node that doesn't have a match in the other graph. The difference value is then normalized by the maximum number of nodes in the two graphs (max($|V|_{G_1}, |V|_{G_2}$)).

As mentioned above, in addition to alpha centrality difference between graphs, their path-level dicrimination is improved by considering their complements also.

**Algorithm 5:** $Total\_Alpha\_Centrality\_Diff(G_1, G_2)$

```
$reg\_diff = alpha\_centrality\_diff(G_1, G_2)$;% Algorithm-3
% Graph Complements $-G_1', G_2'$
$comp\_diff = alpha\_centrality\_diff((G_1', G_2')$;
$total\_\alpha\_diff = reg\_diff + comp\_diff$;
return $total\_\alpha\_diff$;% $\omega => Eq.10$
```

The final term of our graph similarity score computation is formulated using the total alpha centrality difference from the Algorithm 2 as:

$$\omega = exp(-(total\_\alpha\_diff)/2) \tag{10}$$

| Graph-Pair | $\delta$ | $\Delta$ | $\omega$ | sim score |
|---|---|---|---|---|
| Desargues - Dodecahedron | 1 | 1 | 0.9900 | 0.9980 |

TABLE II: Alpha-Centrality Discrimination

### G. The Similarity Score

The final score representing the amount of similarity between two graphs is calculated from the three terms explained in the past subsections. Arbitrary weights $\eta_1, \eta_2, \eta_3$ are assigned to each term such that $(\eta_1 + \eta_2 + \eta_3) = 1$. For our experiments, we picked $\eta_1 = 0.4, \eta_2 = 0.4, \eta_3 = 0.2$. The weight sensitive analysis of the three arbitrary weights are performed and the insights from such experiments are provided in the supplementary section. The values from Eq.1, Eq.2 and Eq.8 are used to compute the final similarity score in the following equation:

$$sim(G_1, G_2) = \eta_1 * \delta + \eta_2 * \Delta + \eta_3 * \omega \tag{11}$$

The following figure depicts $sim(G_1, G_2)$ between various graphs of *C*. Along with isomorphism check, inter-cluster similarity scores are very high (ranging between 0.8 and 0.95), whereas intra-cluster scores are comparatively low (less than 0.5). This indicates that graphs possessing $sim(G_1, G_2) > 0.9$ appear to have subgraphs which possess good candidate for GI or sub-Isomorphism.

**Algorithm 6:** $Similarity\_Score(G_1, G_2)$

```
$\delta\ = Vertex\_Influence\_Diff(G_1, G_2)$; % algo. 1
$\Delta = Node\_Connectivity\_Diff(G_1, G_2)$; % algo. 2
$\omega\ = Total\_Alpha\_Centrality\_Diff(G_1, G_2)$; % algo. 4

return $\eta_1 * \delta + \eta_2 * \Delta + \eta_3 * \omega$; % Eq. 11
```

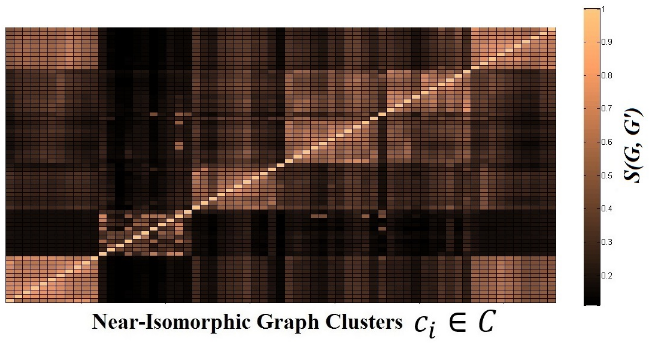


Fig. 11: Near Isomorphic Graph Clusters $c_i \in C$

## IV. Graph Similarity Score - Empirical Analysis, Results and Observation

Our experimental analysis is threefold. In the first set of experiments, our method was examined using the benchmark Graph Kernel Datasets to show how it would classify a real-world graph pair based on our *sim* score. We present a subsection illustrating a summary of our insights and results from this experiment performed on 9 datasets from [38]. Our second setup signified the validity of our method and the results from this setup justified the sim score's aptness for finding near-isomorphic graphs. In the final step of our empirical study, our method showed impeccable strength in predicting perfect isomorphic graphs. The detailed description of our experiments and their respective results are presented in the following subsections. Overall, our method showed impressive results to discriminate between non-isomorphic and find perfect isomorphic graphs.

Our implementations and computations were carried out in MATLAB code. All functionalities of our method are either self-implemented or available by default since no additional packages were used for running our code. We utilized Google Colab to perform our first experiment because of the huge and time-consuming computations. The rest of our studies were performed in Octave software in a computer with *Intel Core i7-10510U CPU @ 1.80GHz* $\times$ *8* processor, *Ubuntu 18.04.6 LTS* OS, and 16GB RAM.

### A. Experiment with benchmark Graph Kernel Datasets

As we are interested in graph isomorphism analysis rather than graph classification, only the topological structures of graphs were considered. Hence, after neglecting the node features, node label, edge features and edge labels, experiments were performed on 2000 graphs from the AIDS dataset from [38]. In our setup, graphs from same classes and across classes were compared, which resulted in a total of 1999000 graph comparisons.

From our results presented in the figure 12, most of the comparisons between two classes (i.e. class-0 and class-1) of the AIDS dataset had similarity scores in the range $< 0.8$ and $0.85-0.85$. We observed a very minute amount of inter-class graph pairs with similarity scores between 0.85 and above.

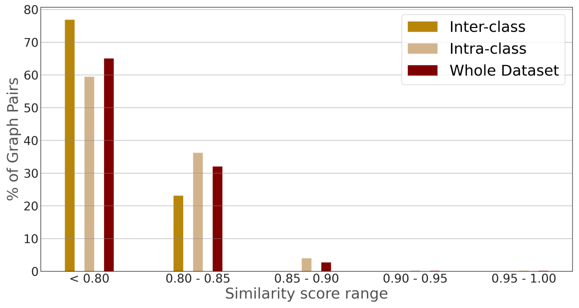


Fig. 12: Observations for the AIDS dataset

From the figure 12, no two graphs across two classes had a similarity score above 0.9. This result confirmed the dissimilarity between graphs from two different classes of the same dataset. Our method showed good performance as far as intra-class graph comparisons are concerned. While only 0.03% of graph comparisons between two classes had similarity scores greater than 0.85, we noticed 4.36% of intra-class graph pairs to have a sim score of 0.85 and above. In plain numbers, around 59252 same class graph pairs in comparison to 192 different class graph pairs produced a sim score of above 0.85. To add more positivity to our method, in the 0.8 and less sim score range, per cent of graph comparisons between classes (76%) was higher than that of the graph pairs of the same type (59%). The high number of near-isomorphic intra-class graph pairs might be due to the presence of more sub-isomorphic structures present between such graphs, which could have resulted in a high score in our analysis. A comparative analysis of the state-of-the-art Machine Learning classifiers and similarity scores from our method will be an interesting study to carry forward in the future. Such study could provide us more insights about well-classified data and outliers and can direct us towards understanding those classifications through our scores.

The results for the nine datasets' inter-class graph comparisons from our isomorphic study are provided in the stacked bar graph (Figure.13). We noted a obervable percentage of inter-class graph pairs with high *sim* score of $0.9-1.00$ in the datasets $MSRC\_9$, $MSRC\_21C$ and $Synthei$, which might be due to the presence of many classes in those datasets. Our intuition is that graphs that are very close in the mathematical space will have a high similarity score *(sim)* and datasets with many classes will be more likely to contain a notable amount of graphs at the separation. The graphs at the separations between classes will be close in the mathematical space, which was reflected in our experiments as inter-class near isomorphic graph pairs with high scores. For instance, 8 classes from $MSRC\_9$ and 20 classes from $MSRC\_21C$ supported our view.

II.

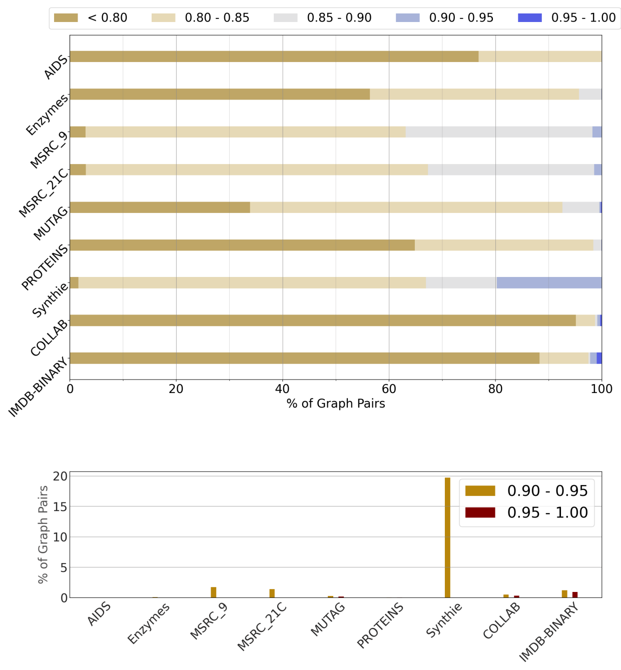


Fig. 13: Inter-class Graph Similarity Score results for the 9 datasets

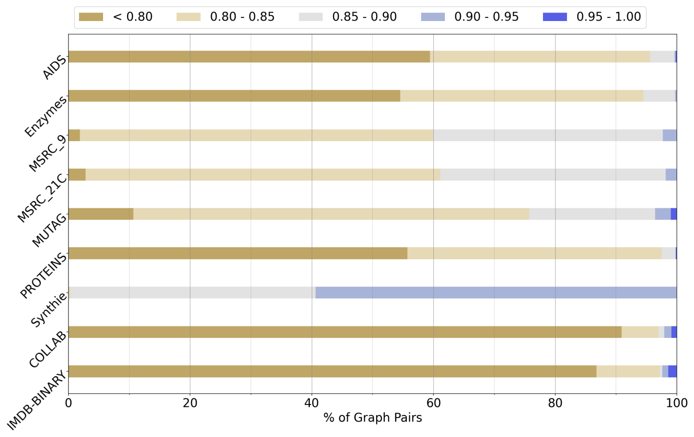


Fig. 14: Intra-class Graph Similarity Score results for the 9 datasets

In both figures 13 and 14, it is clear that a large number of graph pairs can be classified under $< 0.80$ and 0.80-0.85 groups. This is because of our experimental setup where, in every dataset, we compared one graph with all other graphs. For a graph G, the number of graphs that are near to G would be infinitesimal and negligible compared to the number of graphs that are far away from G, which supported our results. Further studies, as mentioned before, could perform graph classifications with Graph Neural Network to provide a plausible explanation for our view about the inter-class graph similarity scores. A summary of our method's evaluation on whole datasets (for the chosen 9) is given in the table

| Dataset | $< 0.80$ | **0.80 - 0.85** | **0.85 - 0.90** | **0.90 - 0.95** | **0.95 - 1.00** |
|---|---|---|---|---|---|
| AIDS | 65.02 | 32.01 | 2.72 | 0.12 | 0.13 |
| Enzymes | 56.08 | 39.43 | 4.36 | 0.13 | 0.0 |
| $MSRC\_9$ | 2.82 | 59.93 | 35.45 | 1.81 | 0.0 |
| $MSRC\_21C$ | 2.99 | 63.81 | 31.75 | 1.45 | 0.0 |
| MUTAG | 21.08 | 62.21 | 14.53 | 1.56 | 0.63 |
| PROTEINS | 60.1 | 37.83 | 1.91 | 0.05 | 0.1 |
| Synthei | 1.23 | 49.05 | 20.07 | 29.64 | 0.0 |
| COLLAB | 93.44 | 4.59 | 0.64 | 0.79 | 0.54 |
| IMDB-BINARY | 87.55 | 9.82 | 0.35 | 1.09 | 1.19 |

TABLE III: Graph Comparison results on Whole Datasets (Results in Percentage)

## B. Experiment with custom dataset

In the second experiment, our custom datasets with 20, 30, 40, 50 and 60 nodes were empl, wherein a graph G from every such cluster was exploited to generate subsequent graphs by removing 1,2,3,4,5 and 10 random edges. In the rest of this subsection, whenever we use the term *reference graph*, we simply mean the graph G of the corresponding dataset. While our first experiment proved the performance and efficiency of our method in graph isomorphism and graph classifications for real-world graphs, our intention in this evaluation was to provide a better picture about the functionality of our method since the previous experiments only revealed the effectiveness of our similarity score. At this point, we are aware that the validity of those results is yet to be established here.

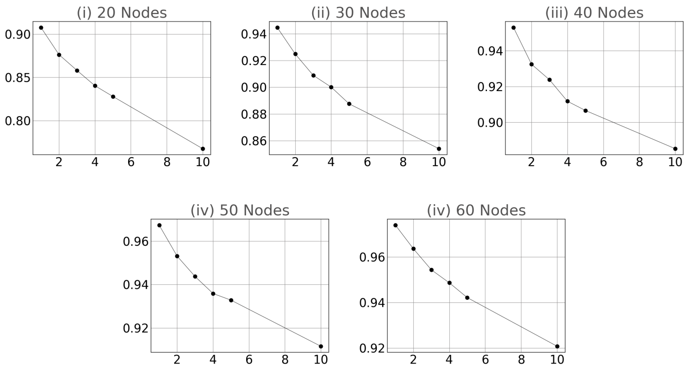


Fig. 15: Results of study with custom datasets. In all plots, x-axis and y-axis respectively represents the number of edges removed from the reference graph and the average similariy score (sim)

To generalize well, for every number of edge removal, viz. 1,2,3,4,5 and 10 edges, we repeated our analysis upto 100 times by removing random edges and comparing them with their reference graph G. Assuming two graphs $G_1$ and $G_2$ with $G_1 = G_2$, one can accept that the percentage of similarity between $G_1$ and $G_2$ decrease with increasing number of edge removals from one of the two graphs. In an effort to confirm if our results are analogous to this expectation, the primary attention was on the mean values of *sim* scores from 100 iterations.

The figure 15 provides information about our experimental results where it can be clearly seen from all the plots that *sim* score dropped from left to right. This confirmed the fact that some of the edge removals shattered certain sub-isomorphic patterns that existed between the reference graphs and the graphs after removing edges. Further, one can notice a connection between the number of nodes, similarity scores and the number of edges removed. For instance, after 10 edge removals, *sim* score in our setup for 20 node graphs and 60 node graphs are 0.76754 and 0.92076 respectively. The fraction of total edges removed was higher in the former than in the latter which resulted in a significant *sim* score difference. On examining the average sim scores from the figure 15 and their corresponding median values, it was obvious that their difference was trivial, which explained the consistency of *sim* scores from 100 iterations. The figure 16 explained this mean and median comparison for *sim* scores from the experiment with graphs having 20 and 60 nodes. Similar oberservations were made with graph comparisons from other clusters also, but for the convenience of reprensentation, only the results for 20 and 60 node graphs were presented here.

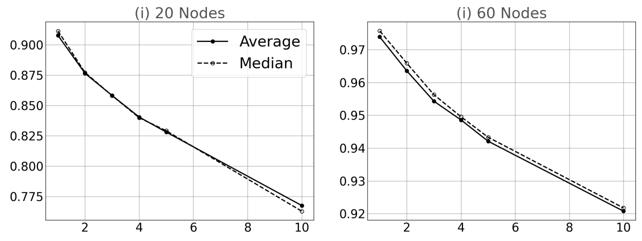


Fig. 16: Mean-Median comparison for similarity scores of 20 node and 60 node graphs. In both plots, x-axis represents the number of edge removals and y-axis represents *sim* scores. Legend and axes labels apply to both plots.

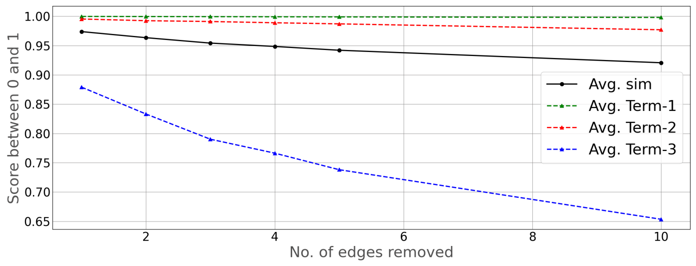


Fig. 17: Similarity score and term-wise split-up for 60 nodes dataset

On further exploration to understand the reason behind the decline in *sim* score with edge removals, we found that the edge removals tend to disturb paths between nodes that had a direct impact on $\omega$ in our method. Clearly, edge removals affected topological structure and vertex closeness in graphs as seen in the figure 17 where the $\delta$ and $\Delta$ values dropped from left to right. However, edge removals had greater effect on node connectivity than topological structure and vertex closeness.

We strongly stress that the results of edge addition will necessarily be similar to that of the edge removal but in entirely reverse order. By stating revere order, we imply that the new edge inclusions could increase vertex closeness and connectivity and eventually increase the dissimilarity between graphs. Hence, leaving edge additions aside, we targeted reference graphs comparison with other graphs containing same number of edges. To retain same number of edges, some of the edges present in reference graphs were altered by removing *n* edges and adding *n* edges. This experiment was carried out for the 60 node graph dataset and the key results after 100 iterations were provided in the figure 18.

The comparison showed in figure 18 depicted the difference between edge removal and update and their effects in graph isomorphism. For the first few edge update, the distance between graph pairs increased heavily compared to edge removal because of the dual update involved in edge update (i.e. edge removal and edge addition) compared to a single update in edge removal. As the process continued to do more edge removal/updates, a change in trend could be noticed. After removing many edges, graphs tend to loss important structural properties and be very different from its reference graph. However, this is not true for edge alterations. The new edges might somehow retain at least a few minimal topological structures and critical graph properties instead of resulting in a sparse graph in the case of edge removal. This behavioural difference proved to be the reason behind the gap between the similarity scores after 100 edge removals and 100 edge updates in our results.

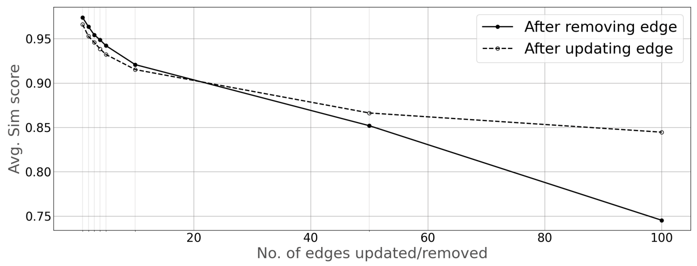


Fig. 18: Comparison between the drop-in similarity scores after edge removal and updation in Graphs having 60 nodes.

The experimental results and related obervations presented demonstrated the functioning of our method and provided a sense on why it would perform well for near-isomorphic graph pairs. We highlight the validity of such results, which can be acknowledged by observing the changes in similarity scores between varying graph properties like the number of nodes and edges.

## C. Experiment with random graphs

In the third experiment, our objective was to ensure that our method would not produce False-Negative results. It is essential to compute a similarity score of 1 for perfect isomorphic graphs, and we wanted to verify this behaviour with high precedence. The isomorphic copies of graphs from Erdös-Rényi (ER) - G(n, p) random model were generated and passed into our calculations for computing their sim score. By using permutation matrix, the isomorphic graphs were produced using the below formula.

$$P * A * P^T \tag{12}$$

where $P \in \mathbb{R}^{(n \times n)}$ is the Permutation matrix.

Here, random graphs with varying $|V|$ = [5, 10, 20, 30, ..., 80, 90, 100] were generated and used to create 100 isomorphic graphs for every vertex size. Since the purpose of this analysis is to ensure perfect isomorphic graphs, comparisons were made between graphs of the same order. A total of 4950 comparisons were performed in every group by comparing a graph with all other graphs from the same group. An expected *sim* score of 1 for perfect isomorphic graph pairs with all three terms of our method precisely taking a value of 1 is highly understandable. Our method certainly performed upto this expectation and resulted in 1 for the generated isomorphic graphs with 100% success rate. Out of 54450 graph pairs in total, no *sim* score from such computations went below the highest value of 1. Concretely, through our results, we highlight the effectiveness of *sim* score in classifying perfect isomorphic graphs.

## D. Experiment with Miyazaki graphs

In an evaluation to check the performance of our method for Miyazaki graph comparison, it was obvious that our method was not able to discriminate between Miyazaki graphs. The effort to enhance *sim* score computation and rectify this, was not taken as a part of this study.

## V. Conclusion

We proposed a new measure *sim* to quantify the amount of similarity between two graphs. While the primary focus of this study was on near-isomorphism in graphs, our similarity measure could also categorize graph pairs as perfect and near isomorphic and non-isomorphic based on our score range. Our evaluations justified the heirarchial ability of our method in distinguishing graphs by utilizing various important and meaningful graph properties. Furthermore, the empirical analysis revealed the strength of *sim* score in computing similaity between real-world graphs. Further experiments justified the legitimacy of our method where we also showed that *sim* score could figure out perfect-isomorphism.

Although the benchmark graph kernel datasets were used, we didn't extensively consider the class of those graphs as our main intention was to compare graph pairs. During our analysis, we noticed the essentiality of comparing graph classification results with *sim* score to explore the outliers and edge data from [38] and how they differ in similarity measure on comparing them with other graphs from the same datasets. Such a study would be interesting since it will show more insights about our method and results. Moreover, future investigations on *sim* score improvement can target approaches to differentiate Miyazaki graphs. Another interesting direction of study that could follow this work would be to show the reason behind high similarity scores of 0.9 and above by proving the existence of many sub-isomorphic patterns shared between the compared graphs.

## ACKNOWLEDGMENT

The authors would like to thank...

## REFERENCES

[1] L. Cordella, P. Foggia, C. Sansone, and M. Vento, "A (sub)graph isomorphism algorithm for matching large graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 10, pp. 1367–1372, 2004.

[2] P. Zhao and J. Han, "On graph query optimization in large networks," *Proc. VLDB Endow.*, vol. 3, no. 12, p. 340351, sep 2010. [Online]. Available: https://doi.org/10.14778/1920841.1920887

[3] J. R. Ullmann, "Bit-vector algorithms for binary constraint satisfaction and subgraph isomorphism," *ACM J. Exp. Algorithmics*, vol. 15, feb 2011. [Online]. Available: https://doi.org/10.1145/1671970.1921702

[4] A. Schrijver, *Combinatorial optimization: polyhedra and efficiency*. Springer Science & Business Media, 2003, vol. 24.

[5] B. Luo and E. Hancock, "Structural graph matching using the em algorithm and singular value decomposition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 10, pp. 1120–1136, 2001.

[6] L. Babai, "Graph isomorphism in quasipolynomial time," *CoRR*, vol. abs/1512.03547, 2015. [Online]. Available: http://arxiv.org/abs/1512.03547

[7] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis, "Frequent substructure-based approaches for classifying chemical compounds," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 8, pp. 1036–1050, 2005.

[8] O. Macindoe and W. Richards, "Graph comparison using fine structure analysis," in *2010 IEEE Second International Conference on Social Computing*, 2010, pp. 193–200.

[9] K. Lu, Y. Zhang, K. Xu, Y. Gao, and R. C. Wilson, "Approximate maximum common sub-graph isomorphism based on discrete-time quantum walk," in *2014 22nd International Conference on Pattern Recognition*, 2014, pp. 1413–1418.

[10] E. Duesbury, J. D. Holliday, and P. Willett, "Maximum common subgraph isomorphism algorithms," 2017.

[11] D. Conte, P. Foggia, and M. Vento, "Challenging complexity of maximum common subgraph detection algorithms: A performance analysis of three algorithms on a wide database of graphs," *J. Graph Algorithms Appl.*, vol. 11, pp. 99–143, 01 2007.

[12] J. Foisy and L. Ludwig, "When graph theory meets knot theory," 01 2009.

[13] T. Schieber, L. Carpi, A. Diaz-Guilera, P. Pardalos, C. Masoller, and M. Ravetti, "Quantification of network structural dissimilarities," *Nature Communications*, vol. 8, p. 13928, 01 2017.

[14] N. Savage, "Graph matching in theory and practice," *Communications of the ACM*, vol. 59, pp. 12–14, 06 2016.

[15] T. Schieber and M. Ravetti, "Simulating the dynamics of scale-free networks via optimization," *PloS one*, vol. 8, p. e80783, 12 2013.

[16] T. Schieber, L. Carpi, A. Frery, O. Rosso, P. Pardalos, and M. Ravetti, "Information theory perspective on network robustness," *Physics Letters A*, vol. 380, 11 2015.

[17] S. Soundarajan, T. Eliassi-Rad, and B. Gallagher, "A guide to selecting a network similarity method," in *SDM*, 2014.

[18] E. M. Loiola, N. M. M. de Abreu, P. O. Boaventura-Netto, P. Hahn, and T. Querido, "A survey for the quadratic assignment problem," *European Journal of Operational Research*, vol. 176, no. 2, pp. 657–690, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221705008337

[19] H. Bunke and K. Shearer, "A graph distance metric based on the maximal common subgraph," *Pattern Recogn. Lett.*, vol. 19, no. 34, p. 255259, mar 1998. [Online]. Available: https://doi.org/10.1016/S0167-8655(97)00179-7

[20] D. Taylor, F. Klimm, H. Harrington, M. Kramr, K. Mischaikow, M. Porter, and P. Mucha, "Erratum: Topological data analysis of contagion maps for examining spreading processes on networks," *Nature communications*, vol. 6, p. 8374, 09 2015.

[21] B. Kena, "The graph isomorphism problem," in *Proceedings of 7th Conference Student FEI 2001*. Brno University of Technology: Brno University of Technology, april 2001, conference paper, pp. 343–347.

[22] T. Gärtner, P. Flach, and S. Wrobel, "On graph kernels: Hardness results and efficient alternatives," in *Learning Theory and Kernel Machines*, B. Schölkopf and M. K. Warmuth, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 129–143.

[23] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt, "Graph kernels," *J. Mach. Learn. Res.*, vol. 11, p. 12011242, aug 2010.

[24] M. Neuhaus and H. Bunke, "Edit distance-based kernel functions for structural pattern classification," *Pattern Recogn.*, vol. 39, no. 10, p. 18521863, oct 2006. [Online]. Available: https://doi.org/10.1016/j.patcog.2006.04.012

[25] N. M. Kriege, P.-L. Giscard, and R. C. Wilson, "On valid optimal assignment kernels and applications to graph classification," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 16231631.

[26] J. Ramon and T. Grtner, "Expressivity versus efficiency of graph kernels," in *Proceedings of the First International Workshop on Mining Graphs, Trees and Sequences*, 2003, pp. 65–74.

[27] S. Menchetti, F. Costa, and P. Frasconi, "Weighted decomposition kernels," in *Proceedings of the 22nd International Conference on Machine Learning*, ser. ICML '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 585592. [Online]. Available: https://doi.org/10.1145/1102351.1102425

[28] S. Hido and H. Kashima, "A linear-time graph kernel," in *2009 Ninth IEEE International Conference on Data Mining*, 2009, pp. 179–188.

[29] J. F. D. M. C. Kriege, Nils M., "A survey on graph kernels," 01 2020.

[30] C. Arndt, *Information Measures*. Springer, Berlin, Heidelberg.

[31] S. Z. Liesen, Jrg, *Krylov Subspace Methods: Principles and Analysis*. Oxford University Press.

[32] A. Shrivastava and P. Li, "A new space for comparing graphs," 04 2014.

[33] T. Ramraj and R. Prabhakar, "Influence of krylov subspace in graph isomorphism for mobile networks," *Mobile Networks and Applications*, vol. 24, pp. 1–13, 04 2019.

[34] A. Mheich, F. Wendling, and M. Hassan, "Brain network similarity: methods and applications," *Network Neuroscience*, vol. 4, no. 3, pp. 507–527, 07 2020. [Online]. Available: https://doi.org/10.1162/netn_a_00133

[35] C. Palihawadana and G. Poravi, "A comparative study of link analysis algorithms," in *2018 8th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, 2018, pp. 100–104.

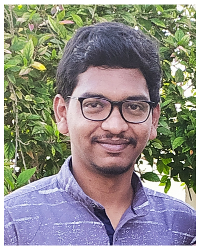
[36] M. Dehmer and F. Emmert-Streib, "Comments to quantification of network structural dissimilarities published by schieber et al," *Mathematical Methods in the Applied Sciences*, vol. 41, 07 2018.

[37] W.-K. Chen, *Linear Networks and Systems: Algorithms and Computer-Aided Implementations*. WORLD SCIENTIFIC, 1993.

[38] K. Kersting, N. M. Kriege, C. Morris, P. Mutzel, and M. Neumann, "Benchmark data sets for graph kernels," 2016. [Online]. Available: http://graphkernels.cs.tu-dortmund.de

**Ramraj Thirupathyraj** received his Masters in Computer Science in 2006 and Ph.D from Anna University, India. His research interests include mathematical modeling for graph quantization, spectral graph matching and Graph-based machine learning approaches. He is currently an Assistant Professor with the Department of Artificial Intelligence and Data Science at Coimbatore Institute of Technology, India. He has published a few research articles in various reputed conferences and journals.

**Prabhakar Ramachandra Gupta** received his B.Tech from IIT- Madras and Ph.D from Purdue University in 1975. His research interests include Optimization, Combinatorial algorithmic complexities and spectral graph matching. He is currently a Professor with the Department of Computer Science at Coimbatore Institute of Technology, India.



**Sivakumar Palanisamy** received his B.Eng in Computer Science in 2016 from Coimbatore Institute of Technology, India. His research interests involve establishing and improving mathematical foundations of supervised Machine Learning and Deep Learning algorithms like Graph Neural Networks, SVM and a few more.