Learning and Nonlinear Models - Journal of the Brazilian Society on Artificial Intelligence (SBIA), Vol. 24, Iss. 1, pp. 18–35, 2026
https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

DATA LABELING WITH LOCAL AND GLOBAL CONSISTENCY USING PHYLOGRAMS

Breno Caetano da Silva
Computer Science Department
Federal Institute of Maranhão, Timon - MA, Brazil
breno.caetano@ifma.edu.br

Jorge Luiz Franco
Institute of Mathematics and Computer Science
University of São Paulo, São Carlos - SP, Brazil
jorge.luiz@usp.br

Hermes Manoel Galvão Castelo Branco
Electrical Engineering Department
Federal University of Piauí, Teresina - PI, Brazil
hermescb@ufpi.edu.br

Alexandre Claudio Botazzo Delbem
Institute of Mathematics and Computer Science
University of São Paulo, São Carlos - SP, Brazil
acbd@icmc.usp.br

Abstract – One of the main challenges in machine learning tasks is unlabeled data. Consequently, semi-supervised learning is an approach commonly used in labeling data tasks. The use of diffusion kernels is one of the powerful methods in this set of learning. This paper introduces Phylogram based Data Labeling (PhyDL), an innovative method that combines the benefits of phylogram analysis with Learning with Local and Global Consistency (LLGC). PhyDL leverages phylogram analysis to act as an aggregator of information from labeled samples, capable of correcting possible labeling errors. PhyDL constructs a phylogram using community detection algorithms, FastGreedy and Edge Betweenness, to identify and leverage the structure of the data, with FastGreedy emerging as the best-performing algorithm among those evaluated. We implemented PhyDL and evaluated its performance on the TwoMoons, MNIST, MUSK, HEART, and SONAR datasets. PhyDL outperformed LLGC on 4 out of 5 evaluated datasets, highlighting its effectiveness in semi-supervised learning scenarios. On the TwoMoons, MNIST, MUSK, and SONAR benchmarks, the method showed steady accuracy gains, with consistent improvements as the number of labeled samples increased. However, on the HEART dataset, standard LLGC achieved better results, indicating that PhyDL's community-based diffusion can be limited by certain underlying spatial structures. Furthermore, the experiments revealed that the Edge Betweenness algorithm is unsuitable for these specific high-dimensional tasks. Ultimately, this study showcases PhyDL's potential—when equipped with the FastGreedy algorithm—to significantly improve data labeling performance in semi-supervised scenarios, effectively leveraging the structural topology of complex datasets.

Keywords – Local and Global Consistency, DAMICORE, Phylogram, Community Detection, Diffusion Kernel, Semi-Supervised Learning.

1 INTRODUCTION

In the modern context of machine learning, acquiring massive amounts of raw data is often trivial, whereas the process of accurately labeling these instances by human experts remains a costly, slow, and labor-intensive bottleneck [1, 2]. Semi-supervised learning (SSL) emerges as a fundamental paradigm to address this challenge by jointly exploiting a limited set of explicitly labeled instances and a comparatively large pool of unlabeled data [3, 4]. Conceptually, SSL operates under strong structural premises—most notably the manifold and smoothness assumptions—which postulate that high-dimensional data points reside on lower-dimensional geometric structures [5, 6]. By modeling the intrinsic topology of the data, often through graph-based representations and diffusion processes [7], SSL algorithms can intelligently propagate known labels across dense regions of the feature space, significantly reducing annotation costs without compromising predictive performance.

Among the various semi-supervised learning techniques, the use of diffusion kernels has been a highlight for its effectiveness in several applications [3, 8, 9]. Diffusion kernels are an essential tool in the analysis of data in networks and graphs. They

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

are used to capture the underlying structure of the data and facilitate the propagation of information through the graph. This propagation process is especially useful in semi-supervised learning, where the information from the few labeled data points can be efficiently extended to the unlabeled data [10, 11].

In semi-supervised learning **using graphs**, the nodes represent data examples and the edges represent the similarity or affinity between these examples. The diffusion kernel is then used to propagate the information from the labeled nodes to the unlabeled ones through the edges of the graph [12]. This approach is efficient in capturing the data distribution and, in many cases, can achieve results comparable to supervised methods, even when labeled data is scarce [13].

The use of diffusion kernels in semi-supervised learning presents several advantages. Firstly, it allows for the incorporation of structural information from the data, which is critical in many practical applications where the data has an intrinsic structure, such as social networks, computational biology, and natural language processing [14]. Secondly, this approach is robust to noise and can handle data irregularities well, as diffusion tends to smooth out abrupt variations in labels [15]. Additionally, the flexibility of diffusion kernels allows for adaptations and extensions to handle different types of data and problems. A notable method in this category is the Learning with Local and Global Consistency (LLGC) [16]. LLGC labels samples by propagating information about labeled samples. However, depending on the relative location between the labeled and unlabeled samples, labeling errors may occur.

Phylogeny and phylogram analysis have roots in biology [17, 18] but have shown interesting results in various application areas [19–22]. In the clustering process carried out in [18], it has been observed in experiments that the generated clusters tend to follow the spatial structure of the data. This fact motivated us to combine its benefits with LLGC in an innovative method, *Phylogram based Data Labeling (PhyDL)*. In the proposed approach, the phylogram analysis acts as an aggregator of information from labeled samples, capable of correcting possible labeling errors.

PhyDL leverages phylogram analysis, initially considering an appropriate distance metric. This approach facilitates a sensible grouping of data points into communities. The subsequent step involves diffusing labeled points across the entire dataset. These *diffusion contributions* are then organized into communities, where points within a community receive the same label.

We implemented PhyDL[1] and evaluated its performance on five datasets: a benchmark dataset for binary classification (Two Moons)[2]; and four real-world datasets: MNIST [23], SONAR [24], HEART [25], and MUSK v1 [26]. The results demonstrated that PhyDL outperformed LLGC in 4 out of 5 scenarios, showing the effectiveness of the proposed approach across multiple domains.

In summary, PhyDL builds upon LLGC, drawing insights from phylogram analysis to achieve excellent results in semi-supervised learning scenarios. Further details on implementation and results follow.

2 MATERIAL AND METHODS

This section presents the theoretical framework behind the proposal developed in this work, demonstrating how the concept of phylogram can be utilized in conjunction with diffusion kernels to improve data labeling in the context of semi-supervised learning.

2.1 Labeling through diffusion kernel

The use of diffusion kernels to label data samples represented by vertices in a graph can be seen as the propagation of information contained in some vertices to the others. For this purpose, the dataset represented by $X = \{x_1, ..., x_l, x_{l+1}, ..., x_n\}$, with n samples, and a set of classes, $L = \{1, ..., c\}$, is taken, where X contains l labeled samples and $n - l$ unlabeled samples. Let F be a matrix with non-negative elements and dimensions $n \times c$, which will represent the diffusion of labels.

An implementation for label propagation in a graph is the LLGC algorithm [16]. It consists of four steps:

1. Construction of the affinity matrix: $W_{ij} = \exp(-||x_i - x_j||^2/2\sigma^2)$
2. Obtaining the normalized Laplacian: $S = D^{-1/2}WD^{-1/2}$, where D is a diagonal matrix, $D_{ij} = \sum_j W_{ij}$
3. Iteration: $F(t+1) = \alpha SF(t) + (1-\alpha)Y$ until the change between two iterations is smaller than a threshold.
4. With F^*, the labeling can be obtained by $y_i = \arg\max_{j \leq c} F_{ij}$

Step 3 is an iterative process, which is resource-intensive. An alternative, proposed in [16], is to obtain the function in the limit, letting $t \to \infty$, which can be visualized in Equation 1.

$$F^* = (I - \alpha S)^{-1} Y \tag{1}$$

where α is a parameter that regulates the step size in the convergence process. S and Y are, respectively, the similarity matrix and a vector containing information about which samples are labeled, taking a class as a reference. The variable Y is a binary vector where the non-zero positions are those where the sample is initially labeled. Thus, each vector Y presented to Equation

[1] Source code and reproducibility scripts available at https://github.com/brenocaetano/PhyDL

[2] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_moons.html

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

1 will represent a class with its respective labeled samples. At the end of the iterative process, F will represent the amount of information (diffusion contribution) each node has received.

The parameter α in the LLGC algorithm plays a crucial role in balancing two key sources of information: the initial labeled data and the graph structure representing data relationships (see step 3). This balance is reflected in how α influences the algorithm's behavior. When α values are low (closer to 0), the algorithm places greater emphasis on the initial labels, relying more heavily on the limited set of labeled examples. Conversely, high α values (closer to 1) give more weight to the information diffused through the graph, leveraging the underlying data structure. In this paper, the authors employ the value of 0.99, as used in the original study on the LLGC algorithm. This approach leverages the underlying structure of the data, allowing the algorithm to make inferences based on the relationships between data points, rather than being constrained by the sparse initial labeling. It's important to note that the authors do not intend to obtain the best α value for the databases analyzed in this study. Instead, their objective is to investigate whether, starting from a label diffusion process, communities obtained from a phylogram can be used to improve the accuracy of labeling.

2.2 Phylogram

The concepts of homology and phylogeny, widely employed in biology, have proven to be extremely useful when adapted for data analysis in various fields [27]. In biology, homology refers to the shared characteristics between different species due to a common ancestor. This concept can be adapted for data analysis, allowing the identification of patterns or similar structures in different samples of a dataset. The identification of homologies in non-biological data can reveal hidden information, such as the detection of patterns or the discovery of recurring subgraphs in complex networks. As a representation of these homologous similarities, the use of a phylogram is common [21, 28].

A phylogram, which in biology refers to the representation of evolutionary relationships between species, can be applied analogously in data analysis to understand the relationships between different entities or variables. The construction of phylograms or phylogenetic trees for non-biological data allows for the visualization and analysis of how different elements of a dataset are related and how they have evolved over time [22].

Among the methods for reconstructing phylograms, those based on a distance matrix enable the creation of an evolutionary tree from a measure of distance between the analyzed elements. A distance matrix details the distances between pairs of elements, calculated using a selected metric. Any type of information can be analyzed from a phylogenetic perspective if a distance matrix between the elements is available. This approach allows for analyzing data based on patterns of evolution, diversification, and hierarchical components that represent their homology, offering a comprehensive view of the relationships and similarities between the elements. By visualizing these relationships, researchers gain deeper insights into the underlying structure within the data.

The most recognized method for reconstructing phylogenetic trees is the *Neighbor Joining (NJ)* method, introduced by [29]. It serves as a benchmark for evaluating other reconstruction methods due to its efficiency and capability to generate accurate trees even with extensive datasets. A significant advantage of NJ is that it does not require a constant rate of evolution, making it adaptable to data where evolutionary rates differ across lineages. This flexibility allows NJ to produce trees that accurately reflect the true evolutionary relationships [17]. As a distance-based approach, NJ's results depend heavily on the accuracy of the distances between elements. In phylogenetics, the distance matrix used is additive, ensuring that the distances between element pairs can be fully represented on a phylogenetic tree without distortion or information loss.

The NJ method can accurately reconstruct the true phylogenetic tree if the correct distance matrix is provided [17]. Although it is not always possible to ensure the exact accuracy of the distance matrix, the NJ method remains effective as long as the noise level does not exceed the difference between the actual matrix and the analyzed matrix by more than a threshold $\epsilon = 1/4$, as specified by:

$$\|D^T - \hat{D}\|_\infty < \epsilon \min_{e \in E(T)} l(e) \tag{2}$$

where D^T and $\hat{D}$ represent the additive matrices of the actual tree and the approximate tree, respectively, $E(T)$ denotes the set of vertices of the tree, $l(e)$ defines the branch length of a given edge e, and ϵ is the proportion relative to the smallest edge of the tree [30]. This characteristic allows NJ to be effective on any type of data, as the quality of the reconstructed tree relies entirely on the accuracy of the distance metric used.

Another aspect that supports the quality of NJ's reconstruction is the quartet consistency theorem [31]. According to this theorem, if NJ successfully reconstructs the quartets locally, it will also succeed globally. In other words, if NJ can derive all the quartets of tree T from a distance matrix ($\hat{D}$) representing T, it will effectively reconstruct T from $\hat{D}$.

2.3 Community Detection

The spatial structure of a phylogram can be visualized as interconnected entities, where the distances between these entities represent the weights or costs of their connections, effectively making the phylogram a network. Grouping nodes of a network into communities is known as community detection [32]. Various authors have proposed algorithms to perform this task efficiently and effectively.

In [33], the authors propose an algorithm called *FastGreedy*, which is an improvement of the *fastAlgorithm* proposed by [34]. FastGreedy uses a greedy strategy in the process of detecting communities in a given input graph. Its operation is a hierarchical

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

agglomerative clustering, producing a dendrogram representing the similarities found between nodes of the input graph, which are grouped into communities. The difference here, compared to the previous section executed by NJ, is that NJ used information from a distance matrix to generate the graph, which can be seen as a network. FastGreedy, using this graph as input, performs a more abstract feature extraction based on the value of the modularity of the grouping. Modularity is a property of the network, used as a criterion for dividing a network into communities. It serves as an indication of how good the grouping is, by observing whether there are many connections between the nodes within a community and few between communities [33].

The FastGreedy algorithm iteratively checks whether merging pairs of communities can improve the modularity value. It is important to note that the process begins with each node as its own community, and the merging process continues until all nodes form a single community. In summary, the *FastGreedy* algorithm is based on a greedy optimization process that aims to maximize the modularity value of the network by iteratively merging communities until no further improvement can be made. This approach has been widely used in community detection due to its effectiveness in finding communities in large-scale networks.

The Edge Betweenness algorithm, proposed by [35], is a divisive method for community detection in networks, differing from traditional algorithms. Instead of seeking the most distinct edge for removal, it focuses on finding the edge with the highest Betweenness, which favors edges whose endpoints are in different communities, as these edges will be more utilized in the shortest paths between pairs of nodes.

2.4 Phylogram-based Data Labeling

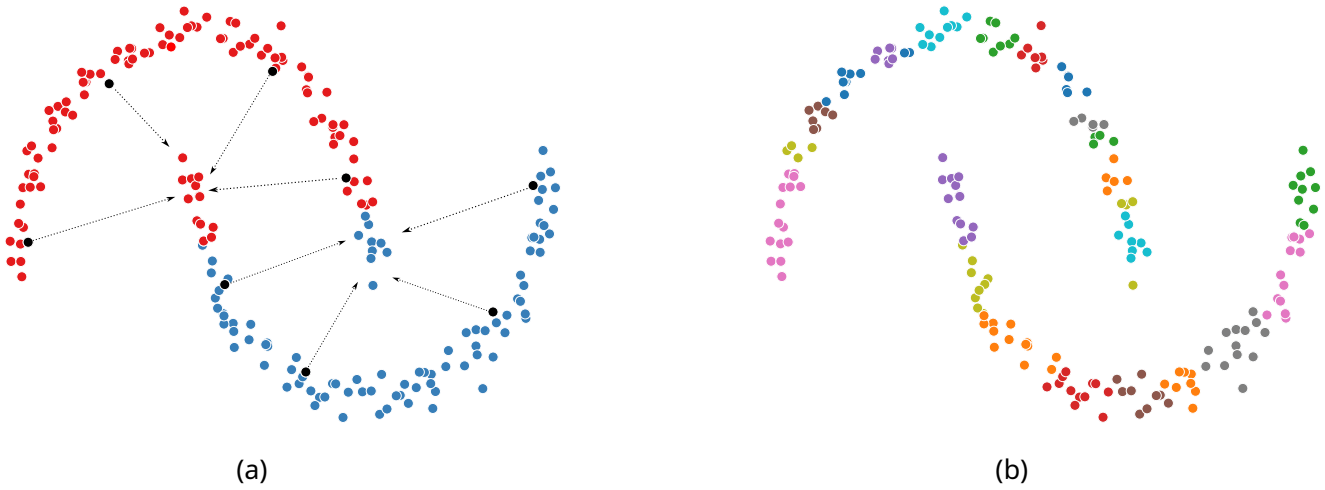


Figure 1: Diffusion of labeled points on the Two Moons dataset in a semi-supervised learning setting. In (a), the colors represent the groupings (and labels) obtained a diffusion kernel (LLGC); and in (b), the communities obtained in the phylogram analysis. In the traditional LLGC (a), the points in black correspond to the initially labeled points and the arrows symbolize the diffusion of the class between the two *moons*, which is undesirable in this case.

As previously presented, the use of a diffusion kernel applied to data labeling requires only a small portion of labeled data. This process can be observed in Figure 1(a). In this image, it can be seen that the central region of the moons exhibits all the errors, indicating a breakdown of local and global (structural) consistency. Local consistency means that nearby samples should have the same label, while global consistency indicates that samples within the same structure should possess the same labels [16]. In panel (a), it is evident that both consistencies are not maintained in the central region of the image.

It can be observed that the detection of communities on a phylogram generates clusters that is guided by the structure of the classes (half-moons). These clusters can be visualized in panel (b) of Figure 1. If we analyze each community in isolation, in the majority of cases, they fulfill the criterion of local consistency, meaning that nearby samples tend to have the same label. These communities were obtained by applying the FastGreedy algorithm to the phylogram of the data. The phylogram, in turn, represents the relationships of the samples in a global arrangement, meaning that when analyzing it, samples on the same branch tend to belong to the same data structure, in this case, one of the moons. Therefore, intuitively, community detection on a phylogram seeks to fulfill the requirements of local and global consistency of the data.

Based on this premise, the application of a diffusion kernel along with community information obtained from a phylogram in the task of labeling unlabeled samples was studied. The iteration of these tools is favorable due to the shared characteristics of local and structural consistency. In Figure 1(a), it is possible to visualize how the propagation of information from the labeled samples performed by the diffusion kernel occurs through the structure of the dataset. In Figure 1(a), only 4 samples from each moon are initially labeled. The rest are labeled by the LLGC algorithm. Next to it, in panel (b) of Figure 1, it is shown how the communities were identified, following the structure of the dataset. Based on these tools, PhyDL is proposed and illustrated in Figure 2.

Learning and Nonlinear Models - Journal of the Brazilian Society on Artificial Intelligence (SBIA), Vol. 24, Iss. 1, pp. 18–35, 2026
https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

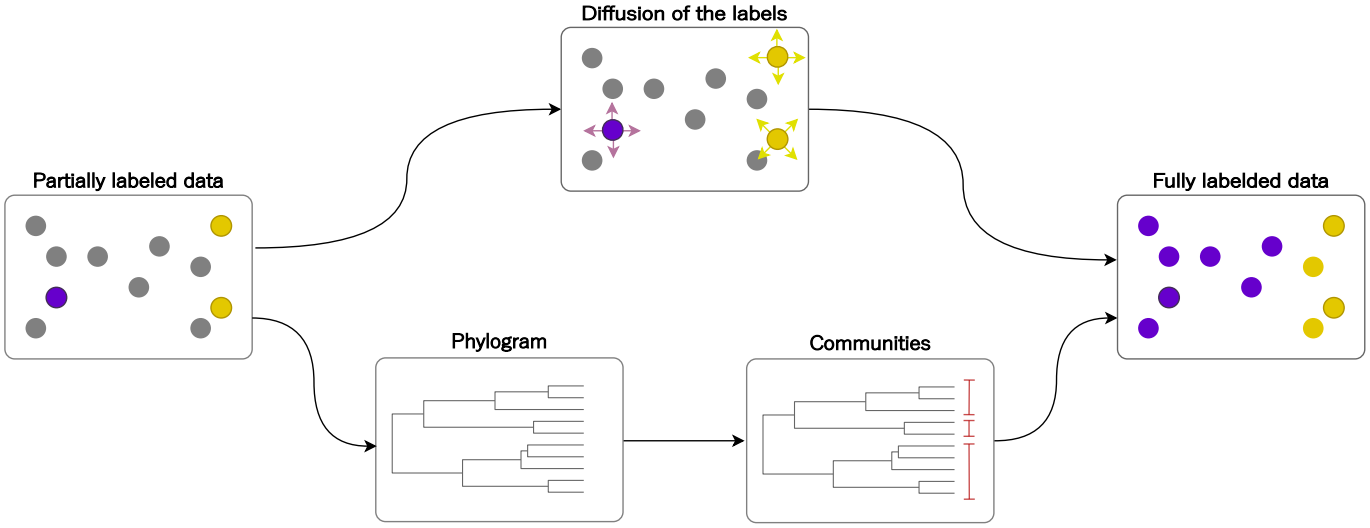


Figure 2: Main components of PhyDL. From the initial partially labeled data, the diffusion contribution of each labeled point to each point in the dataset is computed, as performed in LLGC. From the initial data, a phylogram representing the data is obtained and the communities are identified. Finally, the diffusion contribution of each label to each point is aggregated by community, such that a single label is assigned per community.

In a semi-supervised setting, only a small sample of the dataset is labeled. As can be seen in Figure 2, in the first step of PhyDL, we calculate the diffusion of these labeled points over all the points as in Equation 1. From the initial dataset, we also calculate the pairwise distance between the points. In this step, the information about the initial labels are disregarded. The distance metric is a fundamental factor in this step. The distances between the points allows us to determine in a next step the phylogram of the dataset. This step is followed by the identification of communities in the phylogram. Finally, we combine the results from the label diffusion and the communities, by defining a consensus on the label by community (see Figure 3). In the end, a fully labeled dataset is obtained.

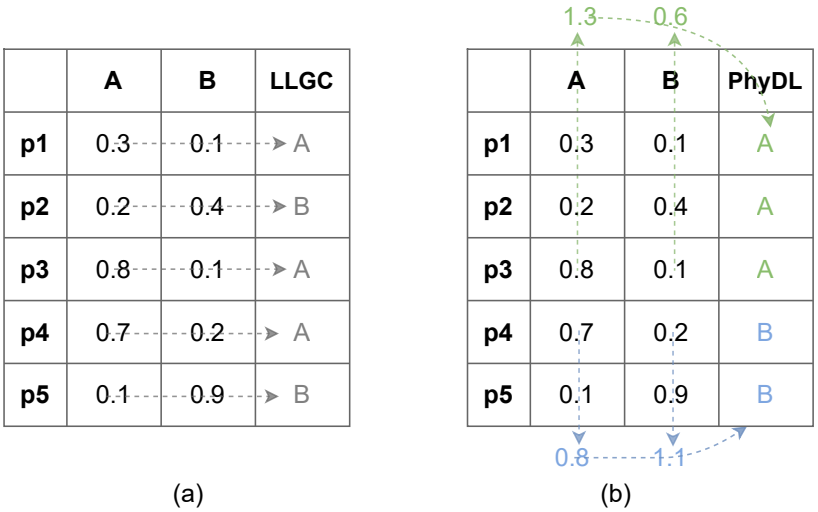


Figure 3: Assignment of new labels using (a) LLGC and (b) PhyDL in a hypothetical semi-supervised setting. In this example, there are two possible labels (A and B) and five points (p1 – p5). The numerical values correspond to the *contribution* received by each labeled point from each class. In the original LLGC (a), the labels are determined by calculating the maximum between the contributions of each class. In PhyDL (b), assuming two communities (p1, p2, p3) and (p4, p5), for each label, the sum over all the points are calculated and the maximum *between the sums* are calculated and all the points receive the same label.

In Figure 3, we illustrate a numerical calculation comparing LLGC and our proposed method, PhyDL. The rows represent individual data points, and the numerical values indicate the diffusion contribution of each label received by each point. This contribution is calculated using Equation 1. In this example, we consider three points and two possible labels. In scenario (a), each points label is determined by the one corresponding to the maximum diffusion contribution. In scenario (b), we assume that

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

points are grouped into a community structure and for each community, the diffusion contributions of each label are summed across the points within that community. Subsequently, a resulting diffusion contribution for each label is obtained. Finally, the label associated with the maximum value among these contributions is assigned to all points within that community.

3 EXPERIMENTS AND RESULTS

To achieve the proposed objectives, a series of experiments were conducted using 5 databases:

1. Twomoons
2. MNIST
3. MUSK v1 (MUSK)
4. HEART Disease (HEART)
5. SONAR

To guarantee a fair evaluation during the distance matrix computation, data preprocessing was performed. All features across the evaluated datasets were normalized to a [0, 1] range using Min-Max scaling. This step ensures that every attribute contributes equally to the Euclidean distance calculations, which is a crucial requirement for both the phylogram reconstruction and the diffusion kernels.

The *TwoMoons* database consists of a synthetic benchmark, composed of two intertwined semi-circles, commonly used in machine learning. A representation of the *TwoMoons* database can be seen in Figure 1. For the experiments, 200 samples were generated (100 for each moon). The MNIST database represents handwritten characters from 0 to 9. This database comes from the United States National Institute of Standards and Technology. It consists of 70,000 handwritten characters, normalized to a bounding box with dimensions of 28x28. Here it is necessary to explain that in the experiments with MNIST just 500 samples (randomly chosen from the database) of each class were used. The MUSK dataset is a molecular classification dataset designed to identify molecules with ”musk” (pleasant odor) or ”non-musk” (without pleasant odor) characteristics. It consists of 92 molecules (47 musks and 45 non-musks) in 476 distinct conformations, using 166 numerical features to describe the structural properties of the molecules, including distance measurements and three-dimensional displacements. The HEART dataset combines information from multiple sources to predict the presence of cardiovascular diseases. Originally developed in 1988, the dataset integrates information from five different databases: Cleveland, Hungary, Switzerland, Long Beach VA, and Statlog, initially totaling 76 attributes, although most research uses only 14 main features. The dataset contains information on 303 patients, classifying them into five classes, with integer values ranging from zero (absence of disease) to four (different levels of disease presence). The distribution of samples among classes is not homogeneous, containing 164, 55, 36, 35, and 13 samples for classes 0, 1, 2, 3, and 4, respectively. Due to its high imbalance and the possibility of transforming this dataset into a binary problem (disease, no disease), classes 1, 2, 3, and 4 were merged into a single class, representing the existence of heart disease in the patient. This transformation is common in the literature. The SONAR dataset, also known as Mines vs. Rocks, is a classic pattern classification dataset originating from studies discriminating between sonar signals reflected by metal cylinders and approximately cylindrical rocks. The dataset contains 208 patterns, with 111 obtained from different reflection angles of metal cylinders and 97 from rock formations, each described by 60 numerical features in the range [0,1]. These features represent the energy within a given frequency band integrated over time, with increasing frequencies for each subsequent attribute. A quantitative summary including the number of samples, classes, features and the execution times for the two community detection algorithms (CDAs) used can be found in Table 1. Notably, the execution times for the LLGC algorithm are not affected by the choice of community detector, with variations likely related to the system resource load during experimental executions. The higher computational cost of PhyDL is attributed to its additional processing steps, such as the construction of the phylogram using the Neighbor-Joining (NJ) algorithm, which has a computational complexity of $O(n^3)$. Additionally, the complexity of the chosen CDA directly impacts the total time; FastGreedy follows a greedy optimization with an approximate complexity of $O(m \cdot d \log n)$, while Edge Betweenness is significantly more resource-intensive, with a complexity of $O(m^2 n)$

Table 1: Comparison of execution times (in seconds) and dataset characteristics using different community detection algorithms.

					PhyDL (s)	
Dataset	**Samples**	**Features**	**Classes**	**LLGC (s)**	**FastGreedy**	**Edge Betweenness**
MNIST	1000	784	10	4.592	416.149	564.014
TwoMoons	200	2	2	0.028	3.074	4.871
Heart	284	13	2	0.0747	10.667	14.8032
Musk	476	166	2	0.301	43.740	63.765
Sonar	208	60	2	0.042	3.353	3.831

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

As described in Section 2, PhyDL can be configured in various ways at any of its stages. Thus, it is possible to test the performance effects of PhyDL with various phylogenetic reconstruction algorithms and community detection algorithms (CDA).

As phylogenetic reconstruction algorithms, only NJ will be used, as it is already a consensus in the literature [19] that it is the best cost-benefit in the process of reconstructing phylogenetic trees. Finally, as CDAs, were used at experiments the FastGreedy [33] and Edge Betweeness (often referred as Girvan-Newman) [32].

The results of these experiments are reported in the next section.

3.1 Experiments and Results using the TwoMoons Database

In all experiments using the TwoMoons database, some samples were selected where the label would be known. This set of samples consists of the labeled samples, with the remaining samples being part of the unlabeled samples. To assess the impact of the number of samples on the labeling process, the number of labeled samples varied from 2% to 20% samples, in steps of 2%. For each variation in the number of labeled samples, 30 executions of the algorithms were carried out, in order to avoid possible combinations of samples that would distort the obtained statistical results. It is also noted that each set of labeled samples was composed of equal parts, in number, of the analyzed classes.

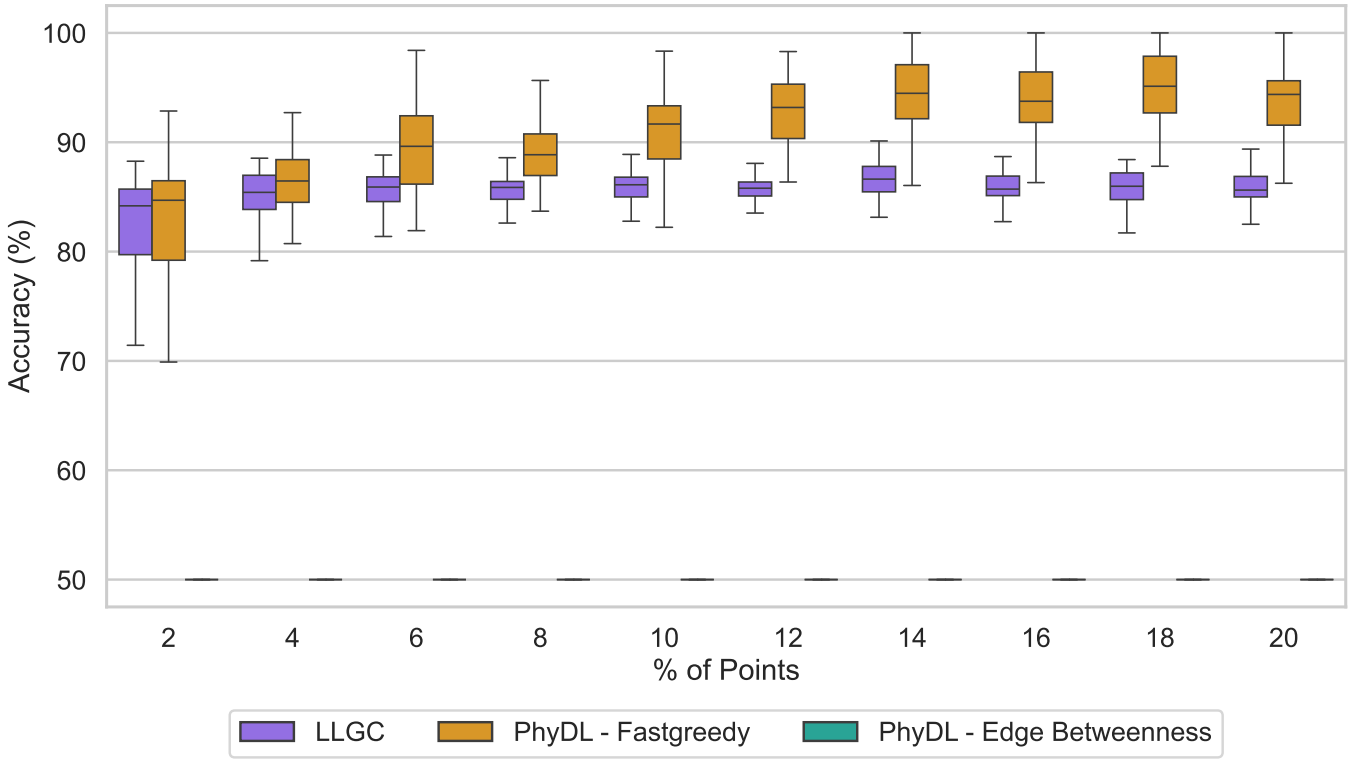


Figure 4: Results obtained on the TwoMoons database by LLGC contrasted with those achieved by PhyDL using the FastGreedy and Edge Betweeness CDAs. The boxplots illustrate the median and the interquartile range across the 30 independent executions.

Regarding the used CDA, PhyDL was used in experiments using FastGreedy and EdgeBetweenness. It is emphasized that both PhyDL used only Euclidean distance in the first stage and NJ in the second stage of in all experiments.

In Figure 4, the accuracy results for the evaluated methods are presented. The LLGC method established a solid baseline, maintaining a median accuracy between 84% and 86% across the varying percentages of labeled samples. However, PhyDL using the FastGreedy algorithm demonstrated a consistent and significant improvement over LLGC. While its advantage is modest at lower percentages (2% to 4%), FastGreedy's accuracy scales efficiently as more data becomes available, reaching a median accuracy of approximately 95% when 20% of the samples are labeled.

In stark contrast, PhyDL using the Edge Betweenness CDA exhibited a severe drop in performance. Its accuracy flatlined at exactly 50% across all evaluated scenarios. Given that TwoMoons is a binary classification problem, this outcome is equivalent to random guessing. This indicates that the Edge Betweenness algorithm completely failed to capture the manifold structure of the dataset in this experimental setup, likely breaking the intertwined half-moons into incorrect disconnected components due to its edge-removal strategy.

Figure 5 illustrates the relative gains of the PhyDL configurations compared to LLGC. The FastGreedy configuration achieved positive and growing gains, starting near 1% with 2% labeled samples and peaking at nearly 10% relative gain at the 18% and 20% marks. Conversely, the inability of Edge Betweenness to model the dataset resulted in massive negative gains, consistently hovering between -38% and -42%. These results highlight that while PhyDL is highly effective, its success is inherently tied to the community detection algorithm's ability to respect the local topology of the data, a task where FastGreedy proved to be remarkably robust.

Learning and Nonlinear Models - Journal of the Brazilian Society on Artificial Intelligence (SBIA), Vol. 24, Iss. 1, pp. 18–35, 2026
https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

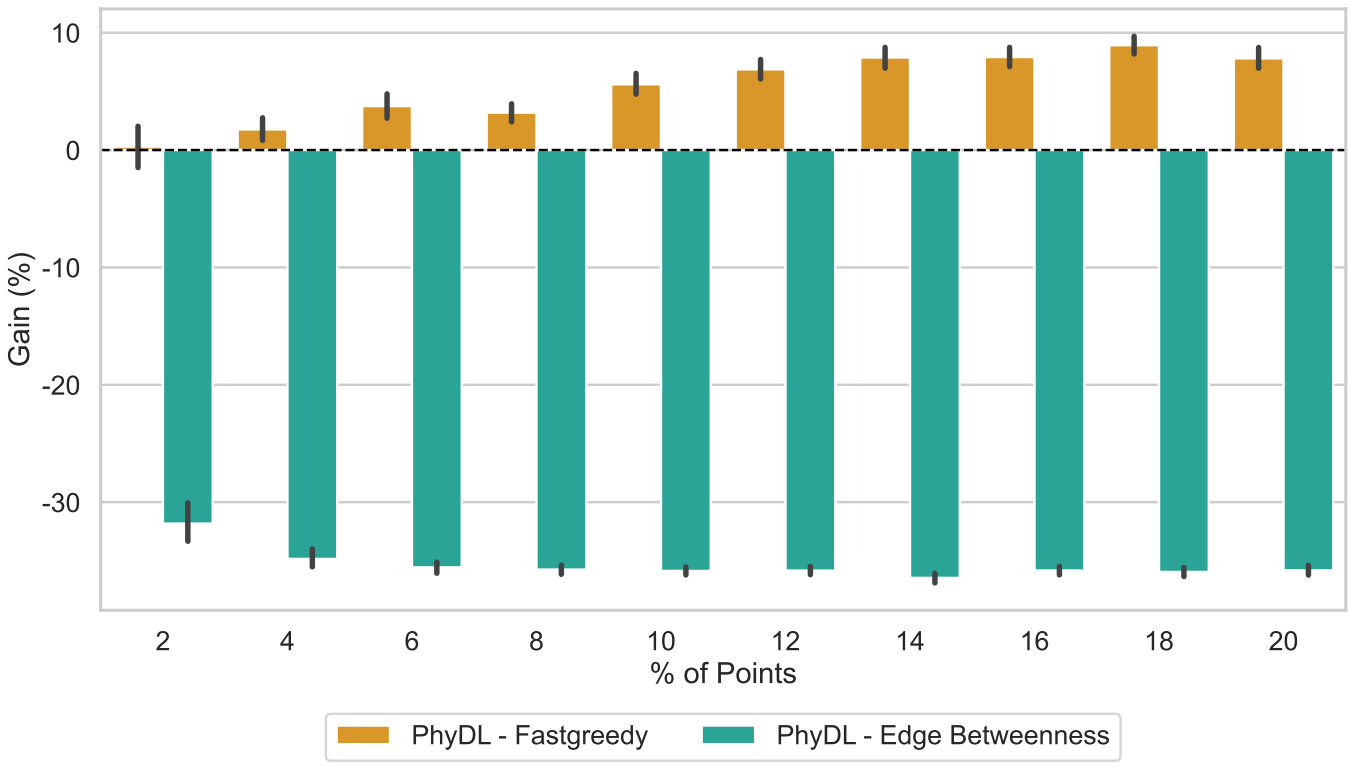


Figure 5: Relative Gain of PhyDL using the two different CDAs over LLGC on the TwoMoons dataset. The bar values represent the mean accuracy across the 30 independent executions.

In Figure 6, we have an example of labeling performed by LLGC, panel (a), and by PhyDL, panel (b), using the FastGreedy CDA over the TwoMoons dataset with 4% of initially labeled samples (represented as white points). As observed in panel (a), the diffusion of the classes across the empty space between the two moons causes a breakdown of local and global consistency, leading to misclassifications, particularly near the extremities. In contrast, panel (b) demonstrates how PhyDL mitigates this behavior. By aggregating the diffusion contributions through the communities identified by the FastGreedy algorithm, PhyDL better respects the spatial structure of the data, significantly reducing the incorrect propagation of labels between the upper and lower moons. While it does not achieve a perfect separation and some labeling errors still occur at the ends of the moons, PhyDL successfully corrects a substantial portion of the samples incorrectly labeled by LLGC. In this specific execution, the accuracy was 84.9% and 91.14%, respectively for LLGC and PhyDL.

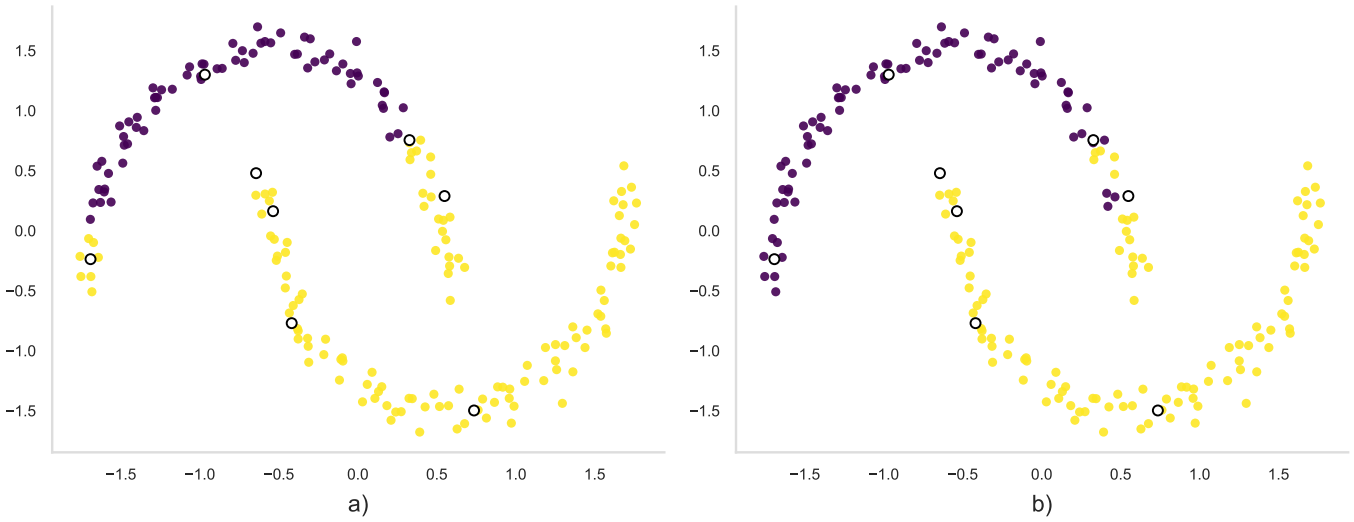


Figure 6: Comparison between LLGC and PhyDL on the TwoMoons dataset. In (a) we have the labeling performed by LLGC. In (b) we have the labeling performed by PhyDL, using Edge Betweenness

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

3.2 Experiments and Results using the MNIST Database

In Figure 7, the accuracy results obtained on the MNIST dataset are presented. This figure reveals a stark contrast between the methods, particularly highlighting the challenges of high-dimensional data. The baseline LLGC algorithm struggled significantly, exhibiting a median accuracy that fluctuated between 12% and 18% across all labeling rates, indicating poor label diffusion across the complex digit classes.

However, incorporating the FastGreedy CDA into PhyDL dramatically transformed this outcome. With only 2% of labeled samples, PhyDL-FastGreedy already achieved a median accuracy of 33%, almost doubling the LLGC baseline. This advantage scaled remarkably as the number of labeled samples increased, reaching a median accuracy of over 63% with 20% of labeled points. In striking contrast, PhyDL using Edge Betweenness failed entirely in this high-dimensional space. Its accuracy flatlined at exactly 10% across all scenarios, which—given the 10 classes of the MNIST dataset—is statistically equivalent to random guessing. This confirms that the divisive nature of Edge Betweenness is highly unsuitable for this dataset's topology.

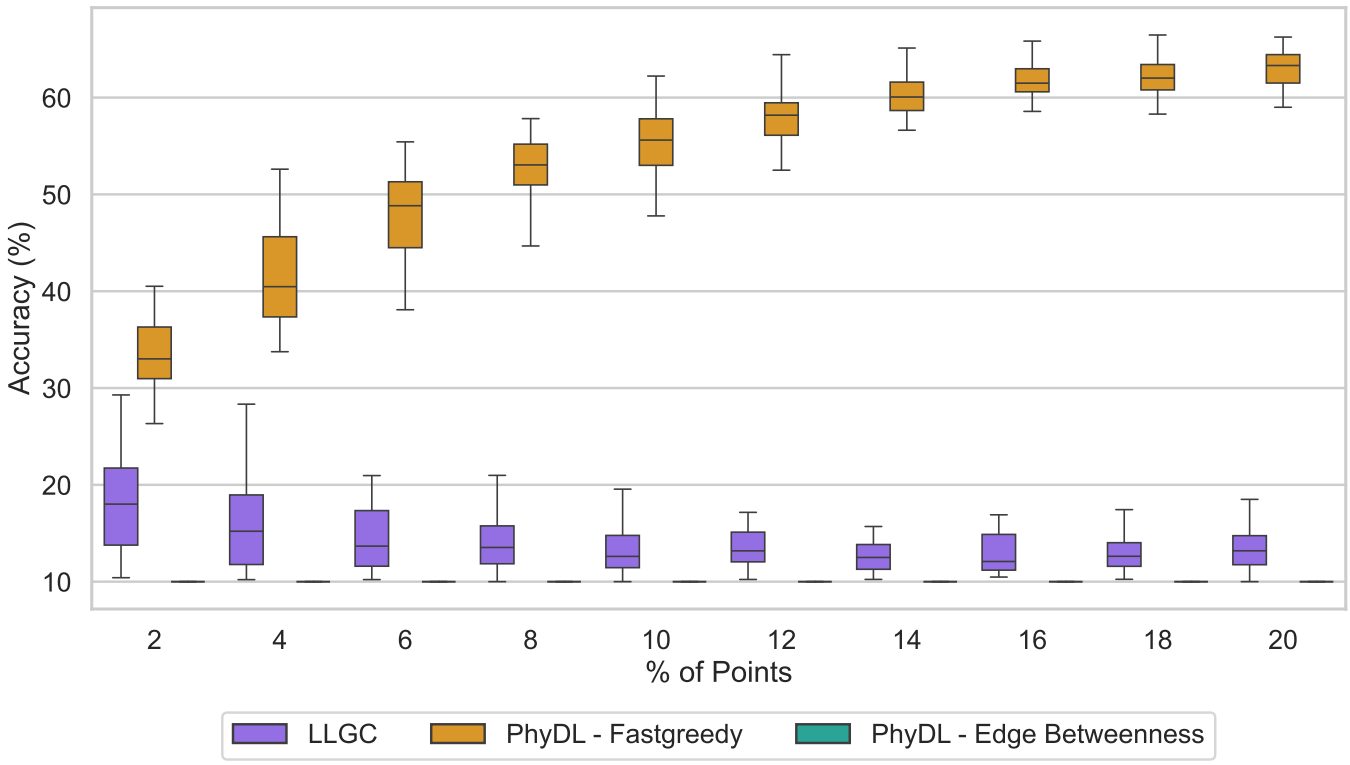


Figure 7: Accuracy results obtained on the MNIST dataset by LLGC contrasted with those achieved by PhyDL using two different CDA (Fastgreedy and Edge Betweenness) . On the x-axis, the ratio of initially labeled points considering the entire dataset. The boxplots illustrate the median and the interquartile range across the 30 independent executions.

Figure 8 illustrates the relative gain in mean accuracy compared to LLGC. The disparity between the two CDAs is once again evident. PhyDL-FastGreedy exhibited massive and continuous growth; starting with a relative gain of approximately 86% with 2% of labeled samples, its gain skyrocketed to an extraordinary peak of over 370% when 14% to 18% of samples were labeled. On the other hand, the complete failure of Edge Betweenness translated into constant negative relative gains (from -21% to -44%). This performance decisively demonstrates that for complex datasets with higher dimensionality and multiple classes, PhyDL equipped with FastGreedy overwhelmingly excels over standard LLGC by effectively mapping and leveraging the intricate structural communities.

In Figure 9a the visualization of MNIST dataset after applying t-SNE [36] is illustrated. Figures 9b and 9c shows how this dataset is labeled according to the methods used in this work. It is observed that when using PhyDL, the labeled data appears much closer to the true labels compared to the data labeled by LLGC, highlighting the advantage of the proposed method.

3.3 Experiments and Results using the MUSK Database

The experiments with the MUSK dataset were conducted in the same manner as the aforementioned experiments. Figure 10 shows that, for this dataset, the choice of CDA drastically affected the results of PhyDL. It can be observed that when using the FastGreedy CDA, PhyDL shows superior performance to LLGC. However, using the Edge Betweenness CDA, PhyDL performed slightly worse or identically to the baseline.

Analyzing the median accuracy, LLGC displayed a declining trend as more samples were labeled, starting at 45.4% (with 2% of labeled samples) and dropping to approximately 41.9% (with 20% of labeled samples). In contrast, PhyDL using the

Learning and Nonlinear Models - Journal of the Brazilian Society on Artificial Intelligence (SBIA), Vol. 24, Iss. 1, pp. 18–35, 2026
https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

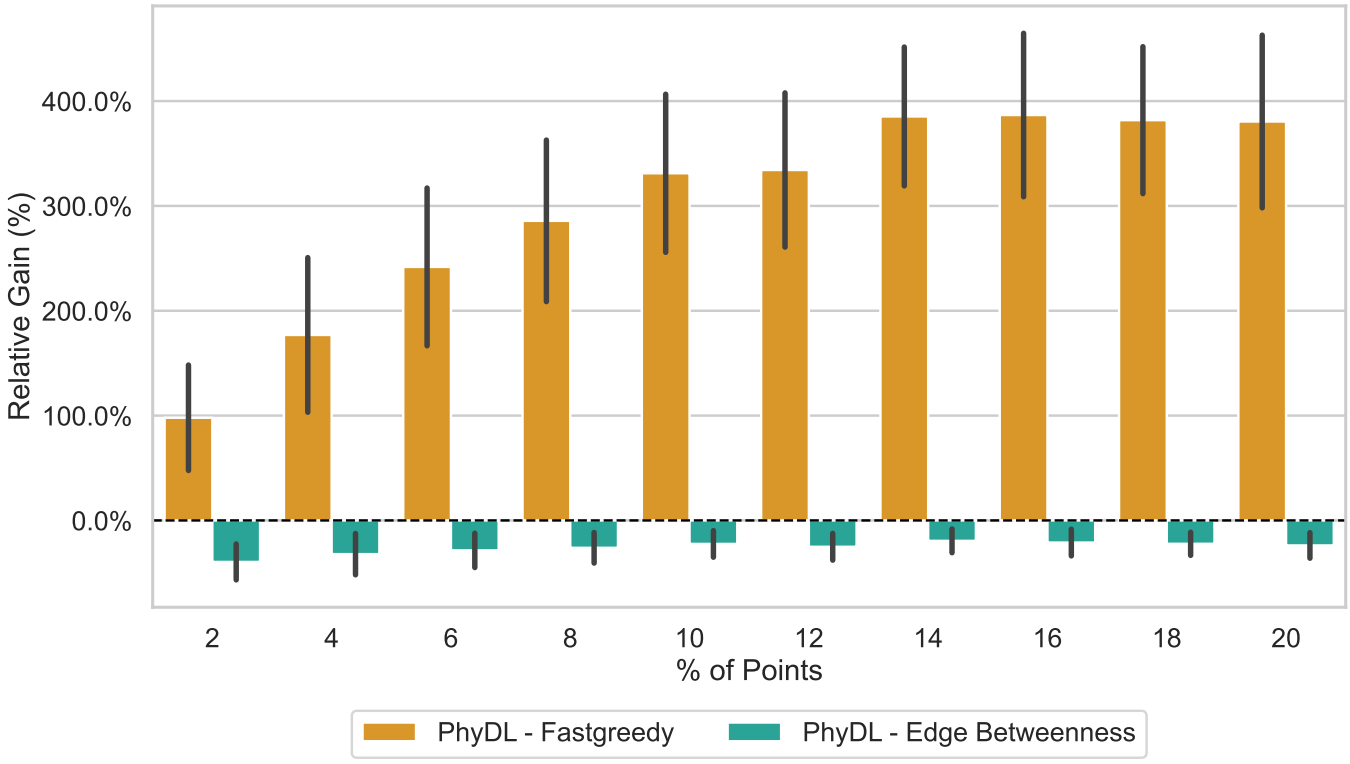
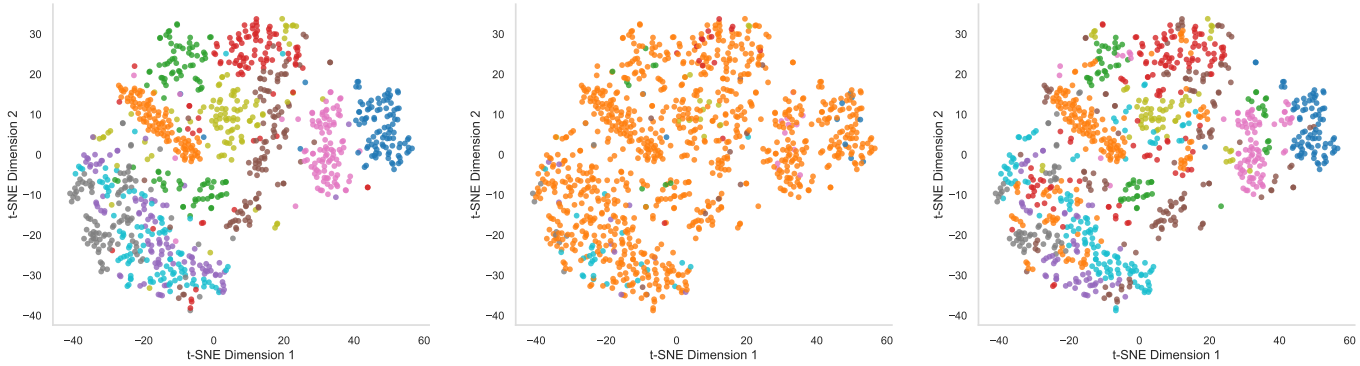


Figure 8: Relative Gain of PhyDL using two CDA (FastGreedy and Edge Betweenness) over LLGC on the MNIST dataset. The bar values represent the mean accuracy across the 30 independent executions.



(a) Ground Truth (b) LLGC (c) PhyDL

Figure 9: t-SNE projections for the MNIST dataset: (a) Original representation where colors indicate the 10 classes; (b) Data labeled by LLGC with an accuracy of 12.22%; and (c) Data labeled by PhyDL using Edge Betweenness with an accuracy of 55.67%. For both (b) and (c) 20% of the total data) were initially labeled (class-balanced).

FastGreedy CDA not only started with a higher median accuracy of 50.0% but also demonstrated continuous growth, reaching nearly 63.9% with 20% of labeled samples. On the other hand, PhyDL using the Edge Betweenness CDA practically mirrored the declining behavior of LLGC, starting slightly lower at 43.4% and converging to the same 41.9% accuracy at the end of the experiment.

Figure 11 illustrates the relative gain of the PhyDL versions over LLGC. Using the FastGreedy CDA in the PhyDL configuration provided substantial and continuous performance gains. With 2% of labeled samples, PhyDL-FastGreedy had an average relative gain of approximately 6.1%. Unlike what was observed in previous older executions, this gain did not decrease; instead, it maintained a robust upward trajectory, peaking at an impressive 52.1% relative gain when 20% of the samples were labeled. Conversely, when using the Edge Betweenness CDA, the relative gain remained slightly negative or negligible across all values. It started at a -6.0% gain with 2% of labeled samples and gradually converged to a neutral 0% (specifically -0.16%) with 20% of labeled samples, indicating that Edge Betweenness failed to aggregate any useful structural information to improve the standard LLGC diffusion on this dataset.

Figure 12a shows the projection of the MUSK dataset using t-SNE with its classes represented. Figures 12b and 12c shows the same projection by LLGC and PhyDL. In 12b, class 1 practically disappears from the final result, while in 12c, the result

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

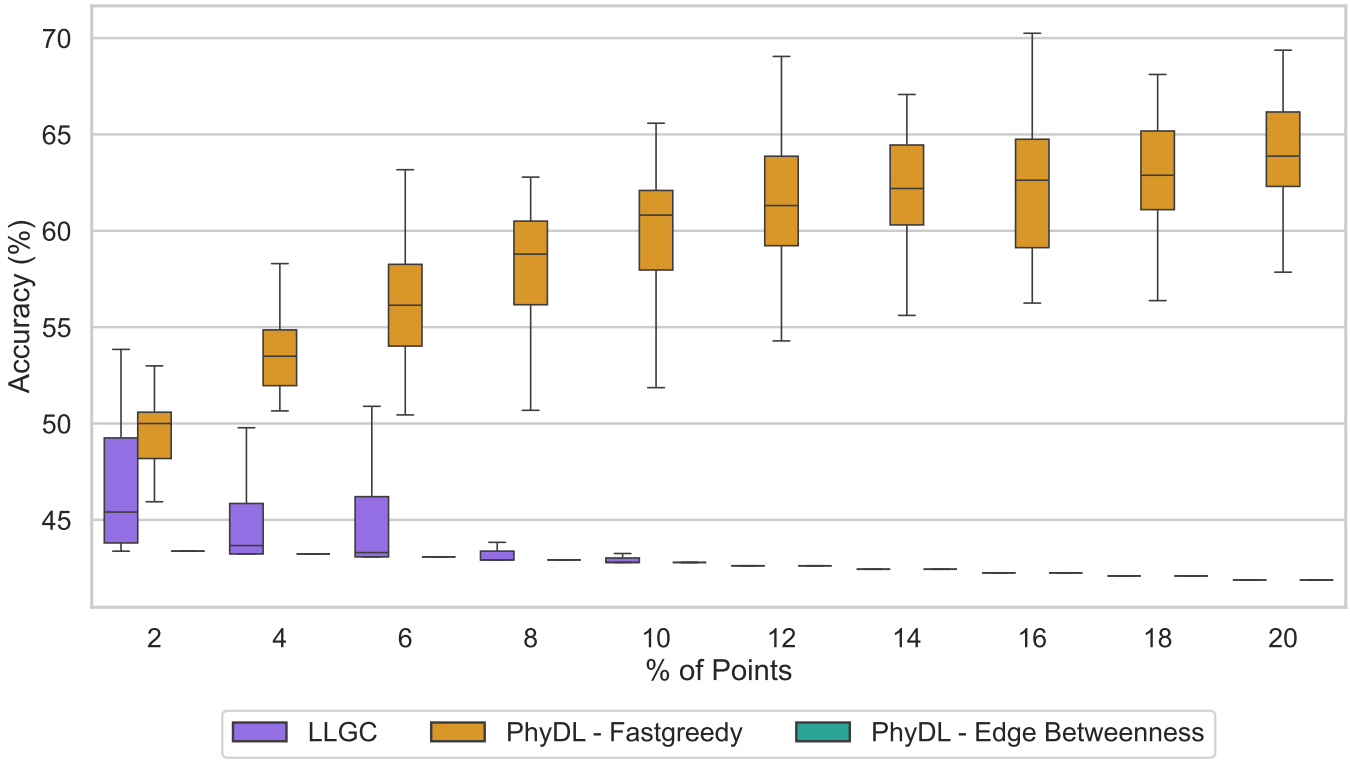


Figure 10: Accuracy results obtained on the MUSK dataset by LLGC contrasted with those achieved by PhyDL using two different CDA (Fastgreedy and Edge Betweenness) . On the x-axis, the ratio of initially labeled points considering the entire dataset. The boxplots illustrate the median and the interquartile range across the 30 independent executions.

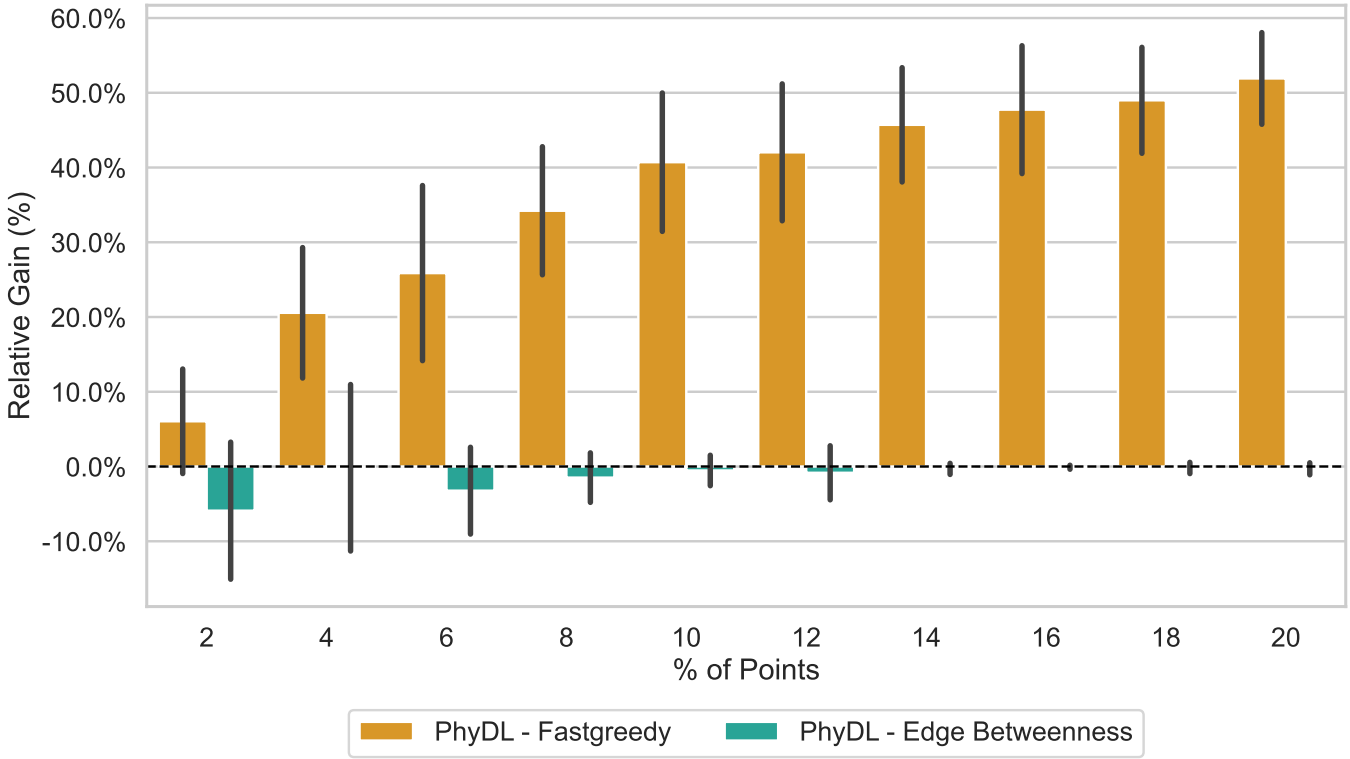
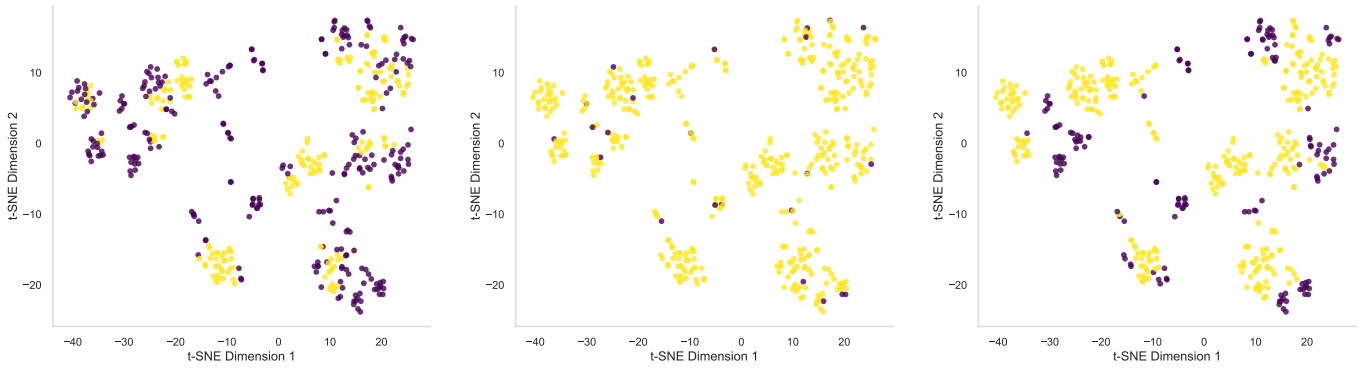


Figure 11: Relative Gain of PhyDL using two CDA (FastGreedy and Edge Betweenness) over LLGC on the MUSK dataset. The bar values represent the mean accuracy across the 30 independent executions.

resembles the original. This demonstrates that our proposal managed to surpass the result presented by LLGC.

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence



(a) Ground Truth (b) LLGC (c) PhyDL

Figure 12: t-SNE projections for the MUSK dataset: (a) Original representation where colors indicate the 2 classes (musk and non-musk); (b) Data labeled by LLGC, showing an accuracy of 42.79% ; and (c) Data labeled by PhyDL using FastGreedy, achieving an accuracy of 62.33%. For both labeling results, 20% of the samples were initially labeled (class-balanced).

3.4 Experiments and Results using the HEART Database

In the experiments conducted on the HEART dataset, the baseline LLGC method consistently demonstrated the best performance among the analyzed methods, indicating that for this specific spatial topology, standard label diffusion without community aggregation is more effective, Figure 13 presents these results. LLGC accuracy showed a steady increase, starting from a median of 65.5% with 2% of labeled points and growing to 78.0% with 20% of labeled points. PhyDL using the FastGreedy CDA maintained an intermediate performance; although it started slightly higher with a median accuracy of 68.1% at 2% of labeled samples, it was subsequently surpassed by the baseline, fluctuating in the following scenarios and reaching 73.8% at the end of the experiment (with a localized peak of 74.7% at 16%). PhyDL using the Edge Betweenness CDA exhibited the lowest performance, remaining relatively constant with a median accuracy hovering around 54% regardless of the increase in labeled data.

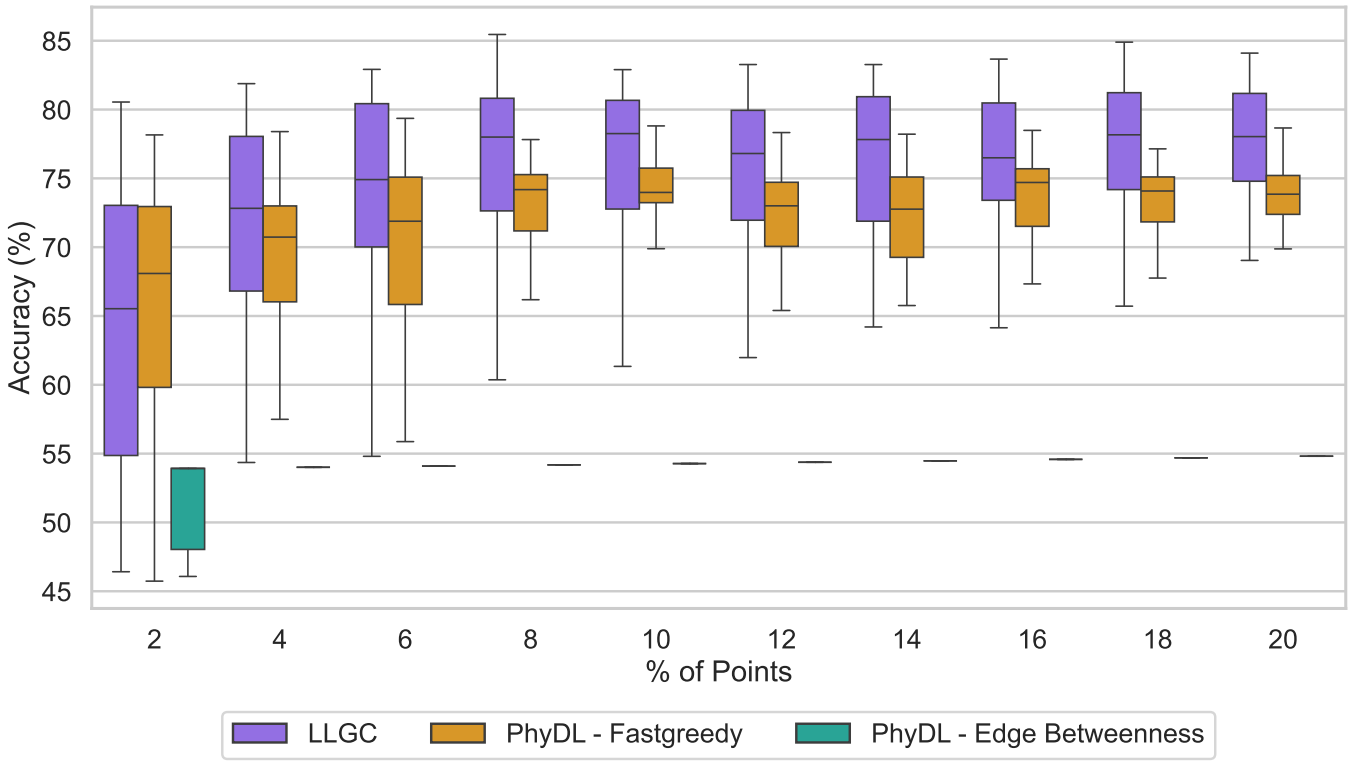


Figure 13: Accuracy results obtained on the HEART dataset by LLGC contrasted with those achieved by PhyDL using two different CDA (Fastgreedy and Edge Betweenness) . On the x-axis, the ratio of initially labeled points considering the entire dataset. The boxplots illustrate the median and the interquartile range across the 30 independent executions.

Regarding the average relative gain per execution (Figure 14), the inclusion of the phylogram aggregation step in PhyDL resulted in a performance decrease on this dataset. Consequently, the relative gains were consistently negative. PhyDL with the

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

FastGreedy CDA exhibited slight but consistent relative losses compared to LLGC, starting at -1.6% with 2% of labeled samples and dropping to -4.6% with 18% of labeled samples, finalizing at -3.8% when 20% of the data was labeled. PhyDL with the Edge Betweenness CDA showed a more substantial performance drop, presenting a relative loss of -20.3% at 2% and worsening to approximately -28.6% with 20% of labeled samples. These results highlight that while PhyDL is effective for datasets with well-defined topological manifolds (as seen in TwoMoons, MNIST, and SONAR), its community-based diffusion constraints can limit performance in datasets where the underlying structure does not align favorably with phylogenetic aggregation.

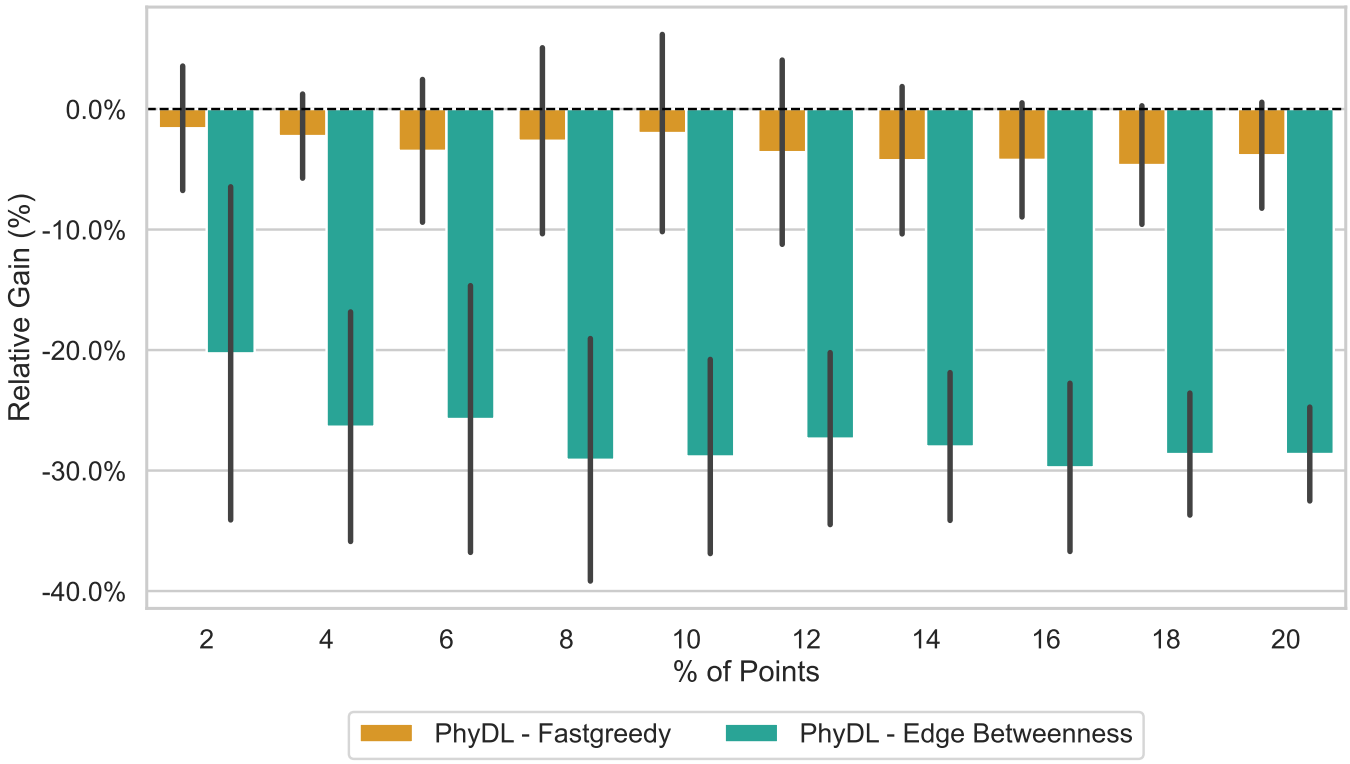
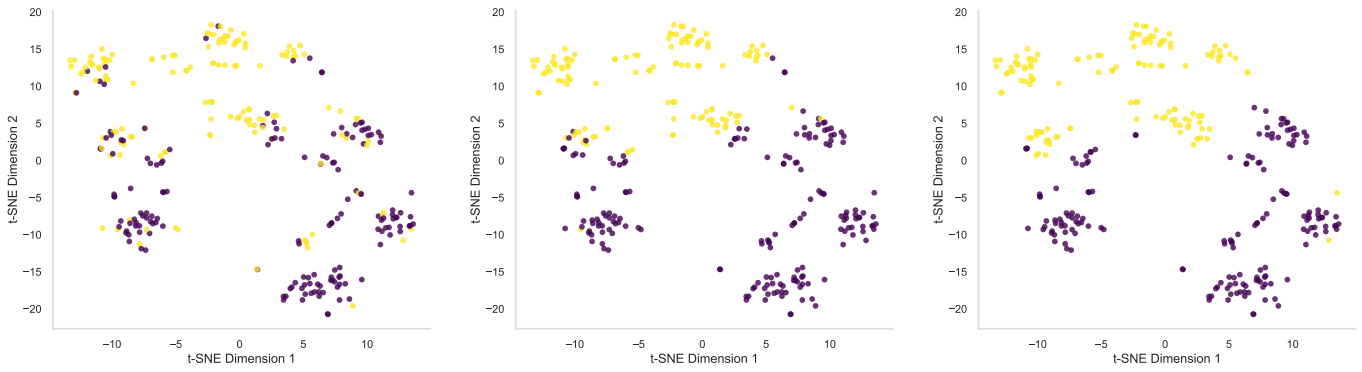


Figure 14: Relative Gain of PhyDL using two CDA (FastGreedy and Edge Betweenness) over LLGC on the HEART dataset. The bar values represent the mean accuracy across the 30 independent executions.



(a) Ground Truth (b) LLGC (c) PhyDL

Figure 15: t-SNE projections for the HEART dataset: (a) Original representation where colors indicate the 2 classes; (b) Data labeled by LLGC, showing an accuracy of 82.9%; and (c) Data labeled by PhyDL using FastGreedy, showing an accuracy of 78.81%. For both labeling results, 20% of the total data were initially labeled (class-balanced).

Figure 15a shows the projection of the HEART dataset using t-SNE with its classes represented. Figures 15b and 15c shows the same projection with the labeling performed by LLGC and PhyDL. Figure 15b has a closer resemblance to the original t-SNE projection, which corroborates the aforementioned results.

3.5 Experiments and Results using the SONAR Database

In the experiments with the SONAR dataset, PhyDL with the FastGreedy CDA demonstrated superior performance. As shown in Figure 16, it exhibited a consistent increase in accuracy, starting at a median of 53.9% with 2% of labeled samples and

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

reaching 66.7% with 20% of labeled samples (peaking at 66.9% when 18% of the data was labeled). This method also exhibited less variability in its results when compared to LLGC. The baseline LLGC showed inferior performance to PhyDL-FastGreedy, with its median accuracy ranging from 49.5% to 58.3%. Similar to PhyDL-FastGreedy, LLGC showed performance gains as the number of labeled samples increased, although these gains were more modest. PhyDL with the Edge Betweenness CDA demonstrated the most limited performance, remaining practically flat at around 46% median accuracy regardless of the number of points used. Despite showing the least variability among the three methods, it was completely unable to improve its predictive performance with more data.

Regarding the average relative gain across all independent executions, Figure 17 show PhyDL with the FastGreedy CDA achieved positive and increasing gains compared to the baseline. Starting with a modest gain of 1.7% using only 2% of labeled samples, the method showed a growing trend, ultimately reaching a relative gain of 14.6% when using 20% of labeled samples. Notably, the method presented a significant localized surge that touched the 20.0% mark when 10% of the samples were labeled, and its performance peaked strongly at an impressive 20.7% relative gain—surpassing the 20% threshold—with 18% of labeled samples. This positive progression suggests that the method effectively leverages the increase in data quantity to improve its performance. On the other hand, PhyDL with the Edge Betweenness CDA exhibited a systematic degradation behavior. The method started with a relative loss of 7.1% with 2% of labeled samples and showed a progressive deterioration, reaching a loss of 16.5% when using 20% of labeled samples. The divergence in the behavior of these methods becomes more pronounced as the percentage of points used increases, creating an ever-widening gap between their performances. While FastGreedy demonstrates the ability to learn and adapt positively with more data, Edge Betweenness shows a negative sensitivity to the increase in information volume in this dataset."

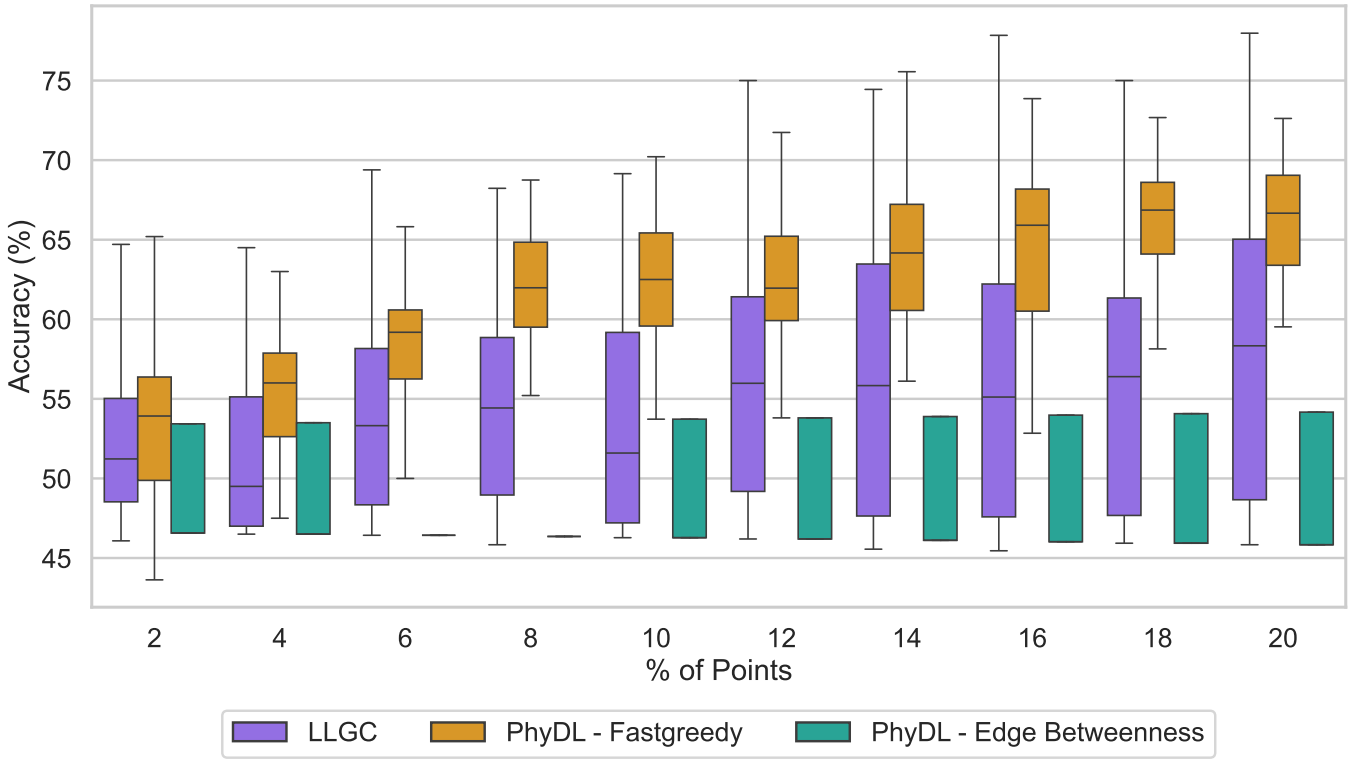


Figure 16: Accuracy results obtained on the SONAR dataset by LLGC contrasted with those achieved by PhyDL using two different CDA (Fastgreedy and Edge Betweenness) . On the x-axis, the ratio of initially labeled points considering the entire dataset. The boxplots illustrate the median and the interquartile range across the 30 independent executions.

Figure 18a shows the projection of the SONAR dataset using t-SNE with its classes represented. Figures 18b and 18c shows the same projection with the labeling performed by LLGC and PhyDL. Figure 18c visibly has a closer resemblance to the original t-SNE projection, which corroborates the presented results.

3.6 Statistical Significance Analysis

To validate the comparative performance of the proposed method against the baseline, a one-sided Wilcoxon signed-rank test was conducted. The alternative hypothesis established was that PhyDL achieves higher accuracy than LLGC ($PhyDL > LLGC$). A standard significance level of $\alpha = 0.05$ was adopted. Table 2 presents the resulting p-values for both CDAs across all datasets, as well as an overall aggregate evaluation. The statistical results corroborate the empirical observations discussed in the previous sections. PhyDL equipped with the FastGreedy algorithm demonstrated statistically significant superiority over LLGC overall (p-value < 0.05), as well as individually in the MUSK, SONAR, TwoMoons, and MNIST datasets. For the HEART

https://doi.org/10.21528/lnlm-vol24-no1-art2 © Brazilian Society on Artificial Intelligence

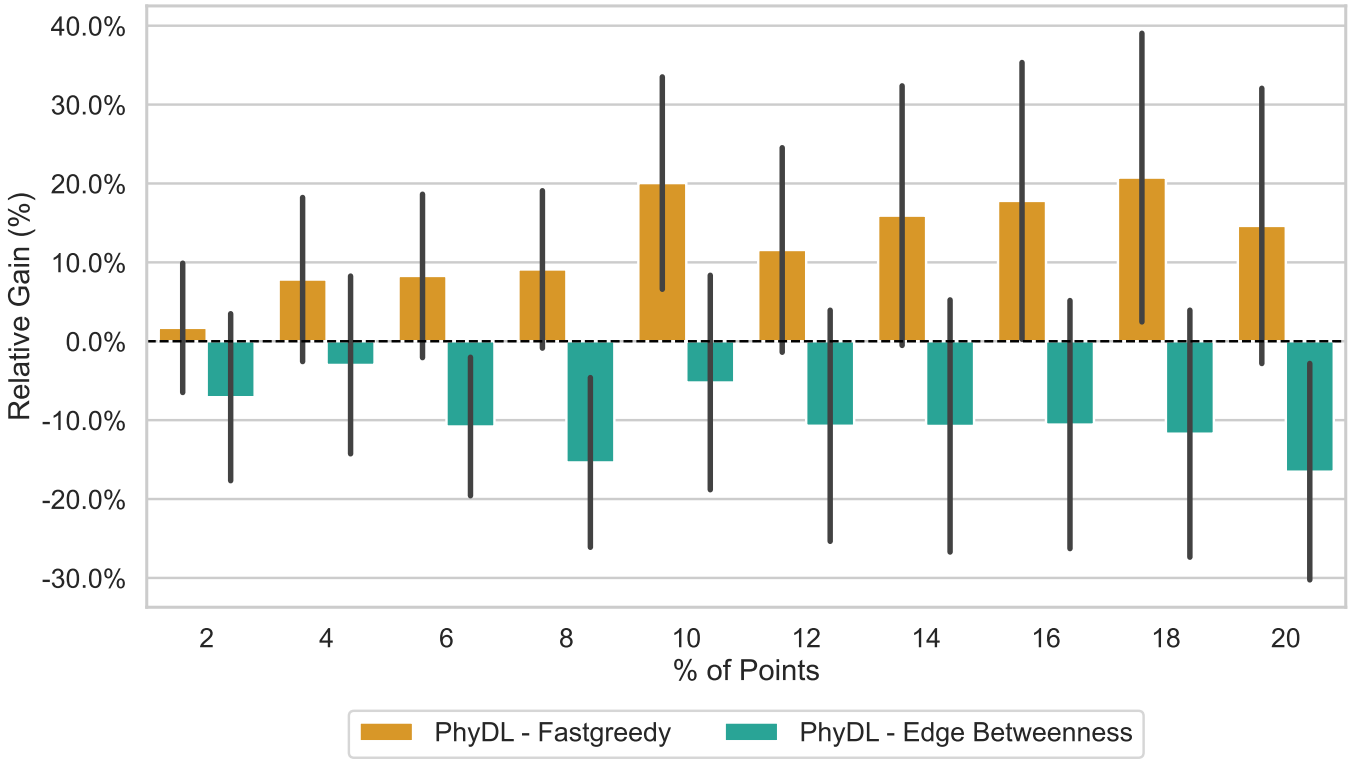
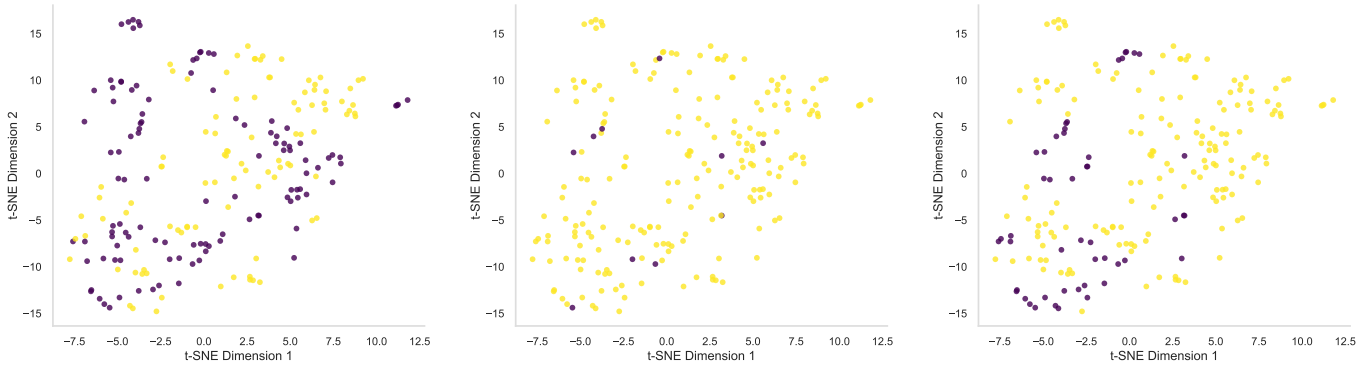


Figure 17: Relative Gain of PhyDL using two CDA (FastGreedy and Edge Betweenness) over LLGC on the SONAR dataset. The bar values represent the mean accuracy across the 30 independent executions.



(a) Ground Truth (b) LLGC (c) PhyDL

Figure 18: t-SNE projections for the SONAR dataset: (a) Original representation with colors representing the 2 classes; (b) Data labeled by LLGC, which showed an accuracy of 46.28%; and (c) Data labeled by PhyDL using FastGreedy CDA, achieving an accuracy of 56.91%. For both (b) and (c), 10% of the total data were initially labeled (class-balanced).

dataset, the test yielded a p-value of 1.0, confirming that PhyDL-FastGreedy does not outperform the baseline in this specific topology. Furthermore, the Edge Betweenness CDA failed to achieve statistical significance in any scenario, confirming its unsuitability for the evaluated datasets. These tests provide robust statistical confidence that PhyDL, when utilizing an appropriate CDA like FastGreedy, delivers significant improvements in semi-supervised data labeling tasks.

4 CONCLUSIONS

From the presented experiments, it is observed that label diffusion performance can be significantly improved by the joint application of phylogram construction and community detection. In most of the evaluated scenarios, PhyDL equipped with the FastGreedy algorithm demonstrated superior robustness and predictive performance compared to the standard LLGC baseline, successfully outperforming it in 4 out of 5 datasets (TwoMoons, MNIST, MUSK, and SONAR).

As expected, an increase in the number of labeled samples generally correlates with higher accuracy rates. Notably, the choice of the community detection algorithm proved to be a critical factor; while FastGreedy effectively leveraged the underlying data structures—achieving relative accuracy gains that exceeded 370% on the MNIST dataset—Edge Betweenness exhibited systematic degradation and failed to capture the necessary structural nuances across the complex datasets. Furthermore, the results from

Table 2: Wilcoxon test p-values for the alternative hypothesis that PhyDL outperforms LLGC ($PhyDL > LLGC$) across different datasets. A significance level of $\alpha = 0.05$ was used.

Dataset	FastGreedy (p-value)	Significant	EdgeBetw. (p-value)	Significant
Overall	6.71×10^{-155}	Yes	1.0	No
HEART	1.0	No	1.0	No
MUSK	8.64×10^{-51}	Yes	1.0	No
SONAR	3.14×10^{-32}	Yes	1.0	No
TwoMoons	3.76×10^{-47}	Yes	1.0	No
MNIST	3.04×10^{-51}	Yes	1.0	No

the HEART dataset highlighted a vital topological condition: for certain spatial distributions, standard label diffusion without community aggregation remains more effective. Although PhyDL presents a higher computational cost than LLGC due to the $O(n^3)$ complexity of the phylogram construction, these activities can be executed independently. Since semi-supervised learning is primarily encouraged for applications where manual labeling is expensive and scarce, the substantial accuracy improvements provided by PhyDL-FastGreedy easily outweigh the additional execution time required.

Observing the results for the TwoMoons databases, we can also infer that LLGC presented responses close to PhyDL when the database has a more compact and well-separated spatial structure regarding the represented classes. Observing the data for the TwoMoons class, which presents samples of the same class very close together and a structural separation between the classes quite distinct, LLGC obtained significant results, although inferior to those found by PhyDL. The fact that LLGC is part of the group of algorithms that use a graph to represent the data provides good results in databases similar to TwoMoons. When applied to the MNIST database, the performance of LLGC drops drastically compared to the result presented when the database used is TwoMoons. For the MNIST database, the use of PhyDL adds a lot of information, favoring correct labeling.

For both the TwoMoons and MNIST datasets, the choice of community detection algorithm to be used by PhyDL is indifferent when the analysis focuses on performance gain compared to LLGC. In these datasets, the classes are well separated, as can be seen in Figures 6 and 9. When using more complex real datasets (MUSK, HEART, and SONAR), it is observed that Edge Betweenness performs worse compared to FastGreedy and LLGC. Figures 12a, 15a, and 18a illustrate that the classes are not as well separated; they undergo a mixture in the projection, unlike the MNIST dataset where each class has a well-defined spatial location.

Although the analyzed real datasets have this characteristic, FastGreedy tends to identify groups of densely connected nodes, prioritizing the union of communities in its iterative process that are already well connected, thus respecting the local topology of the network. Meanwhile, Edge Betweenness focuses exclusively on the betweenness of nodes as a criterion for producing communities. The removal of nodes that act as bridges between communities can quickly divide the network into disconnected components, ignoring the density of connections within communities. Moreover, in real networks with noise, betweenness may not perfectly reflect the structure of communities, leading to premature and potentially incorrect divisions. Given this, FastGreedy is considered the most suitable choice for use by PhyDL.

However, the HEART dataset shows that PhyDL with FastGreedy CDA has limitations and may not outperform LLGC in some cases. Among the 5 datasets analyzed, HEART was the only one where PhyDL with FastGreedy CDA did not surpass LLGC. Not all attributes of the original dataset were used, so the result may suggest that the attributes used do not well represent the data separation using Euclidean distance, which is used in constructing the distance matrix to be used by PhyDL.

Despite its effectiveness, the computational complexity of the phylogram construction, primarily bounded by the $O(n^3)$ cost of the Neighbor-Joining algorithm, poses a scalability challenge for massive Big Data applications. Future work will focus on adapting PhyDL to distributed computing frameworks or exploring heuristic approximations to mitigate this computational bottleneck. Additionally, integrating the phylogram-based structural analysis with state-of-the-art representations—such as high-dimensional embeddings generated by Large Language Models (LLMs) or modern deep clustering architectures—represents a promising avenue to explore richer semantic distances in highly complex domains.

ACKNOWLEDGEMENT

The authors gratefully acknowledge financial support from the Center for Artificial Intelligence (C4AI-USP, FAPESP grant #2019/07665-4), the IBM Corporation, CEPID-CeMEAI/ICMC-USP (CEPID, FAPESP grant #2013/07375-0), the INCT, the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES, Finance Code 001), the FAPEMA (Foundation for Research Support of Maranhão) and IFMA (Federal Institute of Maranhão).

REFERENCES

[1] Y. Grandvalet and Y. Bengio. “Semi-supervised learning by entropy minimization”. In *Advances in neural information processing systems*, volume 17, pp. 529–536, 2005.

[2] J. E. van Engelen and H. H. Hoos. “A survey on semi-supervised learning”. *Machine Learning*, vol. 109, no. 2, pp. 373–440, November 2019.

[3] *Semi-Supervised Learning*. The MIT Press, September 2006.

[4] K. Zhang, Y. Zhang and X. Liu. "Semi-Supervised Graph Learning for Brain Disease Identification". *Journal of Applied Mathematics and Physics*, vol. 11, no. 07, pp. 1846–1859, 2023.

[5] M. Belkin and P. Niyogi. "Semi-supervised learning on Riemannian manifolds". *Machine learning*, vol. 56, no. 1, pp. 209–239, 2004.

[6] H. Zhu and P. Koniusz. "Simple spectral graph convolution". In *International conference on learning representations*, 2021.

[7] J. Gasteiger, S. Weißenberger and S. Günnemann. "Diffusion improves graph learning". *Advances in neural information processing systems*, vol. 32, 2019.

[8] X. Zhu. "Semi-Supervised Learning Literature Survey". Technical report, Computer Sciences Technical Report, 1530, 2005.

[9] E. K. Tokuda, G. B. A. Ferreira, C. Silva and R. M. Cesar. "A novel semi-supervised detection approach with weak annotation". In *2018 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, p. 129–132. IEEE, April 2018.

[10] M. Belkin and P. Niyogi. "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation". *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, June 2003.

[11] Q. Li, Z. Han and X.-M. Wu. "Deeper insights into graph convolutional networks for semi-supervised learning". In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[12] X. Zhu, Z. Ghahramani and J. Lafferty. "Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions". In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, 2003.

[13] A. Blum and S. Chawla. "Learning from Labeled and Unlabeled Data using Graph Mincuts". In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, p. 19–26, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[14] L. Getoor and B. Taskar. *Introduction to Statistical Relational Learning*. The MIT Press, August 2007.

[15] Y. Bengio, A. Courville and P. Vincent. "Representation Learning: A Review and New Perspectives". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, August 2013.

[16] D. Zhou, O. Bousquet, T. N. Lal, J. Weston and B. Schölkopf. "Learning with local and global consistency". In *Advances in neural information processing systems*, pp. 321–328, 2004.

[17] J. Felsenstein. *Inferring phylogenies*. Sinauer Associates, 2004.

[18] B. K. M. Cesar. "Estudo e extensão da metodologia DAMICORE para tarefas de classificação". Ph.D. thesis, Universidade de São Paulo, Agência USP de Gestão da Informação Acadêmica (AGUIA), 2016.

[19] A. H. M. Soares. "Algoritmos de estimação de distribuição baseados em árvores filogenéticas". Ph.D. thesis, Universidade de São Paulo, 2014.

[20] B. d. A. Silva. "Um método de otimização da relação desempenho/consumo de energia para arquiteturas multi-cores heterogêneas em FPGA". Ph.D. thesis, Universidade de São Paulo, 2016.

[21] F. G. Z. Kharrat. "FS-OPA: combinação de otimização baseada em análise de filogramas e regressão de cox aplicada a conjuntos de dados de transtornos mentais". Ph.D. thesis, Universidade de São Paulo, 2019.

[22] A. S. O. Bailão. "Reconhecimento de padrões por processos adaptativos de compressão". Ph.D. thesis, Universidade Federal de Goiás, 2020.

[23] L. Deng. "The mnist database of handwritten digit images for machine learning research". *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[24] T. Sejnowski and R. Gorman. "Connectionist Bench (Sonar, Mines vs. Rocks)". UCI Machine Learning Repository, 1988. DOI: https://doi.org/10.24432/C5T01Q.

[25] S. W. P. M. Janosi, Andras and R. Detrano. "Heart Disease". UCI Machine Learning Repository, 1989. DOI: https://doi.org/10.24432/C52P4X.

[26] D. Chapman and A. Jain. "Musk (Version 1)". UCI Machine Learning Repository, 1994. DOI: https://doi.org/10.24432/C5ZK5B.

[27] B. d. A. Silva, L. A. Cuminato, A. C. Delbem, P. C. Diniz and V. Bonato. "Application-oriented cache memory configuration for energy efficiency in multi-cores". *IET Computers & Digital Techniques*, vol. 9, no. 1, pp. 73–81, January 2015.

[28] R. d. S. Pinto. "Caracterização do perfil de utilização de recursos de programas a partir de arquivos executáveis utilizando mineração de dados". Ph.D. thesis, Universidade de São Paulo, 2017.

[29] N. Saitou and M. Nei. "The neighbor-joining method: a new method for reconstructing phylogenetic trees." *Molecular biology and evolution*, vol. 4, no. 4, pp. 406–425, 1987.

[30] K. Atteson. "The performance of neighbor-joining methods of phylogenetic reconstruction". *Algorithmica*, vol. 25, no. 2, pp. 251–278, 1999.

[31] R. Mihaescu, D. Levy and L. Pachter. "Why neighbor-joining works". *Algorithmica*, vol. 54, pp. 1–24, 2009.

[32] M. Girvan and M. E. J. Newman. "Community structure in social and biological networks". *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, June 2002.

[33] A. Clauset, M. E. J. Newman and C. Moore. "Finding community structure in very large networks". *Physical Review E*, vol. 70, no. 6, December 2004.

[34] M. E. J. Newman. "Fast algorithm for detecting community structure in networks". *Physical Review E*, vol. 69, no. 6, June 2004.

[35] M. E. J. Newman and M. Girvan. "Finding and evaluating community structure in networks". *Physical Review E*, vol. 69, no. 2, February 2004.

[36] L. Van der Maaten and G. Hinton. "Visualizing data using t-SNE." *Journal of machine learning research*, vol. 9, no. 11, 2008.