Relaxed Dissimilarity-based Symbolic Histogram Variants for Granular Graph Embedding

Luca Baldini[1][a], Alessio Martino[2,3][b] and Antonello Rizzi[1][c]
[1]*Department of Information Engineering, Electronics and Telecommunications, University of Rome "La Sapienza", Via Eudossiana 18, 00184 Rome, Italy*
[2]*Institute of Cognitive Sciences and Technologies (ISTC-CNR), Italian National Research Council, Via San Martino della Battaglia 44, 00185 Rome, Italy*
[3]*Department of Business and Management, LUISS University, Viale Romania 32, 00197 Rome, Italy*

Keywords: Structural Pattern Recognition, Supervised Learning, Embedding Spaces, Granular Computing, Graph Edit Distances, Graph Embedding.

Abstract: Graph embedding is an established and popular approach when designing graph-based pattern recognition systems. Amongst the several strategies, in the last ten years, Granular Computing emerged as a promising framework for structural pattern recognition. In the late 2000's, symbolic histograms have been proposed as the driving force in order to perform the graph embedding procedure by counting the number of times each granule of information appears in the graph to be embedded. Similarly to a bag-of-words representation of a text corpora, symbolic histograms have been originally conceived as integer-valued vectorial representation of the graphs. In this paper, we propose six 'relaxed' versions of symbolic histograms, where the proper dissimilarity values between the information granules and the constituent parts of the graph to be embedded are taken into account, information which is discarded in the original symbolic histogram formulation due to the hard-limited nature of the counting procedure. Experimental results on six open-access datasets of fully-labelled graphs show comparable performance in terms of classification accuracy with respect to the original symbolic histograms (average accuracy shift ranging from -7% to +2%), counterbalanced by a great improvement in terms of number of resulting information granules, hence number of features in the embedding space (up to 75% less features, on average).

1 INTRODUCTION

In the last two decades, we witnessed a growing interest of the scientific community in applying pattern recognition (and related fields) techniques to the domain of graph data structures. This interest can be naturally explained by the representation power conveyed by the semantic and topological description that graph domain offers in describing processes where objects or entities interact together. In fact, structured data such as graphs are largely used to model complex phenomena in different application fields such as economic markets, human social interactions, immune systems, disease diffusion and generally in situations where parts of the system under investigation are connected together in a network represented as a graph.

On the other hand, the deployment of pattern recognition techniques in graph domain is a non-trivial operation due its implicit structured and combinatorial nature. Compared to geometric spaces such as the Euclidean one, basic algebraic operations like the computation of similarity or dissimilarity between two graphs are not naturally defined. Furthermore, vector data are static objects with predefined length, whilst different patterns in a set of graphs can show different sizes in terms of number of nodes and/or edges, posing additional limitations in utilizing a plethora of well-known statistical learning algorithms (Bunke, 2003).

It is possible to spot three main approaches in the current literature for dealing with the limitations imposed by the non-geometrical nature of the graph domain. The most intuitive and natural approach consists in defining an ad-hoc dissimilarity measure di-

[a] https://orcid.org/0000-0003-4391-2598
[b] https://orcid.org/0000-0003-1730-5436
[c] https://orcid.org/0000-0001-8244-0015

Baldini, L., Martino, A. and Rizzi, A.
Relaxed Dissimilarity-based Symbolic Histogram Variants for Granular Graph Embedding.
DOI: 10.5220/0010652500003063
In *Proceedings of the 13th International Joint Conference on Computational Intelligence (IJCCI 2021)*, pages 221-235
ISBN: 978-989-758-534-0; ISSN: 2184-2825

rectly in the graph domain, notably *Graph Edit Distances* (GED) (Bunke and Jiang, 2000; Riesen et al., 2010; Bunke and Allermann, 1983). In the latest decades, *kernel methods* have been deeply studied as a valid approach for combining pattern recognition and graph domain (Ghosh et al., 2018; Kriege et al., 2020). Given a kernel function $\kappa : \mathcal{G} \times \mathcal{G}$ where $\mathcal{G}$ is the graph domain, kernel methods measure the pairwise similarities between graphs. When the resulting kernel matrix $\mathbf{K}$ (i.e., the matrix containing all the pairwise kernel evaluations between the available graphs) is positive (semi)definite, κ is valid dot product in an implicit Hilbert space. Opposite to kernel methods, *graph embedding* procedures explicitly map graphs into a geometric embedding space thanks to a suitable embedding function $\Phi : \mathcal{G} \to \mathcal{X}$, where $\mathcal{X} \subseteq \mathbb{R}^n$. This approach results very flexible since it enables the use of all the algorithmic frameworks available in the field of pattern recognition. The main challenge posed by graph embedding approaches is to correctly design a Φ function able to keep topological and semantic properties of the original domain intact in the target domain. For this reason, several mapping functions have been proposed and applied in different application domains. A straightforward method consists in manually collect n relevant descriptive numerical features of the graph into a n-length real-valued vector. Such approach can be tedious and requires a deep (and often not available) knowledge of the underlying process. In (Bunke and Riesen, 2008), the authors introduced a graph embedding procedure based on *dissimilarity representation* (Pękalska and Duin, 2005; Duin and Pękalska, 2012). Given a set $\mathcal{R}$ of n prototype graphs extracted from the training data and a graph G to be embedded, the vector representation of G can be expressed as an n-length vector whose i^{th} component is defined as $d(p_i, G)$, where $p_i \in \mathcal{R}$ and $d : \mathcal{G} \times \mathcal{G}$ is a suitable dissimilarity measure. Major drawbacks of this procedure are the computational cost needed for evaluating $d(\cdot,\cdot)$ between large graphs and/or large datasets and the choice of enough informative prototypes to populate $\mathcal{R}$ (Pękalska et al., 2006). Another interesting human inspired paradigm known as *Granular Computing* (GrC) (Zadeh, 1997; Bargiela and Pedrycz, 2003; Pedrycz, 2001) can be exploited for graph embedding scope. A GrC approach consists in extracting relevant information (known as *information granules*) by observing the problem at different levels of abstraction. In (Bianchi et al., 2014), GrC has been employed for extracting automatically relevant prototype subgraphs in conjunction with the *symbolic histogram* representation for performing the embedding procedure (Del Vescovo and Rizzi, 2007a; Del Vescovo and Rizzi, 2007b). In particular, from the symbolic histogram perspective, the i^{th} component of the embedded graph G is obtained by counting the occurrences of i^{th} relevant substructure (i.e., the granule extracted with a suitable granulation procedure) in G. The process of occurrences counting is accomplished by an hard-limiting function which triggers a counter whether the dissimilarity measure between the specific prototype and a subgraph $g \in G$ is below a given threshold.

In this paper, we investigate a novel approach based on symbolic histograms for granular graph embedding. In particular, we 'relax' the evaluation of the occurrences for the symbolic histogram in favour of three different types of functions, that is, instead of relying on counting the number of occurrences (i.e., an integer-valued embedding vector), we take into consideration the proper dissimilarity values when searching for granules occurrences within the graph to be embedded, following the rationale behind the aforementioned dissimilarity space embedding.

The remainder of the paper is structured as follows: in Section 2 we describe the GrC-based classification system, along with the original symbolic histograms definition; Section 3 contains the main novelty of the paper, as we describe the six 'relaxed' symbolic histograms variants; Section 4 regards the computational experiments, including an exhaustive comparison against state-of-the-art techniques. Finally, Section 5 concludes the paper. The paper also features an Appendix in which we give an example of how GrC-based pattern recognition systems allow a certain degree of interpretability by analyzing the resulting granules of information.

2 GRANULAR GRAPH EMBEDDING WITH SYMBOLIC HISTOGRAMS

In this section, we describe the procedure that allows the embedding and classification of a set of graphs according to the GrC paradigm. The core idea behind this method is to synthesize an *alphabet* of symbols $\mathcal{A} = \{s_1, \ldots, s_n\}$, namely the set of meaningful and relevant subgraphs extracted from the training set under analysis by following a granular approach. The symbols, i.e. granules of information, emerge thanks to a granulation procedure based on the Basic Sequential Algorithmic Scheme (BSAS) free clustering method (Theodoridis and Koutroumbas, 2008), driven by a resolution parameter θ which determines the granularity level of observation. Finally, the gen-

erated alphabet $\mathcal{A}$ is employed for building the symbolic histogram of each graph, hence moving the classification problem towards a metric space where any statistical classifier can be employed. In the seminal paper (Bianchi et al., 2014), this approach has been successfully exploited for building a graph classification system named GRALG.

2.1 Substructures Extraction

Starting from the training set $\mathcal{S}^{(\mathrm{tr})}$, this block extracts a set of subgraphs $\mathcal{S}_g^{(\mathrm{tr})}$. The design of this block is crucial for two different aspects that can undermine the entire embedding procedure. From a memory footprint point of view, an exhaustive extraction of all the subgraphs in $\mathcal{S}^{(\mathrm{tr})}$ can be unfeasible. In fact, the number of subgraphs that can be extracted from a single graph grows combinatorially with the number of its nodes. This poses a serious limitation to the usability of this approach, constricting the application field to problems involving graphs of small size. Additionally, the computational cost of the following blocks, in particular the granulation procedure, are strongly influenced by the number of elements in $\mathcal{S}_g^{(\mathrm{tr})}$, possibly making the entire graph embedding procedure unfeasible. The second aspects regards the choice of the desired topology of the subgraphs to be extracted (e.g., cliques, paths, stars and the like). Even though this point can give flexibility to our approach, the selection of an appropriate traversal algorithm is not trivial and likely influences the performance of a graph classification system driven by finding meaningful subgraphs for graph embedding purposes (like the one we present in this work) since the topological properties of the original graphs must be reflected in the subgraphs to be extracted. In this work, we use the stratified stochastic extraction procedure proposed in (Baldini et al., 2021) in order to limit the cardinality of $\mathcal{S}_g^{(\mathrm{tr})}$ to a user-defined value W. The procedure goes as follows:

1. For a given problem-related class, say the i^{th}, find the number of subgraphs to be extracted as $f = W \cdot \frac{|\mathcal{S}^{(\mathrm{tr},i)}|}{|\mathcal{S}|}$, where $\mathcal{S}^{(\mathrm{tr},i)}$ denotes the subset of $\mathcal{S}^{(\mathrm{tr})}$ containing only patterns belonging to class i
   (a) A single graph G is extracted uniformly at random from $\mathcal{S}^{(\mathrm{tr},i)}$
   (b) The selected graph G is visited using one of the many traversing strategies available in the computer science and graph theory communities–such as Breath First Search (BFS), Depth First Search (DFS) and random walks–until a fixed number of nodes V are visited
   (c) Visited nodes and edges are stored in the resulting subgraph g and collected into $\mathcal{S}_g^{(\mathrm{tr},i)}$
   (d) The procedure goes back to 1a until the cardinality of $\mathcal{S}_g^{(\mathrm{tr},i)}$ has reached f
2. The procedure goes back to 1 until all classes are considered.

The rationale behind the stratified approach is that the bucket of subgraphs $\mathcal{S}_g^{(\mathrm{tr})}$ is actually a bucket-of-buckets $\mathcal{S}_g^{(\mathrm{tr})} = \{\mathcal{S}^{(\mathrm{tr},1)}, \ldots, \mathcal{S}^{(\mathrm{tr},c)}\}$, with c being the number of classes, in such a way that the i^{th} bucket only contains subgraphs extracted from patterns belonging to class i. This assures that, if compared to a purely uniform random extraction (Baldini. et al., 2019), all classes are represented in the bucket, with a stratified number of subgraphs that scales according to the frequency of the class itself (see Step #1), to ensure fairness in the distribution.

2.2 Granulation Method for Alphabet Synthesis

In this subsection, we give a formal description of the process that synthesizes the alphabet $\mathcal{A}$. In the spirit of GrC, each symbol $s_i \in \mathcal{A}$ is an highly informative mathematical entity following the principle of justifiable granularity related to the specific level of abstraction at which the problem has been observed (Pedrycz and Homenda, 2013). In the literature, many authors have given several different definitions of information granule: fuzzy sets, rough sets and shadowed sets (Pedrycz et al., 2015) are just few examples on how granules can be practically formalized. In our work, we follow a clustering-based approach (Wang et al., 2016) for formally defining a granule of information. As anticipated in Section 2, the core of the granulation procedure is the BSAS algorithm working directly on the set of subgraphs $\mathcal{S}_g^{(\mathrm{tr})}$. It relies on four fundamental actors: a threshold of inclusion θ, a clustering representative g^*, a dissimilarity measure $d : \mathcal{G} \times \mathcal{G} \to \mathbb{R}$ and a threshold Q of maximum allowed number of clusters. By varying θ, the clusters resolution will change accordingly, effectively impacting on the granularity level. Thus, we generate a set of partitions $\mathcal{P} = \{\mathcal{P}_{\theta_1}, \ldots, \mathcal{P}_{\theta_t}\}$, where partition $\mathcal{P}_{\theta_i}$ collects each cluster $\mathbf{C}$ emerged by imposing θ_i as the resolution value. In order to account for only well-formed clusters as candidate information granules, we define the following cluster quality index $F(\mathbf{C}, \rho) \in [0,1]$ (the lower is better):

$$F(\mathbf{C}, \rho) = \rho \cdot \Psi(\mathbf{C}) + (1-\rho) \cdot \Omega(\mathbf{C}) \qquad (1)$$

as the linear convex combination between $\Psi(\mathbf{C})$ and $\Omega(\mathbf{C})$, respectively compactness and cardinality of

cluster $\mathbf{C}$, weighted by a trade-off parameter $\rho \in [0,1]$:

$$\Psi(\mathbf{C}) = \frac{1}{|\mathbf{C}-1|} \sum_{g \in \mathbf{C}} d(g^*, g) \tag{2}$$

$$\Omega(\mathbf{C}) = 1 - \frac{|\mathbf{C}|}{|\mathcal{S}_g^{(\mathrm{tr})}|} \tag{3}$$

In this work, we assume g^* be the MinSOD (Del Vescovo et al., 2014; Martino et al., 2019b) element of $\mathbf{C}$, that is the graph that minimizes the pairwise sum of distances among all patterns in the cluster. Clearly, the dissimilarity measure d must be tailored to the input space under analysis (i.e., the graph domain): our choice felt on a weighted GED-based heuristics named node-Best Match First (nBMF). Full details on nBMF, along with detailed pseudocodes, can be found in (Baldini. et al., 2019; Martino and Rizzi, 2021).

Finally, each cluster proves its validity by comparing its own quality $F(\mathbf{C})$ with a threshold $\tau \in [0,1]$. The MinSoDs of the surviving clusters are then collected together in the alphabet $\mathcal{A}$.

Recalling that $\mathcal{S}_g^{(\mathrm{tr})}$ is actually a set-of-sets, the above procedure is independently evaluated on each class-related bucket $\mathcal{S}^{(\mathrm{tr},i)}|_{i=1}^{c}$: this means that c class-aware alphabets $\mathcal{A}^{(i)}|_{i=1}^{c}$ are returned, hence the overall alphabet simply reads as the concatenation of the class-related alphabets $\mathcal{A} = \cup_{i=1}^{c} \mathcal{A}^{(i)}$.

2.3 Embedding with Symbolic Histograms

In this block, the embedding function $\Phi : \mathcal{G} \to \mathcal{X}$ is implemented by following the symbolic histogram paradigm (Del Vescovo and Rizzi, 2007b). This method allows the embedding of a graph G into a feature space $\mathcal{X}$ according to two core sets: the first is a collection of prototype substructures that, in our case, coincides with the alphabet $\mathcal{A}$ obtained in Section 2.2; the second G_{exp} is the set of subgraphs that compose G. In other words, $G_{exp} = \{g_1, \ldots, g_k\}$ is a (possibly non-exhaustive) decomposition of the graph G in k atomic units. The symbolic histogram $\mathbf{h}_G^{\mathcal{A}}$ is then defined as:

$$\mathbf{h}_G^{\mathcal{A}} = [\mathrm{occ}(s_1, G_{exp}), \ldots, \mathrm{occ}(s_n, G_{exp})] \tag{4}$$

By observing Eq. (4), we can see that $\mathbf{h}_G^{\mathcal{A}}$ is an integer-valued vector with n components, where $n = |\mathcal{A}|$. The process of counting occurrences of a symbol $s_i \in \mathcal{A}$ in G_{exp} is performed by the function $\mathrm{occ} : \mathcal{A} \times \mathcal{G} \to \mathbb{N}$, defined as follows:

$$\mathrm{occ}(s_i, G_{exp}) = \sum_{g \in G_{exp}} \Gamma(s_i, g) \tag{5}$$

In Eq. (5), Γ is a function that defines whether a symbol s_i can be matched with a subgraph $g \in G_{exp}$. Exact match between two graphs (also known as graph or subgraph isomorphism) is often an unpractical operation for its hardness from a computational complexity point of view, especially when nodes and edges are equipped with real-valued vectors or even with custom data structures (Emmert-Streib et al., 2016). In order to account for inexactness in the matching procedure, Γ evaluates the degree of dissimilarity $d(s_i, g)$ between s_i and g, then it triggers the counter (i.e., a match is considered a hit) only if this value does not exceed a symbol-dependent threshold ζ_{s_i}:

$$\Gamma(s_i, g) = \begin{cases} 1 & \text{if } d(s_i, g) \leq \zeta_{s_i} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

2.4 Alphabet Optimization

After having synthesized a candidate alphabet $\mathcal{A}$, each graph from the dataset at hand can be embedded toward the resulting embedding space $\mathcal{X} \subseteq \mathbb{R}^n$. A natural question that arise is how to determine the quality of this embedding space in order to evaluate the goodness of $\mathcal{A}$. A possible solution already explored in GRALG consists in training a classifier $\mathcal{H}$ in the embedding space and evaluating a performance measure $\pi : \mathcal{X} \to \mathbb{R}$ of $\mathcal{H}$ in classifying a validation set. Then, π can be interpreted as a critic of $\mathcal{H}$ about the embedding space $\mathcal{X}$. In order to automatize the search for a suitable and informative alphabet, an evolutionary optimization phase (driven, for example, by a differential evolution (Storn and Price, 1997)) whose fitness function relies on π can be employed, aiming at synthesizing ever-improving alphabets.

The genetic code of each individual, summarized in Eq. (7), contains $\mathbf{w} = \{w_{ins}^{node}, w_{del}^{node}, w_{sub}^{node}, w_{ins}^{edge}, w_{del}^{edge}, w_{sub}^{edge}\} \in [0,1]^6$, which are the nodes and edges insertion, deletion, substitution weights for the weighted GED and $\boldsymbol{\gamma}$, the set of parameters driving nodes and edges dissimilarity measures (if applicable). Additionally, the procedure optimizes the relevant parameters for the granulation procedure, namely the maximum number of clusters allowed for BSAS $Q \in [1, Q_{\max}]$, the quality threshold $\tau \in [0,1]$ for promoting symbols to the alphabet and the compactness-vs-cardinality trade-off parameter $\rho \in [0,1]$.

$$[\mathbf{w} \quad \boldsymbol{\gamma} \quad Q \quad \tau \quad \rho] \tag{7}$$

The objective function J_1, to be minimized, is defined as follows:

$$J_1 = \alpha \cdot (1 - \pi) + (1 - \alpha) \cdot \frac{|\mathcal{A}|}{|\mathcal{S}_g^{(tr)}|} \tag{8}$$

Thus, Eq. (8) can be read as a linear convex combination between the performance π (leftmost term) and the penalty (rightmost term), with the latter aiming at fostering sparse alphabets according to a user-defined trade-off parameter $\alpha \in [0,1]$.

More in detail, each individual from the evolving population reads $\mathcal{S}_g^{(\text{tr})}$ (the set of subgraphs extracted from the training set), $\mathcal{S}^{(\text{tr})}$ (the training set), $\mathcal{S}^{(\text{vs})}$ (the validation set) and exploits Q, τ and ρ for driving the granulation procedure which, in turn, relies on the nBMF dissimilarity measure driven by $\mathbf{w}$ and $\boldsymbol{\gamma}$ (if applicable). As the alphabet $\mathcal{A}$ is returned by the granulation procedure (Section 2.2), the embedding takes place (Section 2.3). Therefore, the individual builds two instance matrices, namely $\mathbf{H}^{(\text{tr})}$ and $\mathbf{H}^{(\text{vs})}$, respectively an $|\mathcal{S}^{(\text{tr})}| \times n$ and $|\mathcal{S}^{(\text{vs})}| \times n$ matrix which see the symbolic histograms of the training set and validation set organized as rows. The performance $\pi \in [0,1]$ is chosen as the classification accuracy obtained by $\mathcal{H}$ trained with $\mathbf{H}^{(\text{tr})}$ in classifying $\mathbf{H}^{(\text{vs})}$.

Once the evolutionary strategy is completed, the optimal alphabet $\tilde{\mathcal{A}}$ is retained together with the optimal genetic code which leads to the synthesis of $\tilde{\mathbf{H}}^{(\text{tr})}$, $\tilde{\mathbf{H}}^{(\text{vs})}$, respectively the vector representation of $\mathcal{S}^{(\text{tr})}$ and $\mathcal{S}^{(\text{vs})}$ obtained with $\tilde{\mathcal{A}}$.

2.5 Feature Selection and Test Phase

It is not rare that after the alphabet optimization described in the previous section, the cardinality of the optimal alphabet set $\tilde{n} = |\tilde{\mathcal{A}}|$ can be very large, that is $\tilde{\mathcal{A}}$ may contains a large number of symbols and thus spanning vectors in high-dimensional spaces. There are several problems that can arise and negatively impact the performances of a classification system, including the curse of dimensionality (Tang et al., 2014), longer training and testing times, deteriorated model interpretability (Martino et al., 2020). For this reason, a feature selection phase should be applied to $\tilde{\mathcal{A}}$ in order to remove uninformative, redundant and in general not necessary symbols for the classification task. A wrapper approach based on an evolutionary algorithm has been designed for this purpose, where the genetic code of each individual is a binary mask $\mathbf{m} \in [0,1]^{\tilde{n}}$ which allows the selection of a subset of features. Hence, each individual:

1. reads $\tilde{\mathbf{H}}^{(\text{tr})}$ and $\tilde{\mathbf{H}}^{(\text{vs})}$
2. according to the 0's in $\mathbf{m}$, the corresponding columns of $\tilde{\mathbf{H}}^{(\text{tr})}$ and $\tilde{\mathbf{H}}^{(\text{vs})}$ are removed, leading to projected matrices $\tilde{\mathbf{H}}'^{(\text{tr})} \in \mathbb{R}^{|\mathcal{S}^{(\text{tr})}| \times n'}$ and $\tilde{\mathbf{H}}'^{(\text{vs})} \in \mathbb{R}^{|\mathcal{S}^{(\text{vs})}| \times n'}$ where $n' = \sum_{i=1}^{\tilde{n}} \mathbf{m}_i$
3. a classifier $\mathcal{H}$ is trained on $\tilde{\mathbf{H}}'^{(\text{tr})}$ and its own performance measure ω is computed as the misclassification rate on $\tilde{\mathbf{H}}'^{(\text{vs})}$

The objective function (to be minimized) is then defined as:

$$J_2 = (1-\lambda) \cdot \omega + \lambda \cdot \frac{n'}{\tilde{n}} \tag{9}$$

which, as in the case of Eq. (8), reads as a convex linear combination between the error rate on the validation set (leftmost term) and the ratio of selected symbols (rightmost term), weighted by a user-defined trade-off parameter $\lambda \in [0,1]$.

Once the optimization is completed, the optimal mask $\mathbf{m}^*$ is retained with $\mathbf{H}^{*(\text{tr})} \in \mathbb{R}^{|\mathcal{S}^{(\text{tr})}| \times n^*}$, $\mathbf{H}^{*(\text{vs})} \in \mathbb{R}^{|\mathcal{S}^{(\text{vs})}| \times n^*}$ as well, where $n^* = \sum_{i=1}^{\tilde{n}} \mathbf{m}_i^*$. Accordingly, the optimal alphabet $\mathcal{A}^* \subset \tilde{\mathcal{A}}$ is created by selecting the features indicated by the 1's in $\mathbf{m}^*$.

The reduced alphabet $\mathcal{A}^*$ is the main actor when it comes to address the generalization capabilities of the whole system with a set of test data $\mathcal{S}^{(\text{ts})}$. In fact, $\mathcal{S}^{(\text{ts})}$ is embedded thanks to $\mathcal{A}^*$, hence returning $\mathbf{H}^{*(\text{ts})} \in \mathbb{R}^{|\mathcal{S}^{(\text{ts})}| \times n^*}$, then the classifier $\mathcal{H}$ is trained on $\mathbf{H}^{*(\text{tr})}$ and finally tested on $\mathbf{H}^{*(\text{ts})}$.

3 RELAXED SYMBOLIC HISTOGRAMS

From Section 2.3, let us recall the original symbolic histogram. In short, it aims at representing the graph to be embedded as a vector collecting (in the i^{th} position) the number of occurrences of the i^{th} symbol from the alphabet $\mathcal{A}$ within the graph to be embedded G, with the latter being properly decomposed (G_{exp}) to facilitate the search-and-matching procedure. It is clear that the original symbolic histogram reads as an integer-valued vector, where the proper symbol-to-subgraphs dissimilarity values (i.e., $d(s_i, g)$ in Eq. (6)) are not taken into account. Inspired by dissimilarity spaces (Section 1), where the dissimilarity value amongst data is the core of the embedding procedure, here below we present six different strategies in order to populate the symbolic histogram, while at the same time taking into account the dissimilarities between symbols in the alphabet and the constituent parts of the graph to be embedded.

3.1 Sum

The sum strategy aims at collecting, in the i^{th} position of the symbolic histogram, the sum of distances between the i^{th} symbol in the alphabet and the constituent parts of the graph to be embedded. Formally,

$$\mathbf{h}_G^{\mathcal{A}} = [\text{sum}(s_1, G_{exp}), \ldots, \text{sum}(s_n, G_{exp})] \tag{10}$$

where the sum : $\mathcal{A} \times \mathcal{G} \to \mathbb{R}$ operator reads as

$$\text{sum}(s_i, G_{exp}) = \sum_{g \in G_{exp}} d(s_i, g) \tag{11}$$

3.2 Mean

The sum strategy is characterized by a couple of caveats: a) dissimilarity values are summed up regardless of their magnitude, b) the number of dissimilarities that are summed up is not taken into account. Especially in light of the second caveat, the mean strategy accounts for the mean of distances between the i^{th} symbol in the alphabet and the constituent parts of the graph to be embedded. Formally,

$$\mathbf{h}_G^{\mathcal{A}} = [\text{mean}(s_1, G_{exp}), \ldots, \text{mean}(s_n, G_{exp})] \tag{12}$$

where the mean : $\mathcal{A} \times \mathcal{G} \to \mathbb{R}$ operator reads as

$$\text{mean}(s_i, G_{exp}) = \frac{1}{|G_{exp}|} \sum_{g \in G_{exp}} d(s_i, g) \tag{13}$$

It is worth noting that $|G_{exp}|$ varies on a graph-based fashion (i.e., each symbolic histogram has its own scaling factor as it depends on the graph G to be embedded) and it is not equivalent to a constant scaling of all symbolic histograms.

3.3 Median

It is well known that outliers might have a non-negligible impact on the mean of a set of scalar values. In our case, this reflects on very high or very low dissimilarities that might skew the mean value. In order to mitigate this effect, a more robust statistic based on the median value is considered. Formally,

$$\mathbf{h}_G^{\mathcal{A}} = [\text{median}(s_1, G_{exp}), \ldots, \text{median}(s_n, G_{exp})] \tag{14}$$

where the median : $\mathcal{A} \times \mathcal{G} \to \mathbb{R}$ operator reads as

$$\text{median}(s_i, g) = \begin{cases} \mathbf{d}_{\frac{|G_{exp}|}{2}} & \text{if } |G_{exp}| \text{ is even} \\ \frac{\mathbf{d}_{\frac{|G_{exp}|-1}{2}} + \mathbf{d}_{\frac{|G_{exp}|+1}{2}}}{2} & \text{if } |G_{exp}| \text{ is odd} \end{cases} \tag{15}$$

and $\mathbf{d} \in \mathbb{R}^{1 \times |G_{exp}|}$ is a vector that collects the pairwise dissimilarities between s_i and all items in $g \in G_{exp}$, sorted in ascending order.

3.4 Thresholded-sum

The three strategies discussed so far in Sections 3.1–3.3 aim at aggregating, according to different operators, the pairwise symbol-to-subgraphs dissimilarities for populating a given entry of the symbolic histogram. Yet, as introduced in Section 3.2, all dissimilarities (regardless of their magnitude) are entitled to contribute to a given entry of the symbolic histogram vector. Taking inspiration from the original symbolic histogram (Section 2.3), we let dissimilarities contribute to a given operator if and only if their magnitude is below a given threshold.

As the sum operator is concerned, in Eq. (10), the $\text{sum}(\cdot,\cdot)$ operator, formerly Eq. (11), is replaced by the following thresholded sum (or, for short, t-sum : $\mathcal{A} \times \mathcal{G} \to \mathbb{R}$) operator

$$\text{t-sum}(s_i, G_{exp}) = \sum_{\substack{g \in G_{exp} \\ d(s_i,g) \leq \zeta_i}} d(s_i, g) \tag{16}$$

where, recall, ζ_i is a symbol-aware inclusion threshold.

3.5 Thresholded-mean

The thresholded mean follows the same rationale behind $\text{t-sum}(\cdot,\cdot)$: only dissimilarities below a given threshold are entitled to contribute to the mean value for populating the symbolic histogram entries. That is, the $\text{mean}(\cdot,\cdot)$ operator defined in Eq. (13) to be plugged into the symbolic histogram (see Eq. (12)) is replaced by the following thresholded mean (or, for short, t-mean : $\mathcal{A} \times \mathcal{G} \to \mathbb{R}$) operator

$$\text{t-mean}(s_i, G_{exp}) = \frac{\sum_{\substack{g \in G_{exp} \\ d(s_i,g) \leq \zeta_i}} d(s_i, g)}{|\{g : d(s_i, g) \leq \zeta_i, \forall g \in G_{exp}\}|} \tag{17}$$

3.6 Thresholded-median

Finally, the thresholded median (or, for short, t-median : $\mathcal{A} \times \mathcal{G} \to \mathbb{R}$) reads as the $\text{median}(\cdot,\cdot)$ operator in Eq. (15). The major difference is that the (sorted) vector $\mathbf{d}$ will only contain dissimilarities below the symbol-related thresholds ζ_i.

4 TESTS AND RESULTS

4.1 Datasets Description

In order to test the proposed algorithm and the different embedding strategies, the following six open-access datasets from the IAM Repository (Riesen and Bunke, 2008) are considered:

AIDS: The AIDS dataset is composed by graphs representing molecular compounds showing activity or not against HIV. Molecules are converted into graphs by representing atoms as nodes and the covalent bonds as edges. Nodes are labeled with the number of the corresponding chemical symbol and edges by the valence of the linkage.

GREC: The GREC dataset consists of graphs representing symbols from architectural and electronic drawings. Drawings have been pre-processed and ending points, circles, corners and intersections are considered as nodes labelled with their position and type. Edges connecting such nodes are equipped with a hierarchical data structure that identifies the type of connection and its characteristics.

Letter: The Letter datasets involve graphs that represent distorted letter drawings. Amongst the capital letters of the Roman alphabet, 15 have been selected due to them being representable by straight lines only. Drawings are then converted into graphs by considering lines as edges and ending points as nodes. Nodes are labelled with a 2-dimensional vector indicating their position, whereas edges are unlabelled. Three different amount of distortion (Low, Medium, High) account for three different datasets, hereinafter referred to as Letter-L, Letter-M and Letter-H, respectively.

Mutagenicity: The Mutagenicity dataset is composed by graphs representing molecular compounds showing mutagenicity properties or not. Similar to the AIDS dataset, nodes correspond to atoms (labelled with their chemical symbol) and edges are labelled with the valence of the linkage.

In Table 1 we show some basic statistics about the six datasets. As the nodes and edges dissimilarities are concerned, all of them are customized according to the nodes and edges attributes for each dataset. GREC is the only dataset for which the dissimilarity measures between nodes and edges are parametric themselves: such values populate $\boldsymbol{\gamma}$ which shall be optimized, as described in Section 2.4. Full details on the dissimilarity measures for AIDS, Letter and GREC can be found in (Martino and Rizzi, 2021) and (Baldini et al., 2021). For Mutagenicity, as instead, the dissimilarity measures between nodes and edges are plain non-parametric simple matching between their respective labels.

4.2 Comparison amongst Embedding Strategies

The algorithm parameters are set as follows:

- a simple random walk is employed as graph traversing strategy for mining subgraphs of maximum order $V = 5$ for the extraction phase (i.e., for populating $\mathcal{S}_g^{(\mathrm{tr})}$))[1]
- a modified BFS[2] is employed for extracting subgraphs for the embedding phase (i.e., for building G_{exp})
- $W = 10\%,\ 30\%,\ 50\%$ of $|\mathcal{S}_{\max}^{(\mathrm{tr})}|$, with the latter being the maximum number of subgraphs of maximum order $V = 5$ attainable from the training graphs[3] (i.e., an exhaustive extraction)
- $Q_{\max} = 500/c$, where c is the (dataset-dependent) number of classes
- maximum number of 20 generations for both differential evolution stages (alphabet optimization and feature selection)
- 20 individuals for the population of the first differential evolution (alphabet optimization)
- 100 individuals for the population of the second differential evolution (feature selection)
- $\alpha = 0.9$ in the fitness function J_1 of the first differential evolution (major weight to performance)
- $\lambda = 0.1$ in the fitness function J_2 of the second differential evolution (major weight to performance)
- K-Nearest Neighbours with $K = 5$ as classification system $\mathcal{H}$
- $\zeta_i = 1.1 \cdot \Psi(\mathbf{C}_i)$ as (cluster-dependent) tolerance value for the symbolic histograms evaluation.

In Figures 1, 2 and 3, we report the results obtained by equipping the GRALG classification system presented in Section 2 with the six different symbolic histograms-inspired embedding methods as described in Section 3. Each figure corresponds to either one of the three subsampling rates W in order to address the performances of the system as a function of the candidate number of subgraphs for the granulation phase. In order to account for the stochastic nature of the algorithm, results in the following have been averaged across 10 different runs. We explored the efficiency of our approach under three different point of views, i.e. the classifier performance measured on the test set $\mathcal{S}^{(\mathrm{ts})}$ and the cardinality of the alphabet before ($|\tilde{\mathcal{A}}|$)

[1] In our previous work (Baldini. et al., 2019), we witnessed that there is no clear winner between BFS or DFS in terms of performances and/or running times, so the rationale behind choosing random walks is that there exist the possibility of having both star-like and path-like subgraphs in $\mathcal{S}_g^{(\mathrm{tr})}$.

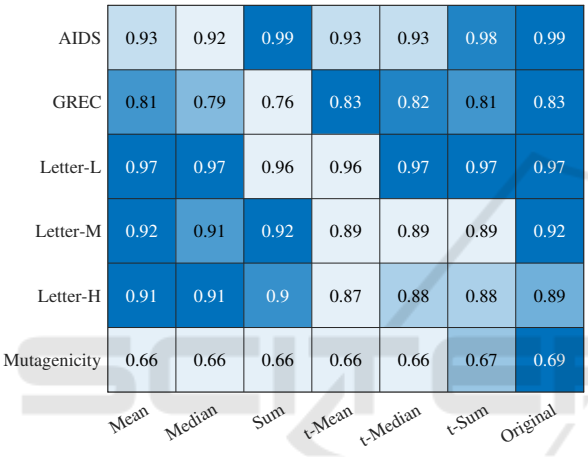
[2] As proposed in (Baldini. et al., 2019), we use BFS to extract subgraphs in such a way that already-visited nodes do not appear as root nodes for following traversals: this ensures a complete coverage of the graph to be embedded whilst, at the same time, keeping a low number of resulting subgraphs.

[3] AIDS: 27589; GREC: 21009; Letter-L: 7975; Letter-M: 8217; Letter-H: 12603; Mutagenicity: 449519.

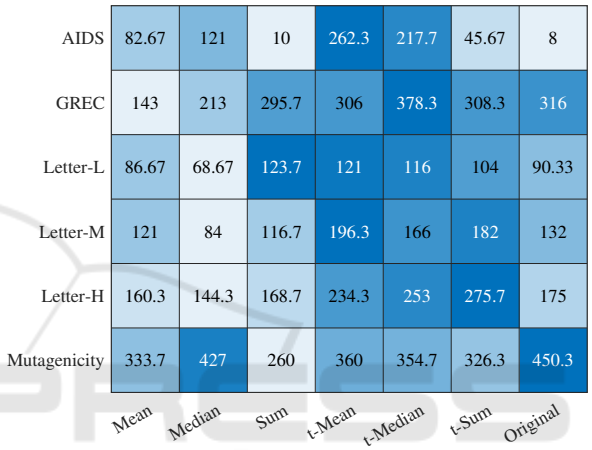
Table 1: Statistics of the considered datasets: number of graphs in training, validation and test set ($|\mathcal{S}^{(\mathrm{tr})}|$, $|\mathcal{S}^{(\mathrm{vs})}|$, $|\mathcal{S}^{(\mathrm{ts})}|$), number of classes ($c$) and type of nodes and edges labels. Adapted from (Riesen and Bunke, 2008).

Dataset	$\lvert\mathcal{S}^{(\mathrm{tr})}\rvert$	$\lvert\mathcal{S}^{(\mathrm{vs})}\rvert$	$\lvert\mathcal{S}^{(\mathrm{ts})}\rvert$	c	Node Labels	Edge Labels
AIDS	250	250	1500	2	string + integer + $\mathbb{R}^2$	integer†
GREC	286	286	528	22	string + $\mathbb{R}^2$	tuple
Letter-L	750	750	750	15	$\mathbb{R}^2$	none
Letter-M	750	750	750	15	$\mathbb{R}^2$	none
Letter-H	750	750	750	15	$\mathbb{R}^2$	none
Mutagenicity	1500	500	2337	2	string	integer

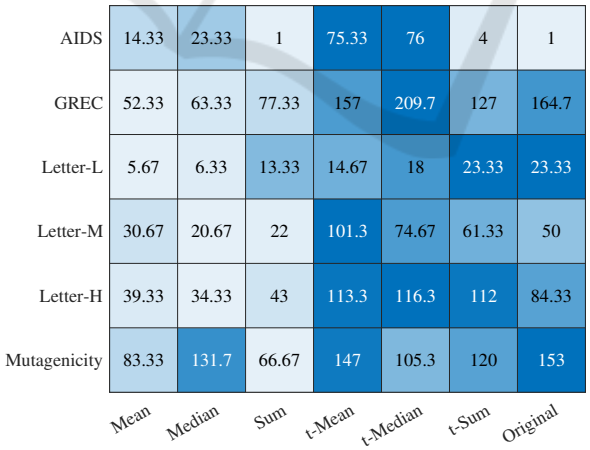
† In our experiments, by following (Bianchi et al., 2014), the edge label has been discarded.



(a) Accuracy on the Test Set.



(b) Embedding space dimensionality before Feature Selection.

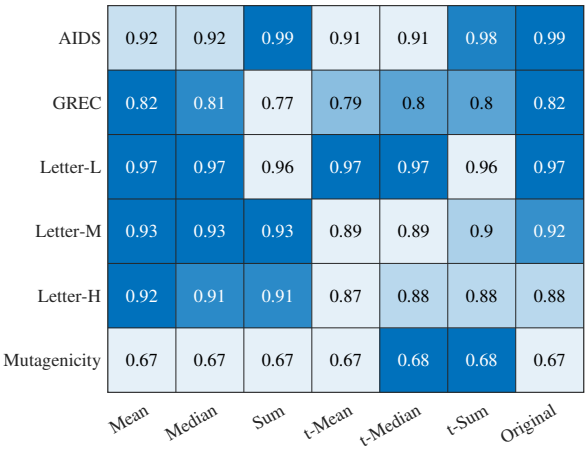


(c) Embedding space dimensionality after Feature Selection.

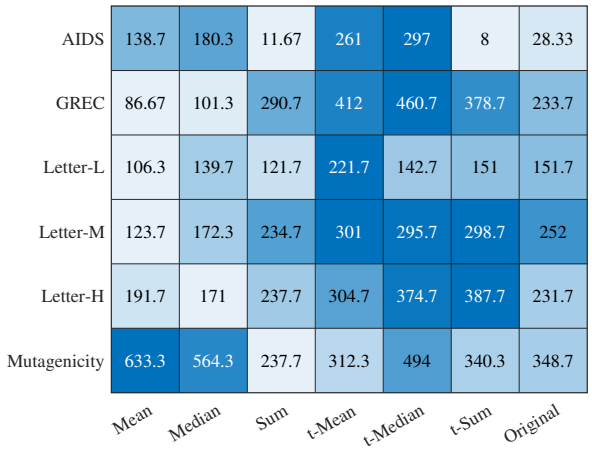
Figure 1: Results at 10% subsampling rate. Color maps are normalized row-wise (i.e., independently for each dataset), with a white-to-blue range mapping smallest-to-largest values.

and after ($|\mathcal{A}^*|$) the feature selection phase (see Section 2.5). It is worth noting that the performances on the test set are obtained by the classifier $\mathcal{H}$ trained with $\mathbf{H}^{*(\mathrm{tr})}$. Recall from Section 2.5 that the vectorial representation $\mathbf{H}^{*(\mathrm{ts})}$ of $\mathcal{S}^{(\mathrm{ts})}$ is obtained thanks to the selected embedding procedure using the optimized alphabet $\mathcal{A}^*$.

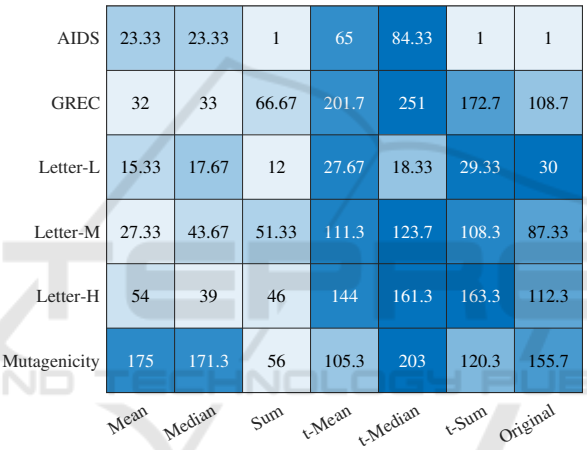
By comparing Figures 1a, 2a and 3a it is pos-



(a) Accuracy on the Test Set.



(b) Embedding space dimensionality before Feature Selection.

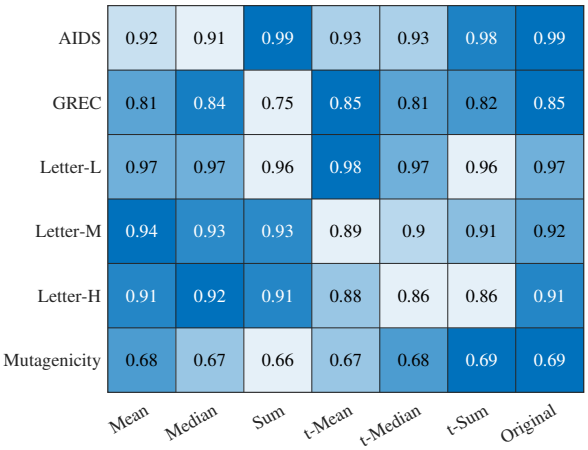


(c) Embedding space dimensionality after Feature Selection.

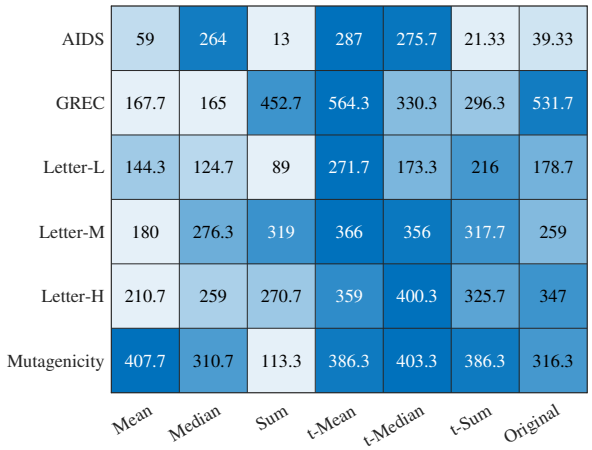
Figure 2: Results at 30% subsampling rate. Color map details in the caption of Figure 1.

sible to spot the differences in terms of accuracy. The results depicted in the first two columns (Mean and Medium) witnessed that the selected embedding methods are reaching comparable performances with respect to the Original symbolic histogram method equipped with the hard-limiting function regardless of the number of candidate information granules W. A note should be mentioned for AIDS, which is arguably attaining lower level of performances (6~7 %) when compared with the Original column. On the other hand, the remaining four approaches, i.e. Sum, t-Mean, t-Median and t-Sum, show (on average) worst performances when compared to Mean, Median and Original. We again can observe an exception for AIDS: when using the Sum aggregation operator, it shows the highest result in terms of accuracy.

An interesting behaviour that emerge from the tests regards the number of symbols that compose the alphabets $\mathcal{A}^*$ and $\tilde{\mathcal{A}}$. With respect to thresholded methods (t-Mean, t-Median, t-Sum and Original), Mean, Median and Sum are by far producing optimal alphabets with lower number of symbols, regardless of W. This can be spotted by observing Figures 1c, 2c, 3c and their counterparts before the feature selection phase (Figures 1b, 2b, 3b), suggesting that such reduction in terms of number of features is not due to the feature selection phase but is a proper characteristic of the aggregation function. With the exception of Mutagenicity, all datasets show a clear-cut difference in terms of alphabet cardinality when not-thresholded methods are exploited. We advance the hypothesis that by using hard-limiting functions in the symbolic histograms, the resulting dynamic of each vector component can be influenced by the choice of the threshold. In fact, the set of parameters (notably those related to the dissimilarity measure, i.e.



(a) Accuracy on the Test Set.



(b) Embedding space dimensionality before Feature Selection.

(c) Embedding space dimensionality after Feature Selection.

Figure 3: Results at 50% subsampling rate. Color map details in the caption of Figure 1.

w, **γ**) explored by the evolutionary algorithm might not be suitable enough for imposing an expressive dissimilarity measure able to fairly compare the symbols with the substructure set G_{exp}. In this particular situation, most of the symbols would not be matched with substructures in G_{exp}, leading to an uninformative embedding space spanned by flat vectors possibly having many null components. On the other hand, the evolutionary algorithm in the attempt to optimize the error rate might be tempted to relieve this issue by exploring granulation parameters (i.e., Q, τ and ρ) which allow larger alphabets with higher chances of scoring matches between symbols and the substructures of the graphs to be embedded. Clearly, this situation does not hold in non-thresholded methods, where each match counts (albeit proportionally to its dissimilarity degree): in turn, this means that a given symbol from the alphabet is (in very plain terms) 'always found' in the graph to be embedded which is therefore completely explored during the embedding procedure.

4.3 Comparison against State-of-the-Art Techniques

In Section 4.2 we focused the computational results on the comparison amongst the six different strategies for populating the symbolic histograms presented in Section 3. Hereinafter, we move the comparison against other state-of-the-art techniques, summarized in Table 2. Competitors span a variety of approaches for graph classification (as briefly reviewed in Section 1), namely:

- classifiers working on the top of pure graph matching similarities (Riesen and Bunke, 2008; Riesen and Bunke, 2009a);

- kernel methods (Da San Martino et al., 2016; Martino and Rizzi, 2020);
- several embedding techniques (Riesen and Bunke, 2009b; Gibert et al., 2011), including GrC-based (Martino et al., 2019a; Baldini. et al., 2019; Martino and Rizzi, 2021; Baldini et al., 2021) and neural-based ones (Bacciu et al., 2018; Martineau et al., 2020).

The accuracy of the last eight methods are obtained from our own experiments. For the remaining competing algorithms, we directly quote results from their respective papers.

Clearly, GRALG also whether equipped with 'relaxed' symbolic histograms, is able to reach state-of-the-art performances for 4 out of 6 datasets while, at the same time, is able to return an interpretable model: the same peculiarity only holds in other GrC-based pattern recognition systems (namely (Martino et al., 2019a; Baldini. et al., 2019; Martino and Rizzi, 2021; Baldini et al., 2021)), whilst it is well-known that other learning paradigms, notably those based on artificial neural networks (Xu et al., 2019), lack any interpretation.

Specifically, for AIDS and Letter-H the performance shift with respect to the most performing technique is below 1%; for Letter-L and Letter-M the gap slightly increases to approximately 2%. Mutagenicity and GREC are the only two datasets for which the shift with respect to the most performing technique becomes larger: approximately 10% for the former and approximately 6% for the latter dataset, if we omit the results obtained via cross-validation.

Table 2: Comparison against state of the art graph classification system in terms of accuracy, expressed in percentage. Best accuracy amongst all subsampling values have been reported for methods based on GRALG. A dash (-) indicates that a given dataset has not been tested in the literature on the corresponding model.

Technique	AIDS	GREC	Letter-L	Letter-M	Letter-H	Mutagenicity	Reference
Bipartite Graph Matching + K-NN	-	86.3	91.1	77.6	61.6	-	(Riesen and Bunke, 2009a)
Lipschitz Embedding + SVM	98.3	96.8	99.3	95.9	92.5	74.9	(Riesen and Bunke, 2009b)
Graph Edit Distance + K-NN	97.3	95.5	99.6	94	90	71.5	(Riesen and Bunke, 2008)
Graph of Words + K-NN	-	97.5	98.8	-	-	-	(Gibert et al., 2011)
Graph of Words + kPCA + K-NN	-	97.1	97.6	-	-	-	(Gibert et al., 2011)
Graph of Words + ICA + K-NN	-	58.9	82.8	-	-	-	(Gibert et al., 2011)
ODD ST_+ kernel	82.06*	-	-	-	-	-	(Da San Martino et al., 2016)
ODD ST_+^{TANH} kernel	82.54*	-	-	-	-	-	(Da San Martino et al., 2016)
CGMM + linear SVM	84.16*	-	-	-	-	-	(Bacciu et al., 2018)
G-L-Perceptron	-	70	95	64	70	-	(Martineau et al., 2020)
G-M-Perceptron	-	75	98	87	81	-	(Martineau et al., 2020)
C-1NN	-	-	96	93	84	-	(Martineau et al., 2020)
C-M-1NN	-	-	98	81	71	-	(Martineau et al., 2020)
Hypergraph Embedding + SVM	99.3†	-	-	-	-	77.0†	(Martino et al., 2019a)
RECTIFIER + K-NN	99.07	95.57	97.12	92.16	91.60	-	(Martino and Rizzi, 2021)
Dual RECTIFIER + K-NN	99.13	96.61	96.40	93.04	91.31	-	(Martino and Rizzi, 2021)
HCK Hypergraph kernel + SVM	89.5*†	-	-	-	-	73.3†	(Martino and Rizzi, 2020)
WJK Hypergraph kernel + SVM	99.5*†	-	-	-	-	82†	(Martino and Rizzi, 2020)
EK Hypergraph kernel + SVM	99.5*†	-	-	-	-	75.4†	(Martino and Rizzi, 2020)
SEK Hypergraph kernel + SVM	99.6*†	-	-	-	-	75.3†	(Martino and Rizzi, 2020)
GRALG (simple random walk) original symbolic histogram	99.16	84.04	96.58	86.58	73.84	74.73	(Baldini. et al., 2019)
GRALG (stratified random walk) original symbolic histogram	99	85	97	92	91	69	This work also in (Baldini et al., 2021)
GRALG (stratified random walk) sum symbolic histogram	99	77	96	93	91	67	This work
GRALG (stratified random walk) mean symbolic histogram	93	82	97	94	92	68	This work
GRALG (stratified random walk) median symbolic histogram	92	84	97	93	92	67	This work
GRALG (stratified random walk) t-sum symbolic histogram	98	82	97	91	88	69	This work
GRALG (stratified random walk) t-mean symbolic histogram	93	85	98	89	88	67	This work
GRALG (stratified random walk) t-median symbolic histogram	93	82	97	90	88	68	This work

* Results refer to cross-validation rather than a separate test set.

† Only chemical symbol type (categorical) as node label. Edge labels are discarded.

5 CONCLUSIONS

In this paper, we proposed six 'relaxed' variants of symbolic histograms for graph classification purposes. Inspired by the dissimilarity space embedding, the 'relaxed' variants take into consideration the proper magnitude of the dissimilarities when matching the pivotal granules of information against the constituent parts of the graphs to be embedded. Three operators (mean, sum and median of distances) have been proposed to aggregate such dissimilarity values, with additional three variants which include a thresholding stage in order to account only for similarities that are 'close enough' with respect to the information granule under analysis.

In order to test these variants against the original symbolic histogram definition, we exploited a GrC-based framework for graph classification, namely GRALG, and plugged the different symbolic histogram variants within the embedding module.

Six open-access datasets of fully labelled graphs corroborate the effectiveness of the 'relaxed' variants. Especially when the thresholding stage is not employed, the resulting set of pivotal symbols is drastically reduced with respect to the thresholded variants (including the original symbolic histogram). In conclusion, the non-thresholded mean and median emerged as the most interesting operators in order to populate the (relaxed) symbolic histogram since they leaded to very small embedding spaces while, at the same time, maintaining interesting performances in terms of accuracy on the test set.

REFERENCES

Bacciu, D., Errica, F., and Micheli, A. (2018). Contextual graph markov model: A deep and generative approach to graph processing. In *35th International Conference on Machine Learning, ICML 2018*, volume 1, pages 495–504.

Baldini., L., Martino., A., and Rizzi., A. (2019). Stochastic information granules extraction for graph embedding and classification. In *Proceedings of the 11th International Joint Conference on Computational Intelligence - NCTA, (IJCCI 2019)*, pages 391–402. INSTICC, SciTePress.

Baldini, L., Martino, A., and Rizzi, A. (2021). Towards a class-aware information granulation for graph embedding and classification. In Merelo, J. J., Garibaldi, J., Linares-Barranco, A., Warwick, K., and Madani, K., editors, *Computational Intelligence: 11th International Joint Conference, IJCCI 2019 Vienna, Austria, September 17–19, 2019, Revised Selected Papers*. Springer International Publishing.

Bargiela, A. and Pedrycz, W. (2003). *Granular computing: an introduction*. Kluwer Academic Publishers, Boston.

Bianchi, F. M., Livi, L., Rizzi, A., and Sadeghian, A. (2014). A granular computing approach to the design of optimized graph classification systems. *Soft Computing*, 18(2):393–412.

Bunke, H. (2003). Graph-based tools for data mining and machine learning. In Perner, P. and Rosenfeld, A., editors, *Machine Learning and Data Mining in Pattern Recognition*, pages 7–19, Berlin, Heidelberg. Springer Berlin Heidelberg.

Bunke, H. and Allermann, G. (1983). Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, 1(4):245 – 253.

Bunke, H. and Jiang, X. (2000). Graph matching and similarity. In Teodorescu, H.-N., Mlynek, D., Kandel, A., and Zimmermann, H.-J., editors, *Intelligent Systems and Interfaces*, pages 281–304. Springer US, Boston, MA.

Bunke, H. and Riesen, K. (2008). Graph classification based on dissimilarity space embedding. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 996–1007. Springer.

Da San Martino, G., Navarin, N., and Sperduti, A. (2016). Ordered decompositional dag kernels enhancements. *Neurocomputing*, 192:92 – 103.

Del Vescovo, G., Livi, L., Frattale Mascioli, F. M., and Rizzi, A. (2014). On the problem of modeling structured data with the minsod representative. *International Journal of Computer Theory and Engineering*, 6(1):9.

Del Vescovo, G. and Rizzi, A. (2007a). Automatic classification of graphs by symbolic histograms. In *2007 IEEE International Conference on Granular Computing (GRC 2007)*, pages 410–410.

Del Vescovo, G. and Rizzi, A. (2007b). Online handwriting recognition by the symbolic histograms approach. In *2007 IEEE International Conference on Granular Computing (GRC 2007)*, pages 686–686. IEEE.

Duin, R. P. and Pękalska, E. (2012). The dissimilarity space: Bridging structural and statistical pattern recognition. *Pattern Recognition Letters*, 33(7):826–832.

Emmert-Streib, F., Dehmer, M., and Shi, Y. (2016). Fifty years of graph matching, network alignment and network comparison. *Information sciences*, 346:180–197.

Ghosh, S., Das, N., Gonçalves, T., Quaresma, P., and Kundu, M. (2018). The journey of graph kernels through two decades. *Computer Science Review*, 27:88–111.

Gibert, J., Valveny, E., and Bunke, H. (2011). Dimensionality reduction for graph of words embedding. In Jiang, X., Ferrer, M., and Torsello, A., editors, *Graph-Based Representations in Pattern Recognition*, pages 22–31. Springer Berlin Heidelberg, Berlin, Heidelberg.

Kriege, N. M., Johansson, F. D., and Morris, C. (2020). A survey on graph kernels. *Applied Network Science*, 5(1):6.

Martineau, M., Raveaux, R., Conte, D., and Venturini, G. (2020). Learning error-correcting graph matching with a multiclass neural network. *Pattern Recognition Letters*, 134:68 – 76.

Martino, A., Frattale Mascioli, F. M., and Rizzi, A. (2020). On the optimization of embedding spaces via information granulation for pattern recognition. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Martino, A., Giuliani, A., and Rizzi, A. (2019a). (hyper)graph embedding and classification via simplicial complexes. *Algorithms*, 12(11).

Martino, A. and Rizzi, A. (2020). (hyper)graph kernels over simplicial complexes. *Entropy*, 22(10).

Martino, A. and Rizzi, A. (2021). An enhanced filtering-based information granulation procedure for graph embedding and classification. *IEEE Access*, 9:15426–15440.

Martino, A., Rizzi, A., and Frattale Mascioli, F. M. (2019b). Efficient approaches for solving the large-scale k-medoids problem: Towards structured data. In Sabourin, C., Merelo, J. J., Madani, K., and Warwick, K., editors, *Computational Intelligence: 9th International Joint Conference, IJCCI 2017 Funchal-Madeira, Portugal, November 1-3, 2017 Revised Selected Papers*, pages 199–219. Springer International Publishing, Cham.

Pedrycz, W. (2001). Granular computing: an introduction. In *Proceedings Joint 9th IFSA World Congress and 20th NAFIPS International Conference*, volume 3, pages 1349–1354. IEEE.

Pedrycz, W. and Homenda, W. (2013). Building the fundamentals of granular computing: a principle of justifiable granularity. *Applied Soft Computing*, 13(10):4209–4218.

Pedrycz, W., Succi, G., Sillitti, A., and Iljazi, J. (2015). Data description: A general framework of information granules. *Knowledge-Based Systems*, 80:98–108.

Pękalska, E. and Duin, R. P. (2005). *The dissimilarity representation for pattern recognition: foundations and applications*. World Scientific.

Pękalska, E., Duin, R. P., and Paclík, P. (2006). Prototype selection for dissimilarity-based classifiers. *Pattern Recognition*, 39(2):189–208.

Riesen, K. and Bunke, H. (2008). Iam graph database repository for graph based pattern recognition and machine learning. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 287–297. Springer.

Riesen, K. and Bunke, H. (2009a). Approximate graph edit distance computation by means of bipartite graph matching. *Image and Vision Computing*, 27(7):950 – 959.

Riesen, K. and Bunke, H. (2009b). Graph classification by means of lipschitz embedding. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(6):1472–1483.

Riesen, K., Jiang, X., and Bunke, H. (2010). Exact and inexact graph matching: Methodology and applications. In *Managing and Mining Graph Data*, pages 217–247. Springer.

Storn, R. and Price, K. (1997). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359.

Tang, J., Alelyani, S., and Liu, H. (2014). Feature selection for classification: A review. In *Data Classification*, pages 37–64. CRC Press.

Theodoridis, S. and Koutroumbas, K. (2008). *Pattern Recognition*. Academic Press, 4 edition.

Wang, X., Pedrycz, W., Gacek, A., and Liu, X. (2016). From numeric data to information granules: A design through clustering and the principle of justifiable granularity. *Knowledge-Based Systems*, 101:100–113.

Xu, F., Uszkoreit, H., Du, Y., Fan, W., Zhao, D., and Zhu, J. (2019). Explainable ai: A brief survey on history, research areas, approaches and challenges. In Tang, J., Kan, M.-Y., Zhao, D., Li, S., and Zan, H., editors, *Natural Language Processing and Chinese Computing*, pages 563–574, Cham. Springer International Publishing.

Zadeh, L. A. (1997). Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. *Fuzzy Sets and Systems*, 90(2):111–127.

APPENDIX

In Section 4.3, we stressed one of the most intriguing aspects of GrC-based pattern recognition systems: the model interpretability. In fact, the resulting set of information granules that populate the alphabet $\mathcal{A}$ is automatically returned by the system during its synthesis, without any intervention by the end-user. Furthermore, it is worth recalling that the alphabet $\mathcal{A}$ contains the set of pivotal granules of information on the top of which the embedding is performed. In plain terms, each information granule 'behaves' as a feature in the embedding space since its recurrences within the graphs to be embedded are the core of the embedding procedure. If, in the so-synthesized embedding space, a given classifier is able to discriminate the embedded graphs, this inevitably suggests that the features that describe the patterns are indeed informative, and so are the underlying information granules.

At this point, one might wonder whether these information granules are meaningful for the problem at hand and validate a-posteriori the optimal alphabet $\mathcal{A}^*$, possibly with the help of field-experts (depending on the application field of the problem). To this end, we selected the best run (amongst the 10)

for the dataset Letter-L. In particular, amongst the results presented in Section 4.2, we selected the 10% subsampling rate with the Mean operator for populating the 'relaxed' symbolic histogram. The rationale behind this choice is three-fold:

1. Letter-L is a dataset composed by capital Roman letter drawings and originally conceived for handwriting recognition tasks; conversely to datasets such as AIDS and Mutagenicity (that pertain to the world of biology) and GREC (that pertains to the world of electronics), the three Letter datasets are more suitable for a broader audience, since no specific background is needed to understand the data and the application under analysis;
2. from Figure 1a, it is possible to observe that the average accuracy is approximately 97%: as for the above discussion, this suggests that the resulting symbols are indeed informative;
3. from Figure 1c, it is possible to see that the resulting symbols is fairly low (approximately 5–6 symbols) and this makes the validation of the symbols less tedious and more comfortable to display.

In Figure 4 we show the 6 resulting symbols in $\mathcal{A}^*$. Let us recall from Section 4.1 that the Letter datasets have unlabelled edges and the nodes are labelled with a 2D vector of $\langle x, y \rangle$ coordinates. For the sake of visualization, all $\langle x, y \rangle$ coordinates are scaled in the unitary hypercube.

Certainly the most unexpected subgraph is depicted in Figure 4f: a single node in the top-right portion of the $\langle x, y \rangle$ plane. Despite it appears trivial, an end-point in the top-right portion is a feature which is common in several capital Roman letters: amongst the 15 letters (classes) it worth mentioning `E`, `F`, `H`, `K`, `M`, `N`, `T` and `Z`.

Figure 4e shows another typical portion of many capital Roman letters: a horizontal top line. This feature is less common with respect to the single node (Figure 4f), yet it is characteristic of letters such as `E`, `F`, `T` and `Z`.

Figures 4a–4d show a series of vertical or slightly oblique lines, where the striking difference is in their position along the x-axis. Many different capital Roman letters can be drawn as a combination of these 'basic' traits: `M`, for example, is a combination of 4 vertical or slightly oblique lines (left to right: vertical, oblique, oblique, vertical) and the same reasoning holds for letters such as `N`, `V`, `X` and `W`.

However, it should be noted that the claim of this a-posteriori validation is *not* that every letter can be drawn by assembling the six symbols as depicted in Figure 4 as-they-are. In fact, we recall that the similarity between graphs follows an *inexact approach*: this means that these symbols are likely to be stretched or somehow moved across the $\langle x, y \rangle$ plane to faithfully 'match' the drawing of capital Roman letters (recall that the dissimilarity between nodes follows the Euclidean distance between their coordinates). If the 'as-they-are conjecture' was true, then it would have been impossible to recognize letters such as `E` (which is one of the 15 letters to be classified)

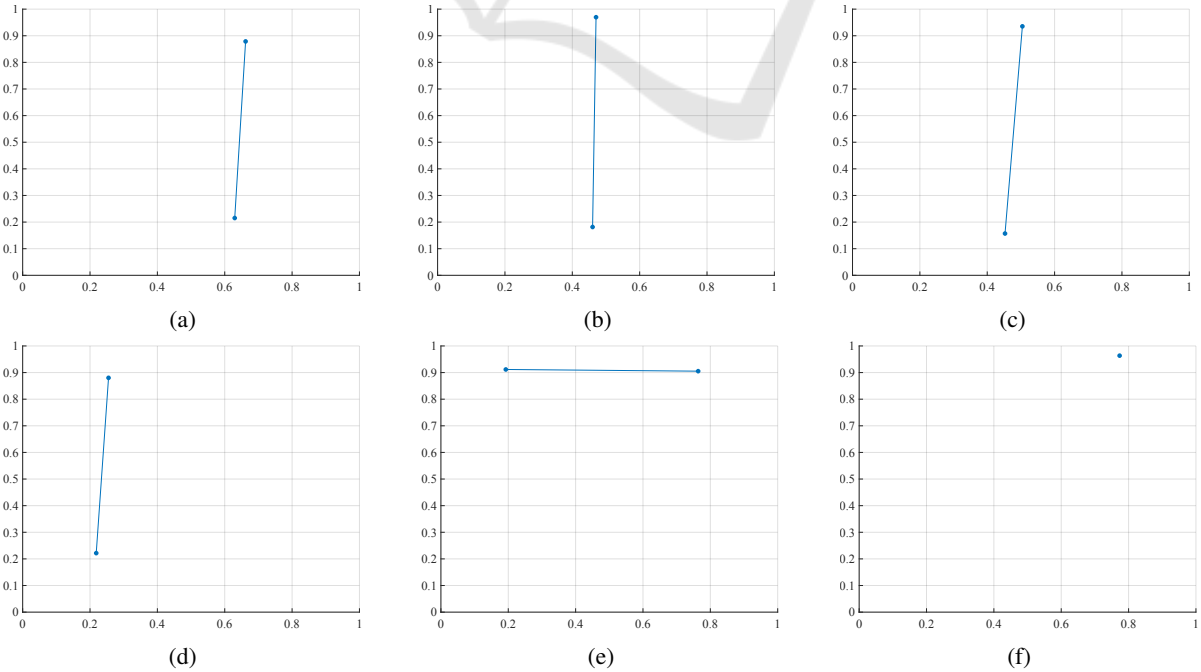


Figure 4: Resulting symbols for Letter-L (10% subsampling rate, Mean operator).

due to the absence of vertical lines positioned in the middle and the bottom of the $\langle x, y \rangle$-plane in the set of symbols. However, there is an horizontal line (Figure 4e) that can easily be used to represent the three horizontal lines in the letter E, counting three distinct matches with different similarity degrees.