The Geodesic Classification Problem on Graphs

Paulo Henrique Macêdo de Araújo, Manoel Campêlo, Ricardo Correa, Martine Labbé

Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 346 (2019) 65–76

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

The Geodesic Classification Problem on Graphs

Paulo Henrique Macêdo de Araújo [1,3]

Departamento de Computação and Campus de Quixadá
Universidade Federal do Ceará
Fortaleza and Quixadá, CE, Brazil

Manoel Campêlo[2,4]

Departamento de Matemática e Estatística Aplicada
Universidade Federal do Ceará
Fortaleza, CE, Brazil

Ricardo C. Corrêa[2,5]

Departamento de Ciência da Computação
Universidade Federal Rural do Rio de Janeiro
Nova Iguaçu, RJ, Brazil

Martine Labbé [6]

Departement D'Informatique
Université Libre de Bruxelles
Bruxelles, Belgium

Abstract

Motivated by the significant advances in integer optimization in the past decade, Bertsimas and Shioda developed an integer optimization method to the classical statistical problem of classification in a multi-dimensional space, delivering a software package called *CRIO* (*Classification and Regression via Integer Optimization*). Following those ideas, we define a new classification problem, exploring its combinatorial aspects. That problem is defined on graphs using the geodesic convexity as an analogy of the Euclidean convexity in the multidimensional space. We denote such a problem by *Geodesic Classification* (*GC*) problem. We propose an integer programming formulation for the *GC* problem along with a branch-and-cut algorithm to solve it. Finally, we show computational experiments in order to evaluate the combinatorial optimization efficiency and classification accuracy of the proposed approach.

Keywords: Classification, Geodesic Convexity, Integer Linear Programming.

https://doi.org/10.1016/j.entcs.2019.08.007

1 Introduction

Supervised learning denotes the automatic prediction of the behavior of unknown data based on a set of samples. It is a tool widely used in many everyday situations of the information society in which we live. In general terms, it can be described as the following two-phase procedure: in the initial phase, or *training phase*, the sample set is analyzed. Each sample consists of an array of encoded attributes that characterize an object of a certain type together with a label that associates a class to the corresponding object. Most commonly, only two classes are considered. A tacit assumption made at this phase is that there is an underlying pattern associated with the samples of each class that sets them apart from the samples of the other classes. Thus, the purpose of the training phase is to determine a mapping from all possible objects into the set of possible classes as an extension of an underlying patterns of the samples. Then, in the second phase, the mapping determined in the training phase is used to respond to queries for the class of objects that do not belong to the sample set.

An optimization problem is usually associated with the training phase. Referred to as *classification problem*, it consists in grouping similar samples so as to get clusters as internally homogeneous as possible. A wide range of solution methods is available, each depending on the coding of the samples and the criterion adopted to express homogeneity. A very popular approach is to encode the samples as vectors in an Euclidean space and to assume that the class patterns can be appropriately characterized by convex sets. In this vein, continuous optimization methods, including linear and quadratic programming, have been developed in the last 40 years [1,6,8,9,14]. More recently, integer linear programming tools started to be used in conjunction with continuous methods [3,13,16,17,18].

Strongly inspired by the version of the classification problem based on Euclidean convexity concepts discussed in [7], the object of study of this paper is the use of integer linear programming formulations for the resolution of a new variation of the classification problem stated in terms of notions of convexity in graphs. The statement of the new classification problem assumes the following hypotheses:

(i) The objects are not encoded numerically. Instead, each object is characterized by its similarities with other objects. The configuration of the objects is thus represented by a similarity graph $G = (V, E)$, connected, where V is the set of all objects and E gives the pairs of similar objects. The objects associated with the sample set constitute a proper subset of V. In addition, it is assumed that there is an underlying samples pattern that can be expressed, or at least

[1] Partially supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

[2] Partially supported by the Brazilian agencies CAPES, CNPq (Proc. 305264/2016-8, Proc. 443747/2014-8) and FUNCAP (PNE-0112-00061.01.00/16).

[3] Email: phmacedoaraujo@lia.ufc.br

[4] Email: mcampelo@lia.ufc.br

[5] Email: correa@ufrrj.br

[6] Email: mlabbe@ulb.ac.be

approximated, by the notion of *geodesic convexity* in graphs, defined with respect to the shortest paths in G (analogously to the definition of Euclidean convexity with respect to the Euclidean distances between points) [15].

(ii) The sample set may contain an arbitrary number of misclassified objects, called *outliers*, resulting from possible sampling errors or due to inherent characteristics of the phenomenon being modeled. From the mathematical point of view, an outlier is that object that causes the underlying pattern of the samples in its class to deviate from the convexity definition. The possible occurrence of outliers poses an additional challenge to any method used to solve the classification problem since they need to be detected and disregarded of accurate solutions.

Considering the hypotheses above, the *Geodesic Classification (GC)* problem tackled in this paper becomes purely combinatorial. However, as detailed later, a solution is neither a covering nor a partitioning of G in convex sets in the sense studied in [2,5]. Thus, the study of the GC problem is of theoretical interest since it refers to the possibility of establishing new problems on graphs which may bring subsidies to resolution methods, including those based on Euclidean convexity. Another motivation with a more practical bias is to allow to encode objects similarities through some binary relation over the sample set. Many applications such as text mining, communities detection in social networks, historic files similarity prediction, recommendation systems, and spam filtering are potential beneficiaries of this approach [10].

To the best of our knowledge, the GC problem was not yet studied in the literature. The main purpose of this work is to propose and study algorithmic methods to solve it. To this end, we state an integer programming formulation and derive a family of facet-defining inequalities. A branch and cut method is then presented. Some computational experiments are described, showing that the solution method proposed is efficient for small and medium size instances. A particular characteristic of the implementation performed was that a lazy constraints scheme and cutting plane algorithm were fundamental to reduce its running time. The accuracy of the geodesic convexity approach is validated by comparing the prediction provided by the algorithm proposed with the one obtained for similar instances with *SVM* and *MLP* (these are two of the most used approaches for the Euclidean convexity classification problem).

This text is organized as follows. After presenting some basic concepts and notation on graphs in Section 2, we formalize the Geodesic Classification Problem. An integer linear formulation is presented in Section 3 together with a family of facet-defining inequalities. In Section 4, we present the developed algorithms and computational experiments. Finally, in Section 5, we give some concluding remarks and future work directions.

2 Problem Statement

Let V be a finite set of objects that are related through a binary relation described by the connected undirected graph $G = (V, E)$, $|V| = n$. Two distinct objects are *similar* if the corresponding vertices are adjacent in G (analogously, two vertices $v, w \in V$, $v \neq w$, are called *similar* if $vw \in E$). We also refer to G as *similarity graph*, and we adopt the standard terminology used in Bondy and Murty's book [4]. In particular, a *path between* $v, w \in V$ is the sequence $\langle v \rangle$, if $v = w$, or a sequence of distinct vertices $P = \langle v = v_1, v_2, \ldots, v_\ell = w \rangle$ such that $v_i v_{i+1} \in E$ for $i = 1, \ldots, \ell - 1$. The *length* of a path is given by its number of edges, *i.e.* the length of $\langle v \rangle$ is zero and that of P is $\ell - 1$. A *geodesic between* $v, w \in V$ is a shortest path between v and w and its length is denoted by $\delta(v, w)$. The closed interval $I[v, w]$ is the set of all vertices lying on a geodesic between v and w. More generally, given $S \subseteq V$, $I[S] = \bigcup_{u,v \in S} I[u, v]$. In this case, if $I[S] = S$, then S is a *convex set*. The *convex hull* of S, denoted by $H[S]$, is the smallest convex set containing S. We also define $D_{vw} = I[v, w] \setminus \{v, w\}$.

For the definition of the 2-class classification problem considered in this paper, we are given two nonempty subsets $V_B, V_R \subseteq V$, $V_B \cap V_R = \emptyset$, so that $V_{BR} = V_B \cup V_R$ represents the sample set and $V_N = V \setminus V_{BR}$. The sets V_B and V_R define the *blue class* and *red class* vertices, respectively. The remaining vertices are called *unclassified vertices*. An example of a similarity graph and its sample set is depicted in Fig. 1. In analogy to the Euclidean version of the problem, we define that G is *linearly separable with respect to* $A_B \subseteq V_B$ *and* $A_R \subseteq V_R$ if

(i) $H[A_B] \cap A_R = \emptyset$,

(ii) $H[A_R] \cap A_B = \emptyset$, and

(iii) $H[A_B] \cap H[A_R] \cap V_N = \emptyset$,

and *linearly inseparable* otherwise. If G is linearly inseparable with respect to some sample set, it can be turned linearly separable if some vertices are disregarded (i.e., considered as outliers). Such a situation occurs in the example of Fig. 1. Indeed, G is linearly inseparable because $v \in H[V_B] \cap H[V_R]$, but it becomes linearly separable if either v_1, v_2, v_3 or v_4 is considered as an outlier. The classification problem associated with G, V_B, and V_R asks for the smallest number of disregarded vertices, as follows.

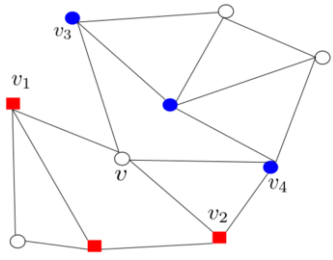


Fig. 1. Example of a similarity graph with blue class (as solid circles) and red class (as solid squares) vertices.

Definition 2.1 Given G, V_B, and V_R, the *Geodesic Classification (GC)* problem asks for groups $A_B \subseteq V_B$ and $A_R \subseteq V_R$ that maximizes $|A_B \cup A_R|$ and makes G linearly separable with respect to A_B and A_R.

It is worth noting that A_B and A_R define a mapping from V onto $\{blue, red\}$ classes which classifies all unclassified in $H[A_B]$ as blue class vertices and all unclassified vertices in $H[A_R]$ as red class vertices. In addition, this mapping arbitrarily classifies the unclassified vertices in $V \setminus (H[A_B] \cup H[A_R])$. Note that the vertices in $(H[A_B] \cap V_R) \cup (H[A_R] \cap V_B)$ are considered as outliers and are not reclassified (that is, moved to the other class). Due to the outliers and the vertices arbitrarily classified, each class does not necessarily define a convex set.

3 Integer Formulation for the GC Problem

To describe an integer formulation, let us define some notation. We denote by $\mathcal{T}_K(v)$ the set $\{S \subseteq V_K \mid v \in H[S] \setminus S\}$, for $K \in \{B, R\}$. By the definition of geodesic convexity, $|S| \geq 2$ for all $S \in \mathcal{T}_B(v) \cup \mathcal{T}_R(v)$.

3.1 Variables and valid inequalities

Two types of binary variables are defined. For every $v \in V_{BR}$, $a_v = 1$ means that vertex $v \in V_B$ ($v \in V_R$) belongs to group A_B (A_R), and $a_v = 0$ if v is disregarded. On the other hand, for every $v \in V_N$, p_v indicates the class of v as follows: if $v \in H[A_B]$ (resp. $v \in H[A_R]$), then $p_v = 0$ (resp. $p_v = 1$); otherwise, if v does not belong to the convex hulls of both groups, then p_v can be set arbitrarily. Thus, the formulation aims at maximizing $\sum_{v \in V_{BR}} a_v$ subject to

$$a_v + \sum_{w \in S} a_w \leq |S|, \qquad (v \in V_B, S \in \mathcal{T}_R(v)) \text{ or } (v \in V_R, S \in \mathcal{T}_B(v)) \tag{1}$$

$$p_v + \sum_{w \in S} a_w \leq |S|, \qquad v \in V_N, S \in \mathcal{T}_B(v) \tag{2}$$

$$-p_v + \sum_{w \in S} a_w \leq |S| - 1, \qquad v \in V_N, S \in \mathcal{T}_R(v) \tag{3}$$

$$a_v, p_u \in \{0, 1\}, \qquad v \in V_{BR}, u \in V_N \tag{4}$$

Constraint (1) is the analogous of the convex-inclusion inequality defined in [7] in the single-group case. It ensures that $v \in V_{BR}$ turns to be an outlier whenever it belongs to the convex hull of a set S of samples in the opposite class or at least one point in S must be an outlier. The *covering constraints* are defined by (2) and (3). If $v \in V_N \cap H[A_B]$ (resp. $v \in V_N \cap H[A_R]$), then p_v must get 0 (resp. 1). Hence, v cannot lie in the convex hull of both groups of opposite classes simultaneously.

Let $\mathcal{P}$ be the convex hull of the points satisfying (1)-(4). We denote by $\mathbf{e}_v$ the n-sized binary vector with 1 at the entry v and 0 otherwise. The entry refers to the value of the variable a_v, if $v \in V_{BR}$, or p_v, if $v \in V_N$. Since the n points $\{\mathbf{e}_v \mid v \in V\}$ clearly represent feasible solutions for $\mathcal{P}$ and are all linearly independent points, $\mathcal{P}$

is full-dimensional. The following rank-1 Chvátal-Gomory inequalities result from a configuration of geodesic convex combinations that cannot occur in a Euclidean space.

Proposition 3.1 *If $v, v' \in V_B$ and $w, w' \in V_R$ are distinct vertices such that $\{v, v'\} \subseteq D_{ww'}$ and $\{w, w'\} \subseteq D_{vv'}$, then*

$$a_v + a_{v'} + a_w + a_{w'} \leq 2 \tag{5}$$

is a facet-defining inequality for $\mathcal{P}$.

Proof. The validity of (5) stems from the fact that it is a $\{0, \frac{1}{3}\}$-Chvátal-Gomory cut with multiplier 1/3 for each constraint of type (1) defined for $(v, \{w, w'\})$, $(v', \{w, w'\})$, $(w, \{v, v'\})$, and $(w', \{v, v'\})$. To show that it defines a facet of $\mathcal{P}$, we consider feasible solutions satisfying (5) at equality. Each one of these solutions is associated with a vertex $u \in V$ and is represented by a n-sized binary vector $\mathbf{a}_u$. The n feasible solutions we consider are the following:

$$\mathbf{a}_u = \begin{cases} \mathbf{e}_v + \mathbf{e}_w, & \text{if } u = v \\ \mathbf{e}_v + \mathbf{e}_{v'}, & \text{if } u = v' \\ \mathbf{e}_{v'} + \mathbf{e}_w, & \text{if } u = w \\ \mathbf{e}_{v'} + \mathbf{e}_{w'}, & \text{if } u = w' \\ \mathbf{e}_v + \mathbf{e}_{v'} + \mathbf{e}_u, & \text{if } u \in V_B \setminus \{v, v'\} \\ \mathbf{e}_w + \mathbf{e}_{w'} + \mathbf{e}_u + \sum\limits_{z \in V_N \cap H[\{u,w,w'\}]} \mathbf{e}_z, & \text{if } u \in V_R \setminus \{w, w'\} \\ \mathbf{e}_v + \mathbf{e}_w + \mathbf{e}_u, & \text{if } u \in V_N \end{cases}$$

To show that they are affinely independent, let $\alpha_u \in \mathbb{R}$, for all $u \in V$, be multipliers such that $\sum_{u \in V} \alpha_u \mathbf{a}_u = 0$. First note that the variable a_u, for all $u \in V_{BR} \setminus \{v, v', w, w'\}$, is set to 1 only in $\mathbf{a}_u$. Thus, $\alpha_u = 0$ in these cases and, consequently, $\alpha_v \mathbf{a}_v + \alpha_{v'} \mathbf{a}_{v'} + \alpha_w \mathbf{a}_w + \alpha_{w'} \mathbf{a}_{w'} + \sum_{u \in V_N} \alpha_u \mathbf{a}_u = 0$. Among the solutions associated with the vertices in V_N, the variable p_u gets 1 only in $\mathbf{a}_u$. Hence, $\alpha_u = 0$ in these cases as well. For the remaining multipliers, we have

$$\begin{aligned} \alpha_v + \alpha_{v'} &= 0 \\ \alpha_{v'} + \alpha_w + \alpha_{w'} &= 0 \\ \alpha_v + \alpha_w &= 0 \\ \alpha_{w'} &= 0 \end{aligned}$$

which implies that $\alpha_v = \alpha_{v'} = \alpha_w = \alpha_{w'} = 0$. Therefore, all n solutions are affinely independent. □

3.2 Checking for Feasibility

Since the number of convex-inclusion and covering constraints may be very large, the following lazy constraint scheme is useful for checking feasibility of integer solutions.

It comprises two steps. First, we look for a pair (v, S) that violates a corresponding convex-inclusion or covering constraint. To this end, it is sufficient to explicitly compute the convex hull of both groups, A_B and A_R, of the solution (a, p) at hand, and to check whether there is:

$v \in V_N \cap H[A_B] \cap H[A_R]$**:** in this case, we define $S = A_B$ if $p_v = 1$ or $S = A_R$ if $p_v = 0$.

$v \in A_B \cap H[A_R]$**:** S is defined to be A_R.

$v \in A_R \cap H[A_B]$**:** S is defined to be A_B.

The second step consists of determining a $W \subseteq S$ that is minimal (*i.e.*, $v \notin H[W \backslash u]$, for any $u \in W$) and such that the corresponding convex-inclusion or covering constraint for (v, W) is still violated. Then, the violated constraint is added to the model as a lazy constraint. The minimal subset W can be determined with the following $O(n^4)$ algorithm. We start with $W = S$ and, as long as there exists a vertex $u \in W$ such that the convex hull of $W \backslash \{u\}$ still reaches v, then we remove u from W. This procedure finishes when W becomes minimal.

To calculate the convex hull S of A_B (or A_R), we start with $S' = A_B$ (or $S' = A_R$) and iteratively update it. At each iteration, we add to S' all vertices in D_{uv}, for every pair $u, v \in S'$ (with at least one of them added to S' in the previous iteration). This step is repeated until S' does not change. At this point, we get $S = S'$. Sets D_{uv} can be determined a priori with the BFS-like algorithm described in the next section.

4 Algorithms and Computational Experiments

In this section, we describe the method used to solve the problem and how it was implemented. We also present some results of computational experiments carried out with an Intel i7-7700 processor with 3.6 GHz, 8 cores and 32 GB RAM memory, running a 64 bits Linux OS. The algorithms were coded in C++, and `CPLEX` 12.8 was the optimization solver used. Essentially, we embedded separation procedures in the default branch-and-cut method provided by `CPLEX`.

4.1 Solution Method

The main steps of our solution method are the following:

(i) Computation of the shortest paths (Complexity $O(n(n+m))$): For each $v \in V$, we determine sets D_{vw}, for all $w \in V \setminus \{v\}$, by applying a breadth-first search algorithm from v. Recall that D_{vw} comprises all internal vertices in all shortest paths from v to w.

(ii) Initial cutoff: Since a trivial solution is obtained by taking all vertices of a class as outliers, $\min\{|V_B|, |V_R|\}$ is provided as a cutoff.

(iii) Initial model configuration: All inequalities (5) are included in the initial model by exhaustive enumeration of all sets computed in Step (i) (none of the constraints (1)-(3) are used initially).

(iv) Partial linear relaxation resolution: At the root node of the branch-and-cut tree, we solve the linear relaxation of the initial model together with constraints (1)-(3), for each pair (v, S) with $|S| = 2$, separated as cuts. The separation procedure is implemented as a Cut Callback macro of `CPLEX`.

(v) Exact model resolution: Starting from the model obtained in Step (iv), we add the integrality constraints (4) and solve the integer formulation by adding (1)-(3) as lazy constraints (with a Lazy Callback procedure).

4.2 Testbed

To test the developed algorithm, two sets of instances were used in the experiments, namely a set of random and a set of real world inspired instances. The random instances were categorized by number of vertices ($n \in \{50, 100, 150, 200, 250\}$), graph density percentage ($d \in \{5, 10, 20, 30, 50, 70\}$) and initially classified vertices percentage ($p \in \{20, 40, 60, 80\}$), where $n = |V|$, $d = 100|E|/(n(n-1)/2)$, and $p = 100|V_{BR}|/n$. For each combination (n, d, p), we generated 10 random instances. Therefore, there are 1200 random instances in total.

The real world inspired instances were derived from two datasets for the Euclidean version of the classification problem, namely Parkinson's disease [12] (195 points with 22 attributes) and cardiac Single Proton Emission Computed Tomography (SPECT) images [11] (267 points with 44 attributes), both available at `https://archive.ics.uci.edu/ml/datasets.html`. Each point in these datasets gives an information of a patient to be used to predict new diagnostics. We derived 10 instances for the GC problem for each instance in the datasets by constructing a graph $G = (V, E)$ and the sets V_B and V_R. The vertex set V represents the samples of the dataset and the edge set is defined using the transformation described in [19]. The vertices in V_B and V_R are randomly chosen such that 20% of the vertices are left unclassified in the instance generated. The predefined classes of the vertices chosen as unclassified are available in the dataset and were used to test the efficiency and accuracy of the algorithms.

4.3 Results for Random Instances

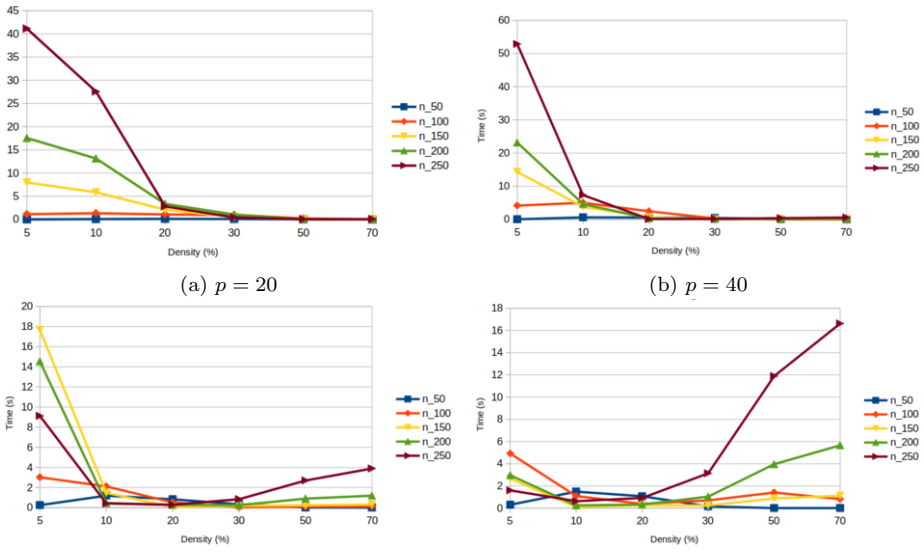
The experiments with random instances were used to analyze the combinatorial aspects of the problem, notably the effect of the constraints added to the model at steps (iii) and (iv). For the sake of comparison, we tested two other versions of the algorithm, each consisting of the elimination of Step (iii) or Step (iv). The results observed are summarized in Tab. 1, showing the performance of the three versions tested with respect to a standard implementation with no constraints added (*i.e.*., eliminating both steps (iii) and (iv)). We could note that the inequalities (5) were extremely effective: on average, there were $2.8|V|$ inequalities added and they reduced 81% of the running time. The constraints (1)-(3) added when solving the root node at Step (iv) were very effective as well. It was much better than setting all of them in the initial model. On average, there were $18|V|$ constraints added and they reduced 75% of the running time. However, using the separator for constraints

B&C Algorithm	N. of constraints (1)-(3)	N. of inequalities (5)	Time Reduction
Step (iii) only	0	$2.8\|V\|$	81%
Step (iv) only	$18\|V\|$	0	75%
Step (iii) and Step (iv)	$2.1\|V\|$	$2.8\|V\|$	84%

Table 1
Performance of the separation algorithms.

(1)-(3), for each pair (v, S) with $|S| = 3$ did not give better results. For $|S| > 3$, it showed to be not practical to use a separation algorithm for them.

Regarding the lazy constraints scheme presented in Section 3.2, its application at Step (v) was fundamental to reduce the running time (with respect to an implementation with all constraints (1)-(3) added to the initial model). Actually, it is impractical to solve the problem without the lazy constraints scheme since the number of constraints (1)-(3) is potentially exponential. Considering all random instances, the average running time of algorithm was about a few seconds, so it shows to be very good even for medium size instances. In general, random instances with density between 10% and 30% are harder to solve, although the worst running time was of 52 seconds for an instance with 250 vertices and 5% of density. Moreover, it seems that, for $p \leq 40$ and for sufficient large n, the running time decreases exponentially as the density increases. This is an evidence that the proposed algorithm works extremely well, especially for dense, medium sized instances.



(a) $p = 20$ (b) $p = 40$

(c) $p = 60$ (d) $p = 80$

Fig. 2. Average time versus density.

4.4 Realistic Instances

In order to evaluate the accuracy of our classification method (GC), we tested it with the Parkinson's disease and SPECT instances. Besides, we compared its per-

Instance	Diam	Dgmin	Dgmax	OPT	$T_{ILP1}(s)$	$Acu_{GC}(\%)$	$Acu_{SVM}(\%)$	$Acu_{MLP}(\%)$
parkinsons-n195-d5-p80-1	10	1	18	39	75.75	**79.49%**	**79.49%**	71.79%
parkinsons-n195-d5-p80-2	10	1	18	38	105.43	**74.36%**	**74.36%**	71.79%
parkinsons-n195-d5-p80-3	10	1	18	33	73.47	64.10%	64.10%	**74.36%**
parkinsons-n195-d5-p80-4	10	1	18	34	96.85	64.10%	64.10%	**76.92%**
parkinsons-n195-d5-p80-5	10	1	18	38	41.73	76.92%	76.92%	**82.05%**
parkinsons-n195-d5-p80-6	10	1	18	39	178.63	**76.92%**	**76.92%**	**76.92%**
parkinsons-n195-d5-p80-7	10	1	18	38	104.85	74.36%	**79.49%**	71.79%
parkinsons-n195-d5-p80-8	10	1	18	33	59.08	61.54%	61.54%	**69.23%**
parkinsons-n195-d5-p80-9	10	1	18	39	193.72	**79.49%**	23.08%	**79.49%**
parkinsons-n195-d5-p80-10	10	1	18	38	45.49	79.49%	**87.18%**	79.49%
AVERAGE	-	-	-	-	97.50	73.08%	68.72%	**75.38%**

Table 2
B&C Algorithm, *SVM* and *MLP* comparison for Parkinson's instances.

Instance	Diam	Dgmin	Dgmax	OPT	$T_{ILP1}(s)$	$Acu_{GC}(\%)$	$Acu_{SVM}(\%)$	$Acu_{MLP}(\%)$
spectf-n267-d5-p80-1	8	1	36	47	**0.22**	**86.79%**	67.92%	75.47%
spectf-n267-d5-p80-2	8	1	36	42	**0.16**	77.36%	**83.02%**	77.36%
spectf-n267-d5-p80-3	8	1	36	47	**0.33**	**86.79%**	71.70%	83.02%
spectf-n267-d5-p80-4	8	1	36	43	**0.16**	**79.25%**	75.47%	**79.25%**
spectf-n267-d5-p80-5	8	1	36	42	**0.18**	**77.36%**	49.06%	**77.36%**
spectf-n267-d5-p80-6	8	1	36	42	**0.17**	77.36%	**79.25%**	77.36%
spectf-n267-d5-p80-7	8	1	36	46	**0.35**	**83.02%**	81.13%	**83.02%**
spectf-n267-d5-p80-8	8	1	36	41	**0.17**	**75.47%**	73.58%	**75.47%**
spectf-n267-d5-p80-9	8	1	36	43	**0.18**	**77.36%**	75.47%	**77.36%**
spectf-n267-d5-p80-10	8	1	36	41	**0.16**	75.47%	79.25%	**86.79%**
AVERAGE	-	-	-	-	**0.21**	**79.62%**	73.59%	79.25%

Table 3
B&C Algorithm, *SVM* and *MLP* comparison for *SPECT* instances.

formance with those obtained by the classic *Support Vector Machine* (*SVM*) and *Multi-Layer Perceptron* (*MLP*) methods. These are approaches for the classification problem in the Euclidean space. We used the implementations of SVM and MLP available at http://scikit-learn.org/stable/modules.html. It is worth observing that SVM and MLP were applied to the original Parkinson's desease and SPECT instances (before the transformation to graph instances).

Tables 2 and 3 summarizes the computational results for for the Parkinson's disease and SPECT instances, respectively. The following information is displayed for each instance: (i) instance identification in the format <dataset>-n<n>-d<d>-p<p>-<sequential-number>, (ii) the diameter of the similarity graph (Diam), (iii) minimum and maximum degree of a vertex (Dgmin and Dgmax), (iv) minimum number of outliers (OPT), (v) running time of our B&C algorithm in seconds ($T_{GC}(s)$), (vi) percentage of the unclassified vertices/points correctly classified by GC ($Acu_{GC}(\%)$), SVM ($Acu_{SVM}(\%)$) and MLP ($Acu_{MLP}(\%)$).

A first remark concerns the high number of outliers in the optimal solution. Remember that we are using linear separation and these real world inspired instances are more unlikely to have such a property.

Regarding the Parkinson's disease instances, we can see that GC obtained the best accuracy in 4 of them, while 5 and 6 were the corresponding scores for SVM and MLP, respectively. However, on average, SVM got the worst accuracy, due to the poor performance in the ninth instance. GC and MLP obtained similar average accuracy with a slight advantage to the latter. For these instances, we could observe higher running times in comparison with the random instances. This was possibly due to the larger diameters and lower maximum degrees of the input graphs.

The results for the SPECT instances were similar to those for the random instances. We can observe that the running times of our B&C algorithm are much smaller than those for the Parkinson's instances. In the SPECT graphs, the diameters are lower and the maximum degree are higher. The GC approach presented the best accuracy in 7 SPECT instances, while SVM and MLP did it in 2 and 6 instances, respectively. On average, GC got also the best accuracy, similar to the one by MLP.

5 Concluding Remarks

In this work, we defined the Geodesic Classification (GC) problem as the analogous, on graphs, of the Euclidean Classification problem. In addition, we proposed an integer programming formulation for this new combinatorial optimization problem, as well as a family of facet-defining inequalities and a branch and cut method to solve it. An interesting point of these results is that the family of facet-defining inequalities is related to a structure of mutual convex combinations that are not possible in the Euclidean space. Results with some computational experiments show that the solution method proposed is very promising since the algorithm for the integer formulation proved to be very efficient, even for medium size instances. A particular characteristic of the implementation performed was that a lazy constraints scheme and cutting plane algorithm were fundamental to reduce its running time. We validated the accuracy of the geodesic convexity approach by comparing the prediction provided by the algorithm proposed with two of the most used approaches for the Euclidean convexity classification problem, namely, *SVM* and *MLP*. The results are very promising, since the prediction accuracy of the GC approach showed to be stable and as good as such classic linear separation algorithms for the multidimensional space. As future works, we intend to carry on a deeper polyhedral study in order to improve the performance of the algorithm.

References

[1] Arthanari, T. S. and Y. Dodge, "Mathematical Programming in Statistics," Wiley-Interscience, New York, 1993.

[2] Artigas, D., S. Dantas, M. C. Dourado and J. L. Szwarcfiter, *Partitioning a graph into convex sets*, Discrete Mathematics **311** (2011), pp. 1968–1977.

[3] Bertsimas, D. and R. Shioda, *Classification and regression via integer optimization*, Operations Research **55** (2007), pp. 252–271.

[4] Bondy, J. and U. Murty, "Graph Theory," Springer, 2008.

[5] Buzatu, R. and S. Cataranciuc, *Convex graph covers*, The Computer Science Journal of Moldova **23** (2015), pp. 251–269.

[6] Carrizosa, E. and D. R. Morales, *A mixed integer optimisation model for data classification*, Computers & Operations Research **40** (2013), pp. 150–165.

[7] Corrêa, R. C., D. D. Donne and J. Marenco, *On the combinatorics of the 2-class classification problem*, Discrete Optimization **31** (2019), pp. 40–55.

[8] Cortes, C. and V. Vapnik, *Support-vector networks*, in: *Machine Learning*, 1995, pp. 273–297.

[9] Freed, N. and F. Glover, *Evaluating alternative linear programming models to solve the two group discriminant problem*, Decision Sciences **17** (2007), pp. 151–162.

[10] Hong, L. and B. D. Davison, *Empirical study of topic modeling in twitter*, in: *Proceedings of the First Workshop on Social Media Analytics*, SOMA '10 (2010), pp. 80–88.

[11] Kurgan, L. A., K. J. Cios, R. Tadeusiewicz, M. Ogiela and L. S. Goodenday, *Knowledge discovery approach to automated cardiac spect diagnosis*, Artificial intelligence in medicine **23** (2001), pp. 149–169.

[12] Little, M. A., P. E. McSharry, S. J. Roberts, D. A. Costello and I. M. Moroz, *Exploiting nonlinear recurrence and fractal scaling properties for voice disorder detection*, BioMedical Engineering OnLine **6** (2007), p. 23.

[13] Maskooki, A., *Improving the efficiency of a mixed integer linear programming based approach for multi-class classification problem*, Comput. Ind. Eng. **66** (2013), pp. 383–388.

[14] Pardalos, P. M. and P. Hansen, **45**, American Mathematical Society, Providence, RI, 2008.

[15] Pelayo, I. M., "Geodesic Convexity in Graphs," Springer-Verlag, New York, 2013.

[16] Sun, M., *A mixed integer programming model for multiple-class discriminant analysis*, International Journal of Information Technology and Decision Making **10** (2011), pp. 589–612.

[17] Uney, F. and M. Turkay, *A mixed-integer programming approach to multi-class data classification problem*, European Journal of Operational Research **173** (2006), pp. 910–920.

[18] Xu, G. and L. G. Papageorgiou, *A mixed integer optimisation model for data classification*, Comput. Ind. Eng. **56** (2009), pp. 1205–1215.

[19] Zaki, M. J. and W. M. Jr, "Data Mining and Analysis: Fundamental Concepts and Algorithms," Cambridge University Press, New York, NY, USA, 2014.