Combining Dialectical Optimization and Gradient Descent Methods for Improving the Accuracy of Straight Line Segment Classifiers

Rosario A. Medina Rodriguez and Ronaldo Fumio Hashimoto
University of Sao Paulo
Institute of Mathematics and Statistics, IME-USP
Department of Computer Science
Rua do Matao, 1010 - Sao Paulo, Brazil
rosarior@ime.usp.br and ronaldo@ime.usp.br

***Abstract*—A recent published pattern recognition technique called Straight Line Segment (SLS) uses two sets of straight line segments to classify a set of points from two different classes and it is based on distances between these points and each set of straight line segments. It has been demonstrated that, using this technique, it is possible to generate classifiers which can reach high accuracy rates for supervised pattern classification. However, a critical issue in this technique is to find the optimal positions of the straight line segments given a training data set. This paper proposes a combining method of the dialectical optimization method (DOM) and the gradient descent technique for solving this optimization problem. The main advantage of DOM, such as any evolutionary algorithm, is the capability of escaping from local optimum by multi-point stochastic searching. On the other hand, the strength of gradient descent method is the ability of finding local optimum by pointing the direction that maximizes the objective function. Our hybrid method combines the main characteristics of these two methods. We have applied our combining approach to several data sets obtained from artificial distributions and UCI databases. These experiments show that the proposed algorithm in most cases has higher classification rates with respect to single gradient descent method and the combination of gradient descent with genetic algorithms.**

***Keywords*- straight line segments; gradient descent technique; dialectical optimization; genetic algorithms; pattern recognition.**

I. INTRODUCTION

In Image Processing and Computer Vision areas we frequently find many Pattern Recognition problems such as optical character recognition (OCR), handwritten recognition, human face image recognition, industrial inspection and medicine diagnoses [1], [2]. Recently a new technique for Pattern Recognition (focused on supervised binary classification),called as Straight Line Segment (SLS) classifier [3], [4], [5] was published. The key issue in this technique is to find optimal positions of the straight line segments given a training data set. An algorithm for finding these optimal positions is called *training algorithm*.

In this paper, we combine the traditional gradient descent method with a novel evolutionary algorithm [6], [7] called Dialectical Optimization Method (DOM), for solving this optimization problem.

On one hand, the main advantage of DOM is the capability of escaping from local optimum by multi-point stochastic searching. On the other hand, the strength of gradient descent method [8] is the ability of finding local optimum by pointing the direction that maximizes the objective function. Thus, based on these characteristics, in this paper we propose a technique that combines the gradient descent and dialectical optimization methods for solving optimization problem in the training algorithm of the SLS classifier.

In order to verify the viability of our algorithm, we have applied it to several data sets which were obtained from both artificial probability distributions (with known probability density functions) and UCI databases. Based on the results of these experiments, the proposed algorithm obtained in most of the cases higher classification rates when compared to: the solutions obtained from using just one single gradient descent method and the solutions obtained from the combination of gradient descent with genetic algorithms. As a conclusion, this hybrid technique improves the accuracy of the SLS classifiers in order to provide better solutions for supervised Pattern Recognition problems that appear in Image Processing or Computer Vision areas.

Following this brief introduction, Sections II and III briefly recall, respectively, the SLS classifier and the Dialectical Optimization Method. In Section IV, we present our hybrid approach. Section V shows the experimental results we have conducted. Finally, we give some conclusions and directions for future work in Section VI.

II. CLASSIFIER BASED ON STRAIGHT LINE SEGMENTS

A recent publication on Pattern Recognition presents a new technique based on straight line segments [3], [4], [5]. Its main contribution is to introduce a new type of classifier based on distances between a set of points and two sets of straight line segments. So far, this technique is focused on *binary* classification, so there is only two possible classes in the data

set. For the sake of completeness, we briefly describe in this section the SLS classifier.

A. Basic Definitions for SLS Classifiers

Let p and $q \in \mathbb{R}^{d+1}$. The straight line segment (SLS) with extremities p and q is defined as:

$$L_{p,q} = \{x \in \mathbb{R}^{d+1} : x = p + \lambda \cdot (q - p), 0 \leq \lambda \leq 1\} \quad (1)$$

Given a point $x \in \mathbb{R}^d$, let $x_e = (x, 0)$ denote the extension of x to $\mathbb{R}^{d+1}$.

Given a point $x \in \mathbb{R}^d$ and $L_{p,q} \subseteq \mathbb{R}^{d+1}$. The *pseudo-distance* between x and L is given by:

$$distP(x, L) = \frac{dist(x_e, p) + dist(x_e, q) - dist(p, q)}{2} \quad (2)$$

where $dist(a, b)$ denotes the Euclidean distance between two points $a, b \in \mathbb{R}^{d+1}$.

Let $\mathcal{L}$ denote a set of SLSs, defined as:

$$\mathcal{L} = \{L_{p_i,q_i} : p_i, q_i \in \mathbb{R}^{d+1}, i = 1, ..., m\} \quad (3)$$

where m represents the number of SLSs for each class.

Given a point $x \in \mathbb{R}^d$, the discriminative function is defined as:

$$T_{\mathcal{L}_0,\mathcal{L}_1}(x) = \sum_{L \in \mathcal{L}_1} \frac{1}{distP(x, L) + \varepsilon} - \sum_{L \in \mathcal{L}_0} \frac{1}{distP(x, L) + \varepsilon} \quad (4)$$

where ε is a small positive constant to avoid division by zero.

Considering the discriminative function, the classification function is defined as:

$$F_{\mathcal{L}_0,\mathcal{L}_1}(x) \quad \begin{cases} 0, & if \ \mathcal{S}_{\mathcal{L}_0,\mathcal{L}_1}(x) < 0.5; \\ 1, & \text{otherwise} \end{cases} \quad (5)$$

where $\mathcal{S}_{\mathcal{L}_0,\mathcal{L}_1}(x)$ is a sigmoid function denoted by:

$$S_{\mathcal{L}_0,\mathcal{L}_1}(x) = \frac{1}{1 + e^{-g(T_{\mathcal{L}_0,\mathcal{L}_1}(x))}} \quad (6)$$

The following relationships can be found among the discriminative, sigmoid and classification functions:

- If x has approximately the same "distance" from both $\mathcal{L}_0$ and $\mathcal{L}_1$ (that is, $\forall L \in \mathcal{L}_0, \exists L' \in \mathcal{L}_1 : distP(x, L) \approx distP(x, L')$ and vice-versa) then, by Eq. 4, $T_{\mathcal{L}_0,\mathcal{L}_1}(x) \approx 0$ and consequently, by Eq. 6, $S_{\mathcal{L}_0,\mathcal{L}_1}(x) \approx 0.5$. So, the more x is equally separated from $\mathcal{L}_0$ and $\mathcal{L}_1$, the more it is difficult to discriminate whether x belongs to class 0 or 1.
- The closer x is to $\mathcal{L}_1$ (that is, $\exists L \in \mathcal{L}_1 : distP(x, L) \approx 0$) and, at the same time, x is farther from $\mathcal{L}_0$ (that is, $\forall L \in \mathcal{L}_0 : distP(x, L) \approx +\infty$) then, by Eq. 4, the bigger is $T_{\mathcal{L}_0,\mathcal{L}_1}(x)$ (that is, it tends to $+\infty$) and, consequently, by Eq. 6, the closer $S_{\mathcal{L}_0,\mathcal{L}_1}(x)$ is to 1, and, therefore, by Eq. 5, the closer is more x can be discriminated as belonging to class 1 (that is, the closer the classification function $F_{\mathcal{L}_0,\mathcal{L}_1}(x)$ is to 1).
- Analogously, the closer x is to $\mathcal{L}_0$ and (at the same time) farther from $\mathcal{L}_1$, the closer the classification function $F_{\mathcal{L}_0,\mathcal{L}_1}(x)$ is to 0.

B. Training Algorithm

Given a set of n examples $E_n = \{(x_i, y_i) \in \mathbb{R}^d \times \{0, 1\} : i = 1, 2, .., n\}$, the main purpose of the training algorithm in the context of SLS classifiers, is to find two sets of SLSs $\mathcal{L}_0$ and $\mathcal{L}_1$ in order to minimize the mean square error function defined as:

$$E(F_{\mathcal{L}_0,\mathcal{L}_1}) = \frac{1}{n} \sum_{i=1}^{n} [F_{\mathcal{L}_0,\mathcal{L}_1}(x_i) - y_i]^2 \quad (7)$$

This algorithm can be divided into two phases [4]:

1) *Placing:* This phase consists of pre-allocating (finding initial positions of) the SLSs (in $\mathcal{L}_0$ and $\mathcal{L}_1$) based on the fact that points x closer to $\mathcal{L}_0$ (or $\mathcal{L}_1$, respectively) and farther from $\mathcal{L}_1$ (or $\mathcal{L}_0$, respectively) lead the classification function $F_{\mathcal{L}_0,\mathcal{L}_1}(x)$ to 0 (or 1, respectively). To achieve this goal, the set of examples E_n is split into two groups, $X_i = \{x \in \mathbb{R}^d : (x, y) \in E_n \text{ and } y = i\}$ (for $i = 0, 1$), and then the clustering algorithm k-means is applied to each group. Later, with the objective to obtain the initial extremities of the SLSs for each cluster, the k-means algorithm (with $k = 2$) is applied again, but at this time to each cluster obtained from the previous k-means application.
2) *Tuning:* The purpose of this phase is to minimize the mean square error function. One possible way to accomplish this task is to use the gradient descent technique [8] to find the positions of the SLSs in $\mathcal{L}_0$ and $\mathcal{L}_1$ such that the mean square function derivate is equal to zero. Despite of the gradient descent method does not guarantee the global minimum and the final solution (positions of the SLSs) depends on the initial placing phase, this method was successfully applied in [4].

III. Dialectical Optimization Method

The Dialectical Optimization Method (DOM) was introduced in [6], [7] and it is an evolutionary method based on the materialist dialectics for solving search and optimization problems. DOM is based on the dynamics of contradictions between their integrating dialectical poles [7], where each pole is considered as a possible solution for the problem. These poles are involved in a *pole struggle* and they are affected by *revolutionary crises* where some poles may disappear or may be absorbed by another poles and new poles may come up from new periods of revolutionary crises. These two processes make the system tend to stability [6].

Since this method is based on philosophical concepts, their definitions shown in [6], are briefly described in this section.

A *pole* is the fundamental integrating unit of a dialectical system and corresponds a candidate solution to the problem and it is represented by a *vector of conditions* with n inputs.

Given a set of poles $\Omega = \{w_1, w_2, ..., w_m\}$, each pole w_i is associated to a *vector of weights* defined as:

$$w_i = (w_{i,1}, w_{i,2}, ..., w_{i,n})^T \quad (8)$$

where m is the number of poles and n is the system dimensionality.

The *social force* of each pole w_i is associated to the objective function of the optimization problem and from now on, it will be represented by $f(w_i)$. This is the fundamental idea of the dialectical method.

The *contradiction* between two poles w_p and w_q is defined as:

$$\delta_{p,q} = dist(w_p, w_q) \tag{9}$$

where $dist(a, b)$ denotes the Euclidean distance between two points a and b.

A *historical phase* consists of two stages: (i) *evolution* and (ii) *revolutionary crisis*. These states correspond to the dynamics between poles through *contradiction* evaluations.

At the *pole struggle* stage, the pole w_i has the *hegemony* at instant t when

$$f_P(t) = f(w_i(t)) = \max_{1 \leq j \leq m} f(w_j(t)) \tag{10}$$

and $f_P(t)$ is known as the *present hegemony* at time t . In the same way, the *historic hegemony* up to time t, denoted by $f_H(t)$, and defined as the maximum *social force* value obtained in each historical phase up to time t.

Given a pole w_i such that $a \leq w_i \leq b$, where $a, b \in \mathbb{R}$, the *opposite* or the *absolute antithesis* of w_i is defined as:

$$\breve{w}_i = b - w_i + a \tag{11}$$

According to the dialectical method, the *synthesis* of two poles is the resolution of the contradiction between them (*thesis* and *antithesis*). As a result, it is obtained one new pole with characteristics inherited from both of them.

The set of parameters that must be provided by the user is: the number of poles (m), the number of historical phases (η_P) and the duration of each historical phase (η_H). As can bee seen in the pseudocode presented on Algorithm 1, the parameters η_P and η_H work as thresholds for the loops described at Lines 2 and 9. In addition, the two phases of DOM (evolution and revolutionary crisis) are represented at Lines $3-8$ and Lines $10-15$, respectively.

Algorithm 1 Dialectical Optimization Method Algorithm

Require: m, η_P, η_H: set of initial parameters.
Ensure: $dGlobalWinValue$: optimized value.

```
1: Generation of m initial poles → (w[0 : m])
2: while (iNumPhases < ηP) do
3:    if (iNumPhases > 0) then
4:       deleteSimilarPoles()
5:       generateSynthesis()
6:       generateAntithesis()
7:       addExternalEffects()
8:    end if
9:    while (iPhaseDuration < ηH) do
10:      while (iNumPoles < m) do
11:         iWinPole ← 0
12:         if (f(w_iNumPoles) < f(w_iWinPole)) then
13:            iWinPole ← iNumPoles
14:            dGlobalWinValue ← f(w_iWinPole)
15:         end if
16:      end while
17:      addCrisisEffect()
18:   end while
19: end while
```

IV. Hybrid of Dialectical Optimization and Gradient Descent Methods

On one hand, the main advantage of DOM is the capability of escaping from local optimum by multi-point stochastic searching. On the other hand, the strength of gradient descent method [8] is the ability of finding local optimum by pointing the direction that maximizes the objective function. Thus, based on these properties, the combination of gradient descent and dialectical optimization methods may improve the solution "quality" of optimization problems. The idea to build a hybrid method is showed in Fig. 1, where the main goal of DOM is to assist the gradient descent method by providing to it a new set of initial positions (the output of the dialectical optimization method) for the two sets of SLSs $\mathcal{L}_0$ and $\mathcal{L}_1$.

The following steps resume the idea of this hybrid method:

1) Generate initial poles as described in DOM.
2) For each phase of DOM, apply the gradient descent to each pole in the population in order to obtain one optimum local for each pole.
3) Proceed with the next steps of DOM (i.e.*pole struggle, synthesis, etc.*) as referred in Section III.

Our approach uses an evolutionary method to scape from the local optimal provided by the gradient descent technique to find another value (even better to the previous one) which may be the new *global optimum*. Then, at the moment when the gradient descent stops at the minimum local, the DOM provides a new start point helping the gradient to scape from that "*valley*".

It is important to emphasize that the initial population (set of poles) includes the solution obtained from one single application of gradient descent so that our hybrid approach can generate a solution that is equal to or better than the one obtained by just using the gradient descent method. In this sense, our approach is conservative.

A. Adaptations to DOM concepts

Some modifications have been done to the previous DOM concepts shown in Section III to adapt the problem of finding the best the positions of the SLSs in $\mathcal{L}_0$ and $\mathcal{L}_1$.

Since a pole corresponds to a possible solution, we define a pole as a vector consisting of the extremities of the SLSs belonging to both $\mathcal{L}_0$ and $\mathcal{L}_1$. Thus, a pole can be represented by a vector obtained from the concatenation of $\mathcal{L}_0$ with $\mathcal{L}_1$ (i.e. $[\mathcal{L}_0|\mathcal{L}_1]$).

The initial population is built in the following way: half of the initial population is randomly generated with data between the range of the minimum and maximum values from the

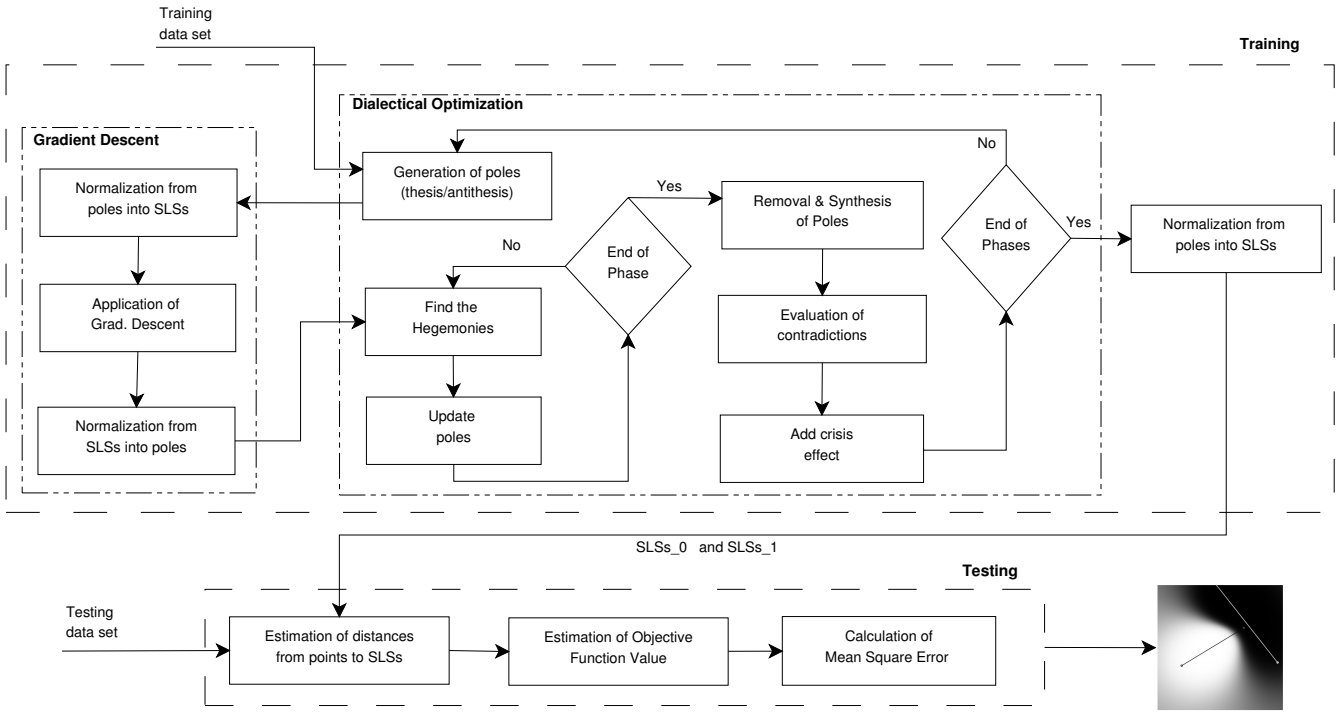


Fig. 1. Data flow representation of the proposed hybrid method, where each block represents a process. The system input is a training data set and the system output is a set of two SLSs and the classified data set.

training data set. The other half part of the initial population is generated with the opposite poles (*antithesis*).

The antithesis of a pole $[\mathcal{L}_0|\mathcal{L}_1]$ (described in Section III) is redefined as $[\mathcal{L}_1|\mathcal{L}_0]$. Since our problem deals with a set of SLSs which divides two classes of points, the opposite vector is equivalent to invert classes of the SLSs positions, meaning that the SLSs in $\mathcal{L}_0$ and $\mathcal{L}_1$, now represents class 1 e 0, respectively.

The social force is represented by the MSE defined as

$$R_{E_n}(\mathcal{L}_0, \mathcal{L}_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - F_{\mathcal{L}_0,\mathcal{L}_1}(x_i))^2 \qquad (12)$$

where $F_{\mathcal{L}_0,\mathcal{L}_1}(x_i)$ is the classification function from the SLS classifier (defined in Eq. 5) and y_i is the class value.

The contradiction of two poles was defined in Section III as their Euclidian distance. In our work, the contradiction is redefined as the difference in absolute value between the MSE obtained for each involved pole. Therefore, the contradiction means the difference of classification errors between two poles.

The pseudocode for this hybrid method is the same as presented in Algorithm 1, with the only difference that right before Line 3, the gradient descent method is applied to each pole before executing the steps of DOM's method.

By looking at Fig. 1, it is easier to understand this hybrid method, folowing the data flow which is divided into two process. First, given a training data set, the population (thesis and antithesis) is generated. Then, it is applied the gradient descent method to each member of the population. Finally the remaining DOM functions, grouped as Evolution (first loop) and Revolutionary Crisis (second loop), are applied. At the end of the training phase, two sets of SLSs are obtained, which are the ones with minimum MSE error. Second, given a test data set, the distances between the new examples and the obtained SLSs from the training set are computed to obtain the MSE error. As a final result, an image of the examples classification is obtained, showing the SLSs positions.

V. Experimental Results and Discussion

In order to evaluate the proposed hybrid method for training SLS classifiers, we conducted two experiments. In the first one, using artificial data, we have applied our hybrid method on data sets drawn from four artificial probability density functions named here as F, S, Simple and X and described in [4].

A. Artificial Data Sets

In these data sets, each region has a uniform probability density function. This property makes possible to apply the Bayes classifier [1] to obtain the ideal classification rate and compare it with our result as it was proposed in [4]. Since the probability density function is known, it is possible to use numerical integration to calculate the classification rate of the SLS classifier that was obtained by applying our hybrid method in the training phase.

For this experiments, the parameters used for the gradient descent method are the same used in [4]:

- Number of iterations = 1000,

- Initial Value = 0.1,
- Displacement decrement = 0.5,
- Displacement increment = 0.1,
- Minimum value = 10^{-5},

and, the parameters for DOM are:

- Number of poles = 30,
- Number of phases = 20,
- Number of iterations = 15,
- Minimum Value = 10^{-3},
- Learn Rate = 0.99,
- Crisis Effect value = 0.2.

We have generated four samples with 100, 200, 400 and 800 examples, respectively. For each sample, we have applied our hybrid method to 1, 2, 3 and 4 SLSs for each class.

To show the results, we use a graphic representation (a grayscale image) of the classification regions computed by the classification function $F_{\mathcal{L}_0,\mathcal{L}_1}(x_i)$ of the SLS classifier. For each image, the black and white regions represent the regions for classes 0 and 1, respectively. The SLSs in $\mathcal{L}_0$ and $\mathcal{L}_1$ are also indicated in the opposite color (i.e. black regions with white SLSs and white regions with black SLSs). The SLSs are in $\mathbb{R}^{d+1}$. For the sake of visualization, when the SLSs are in $\mathbb{R}^3$, they are projected into $\mathbb{R}^2$ with the purpose of showing them in the image results.

In order to evaluate the hybrid method, we have computed the classification rate based on the Bayes classifier for each sample. The bold numbers indicates the best classification rate for each sample (i.e. the best classifier with "n" SLSs per class). It is important to emphasize that all the results are close to the ideal classification rate and, in most cases, they are better than the results obtained in [4].

B. Discussion

The probability density function F is represented on the first row in Fig. 2. By looking at it, we can observe that regions and SLSs positions are better fitted on the probability density function, it is possible to see the improvement from GD in second column, GD with genetic algorithms (GD-AG) in third column and our method in fourth column. In the last three columns we present the best results obtained for each method GD, GD-AG and GD-DOM. It can be seen that GD and GD-DOM results are very similar but GD uses 3-SLS and GD-DOM only 2-SLS which is better.

We can also observe that among the classifications rate presented at Table I, the results for larger amount of examples, such as 400 and 800, the best classification rate is obtained from a SLS classifier with 3 SLSs for each class. The classification rate is about 87.14% for 800 examples which is very close to the ideal classification rate (with Bayes classifier) 87.65%.

In the same way, the probability density function S is represented on the second row in Fig. 2, and once again the best representation of the data distribution is in column *d* which is the result from our hybrid method. From Table I, we can say that the best classification rates were reached using 3 SLSs for samples with 200 and 400 examples (columns *d* and *e* from Fig. 2); while for samples with 800 examples, the best classifier uses 2 SLSs (column *f*). In all results, the classification rates are close to the ideal classification rate.

The Simple-distribution is the one with the best result among the four distributions proposed in [4] and as can be seen on Fig. 2 and compared with the classification rates in Table I. We can highlight that all percentages of classification rate are very close to the ideal classification rate (93.90%) and even one of them is only 0.40% percent lower than the ideal one. This corresponds to the case of the SLS classifier with 3 SLSs obtained from a sample with 800 examples. It can also be seen that for samples with 100 and 400 examples, the percentage of correct classification is about 93%.

In the opposite way, the X-distribution presents the weakest classification rate among the others for our proposed method as can be seen in Table I and it is the well known "XOR" distribution. For this distribution, we did not obtain a big classification rates improvement (sometimes they are equal to the gradient descent method, and sometimes they are worst than it). As can be seen on Fig. 2 (last three columns) the best results for this distribution were reached with just 1-SLS for GD and GD-DOM, but using GD-AG were used 3-SLS to reach the best classification rate.

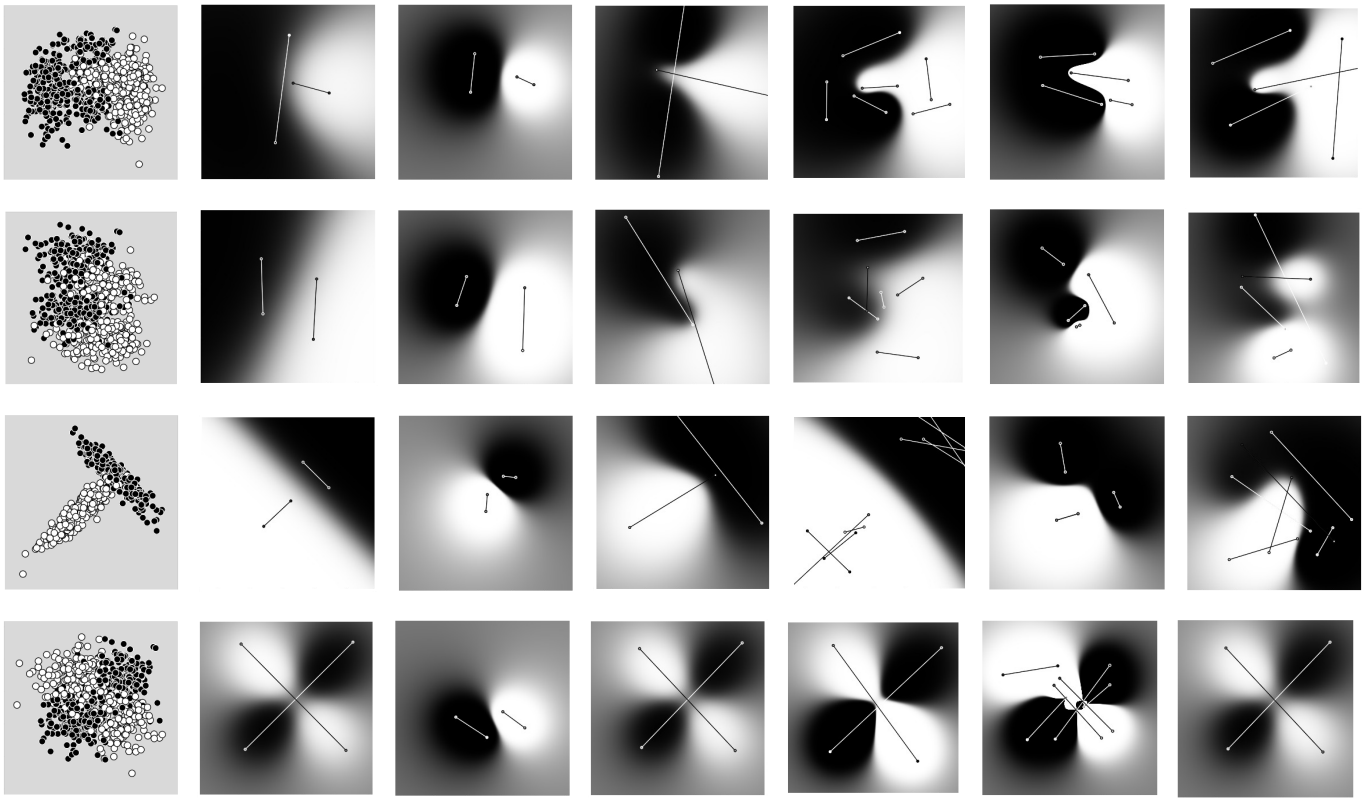
C. Gradient Descent vs. Gradient Descent with DOM

In this section, we present a line diagram in Fig. 3, in order to compare the classification rates obtained for both methods, GD and GD-DOM when a SLS classifier with 3-SLS is applied on the four distributions F, S, X and Simple presented previously and drawn in different colors. As can be seen, the gradient descent method is represented with pointed lines and our hybrid method with contonuous lines. In the figure are represented the best classification rates obtained in the experiment conduced on this work with 100, 200, 400 and 800 examples.

Using this graphic is much easier to compare the gradient descent with the proposed method. Thus, for F-distribution (using plus signs), the improvement by using our approach is between 1% e 4%. In the same way, for S and Simple distributions (using circles and asterisk, respectively) the improvement of the classification rates varies between 0.50% e 8%. Finally, for X-distribution (using letter x), the most complicated for our method the percentage of improvement is between 1% e 4%.

Furthermore, the graph suggests that for this work the distribution with the best classification rate is Simple-distribution, with percentages for both GD and GD-DOM over 90%. Whereas the F-distribution has a percentage of correct classification between 80% and 90%, and for the other two distributions, S and X, the percentages of correct classification are between 60% e 70%.

It is worth noting that for all the distributions presented in this work, our method increases the percentage of correct classification when the number of examples is greater, as can be seen in Fig. 3, where for 400 and 800 examples per sample



(a) Distribution (b) GD (1-SLS) (c) GD-AG (1-SLS) (d) GD-DOM (1-SLS) (e) GD (best) (f) GD-AG (best) (g) GD-DOM (best)

Fig. 2. Graphic representation of the results obtained for the distributions before mentioned. The first four columns were obtained using 800 examples and represent: (a) Data distribution (F, S, Simple and X), (b),(c) and (d) Results with the three methods used in this work. The last columns represent the best classification rate obtained for each distribution using (e) GD, (f) GD-AG and (g) GD-DOM, respectively.

TABLE I

CLASSIFICATION RATES OBTAINED USING: (i) GD (GRADIENT DESCENT); (ii) GD-DOM (GD WITH DIALECTICAL OPTIMIZATION) AND (iii) GD-AG (GD WITH GENETIC ALGORITHMS), FOR EACH DISTRIBUTION WITH AN IDEAL CLASSIFICATION RATE REPRESENTED IN PARENTHESIS.

		NUMBER OF STRAIGHT LINE SEGMENTS PER CLASS											
		1			2			3			4		
	EXAMPLES	GD	GD-DOM	GD-AG	GD	GD-DOM	GD-AG	GD	GD-DOM	GD-AG	GD	GD-DOM	GD-AG
F **(87.65%)**	100	82.89%	**83.32%**	59.94%	**85.99%**	84.84%	84.48%	**86.75%**	85.49%	73.63%	84.14%	**86.46%**	67.19%
	200	75.02%	**86.24%**	73.29%	86.88%	**87.17%**	80.94%	**87.06%**	86.94%	81.76%	83.30%	**87.13%**	81.71%
	400	74.71%	**86.64%**	74.52%	**86.95%**	86.72%	85.02%	82.23%	**86.77%**	79.30%	83.95%	**86.51%**	75,04%
	800	76.86%	**84.69%**	76.77%	86.34%	**86.99%**	82.15%	86.71%	**87.14%**	75.70%	86.36%	**86.55%**	78.61%
S **(68.78%)**	100	59.09%	**61.79%**	59.72%	**66.03%**	60.13%	56.27%	**65.48%**	62.84%	60.13%	62.22%	59.01%	**62.74%**
	200	59.09%	**65.37%**	60.27%	**66.87%**	65.32%	64.80%	62.89%	**65.90%**	63.00%	63.90%	**65.22%**	58.92%
	400	59.01%	**66.38%**	60.16%	66.82%	**66.86%**	60.94%	66.41%	**66.99%**	62.52%	60.54%	**66.90%**	58.82%
	800	58.93%	**66.72%**	59.45%	59.81%	**67.90%**	66.15%	59.76%	**67.66%**	65.91%	59.52%	**67.78%**	62.28%
SIMPLE **(93.90%)**	100	92.78%	**92.92%**	86.92%	92.78%	**92.99%**	91.98%	92.81%	**93.01%**	92.48%	92.78%	**92.96%**	88.92%
	200	92.51%	**92.66%**	89.30%	92.55%	**92.62%**	91.82%	**92.49%**	92.02%	92.22%	92.41%	**92.66%**	92.10%
	400	92.53%	**93.19%**	89.60%	92.61%	92.77%	**93.59%**	92.68%	**93.46%**	91.81%	92.68%	**93.45%**	93.08%
	800	91.75%	**93.50%**	90.02%	92.56%	**93.56%**	93.13%	92.39%	**93.59%**	93.04%	92.70%	**93.51%**	92.92%
X **(66.70%)**	100	65.13%	**65.75%**	40.79%	**66.04%**	64.99%	38.53%	**65.33%**	64.03%	64.83%	**66.24%**	63.69%	58.52%
	200	**66.26%**	65.92%	37.41%	**66.29%**	64.48%	62.70%	**66.09%**	62.98%	61.70%	**66.18%**	60.96%	62.77%
	400	65.91%	**65.98%**	29.65%	65.11%	**64.57%**	62.08%	63.85%	64.06%	**65.56%**	63.66%	**63.36%**	60.76%
	800	**66.50%**	66.46%	19.96%	**66.31%**	65.90%	65.25%	65.93%	**66.21%**	64.91%	**66.35%**	65.81%	63.71%

our method obtained a percentage equal to or greater than the one obtained by the gradient descent method.

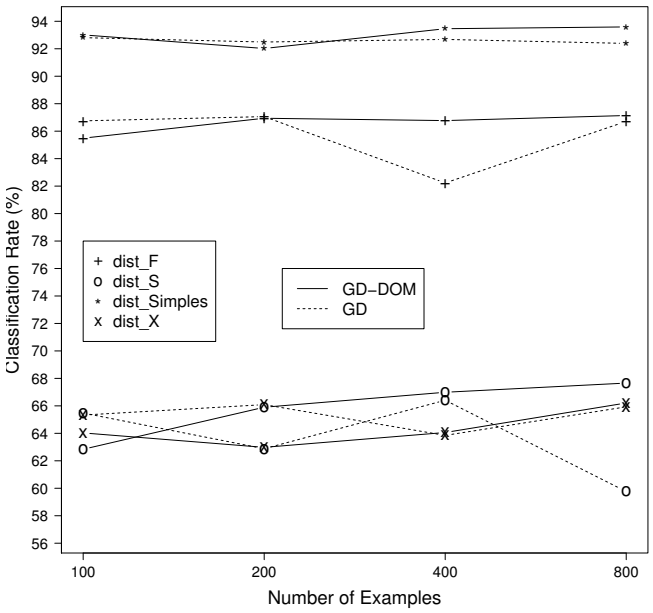


Fig. 3. Comparison between both methods GD and GD-DOM using the classification rates obtained from applying the SLS classifier with 3-SLS on the four distributions described before.

D. Gradient Descent with DOM vs. Gradient Descent with Genetic Algorithms

Some experiments were performed to compare our hybrid method with another optimization algorithms, in this case we chose Genetic Algorithms [9], [10], [11]. These techniques are commonly applied to find approximate solutions in optimization problems and they belong to a particular class of evolutionary algorithms which are inspired on evolutionary biology such as natural selection, mutation and inheritance.

The algorithm operates by iteratively updating a set of individuals, called the population. The population consists of many individuals called chromosomes. All members of the population are evaluated by the *fitness* function on each iteration. A new population is then generated by probabilistically selecting the most fit chromosomes from the current population. Some of the selected chromosomes are added to the new generation while others are selected as parent chromosomes. Parent chromosomes are used for creating new offsprings by applying genetic operators such as crossover and mutation [12].

Here the population is generated in the same way as DOM, each chromosome represents a potential solution to the optimization problem (i.e. $[\mathcal{L}_0|\mathcal{L}_1]$). Consequently, a population is a set of possible solutions and initially is randomly generated. Like in DOM method, there are some parameters in genetic algorithms that must be properly tuned. In particular the number of chromosomes for initial population (set to 200) and the number of iterations (set to 100).

In Table I, can be seen the classification rates obtained from applying the SLS classifier using in this case a combination of gradient descent and genetic algorithms. For F-distribution the rates have a difference of about 5% or 7% percent from the classification rates obtained with GD-DOM. In S-distribution, this new combination get the highest rate among all the other methods, but only for 100 examples and 4-SLS. It was the same situation with Simple-distribution, but this time using 400 examples and 2-SLS. In addition, for Simple and X distributions the rates were closer to the ideal classification rate in some cases. These methods were combined in the same way as gradient descent with DOM: (i) generate population; (ii) apply gradient descent to each chromosome in population and (iii) the remaining AG functions.

The results suggests that our proposed method obtained better classification rates than the combination of gradient descent with genetic algorithms, thus DOM has more iterations with the possible solutions and the operators in DOM were redefined in order to adapt them to our problem.

E. Public Data Set

The proposed hybrid method was also applied to one public data set from the UCI Machine Learning Repository [13]: the Breast Cancer Wisconsin (Diagnostic) Dataset, with 2 classes (B for Benign and M for Malign) and 10 attributes (features).

The experiment was done using a sample with 400 examples for training and a sample with 300 examples for testing. The applied SLS classifiers have 1, 2, 3 and 4 SLSs per class. Table II presents and compares the results obtained by our method with the ones obtained by [4], for the same data set, but with a sample with 682 examples when using the gradient descent method.

As we can see in Table II, our hybrid method had a better performance using SLS classifiers with 3-SLSs and 2-SLSs. It was not possible to obtain some figures to represent the classification for this data set, because it is 10-dimensional data set.

TABLE II
CLASSIFICATION RATE FOR BREAST CANCER DATA SET

METHOD	NUMBER OF SLSS PER CLASS			
	1	**2**	**3**	**4**
GradDesc	**96.78%**	96.92%	96.34%	**96.78%**
GradDescDOM	96.66%	**97.32%**	**96.99%**	82.94%

VI. CONCLUSIONS

This paper presents a new method for training SLS classifiers, with a hybrid between gradient descent and dialectical optimization methods. The goal is to find the best position of the SLSs in order to minimize the MSE error. For that, in [4], it was applied the gradient descent method to optimize the search of this optimal positions. In this work, our main contribution is to improve the training phase (optimization

of SLSs positions), combining the gradient descent with an evolutionary method for optimization based on philosophical concepts such as dialectics.

Comparing the methods previously mentioned, all of them have a high classification rate, but on the hybrid method case (GD-DOM), the distributions are better fitted with the obtained SLSs positions. In addition, we can say that the best SLS classifier for the four proposed artificial distributions and the public data set, must have 3 SLS per class, because it has improved the classification rate in an average of 2%.

The proposed hybrid method has shown a very good classification for F and S distribution, although the classification rate was not improved for X-distribution, which has the lowest improvement percentage about 0.5%.

While this method improves the classification rate, the computation time for the training algorithm increases because of the multiple iterations of evolutionary DOM and gradient descent method. In addition it has been studied the use of threads on the implementation to reduce the training time.

Although, in this paper we use some predefined distributions and one public dataset, the presented results indicate that the SLS classifier using the proposed hybrid method can be potentially used in Computer Vision problems. We plan to do this analysis for future work and also extend the SLS binary classifier to a multiclass classifier.

ACKNOWLEDGMENT

Financial support for this research has been provided by FAPESP and CNPq. The authors are grateful to anonymous reviewers for the critical comments and valuable suggestions.

REFERENCES

[1] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2001.

[2] A. Jain, R. Duin, and J. Mao, "Statistical pattern recognition: a review," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 1, pp. 4 –37, Jan. 2000.

[3] J. Ribeiro and R. Hashimoto, "A new machine learning technique based on straight line segments," in *ICMLA 2006: 5th International Conference on Machine Learning and Applications, Proceedings*. IEEE Computer Society, 2006, pp. 10–16.

[4] ——, "A new training algorithm for pattern recognition technique based on straight line segments," in *Computer Graphics and Image Processing, 2008. SIBGRAPI '08. XXI Brazilian Symposium on*, oct. 2008, pp. 19 –26.

[5] ——, *Pattern Recognition, Recent Advances*. I-Tech, 2010, ch. Pattern Recognition Based on Straight Line Segments, book Chapter.

[6] W. D. Santos and F. D. Assis, "Optimization based on dialectics," in *IJCNN*, 2009, pp. 2804–2811.

[7] W. D. Santos, F. D. Assis, R. D. Souza, P. Mendes, H. Monteiro, and H. Alves, "Dialectical non-supervised image classification," in *Proceedings of the Eleventh conference on Congress on Evolutionary Computation*, ser. CEC'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 2480–2487. [Online]. Available: http://portal.acm.org/citation.cfm?id=1689599.1689929

[8] D. Michie, D. Spiegelhater, and C. Taylor, "Introduction to optimization theory," 1973.

[9] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[10] R. Haupt and S. Haupt, *Practical Genetic Algorithms*. Wiley-Interscience, 2004.

[11] J. Holland, *Adaptation in natural and artificial systems*. Cambridge, MA, USA: MIT Press, 1992.

[12] R. Chandra and C. Omlin, "The comparison and combination of genetic and gradient descent learning in recurrent neural networks: An application to speech phoneme classification," in *Artificial Intelligence and Pattern Recognition*, 2007, pp. 286–293.

[13] A. Asuncion and D. J. Newman, "UCI Machine Learning Repository," http://archive.ics.uci.edu/ml/, 2007.