Coupling Recursive Hyperspheric Classification with Linear Discriminant Analysis for Improved Results

Salyer B. Reed[1], Tyson R. C. Reed[2], and Sergiu M. Dascalu[3]
Department of Computer Science & Engineering
University of Nevada, Reno
Reno, Nevada USA
[1]sreed@cse.unr.edu, [2]TysonRCReed@gmail.com, [3]dascalus@cse.unr.edu

Abstract— **Recursive Hyperspheric Classification (RHC) can accurately and succinctly classify large datasets by dissecting labeled vectors into their constituent groups, or hyperspheres. While RHC is a powerful classification tool, coupling RHC with other linear classifiers enhances the ability and accuracy of the classification system, improving recognition of unlabeled vectors. In this paper, RHC is paired with Linear Discriminant Analysis (LDA) for improved classification and recognition rates.**

Keywords- Recursive Hyperspheric Classification; Machine Learning; Linear Discriminant Analysis; Dimensional Reduction;Wine Dataset

I. INTRODUCTION

Learning, in the general sense, is the ever-evolving process of improving at a given task. As a task is performed, the learned behavior – frequently – can be measured and assessed, indicating improvement. In machine learning – a branch of Artificial Intelligence – the process is no different; computers develop behaviors by modifying and adapting their prior behaviors, endeavoring to excel at the specified task. Ultimately, a generalized, or sensible, conclusion should be ascertained.

Supervised learning, a subset of machine learning, involves learning from exemplars, which are labeled feature vectors. There exist many robust, supervised learning algorithms, including support vector machines (SVM) as well as linear discriminant analysis (LDA). In fact, both of these methods are linear classifiers that boast of superb results for binary, linearly separable datasets.

The Support Vector Machine, initially conceived by Vapnik, is a robust algorithm that strives to separate contrasting classes with the aid of a hyperplane [3, 10]. Kernels, when coupled with the SVM, assist in transforming the dimensional space, attempting to more accurately fit the hyperplane between the respective classes; they project the data into a high dimensional space where the classes are separable. When fitting the linear classifier, the margin – the region separating the support vectors – is maximized, resulting in an optimal classification hyperplane. Finally, although the SVM is frequently utilized in binary classification, it has exhibited notable performance in multiclass classification; the SVM algorithm is flexible and robust [5].

Linear discriminant analysis is traditionally used in dimensional reduction; however, when paired with additional statistical methods, it has enjoyed success as an accepted linear classifier that is capable of separating two or more classes [2, 4]. LDA is a supervised learning technique, which calculates a hyperplane that – when all vectors are projected onto it – will maximize the separability of the projections. During this process, a set of vectors are calculated, which minimizes the within-class scatter yet maximizes the between-class scatter. Ultimately, LDA exploits the fact that there are multiple known classes in the data and finds the features, or combinations of them, that separate the data in fewer dimensions in a subspace. This new subspace can then be assessed and classified accordingly.

Recursive Hyperspheric Classification – a novel classification technique – excels at classifying noisy datasets by partitioning, or dividing, the space into labeled hyperspheres [8, 9]. Initially, a root hypersphere is produced, which encompasses all sample feature vectors. During each epoch, additional hyperspheres are spawned by grouping the contained vectors into sets – based on their class label – and calculating a new centroid for the child. Ultimately, the process of decomposing the space into hyperspheres yields excellent recognition rates as well as a structured hierarchy of the data.

This paper proposes coupling LDA with RHC to assist and improve the classification and recognition processes. Section 2 details the RHC algorithm as well as the recognition process. Section 3 describes the morphing of the algorithm by uniting RHC and LDA. Sections 4 and 5 validate and discuss the results of the hybridized algorithm. Finally, the conclusion and future results are detailed in Section 6.

II. RECURSIVE HYPERSPHERIC CLASSIFICATION

Recurisve Hyperspheric Classification (RHC) is the meticulous process of decomposing a space into labeled hyperspheres, attempting to estimate a distribution function of the defined space. These hyperspheres form a tree structure, or hierarchy. During pattern recognition, these hyperspheres – when traversed – designate labels for nondescript data.

A. Components

RHC, as was mentioned, is a compilation, or hierarchy, of hyperspheres. Each hypersphere has a set of intrinsic properties, including the radius and the centroid, which are

commonly referred to as the center of gravity, or COG. The radius and centroid define the locus of the hypersphere.

Given these two geometric properties, the volume of the hypersphere can be computed. Also, as the volume is defined, the relative position of any vector can be ascertained: it is outside or contained within the hypersphere. Overall, in mathematical terms, a hypersphere is nothing more than a ball.

B. The Algorithm

Amassing a diverse sample dataset is imperative to any classification algorithm, and it is no different for RHC. From the comprehensive dataset, scaling is applied. There exist many scaling techniques, including standardization and Winsorizing; however, as a sphere is symmetrical in all dimensions, normalization is applied, constraining all dimensions to the range [0, 1]:

$$x_i \leftarrow \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}.$$

1) The Root Hypersphere

Having fashioned a diverse, normalized sample dataset, the root – or first – hypersphere can be created for the sample space. As all dimensions are constrained to the interval [0, 1], the space's centroid, which is a vector, is determined to be:

$$\mathbf{c}_{root} = [0.5_1\ 0.5_2\ \cdots\ 0.5_n],$$

where n is the dimensionality of the space. The root hypersphere assumes this vector as its own centroid. Next, the radius is computed. As the root hypersphere must encapsulate all potential vectors in the space, the radius is defined to be the distance between the centroid and the furthest, possible vector:

$$r_{root} = \left|\mathbf{c}_{root} - \mathbf{v}_f\right|.$$

Because the all dimensions are normalized and if using Euclidean distance as the distance measure, the radius can be simplified:

$$r_{root} = \sqrt{\sum (0.5 - 0.0)^2} = \sqrt{\sum 0.5^2}\ .$$

Finally, as mentioned, the hypersphere must assume a class label, which will be selected from the set of known classes in the dataset:

$$class_{root} \in \{class_1, class_2, \cdots, class_n\}.$$

Empirical evidence suggests the class of the root should assume the label of the vector furthest from the centroid. Linearly iterating through the set, the algorithm is able to ascertain this vector, altering the label of the root hypersphere. In all, the root hypersphere is now defined.

2) Classifying the Space

The root hypersphere encompasses all vectors in the sample set; however, multiple hyperspheres must be spawned, or generated, to accurately classify the space. As such, the RHC algorithm recursively spawns additional hyperspheres, called children, starting with the root hypersphere. Ultimately, order and structure emerge from this recursive process, and a hierarchy, or treelike structure, stems from the data.

During each epoch, all hyperspheres will – potentially – generate additional children. For any given hypersphere the process begins by separating all encompassed vectors – that are not contained by a child – into labeled sets. Then, for each set whose label is different than the hypersphere, proceed to spawn a child.

Spawning a child hypersphere involves calculating the centroid, or mean vector, of the amassed labeled set. Once this vector is calculated for the set, and the child assumes it as its centroid:

$$\mathbf{c}_{child} = \mathbf{x}^{(l)},$$

where $\mathbf{x}^{(l)}$ is the mean vector for the labeled set, and $\mathbf{c}_{\text{child}}$ is the centroid of the child.

The radius of the newly spawned child is simply the radius of the parent less the distance between the centers of the parent and the child:

$$r_{child} = r_{parent} - \left|\mathbf{c}_{parent} - \mathbf{c}_{child}\right|,$$

where r_{child} is the radius of the child, $\mathbf{c}_{\text{parent}}$ is the centroid of the parent, and $\mathbf{c}_{\text{child}}$ is the centroid of the child.

Interestingly, the process of spawning children creates a hierarchy of hyperspheres with diminishing radii. Traditionally, searches whittle the space, endeavoring to find a solution, or label; these searches are utilizing a divide-and-conquer approach. Perhaps the most popular technique is the binary search algorithm, which continually divides the space until a solution is found. This process is emulated in RHC by spawning children with tapering radii; as the radius of the child diminishes, the search space, too, shrinks.

3) Recursion Stopping Condition

The process of spawning is continued, recursively diving into children, creating a hierarchy of hyperspheres. The space is considered mapped, or classified, when no additional hyperspheres are spawned. Also, the spawning process is guaranteed to have a finite number of iterations for the radius of each child is smaller than that of its parent.

Once the system has stopped spawning, it is considered classified, and the hierarchical structure of hyperspheres can be used for recognition of unlabeled feature vectors.

4) Pattern Recognition

Pattern recognition, simply, is the process of recognizing, or assigning, a label to unlabeled feature vectors. The system establishes a vector's membership to a specific,

known subclass. Because RHC constructs a hierarchy of hyperspheres, tree traversal algorithms are employed in the process of recognition.

First, the unlabeled vector is mapped onto the space coordinates. Presumably, if the coordinates have been normalized, the vector will reside within the root hypersphere. Because the vector is contained by the root hypersphere, the root is marked as a possible candidate for label selection. However, the scope of the root hypersphere is too vague, for it is extremely large; other hyperspheres, or children, may be more descriptive than the root hypersphere, and they should be parsed for accuracy.

Therefore, starting with the root hypersphere, iterate through each direct descendant, or child, determining if the vector resides within the volume of the hypersphere. If it does, once again mark the child as a possible candidate and, too, parse that child's children. If a child does not encapsulate the vector, it and its descendants can be trimmed from the tree.

Upon completing the exhaustive, less trimmed, search on the tree, a set of candidates will have emerged. From the possible candidates, the system should assign the unlabeled vector the designation of the hypersphere with the smallest radius, for that sphere is most descriptive of the unlabeled feature vector.

5) Distance Measures

Previously, it was mentioned that RHC should utilize a distance measure for calculating the radii of the hyperspheres; however, there exist many distance measures, and choosing the correct measure is imperative to classification and recognition rates.

Perhaps the most popular distance measure is the Euclidean distance, which calculates a scalar value between two points utilizing the Pythagorean theorem; this value is defines the proximity of the two objects, which is the distance. However, while the Euclidean distance is the most beloved measure, it is – perhaps – not optimal in estimating the distribution function that RHC produces. It has been shown that the squared Euclidean distance does, in fact, produce improved results [9].

In short, the squared Euclidean distance provides superior results because radii of children contract, or shrink, at a smaller rate than that of a system utilizing the Euclidean distance [9].

III. Coupling RHC with Linear Discriminant Analysis

Using a sample of exemplar vectors, Recursive Hyperspheric Classification can separate a dimensional space into its constituent parts. However, recognition rates can be, and are, improved when paired with a linear classifier; a synergetic relationship is formed.

A. Justification

Consider the two hyperspheres and exemplars in Fig. 1. If an unlabeled feature vector is inside the parent, or large, hypersphere, the vector would assume the label of this sphere unless it, too, were inside the child, or small, hypersphere; where it would assume the class of the child. Nonetheless, while a label can be inferred, the optimal separation of the space can be questioned.

In the figure, two additional unlabeled vectors are presented to the system for classification; they are identified by the question marks and reside near the child hypersphere. Presently, the system would assign the parent hypersphere's label to the unlabeled vectors. However, as mentioned, the validity of the designated class should be challenged, for the scatter for the square class is very tight, or small, and the unlabeled vectors reside near the grouping.

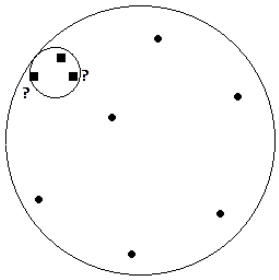

Figure 1. Two hyperspheres, separating two classes. Two unlabeled vectors are presented to the system for classification and are identified by question marks.

As such, it would be advantageous to apply a linear classifier, endeavoring to increase accuracy.

Ignoring the RHC algorithm, one might assume the best classification apparatus would be a single line that partitions the two classes and is comparatively equidistant from the competing vectors. This buffer, or margin, provides adequate room for noise in the testing and validation datasets. Comparing this linear separator with the separating boundary of the hypersphere radius, the radius boundary is too constricting and does not have a sufficient buffer; vectors of the same class may fall outside the classification boundaries set forth, or proposed, by the algorithm. As such, it is advantageous to construct a buffer. This is accomplished using LDA.

LDA, in short, will discover a line that – when data are projected onto it – partitions the classes, ensuring maximum separation, which is illustrated in Fig. 2.

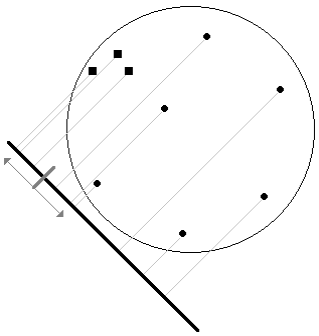

Figure 2. A two-dimensional hypersphere containing two classes showing the projections and midpoint.

B. Modifications to the Algorithm

Coupling a linear classifier with RHC involves modifying the RHC algorithm. First, a single linear classifier is able to separate, at most, two distinct classes. Therefore, if a hypersphere encapsulates three or more class labels, continue to spawn children and partition the space using the RHC algorithm. On the other hand, if the hypersphere encloses two distinct classes, try to separate the classes by applying a linear classifier.

The linear classifier applied in this paper is a derivative of Linear Discriminant Analysis (LDA). LDA is an attractive algorithm used in dimensional reduction, for – unlike unsupervised methods – it exploits the information encoded or observed in class labels of the dataset.

First, in applying the LDA algorithm, the mean vectors are computed for each class as well as the amassed dataset. Moreover, in LDA, there is a large emphasis on the calculated means, for the discriminant function is computed from the means and, subsequently, the projected means. It is noted that if using only the distance between the projected means as a measure of separation, the solution yields unsatisfactory results; the greatest distance between the projected means may not produce the optimal solution, for the proposed separation does not consider the variances of the classes. As a result, Ronald Fisher suggested maximizing the projected means, which are normalized by the scatter, or variance:

$$J(\overline{w}) = \frac{(\hat{\mu}_1 - \hat{\mu}_2)^2}{\hat{s}_1^{\,2} + \hat{s}_2^{\,2}},$$

where $\hat{\mu}_i$ are the projected means and $\hat{s}_i$ are the scatters of the respective classes. This is the objective function, which is to be maximized.

Scatter, essentially, is the variance, or spread, within the data, which is centered around the mean of the dataset. Being that the means are computed, one may compute the covariance for all classes, which indicates the scatter of the respective class. As such, the scatter is calculated as follows:

$$S_c = \sum (\overline{x}_i - \overline{\mu}_c)(\overline{x}_i - \overline{\mu}_c)^T,$$

where $\overline{x}_i$ is an exemplar and $\overline{\mu}_c$ is the mean of the designated class.

Finally, the within-class variance must be computed:

$$S_w = \sum S_c = S_1 + S_2,$$

where S_1 is the variance of the first class, and S_2 is the variance of the second class. Also, the between-class scatter must be computed:

$$\begin{aligned} S_B &= \sum (\overline{\mu}_c - \overline{\mu})(\overline{\mu}_c - \overline{\mu})^T \\ &= (\overline{\mu}_1 - \overline{\mu}_2)(\overline{\mu}_1 - \overline{\mu}_2)^T, \end{aligned}$$

where $\overline{\mu}$ is the mean of the entire dataset, and $\overline{\mu}_c$ is the mean of the respective class.

In order to maximize the objective function, the projected scatters must be written in terms of S_B and S_w. Therefore, it is noted that:

$$\hat{s}_c^{\,2} = \sum_i \left(\overline{w}^T \cdot \overline{x}_i - \hat{\mu}_c\right) = \overline{w}^T S_W \overline{w},$$

and

$$(\hat{\mu}_1 - \hat{\mu}_2) = \overline{w}^T S_B \overline{w}.$$

Having now the numerator and denominator, the objective function can be written as:

$$J(\overline{w}) = \frac{(\hat{\mu}_1 - \hat{\mu}_2)^2}{\hat{s}_1^{\,2} + \hat{s}_2^{\,2}} = \frac{\overline{w}^T S_W \overline{w}}{\overline{w}^T S_B \overline{w}}.$$

Taking the derivative the objective function with respect to $\overline{w}$ and equating it to zero will determine the optimal line for the dataset:

$$\begin{aligned} \frac{d}{dw} J(\overline{w}) &= \frac{d}{dw} \frac{\overline{w}^T S_W \overline{w}^T}{\overline{w}^T S_B \overline{w}^T} \\ 0 &= \frac{\left(\frac{d}{dw}\overline{w}^T S_B \overline{w}\right)(\overline{w}^T S_W \overline{w}) - \left(\frac{d}{dw}\overline{w}^T S_B \overline{w}\right)(\overline{w}^T S_B \overline{w})}{(\overline{w}^T S_W \overline{w})^2} \\ 0 &= (2S_B \overline{w})(\overline{w}^T S_W \overline{w}) - (2S_W \overline{w})(\overline{w}^T S_B \overline{w}) \\ &\vdots \\ (S_W \overline{w})(\overline{w}^T S_B \overline{w}) &= (S_B \overline{w})(\overline{w}^T S_W \overline{w}) \\ \frac{S_W \overline{w}(\overline{w}^T S_B \overline{w})}{\overline{w}^T S_W \overline{w}} &= S_B \overline{w}. \end{aligned}$$

It is noted that $\frac{(w^T S_B w)}{(w^T S_W w)}$ is scalar, which means the equation is a generalized eigenvalue problem, $A\overline{v}=\lambda\overline{v}$, and can be solved as such:

$$S_B S_W^{-1} \overline{w} = \lambda \overline{w}.$$

In attaining the solution, it is noted that the between-class scatter has been reduced to $(\overline{\mu}_1 - \overline{\mu}_2)(\overline{\mu}_1 - \overline{\mu}_2)^T$ because there are only two classes in the hypersphere. As such, the direction of $S_B\overline{w}$ is in the direction of $(\overline{\mu}_1 - \overline{\mu}_2)$; hence, $\overline{w}$ is in the direction of $(\overline{\mu}_1 - \overline{\mu}_2)$, ignoring the scalar ratio, for it does

not affect direction. Wherefore, the eigenvalue problem was reduced to:

$$\overline{w} = S_W^{-1}\left(\overline{\mu}_1 - \overline{\mu}_2\right).$$

Producing a line in the optimal direction is complete and can be utilized to compute the projected values from the dataset.

C. *Utilizing the Projections – Defining the Decision Boundary*

After applying the LDA algorithm, the exemplars are projected onto the optimal line, which was calculated by maximizing the objective function, which was normalized by the scatter. It is emphasized that the projection is, indeed, a line, which is one dimensional and produces scalar results in the subspace.

Because the projections are scalar, a simple assessment of the projected line is needed for classification. Again, the results consist of two class projections; as such, if the classes can be perfectly separated with a single point, then the space within the hypersphere is considered classified. This implies that there is no overlap between the corresponding class projections. On the other hand, if a single point cannot possibly separate the classes – because there is overlap – continue to employ the RHC algorithm, separating the two classes into sets and spawning additional hyperspheres as well as applying LDA on the resulting children.

Traditionally, defining the decision boundary between classes involves probabilistic classification methods such as Bayesian methods or Support Vector Machines. However, because it is a line and if there is no overlap, simple observation of the subspace will allow one to fit an optimal point, separating the two, distinct classes. Borrowing the principle of maximizing the margin between classes from SVM, the optimal choice – in the absence of additional information – would be the midpoint between the two classes. Nonetheless, any point between the two classes will suffice if there is no overlap.

Overall, RHC and LDA form a synergetic relationship. If the dataset is noisy or includes complex distributions, RHC continues to apply its divide-and-conquer strategy until the space is completely broken down into its constituent components – or hyperspheres – or LDA is able to obtain a projection where there is no overlap.

IV. Results

To show improvements, RHC-LDA has been verified and validated using the wine dataset while being compared to classical RHC as well as benchmarked using other classification techniques such as the Fuzzy Classifier Ensemble, the Robust Piecewise-Nonlinear M-SVM, and Evolutionary Computation Genetic Algorithm

The wine dataset is a diverse set of complete, multivariate exemplars. In all, there are 178 labeled vectors with thirteen features: alcohol, malic acid, ash, alkalinity of ash, magnesium, total phenols, flavanoids, nonflavanoid phenols, proanthocyanins, color intensity, hue, OD280/OD315, and praline. Associated with each vector is a class label, which represents the cultivar of the associated wine; there are three cultivars. It is the objective to select the correct cultivar given an unknown feature vector.

After classifying and partitioning the exemplars, RHC algorithmically traverses the hyperspheres, endeavoring to find the smallest hypersphere. If the smallest hypersphere has an associated LDA projection, the projection as well as the position of the midpoint, determine the class label. On the other hand, if the hypersphere does not have an LDA projection – because during its creation it only had a set of vectors with the same label – the unlabeled vector assumes the class of the hypersphere. The results of the benchmark, including ten-fold cross validation, are given in Table I.

TABLE I. Wine Dataset Results

Algorithm	Correct	Incorrect
RHC-LDA Sq. Euclidean	98.40%	1.60%
RHC Sq. Euclidean	97.90%	2.10%
RHC Euclidean	85.60%	14.40%
EC-GA	94.80%	5.20%
Robust Piecewise-Nonlinear M-SVM	90.00%	10.00%
Fuzzy Classifier Ensemble	92.70%	7.30%

V. Discussion

RHC coupled with LDA does, in fact, outperform classical RHC. By projecting the contained vectors in a hypersphere onto a line, an optimal midpoint can be obtained for the subset. If this midpoint can partition the two classes into distinct regions, the midpoint is the classifier for the hypersphere. On the other hand, if the midpoint cannot separate the projections, continue to apply the RHC algorithm until the exemplars can be classified by either a single hypersphere or another projection.

Also, the LDA algorithm assumes the data is a Gaussian and unimodal; as such, complex distributions would – traditionally – inhibit and skew the projections; there would be severe overlap of the projected data. However, RHC – when coupled with LDA – eloquently handles the overlap, for the algorithm will spawn children with diminishing radii, which, too, have LDA applied to their containing vectors.

Finally, it is noted that the projection and midpoint offer a more optimal solution to the subset, for the midpoint solution offers a more generalized solution than the obtuse geometry of the hypersphere; the hypersphere's geometry contours a portion of the space, while LDA reduces the dimensionality of the space and finds the optimal solution that separates the two classes.

VI. Conclusion and Future Work

RHC, while a popular and effective classifier, can be successfully coupled with LDA to produce a more optimal classification of a space. By projecting the exemplars onto a line and determining the midpoint, the classification and recognition process are improved. If the two class projections can be perfectly separated with a single midpoint point, then the space within the hypersphere is considered classified.

In retrospect, finding this midpoint of the line is synonymous to an SVM. An SVM can easily find the optimal classification point because it will maximize the margin between the two classes, and – given no additional information – will fit the midpoint between the two classes, producing an optimal midpoint, which bisects the two classes.

Future work includes the use of other linear classification techniques such as the SVM. Again, using the principle of perfect separation, an SVM can be applied to the vectors contained within a hypersphere. If the SVM can perfectly separate the classes, the space is considered classified. However, if it cannot, further application of the RHC and SVM algorithms can be applied until the space can be broken down into its constituent parts. Also, a kernel could be applied, transforming the data, making it linearly separable.

REFERENCES

[1] C. L. Blake and C. J. Merz, UCI Machine Learning Repository. University of California, Irvine. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[2] N. Boulgouris, K. Plataniotis, and E. Micheli-Tzanakou, "Discriminant Analysis for Dimensionality Reduction: An Overview of Recent Developments." *Wiley-IEEE Press*. 2010, pp. 1 – 19.

[3] M. A. Hearst, et al, "Support Vector Machines." *Intelligent Systems and their Applications, IEEE*. 1998, pp. 18 – 28.

[4] A. M. Martinez and A. C. Kak. "PCA Versus LDA." *Pattern Analysis and Machine Intelligence*. 2001, pp. 228 – 334.

[5] A. Mathur, "Multiclass and Binary SVM Classification: Implications for Training and Classification Users." *Geoscience and Remote Sensing Letters, IEEE*. 2008, pp. 241 – 5.

[6] T. Nakashima, G. Nakai, and Hisao Ishibuchi, "Constructing Fuzzy Ensembles for Pattern Classification Problems," *IEEE International Conference on Systems, Man, and Cybernetics*. 2003, pp. 3200 – 5.

[7] M. L. Raymer, et al, "Knowledge Discovery in Medical and Biological Datasets Using a Hybrid Bayes Classifer / Evolutionary Algorithm," *IEEE Transactions on Systems, Man, and Cybernetics*. 2003, pp. 802 – 13.

[8] S. B. Reed, C. Looney, and S. M. Dascalu, "A Recursive Hyperspheric Classification Algorithm." *21st International Conference on Computer Applications in Industry and Engineering (CAINE)*. 2008, pp. 156 - 60.

[9] S. B. Reed, T. R. C. Reed, and S. M. Dascalu, "High Dimensional Pattern Recognition Using the Recursive Hyperspheric Classification Algorithm." *World Automation Congress*. 2010, pp. 1 – 8.

[10] V. Vapnik, "The Nature of Statistical Learning Theory." *Springer*, Berlin, 1995.

[11] P. Zhong and M. Fukushima, "A Regularized Nonsmooth Newton Method for Multi-class Support Vector Machines," *Optimization Methods and Software*. 2007, pp. 225 – 36.