Extending Granular Ball Twin Support Vector Machine for Multi-class Classification

Hung.C Nguyen[1][0009-0003-2194-3876] and Quang-Thinh Bui[2][0000-0002-7357-2574]

[1] HUTECH University, Ho Chi Minh City, Vietnam
[2] Tien Giang University, Tien Giang, Vietnam
buiquangthinh@tgu.edu.vn

Abstract. Multi-class classification is a key area in machine learning, with Support Vector Machines (SVMs) and their variant, Twin SVM (TSVM), showing significant effectiveness in various applications. Recently, Granular Ball Twin SVM (GBTSVM) was introduced, combining Granular Computing with TSVM to reduce data noise and enhance performance, and it has been applied in several prior studies to multi-label classification tasks. This paper proposes Granular Ball Twin Support Vector Machine for Multi-class Classification (GBTSVM_MC), a novel method to extend GBTSVM for multi-class tasks. GBTSVM_MC employs two decomposition strategies, One-vs-One (OvO) and One-vs-Rest (OvR), to handle multi-class datasets. It leverages TSVM's strong classification ability and the computational efficiency of granular balls while improving computation for multi-label issues. To experimental evaluation of the performance of GBTSVM_MCC on diverse datasets along with additional testing on large datasets, thereby verifying the scalability and practical applicability of the model.

Keywords: Key Word: Multi-class Classification, Support Vector Machine, Twin Support Vector Machine, Granular Ball Computing, Noise Resistance, Large Scale Dataset

1 Introduction

In the field of machine learning, classification algorithms have made significant strides, with Support Vector Machines (SVMs)[1] standing out due to their effectiveness in handling multidimensional data and their strong theoretical foundation in statistical learning theory. SVMs have demonstrated success in diverse applications, including disease diagnosis, anomaly detection, and image classification[2–4]. Originally designed for binary classification, SVMs solve a large quadratic programming problem (QPP) to find the optimal separating hyperplane. However, this design limits their ability to directly address multiclass scenarios and large-scale datasets, prompting the development of advanced variants such as Twin Support Vector Machines (TSVMs)[5]. TSVMs enhance efficiency by solving two smaller QPPs instead of a single large one, achieving computation speeds up to four times faster than traditional SVMs. Building on this, the Granular Ball Twin Support Vector Machine (GBTSVM)[6] incorporates

granular computing to represent data as granular balls rather than individual points, improving noise immunity and computational efficiency.

The motivation to extend binary classification methods to multiclass settings arises from the growing complexity of real-world datasets, where data often span multiple categories, such as in image segmentation or medical diagnosis. Traditional binary classifiers struggle with challenges like the curse of dimensionality and high computational costs when applied to multiclass problems, necessitating efficient strategies such as One-vs-One (OvO) and One-vs-Rest (OvR)[7]. To address these issues, several multiclass models for SVMs[8–10] have emerged. Among them, the Granular Ball K-Class Twin Support Vector Classifier (GB-TWKSVC)[11] tackles multiclass problems by combining GBTSVM with the One-versus-One-versus-Rest (OvOvR) strategy. Despite its advancements, there remains a need for more adaptive and computationally efficient approaches to handle large-scale datasets effectively.

This paper proposes a novel framework, Granular Ball Twin Support Vector Machine for Multi-class Classification (GBTSVM_MC), which extends GBTSVM to multiclass tasks by employing OvO and OvR decomposition strategies instead of the more complex OvOvR approach. Our method integrates the computational efficiency of TSVM with the robust data representation of granular balls to tackle key challenges in multiclass classification, such as scalability to large-scale data and noise immunity. The contributions of this work are outlined as follows:

- **Improved Computational Efficiency for Large-Scale Data**: GBTSVM_MC builds on the TSVM framework by breaking down the classification problem into smaller QPPs and uses a fine-grained granular ball representation to reduce the number of input data points. This combined approach significantly lowers computational costs, making the model well-suited for large-scale datasets, as evidenced by experiments on datasets like Global Cancer, which includes up to 50,000 samples.
- **Flexible Multiclass Processing with OvO and OvR Strategies**: Unlike GB-TWKSVC, which relies on the OvOvR strategy, GBTSVM_MC adopts OvO and OvR methods to decompose multiclass problems into manageable binary subproblems. This flexibility enables the model to adapt to diverse dataset structures, enhancing both accuracy and training speed, as shown by comparative results on various UCI and Kaggle datasets.
- **Enhanced Noise Robustness and Scalability**: By using granular balls as input units, the model improves resilience to noise and outliers, which is critical for real-world applications with imperfect data. Additionally, extensive testing on subsets of the Global Cancer dataset (ranging from 1,000 to 50,000 samples) confirms the model's scalability and practical applicability across different datasets and domains.

The structure of this paper is organized as follows. Section 2 provides a comprehensive background and mathematical foundation of related models, including TSVM, GBTSVM, and GB-TWKSVC, which form the basis for the advancements we propose. Section 3 elaborates on the GBTSVM_MC methodology, covering granular ball generation, multiclass classification strategies, and algorithm design. Section 4 presents the experimental setup and results, highlighting the efficiency and scalability of

GBTSVM_MC across multiple datasets. Finally, Section 5 discusses the implications of our findings and identifies potential directions for future research.

2 Background and Related work

2.1 Granular Ball Twin Support Vector Machines (GBTSVM)

GBTSVM (Granular Ball Twin Support Vector Machines) was proposed by A. Quadir[6] based on the idea of combining the stability of the granular ball computing method and the superior computational efficiency of TSVM. This idea helps the model increase its noise tolerance, thanks to performing calculations based on particles instead of individual data points. At the same time, the computational time is significantly reduced because only a small quadratic programming problem (QPP) needs to be solved for each label, instead of having to handle a large QPP problem for the entire data set.

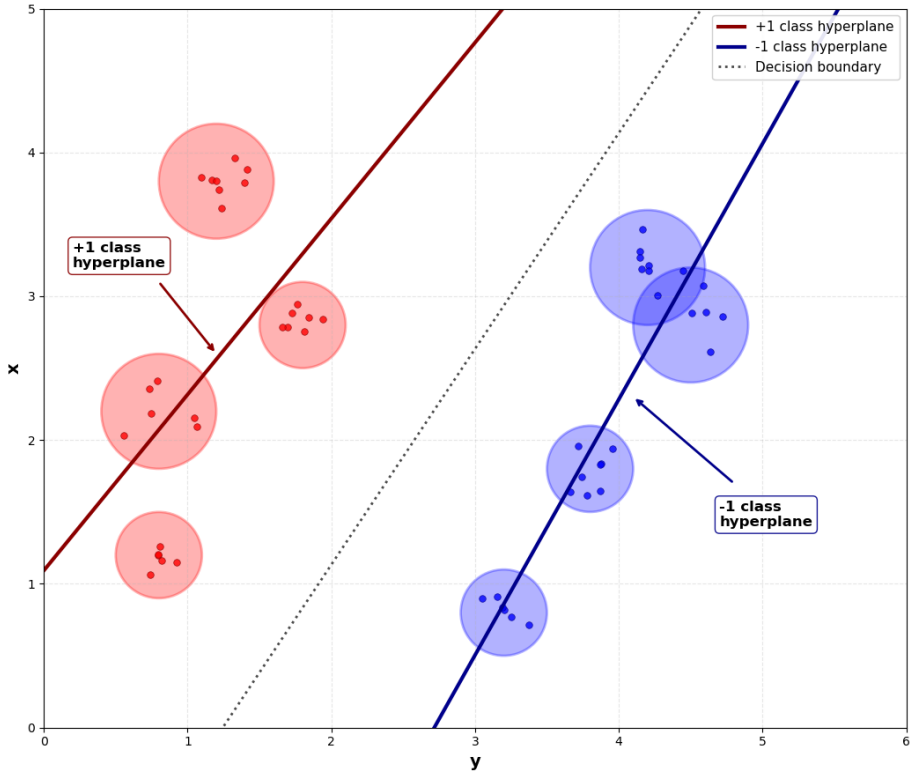

Fig. 1. Visualization of GBTSVM

The mechanism of Granular Ball Computing is a data processing technique to solve the scalability problem with high-dimensional data, by using Granular Balls (GB) defined by the center and radius instead of the entire data points, to reduce the amount of information to be processed and increase the computational efficiency.

To explain it mathematically, let the set of generated GBs be denoted as $S = GB_i, i = 1,2,\dots,p = \big((c_i, r_i), y_i\big), i = 1,2,\dots,p$, where c_i denotes the center, r_i denotes the

radius, and y_i is the label of the $i-th$ GB. This mechanism is represented in the form of an optimization formula as follows:

$$\begin{aligned} &min\ \lambda_1 \frac{n}{\sum_{i=1}^{p} GB_i} + \lambda_2 p \\ &s.t.\, pur\big(GB_j\big) \geq T, j = 1,2,\dots,p \end{aligned} \tag{1}$$

where λ_1 and λ_2 are weighting coefficients, and T is the purity threshold, which represents the proportion of samples in the GB with the same label that are in the majority. After determining the GBs for each layer, the center and radius of these GBs will be provided as input to the GBTSVM in the following manner:

$$\begin{aligned} &\min_{w_1,\, b_1} \frac{1}{2} \parallel C_1 w_1 + e_1 b_1 \parallel^2 + d_1 e_2^T \xi_2, \\ &\text{s.t.} -(C_2 w_1 + e_2 b_1) + \xi_2 \geq e_2 + R_2, \xi_2 \geq 0. \end{aligned} \tag{2}$$

$$\begin{aligned} &\min_{w_2,\, b_2} \frac{1}{2} \parallel C_2 w_2 + e_2 b_2 \parallel^2 + d_2 e_1^T \xi_1, \\ &\text{s.t.} (C_1 w_2 + e_1 b_2) + \xi_1 \geq e_1 + R_1, \xi_1 \geq 0. \end{aligned} \tag{3}$$

where C_1 and C_2 are the center matrix of GBs for layers +1 and -1, and R_1 and R_2 are the radius of GBs.

By applying the Lagrange method and utilizing the Karush–Kuhn–Tucker (K.K.T) conditions[12], deriving the dual form of problem (2) as follows:

$$\begin{aligned} &\max_{\alpha}\ \alpha^T (e_2 + R_2) - \frac{1}{2} \alpha^T G (H^T H + \delta I)^{-1} G^T \alpha, \\ &\text{s.t. } 0 \leq \alpha \leq d_1 e_2. \end{aligned} \tag{4}$$

where $G = [C_1 \quad e_1], H = [C_2 \quad e_2]$ and δI is the regularization term to avoid the matrix inversion problem.

Similarly, the Wolfe dual for (3) can be derived as follows:

$$\begin{aligned} &\max_{\gamma}\ \gamma^T (e_1 + R_1) - \frac{1}{2} \gamma^T H (G^T G + \delta I)^{-1} H^T \gamma, \\ &\text{s.t. } 0 \leq \gamma \leq d_2 e_1. \end{aligned} \tag{5}$$

As a result, the vectors $u_1 = [w_1; b_1]$representing the +1 class and $u_2 = [w_2; b_2]$ representing the −1 class can be set up as follows:

$$u_1 = -(H^T H + \delta I)^{-1} G^T \alpha \tag{6}$$

$$u_2 = (G^T G + \delta I)^{-1} H^T \gamma \tag{7}$$

From here, when a new data point x is classified based on its closest distance to two hyperplanes as described below:

$$\text{class}(x) = \arg\min_{i \in \{1,2\}} \frac{|w_i^T x + b_i|}{\|w_i\|} \tag{8}$$

2.2 Twin k-class Support Vector Machines (Twin-KSVC)

Aiming to apply multi-label classification to TSVM, Twin-KSVC was also introduced by Yitian Xu[13]. Based on the idea of the "1-versus-1-versus rest" (OvOvR) method, two separate sets of samples are selected from k classes and treated as core partitions. This algorithm maps the remaining samples to an intermediate region between these non-parallel hyperplanes, resulting in a ternary output system {-1, 0, +1}.

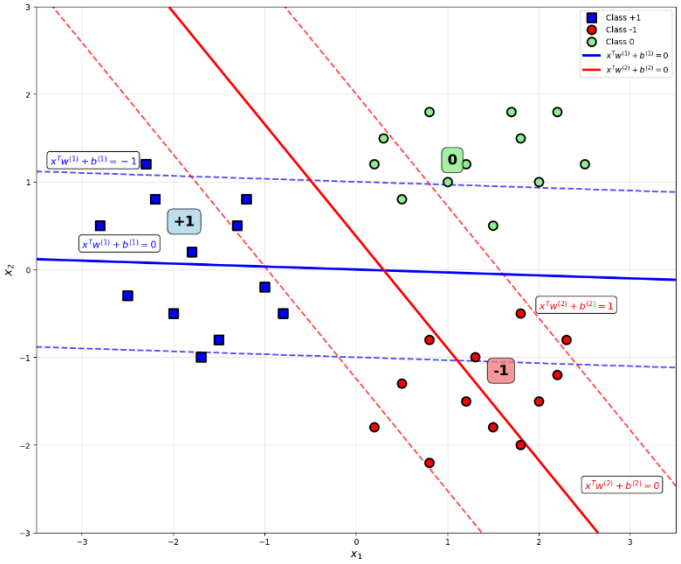

Fig. 2. Visualization of Twin-KSVC

Consider $A \in \mathbb{R}^{n_1 \times d}$ and $B \in \mathbb{R}^{n_2 \times d}$ representing two sets of centroid samples labeled +1 and -1 respectively. The remaining samples are represented by $C \in \mathbb{R}^{n_3 \times d}$ and labeled 0. The mechanism for computing these hyperplanes is obtained by solving the refined QPP pair as follows:

$$\begin{aligned} &\min_{w_1, b_1} \frac{1}{2} \| Aw_1 + e_1 b_1 \|^2 + c_1 e_2^T \xi + c_2 e_3^T \eta, \\ &\text{s.t.} \ -(Bw_1 + e_2 b_1) + \xi \geq e_2, \xi \geq 0, \\ &-(Cw_1 + e_3 b_1) + \eta \geq (1 - \varepsilon) e_3, \eta \geq 0 \end{aligned} \tag{9}$$

$$\begin{aligned} &\min_{w_2, b_2} \frac{1}{2} \| Bw_2 + e_2 b_2 \|^2 + c_3 e_1^T \xi^* + c_4 e_3^T \eta^*, \\ &\text{s.t.} \ -(Aw_2 + e_1 b_1) + \xi^* \geq e_1, \xi^* \geq 0, \\ &-(Cw_2 + e_3 b_1) + \eta^* \geq (1 - \varepsilon) e_3, \eta^* \geq 0 \end{aligned} \tag{10}$$

where η and $\eta^* \in \mathbb{R}^{n_3 \times 1}$, and $\xi \in \mathbb{R}^{n_2 \times 1}$ and $\xi^* \in \mathbb{R}^{n_1 \times 1}$

By using the Lagrange function and the K.K.T) condition, the dual formulation of (9) and (10) are established as shown below:

$$\max_{\gamma} -\frac{1}{2}\gamma^T T(U^T U)^{-1}T^T\gamma + e_4^T\gamma, \quad \text{s.t. } 0 \leq \gamma \leq E. \tag{11}$$

$$\max_{\rho} -\frac{1}{2}\rho^T P(V^T V)^{-1}P^T\rho + e_5^T\rho, \quad \text{s.t. } 0 \leq \rho \leq E^*. \tag{12}$$

for (11), the symbols are defined as $U = [\,A \quad e_1], V = [B \quad e_2], G = [C \quad e_3], T = [V \quad G], \gamma = [\alpha \quad \beta], e_4 = [e_2 \quad e_3(1-\epsilon)]$ and $E = [c_1 e_2 \quad c_2 e_3]$ and likewise with (12).

Therefore, the solution vector $u = [w_1; b_1]$ and $u^* = [w_2; b_2]$ can be obtained as:

$$u = -(U^\top U + \delta I)^{-1}(V^\top \alpha + G^\top \beta) \tag{13}$$

$$u^* = -(V^\top V + \delta I)^{-1}(U^\top \alpha + G^\top \beta) \tag{14}$$

where δI is regularization term with δ is a very small positive number.

For a new checkpoint x_i, to determine its class label through the decision function as follows:

$$f(x_i) = \begin{cases} 1, if\, x_i^\top \mu_1 + eb_1 > -1 + \epsilon \\ -1, if\, x^\top \mu_2 + eb_2 < 1 - \epsilon \\ 0, \text{otherwise} \end{cases} \tag{15}$$

This Twin-KSVC formula provides an optimal solution for multi-class classification, significantly improving computational efficiency and accuracy when processing complex datasets.

2.3 Granular Ball K-Class Twin Support Vector Classifier (GB-TWKSVC)

GB-TWKSVC, introduced by MA. Ganaie[11], is an improved version of GBTSVM, which moves from binary to multi-class classification using the OvOvR strategy, incorporating GBs representation to increase noise immunity and computational efficiency. In OvOvR, for each pair of classes, two non-parallel hyperplanes are constructed to separate the two selected classes, while the other classes are placed in an "epsilon tube" region around the hyperplanes.

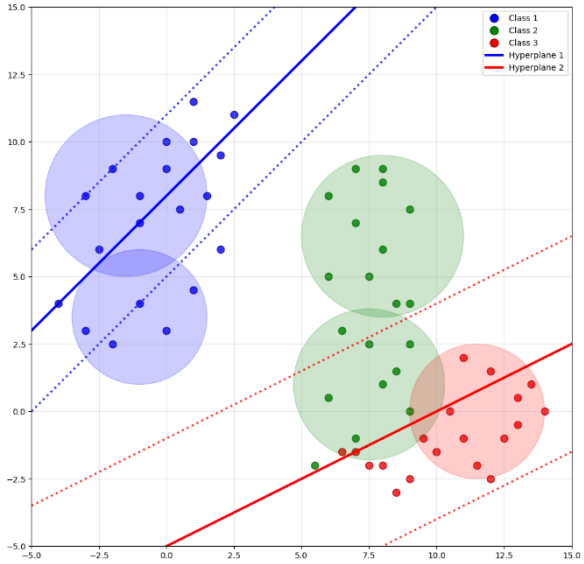

Fig. 3. Visualization for GB-TWKSVC

In GB-TWKSVC, the training dataset is denoted as $\tilde{I}$, which consists of GBs sets I_k corresponding to each class k.

$$\tilde{I} = \cup_{k=1}^{K} I_k \tag{16}$$

Then, to solve the QPP problem, GB-TWKSVC must improve GBTSVM to suit OvOvR is as follows:

$$\begin{aligned} \min_{w_i,b_i,\eta_i,\xi_j,\xi_r} & \frac{1}{2}c_1(\| w_i \|^2 + b_i^2) + \frac{1}{2}\eta_i^T\eta_i + c_2 e_j^T\xi_j + c_3 e_r^T\xi_r, \\ \text{s.t. } & C_i w_i + e_i b_i = \eta_i, \\ & -(C_j w_i + e_j b_i) + \xi_j \geq e_j + R_j, \xi_j \geq 0, \\ & | C_r w_i + e_r b_i | + \xi_r \geq e_r + R_r, \xi_r \geq 0. \end{aligned} \tag{17}$$

$$\begin{aligned} \min_{w_j,b_j,\eta_j,\xi_i,\xi_r} & \frac{1}{2}c_4(\| w_j \|^2 + b_j^2) + \frac{1}{2}\eta_j^T\eta_j + c_5 e_i^T\xi_i + c_6 e_r^T\xi_r, \\ \text{s.t. } & C_j w_j + e_j b_j = \eta_j, \\ & (C_i w_j + e_i b_j) + \xi_i \geq e_i + R_i, \xi_i \geq 0, \\ & | C_r w_j + e_r b_j | + \xi_r \geq e_r + R_r, \xi_r \geq 0. \end{aligned} \tag{18}$$

Similar to the Twin-KSVC approach, with R_1, R_2, R_3 being the set of radii of the corresponding class, the dual formulation of (17) and (18) is formulated as follows:

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\gamma^T V(H^T H)^{-1} V^T \alpha + e_4^T \gamma, \\ \text{s.t.} \quad & 0 \leq \alpha \leq F \end{aligned} \tag{19}$$

$$\begin{aligned} \max_{\rho} \quad & -\frac{1}{2}\rho^T R(G^T G)^{-1} R^T \rho + e_4^T \rho, \\ \text{s.t.} \quad & 0 \le \rho \le F^*. \end{aligned} \tag{20}$$

for (19), where $H = [\,A \quad e_1], G = [B \quad e_2], J = [C \quad e_3], V = [G \quad J], \gamma = [\alpha \quad \beta], e_4 = [e_2 + R_2 \quad e_3(1-\epsilon) + R_3]$ and $F = [c_1 e_2 \quad c_2 e_3]$ and the same calculation for (20).

In this way, the decision vectors $\vartheta_1 = [w_1; b_1]$ and $\vartheta_2 = [w_2; b_2]$ can be calculated by the formula:

$$\vartheta_1 = -(H^\top H)^{-1}(G^\top \alpha + J^\top \mu) \tag{21}$$

$$\vartheta_2 = -(H^\top H)^{-1}(G^\top \alpha + J^\top \mu) \tag{22}$$

The classification of new sample $z \in R^n$ is done based on the closest distance to the hyperplanes:

$$h(z) = \min_{1,2} \{\delta^1(z), \delta^2(z)\} \tag{23}$$

where $\delta^1(z) = | z^T \omega^{(1)} + b^{(1)} |, \delta^2(z) = | z^T \omega^{(2)} + b^{(2)} |$ with $|.|$ represents the normal distance of point z to the hyperplanes.

3 Proposed Model: Twin Support Vector Machine for Multi-class Classification (GBTSVM_MCC)

3.1 Model Overview

The Granular Ball Twin Support Vector Machine for Multi-class Classification (GBTSVM_MCC) introduces an innovative approach to handle multi-class problems by integrating Twin Support Vector Machines (TSVM) with Granular Ball Computing. This model extends the binary classification framework of GBTSVM to multi-class scenarios through two well-established decomposition strategies: One-vs-One (OvO) and One-vs-Rest (OvR). By utilizing granular balls as input units, defined by their center and radius, GBTSVM_MCC enhances noise resistance and computational efficiency, making it suitable for large-scale datasets.

In the OvO strategy, GBTSVM_MCC trains $K(K-1)/2$ binary GBTSVM classifiers for a dataset with K classes, each addressing a unique pair of classes (e.g., class i vs. class j), using GBs specific to those classes. The final classification for a new data point is determined by a voting mechanism, where the class with the most votes across all pairwise classifiers is selected. Conversely, the OvR strategy trains K binary GBTSVM classifiers, each distinguishing one class from all others (e.g., class i vs. not class i), again leveraging granular balls for the respective groups. The decision for a new data point is based on the highest confidence score or distance from the separating hyperplanes among the classifiers. These strategies ensure flexible and efficient multi-class classification tailored to diverse dataset structures.

To visually illustrate the two proposed methods, Figure 4 shows the comparison between the OvO and OvR methods when applied to the GBTSVM model. In the figure, the 3-class dataset is represented as GBs corresponding to each data class. For the OvO method, the non-parallel hyperplanes pass through the corresponding regions

representing each binary classification pair. As for the OvR method, it is represented by non-parallel hyperplanes, in which one plane represents a class chosen as the positive class and the remaining planes represent the separation of that class from the set of all remaining classes.

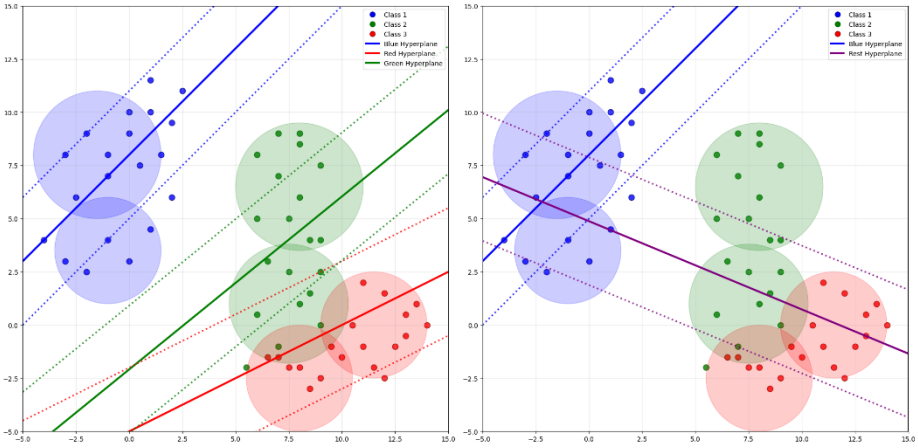

Fig. 4. Comparison of multi-class classification strategies: (Left) One-vs-One (OvO), (Right) One-vs-Rest (OvR). The colored GBs represent decision for three-class classification problem.

3.2 Granular Ball Generation

The first stage of the proposed model is to generate Granular Ball using the K-means algorithm. This process supports the generation of input data for the system. The calibration is performed based on two main parameters: purity threshold and minimum number in GB. The algorithm is implemented in the following steps.

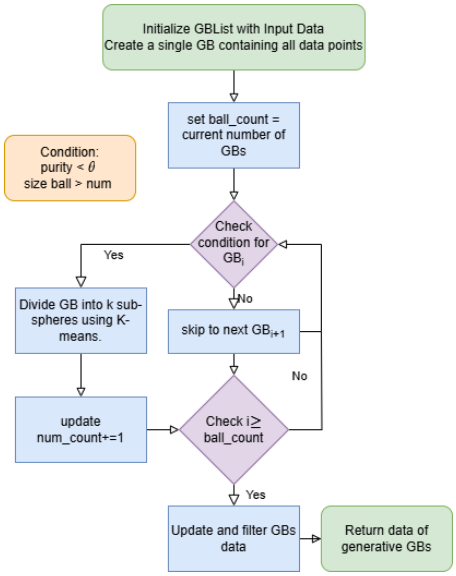

Fig. 5. Process of the GB generation.

1. Create an initial GBs containing a single GB, where this GB contains the entire set of data points.
2. Loop through each GB in the GBs, check the following two conditions:
   a. purity < θ
   b. number of points in the GB > num
   If either condition is not met, move on to the next GB.
3. If both the above conditions are true, perform the partition of the GB into k sub-GBs by applying the K-means algorithm. Then, update the GBs by replacing the original GB with the new sub-GBs and update the total number of GBs in the list.
4. Continue performing steps 2 and 3 until there are no more GBs in the GBs that satisfy the condition for partitioning.
5. Calculate and update the center, radius, label for each final GB, then filter and return the list of generated GBs.

The algorithm of granular ball generation is shown in detail as follows:

Algorithm 1 Granular Ball Generation

Input: Dataset $\mathcal{D} = (\boldsymbol{x}_i, y_i)_{i=1}^{N}$,where $\boldsymbol{x}_i \in R^d$ and label y_i
Purity threshold θ, minimum ball size n_{min}
Output: List of granular balls $\mathcal{G} = \{GB_j\} = \{(c_j, r_j, l_j)\}_{j=1}^{M}$

Initialize $\mathcal{G} \leftarrow \{GB_1 = \mathcal{D}\}$
for each GB_j in $\mathcal{G}$ **do**
  if $purity(GB_j) <$ θ **and** $|GB_j| > n_{min}$ **then**
    $\mathcal{G} \leftarrow (\mathcal{G}\{GB_j\}) \cup \mathrm{K-Means(GB_j, k)}$
  else
    contiune
end for
for each GB_j in $\mathcal{G}$ **do**
  $c_j \leftarrow \frac{1}{|GB_j|}\sum_{\boldsymbol{x}_i \in GB_j} \boldsymbol{x}_i$
  $r_j \leftarrow \frac{1}{|GB_j|}\sum_{\boldsymbol{x}_i \in GB_j} |\boldsymbol{x}_i - c_j|_2$
  $l_j \leftarrow arg\ \max_y |\{\boldsymbol{x}_i \in GB_j : y_i = y\}|$
end for
$\mathcal{G} \leftarrow \{GB_j \in \mathcal{G} : purity(GB_j) \geq \theta \wedge |GB_j| \geq n_{min}\}$
return $\mathcal{G}$

Each granular ball is characterized by a centroid and a radius. The centroid of the granular ball, denoted by c_j, is computed as the average of all feature vectors belonging to that granular ball, forming a d-dimensional vector with d being the number of features. The radius r_j of the granular ball is defined as the average of the L2 (Euclidean) distances from each data point to the centroid cj. In the final step, each granular ball is

labeled according to the class that is the majority in the set of data points belonging to that ball, i.e., the label l_j is the label that appears most frequently in the granular ball.

3.3 Multi-class Classification Strategy

OvO Method.

With data set $\mathcal{D}$ having k classes belonging to label space $C = C_1, C_2, \ldots, C_k$, construct a total of $\binom{K}{2} = \frac{K(K-1)}{2}$ corresponding GBTSVM binary classifiers for each pair. Specifically, $\forall i, j \in \{1,2,\ldots,k\}$ and $i \neq$ j, have the definition of the classifier $f_{i,j}: R \rightarrow \{C_i, C_j\}$ trained on the data subset:

$$\mathcal{D}_{i,j} = \left\{(x_k, y_k) \in \mathcal{D}: y_k \in \{C_i, C_j\}\right\} \tag{24}$$

where $\mathcal{D}_{i,j} \subset \mathcal{D}$ contains only samples belonging to class pair (C_i, C_j).

Each classifier $f_{i,j}$ is optimized to solve the GBTSVM binary classification problem as follows:

$$\begin{aligned} &min_{w_i j^1, b_i j^1} \tfrac{1}{2}\left||C_i w_i j^1 + e_i b_i j^1|\right|^2 + d_1 e_j^T \xi_j^{ij} \\ &s.t. -\left(C_j w_i j^1 + e_j b_i j^1\right) + \xi_j^{ij} \geq e_j + R_j, \xi_j^{ij} \geq 0 \end{aligned} \tag{25}$$

$$\begin{aligned} &min_{w_i j^2, b_i j^2} \tfrac{1}{2}\left|\left|C_j w_i j^2 + e_j b_i j^2\right|\right|^2 + d_2 e_i^T \xi_i^{ij} \\ &s.t. (C_i w_i j^2 + e_i b_i j^2) + \xi_i^{ij} \geq e_i + R_i, \xi_i^{ij} \geq 0 \end{aligned} \tag{26}$$

where $C_i, C_j \in R^{p_i \times m}, R^{p_j \times m}$ is the center matrix of the GBs of class i and j with p_i, p_j are the number of GBs generated for layers i and j, R_i, R_j are the radius vectors of the corresponding GBs, $e_i, e_j \in R^{p_i}, R^{p_j}$ are the vector of one and ξ^{ij} is slack variables for $f_{i,j}$.

From this expression, through the Lagrange function and the K.K.T condition, we obtain the dual form for (25) and (26) which is explained as follows:

$$\begin{aligned} &max_{\alpha^{ij}} \alpha^{ij,T}\left(e_j + R_j\right) - \tfrac{1}{2}\alpha^{ij,T} G^{ij} (H^{ij,T} H^{ij} + \delta I)^{-1} G^{ij,T} \alpha^{ij} \\ &s.t. 0 \leq \alpha^{ij} \leq d_1 e_j \end{aligned} \tag{27}$$

$$\begin{aligned} &max_{\gamma^{ij}} \gamma^{ij,T}(e_i + R_i) - \tfrac{1}{2}\gamma^{ij,T} H^{ij} (G^{ij,T} G^{ij} + \delta I)^{-1} H^{ij,T} \gamma^{ij} \\ &s.t. 0 \leq \gamma^{ij} \leq d_2 e_i \end{aligned} \tag{28}$$

where $H^{ij} = [C_i, e_i], G^{ij} = \left[C_j, e_j\right]$.

Once the dual form is in place, when a new data point x is given, the computation of the hyperplanes is solved by:

$$\delta_1^{ij}(x) = \frac{|w_i j^{1,T} x + b_i j^1|}{||w_i j^1||} \tag{29}$$

$$\delta_2^{ij}(x) = \frac{|w_i j^{2,T} x + b_i j^2|}{||w_i j^2||} \tag{30}$$

Subsequently, the prediction output for each binary classifier f_{ij}, which distinguishes between class i and class j in the One-vs-One multi-class classification framework, is formulated as:

$$pred^{ij}(x) = \begin{cases} i, if\, \delta_1^{ij}(x) < \delta_2^{ij}(x) \\ j, otherwise \end{cases} \tag{31}$$

Finally, upon completion of all pairwise binary classification computations, the ultimate class label assignment for the input sample x is determined through a comprehensive voting mechanism that aggregates the individual predictions from each binary classifier:

$$class(x) = argmax_{k=1,\dots,K} \sum_{j \neq k} I(pred^{kj}(x) = k) \tag{32}$$

OvR Method.

Unlike OvO, OvR strategy in multiclass classification problem sets up K independent GBTSVM binary classifiers $f_1, f_2, \dots, f_K$ corresponding to K classes in the dataset. Specifically, given a data set with K classes $C = C_1, C_2, \dots, C_K$ each classifier f_i with $i \in 1,2, \dots, K$ is trained to solve the binary classification problem $f_i: X \to \{+1, -1\}$in which the positive class (+1) includes samples belonging to class C_i and the negative class (-1) includes samples belonging to the set of all remaining classes $U_{i \neq j} C_j$. For each classifier f_i, the training set is constructed according to the formula:

$$\mathcal{D}_i = \left\{ \left(x_j, y_j^{(i)}\right) : j = 1,2, \dots, n \right\} \tag{33}$$

The f_i classifier is optimized to solve the GBTSVM binary classification problem through the following problem:

$$\begin{gathered} min_{w_i^1, b_i^1} \frac{1}{2} \left| |C_i w_i^1 + e_i b_i^1| \right|^2 + d_1 e_{\neg i}^T \xi_{\neg i}^i \\ s.t. -(C_{\neg i} w_i^1 + e_{\neg i} b_i^1) + \xi_{\neg i}^i \geq e_{\neg i} + R_{\neg i}, \xi_{\neg i}^i \geq 0 \end{gathered} \tag{34}$$

$$\begin{gathered} min_{w_i^2, b_i^2} \frac{1}{2} \left| |C_{\neg i} w_i^2 + e_{\neg i} b_i^2| \right|^2 + d_2 e_i^T \xi_i^i \\ s.t. (C_i w_i^2 + e_i b_i^2) + \xi_i^i \geq e_i + R_i, \xi_i^i \geq 0 \end{gathered} \tag{35}$$

where $C_{\neg i}$ is the center matrix of all GBs not belonging to class i and $R_{\neg i}$ is the corresponding radius vector

Similar to OvO, the optimization problems are also passed through the Lagrange function and the K.K.T condition to obtain the dual form of (34) and (35) as follows:

$$\begin{gathered} max_{\alpha^i} \alpha^{i,T} (e_{\neg i} + R_{\neg i}) - \frac{1}{2} \alpha^{i,T} G^i (H^{i,T} H^i + \delta I)^{-1} G^{i,T} \alpha^i \\ s.t. 0 \leq \alpha^i \leq d_1 e_{\neg i} \end{gathered} \tag{36}$$

$$\begin{gathered} max_{\gamma^i} \gamma^{i,T} (e_i + R_i) - 1/2 \gamma^{i,T} H^i (G^{i,T} G^i + \delta I)^{-1} H^{i,T} \gamma^i \\ s.t. 0 \leq \gamma^i \leq d_2 e_i \end{gathered} \tag{37}$$

where $H^i = [C_i, e_i], G^i = [C_{\neg i}, e_{\neg i}]$.

For a newly introduced data point x, the calculation of the confidence value corresponding to each class i in the set of K classes of the classification model will be performed:

$$score_i(x) = max\left\{\frac{-\left(w_i^{1,T}x+b_i^1\right)}{\left\|w_i^1\right\|}, \frac{\left(w_i^{2,T}x+b_i^2\right)}{\left\|w_i^2\right\|}\right\} \tag{38}$$

From here, the final prediction result is given as follows:

$$class(x) = argmax_{i=1,\ldots,K} score_i(x) \tag{39}$$

3.4 Algorithm Design and Time Complexity

The GBTSVM_MC algorithm employs a combination of K-Means clustering and granular ball generation to efficiently partition and classify multi-class data. The clustering method used in GBTSVM_MC incorporates K-Means to iteratively partition data into clusters based on purity thresholds and minimum size constraints. Each cluster is formed to ensure a high degree of label homogeneity, resulting in final granular balls that accurately reflect class separations.

The GBTSVM_MC framework is structured as follows:

- **Granular Ball Generation:** The process begins with K-Means clustering to partition the dataset into granular balls (**Algorithm 1**). This step ensures that the data is split into homogeneous subsets based on a purity threshold and a minimum number of data points per ball. Each granular ball is defined by its centroid, radius, and majority label. These granular balls serve as the foundation for constructing hyperplanes in the classification phase, reducing computational complexity and enhancing noise resistance.
- **Multi-class Classification Strategies**: GBTSVM_MC adopts two decomposition strategies, One-vs-One (OvO) and One-vs-Rest (OvR), to handle multi-class data. In the OvO approach, the algorithm constructs binary classifiers for each pair of classes, training on granular balls specific to those pairs, and determines the final class through a voting mechanism. In the OvR approach, a binary classifier is trained for each class against all others, using confidence scores to assign the final label. These strategies ensure flexibility and efficiency in managing diverse dataset structures (**Algorithm 2** and **Algorithm 3**).
- **GBTSVM_MC Classification**: Using the granular balls, the GBTSVM_MC algorithm constructs hyperplanes by solving smaller quadratic programming problems (QPPs) for each binary subproblem. For each classifier, whether in OvO or OvR mode, centroids and radii of the granular balls are extracted to formulate the QPPs and derive hyperplane parameters (w1, b1) and (w2, b2). These hyperplanes are then used to classify new data points, either through a voting system in OvO or by selecting the highest confidence score in OvR.

Algorithm 2 GBTSVM_MC with OvO Strategy

Input: Granular Balls $G = GB_j = \left(c_j, r_j, l_j\right)_{j=1}^{M}$ from K classes
Hyperparameters: d_1, d_2, δ
Output: Trained classifiers $F_{OvO} = \left\{f_{i,j} : i, j \in \{1, \dots, K\}, i \neq j\right\}$
Decision function class(x)

Initialize $F_{OvO} \leftarrow \emptyset$
for $i = 1$ **to** K **do**
  for $j = i + 1$ **to** K **do**
    Extract $G_{i,j} \leftarrow \left\{GB_k \in G : l_k \in \{i, j\}\right\}$
    Construct C_i, C_j, R_i, R_j from $G_{i,j}$
    Solve dual problems (27) and (28) to get α^{ij}, γ^{ij}
    Compute hyperplane parameters:
      $u^{1ij} \leftarrow -\left(H^{ij,T}H^{ij} + \delta I\right)^{-1} G^{ij,T}\alpha^{ij}$
      $u^{2ij} \leftarrow \left(G^{ij,T}G^{ij} + \delta I\right)^{-1} H^{ij,T}\gamma^{ij}$
    $F_{OvO} \leftarrow F_{OvO} \cup f_{i,j}\left(u^{1ij}, u^{2ij}\right)$
  end for
end for
return F_{OvO}

Prediction for new sample x:
for each $f_{i,j} \in F_{OvO}$ **do**
  Compute $pred^{ij}(x)$ using equation (31)
end for
return $class(x) = argmax_{k=1,\dots,K} \sum_{j \neq k} I\left(pred^{kj}(x) = k\right)$

Algorithm 3 GBTSVM_MC with OvR Strategy

Input: Granular Balls $G = GB_j = \left(c_j, r_j, l_j\right)_{j=1}^{M}$ from K classes
Hyperparameters: d_1, d_2, δ
Output: Trained classifiers $F_{OvR} = \left\{f_i : i \in \{1, \dots, K\}\right\}$
Decision function class(x)

Initialize $F_{OvR} \leftarrow \emptyset$
for $i = 1$ **to** K **do**
  Construct positive class: $G_i \leftarrow GB_k \in G : l_k = i$
  Construct negative class: $G_{\neg i} \leftarrow GB_k \in G : l_k \neq i$
  Form $C_i, C_{\neg i}, R_i, R_{\neg i}$ from G_i and $G_{\neg i}$
  Solve dual problems (36) and (37) to get α^i, γ^i
  Compute hyperplane parameters:
    $u^{1i} \leftarrow -\left(H^{i,T}H^{i} + \delta I\right)^{-1} G^{i,T}\alpha^{i}$
    $u^{2i} \leftarrow \left(G^{i,T}G^{i} + \delta I\right)^{-1} H^{i,T}\gamma^{i}$
  $F_{OvR} \leftarrow F_{OvR} \cup f_i\left(u^{1i}, u^{2i}\right)$

9: **end for**

10: **return** F_{OvR}

Prediction for new sample x:

12: **for** each $f_i \in F_{OvR}$ **do**

13: Compute $score_i(x)$ using equation (31)

14: **end for**

15: **return** $class(x) = argmax_{i=1,\ldots,K} score_i(x)$

The computational complexity of GBTSVM_MC varies significantly depending on the multiclass extension strategy employed. This section provides a detailed analysis of both OvO and OvR approaches, examining their training and prediction complexities.

- **OvO Strategy**:
  - **Training Phase**: The algorithm constructs $\binom{K}{2} = \frac{K(K-1)}{2}$ binary GBTSVM classifiers, where each classifier is trained exclusively on granular balls from two specific classes. Given that the average number of granular balls per class pair is $\bar{M}_{ij}$, the overall training complexity is expressed as:
$$T_{OvO}^{train} = O\left(\frac{K(K-1)}{2} \times \bar{M}_{ij}^3\right) = O(K^2\bar{M}^3)$$
  - **Prediction Phase**: During inference, each test sample must be evaluated by all $\frac{K(K-1)}{2}$ binary classifiers to determine the final class assignment through majority voting. This results in a prediction complexity of:
$$T_{OvO}^{pred} = O(K^2\bar{M})$$
- **OvR Strategy**:
  - Training Phase: The method constructs K binary classifiers, where each classifier utilizes the complete set of M granular balls. The training complexity is therefore:
$$T_{OvR}^{train} = O(K \times M^3) = O(KM^3)$$
  - Prediction Phase: Classification requires evaluation by only K classifiers, yielding a prediction complexity of:
$$T_{OvR}^{pred} = O(KM)$$

To provide a comprehensive comparison of computational complexity between GBTSVM_MC and SOTA multiclass classification methods, Table 3 is presented in detail and clearly as follows:

Table 1. comparison of computational complexity between GBTSVM_MC, GB-TWKSVC and Twin-KSVC

Method	Training Complex	Prediction Complex	Number of Classifiers
GBTSVM_MC (OvO)	$O(K^2\bar{M}^3)$	$O(K^2\bar{M})$	$\frac{K(K-1)}{2}$
GBTSVM_MC (OvR)	$O(KM^3)$	$O(KM^3)$	K

GB-TWKSVC	$O(K^3M^3)$	$O(K^3M)$	$\binom{K}{2}$
Twin-KSVC	$O(K^2N^3)$	$O(K^2\mathrm{N})$	$\binom{K}{2}$

The analysis shows that GBTSVM_MC with both proposed strategies achieve the most favorable complexity configuration, especially for scenarios with large number of layers.

4 Experiments Result

4.1 Experimental setup

The experiment was conducted on a personal computer with an Intel Core i5-11300H CPU configuration at 3.20GHz, 24GB RAM and Windows 11 operating system. The programming environment used Python 3.11.7 with supporting libraries such as scikit-learn, NumPy and pandas. The datasets were randomly divided into 80% for the training set and 20% for the test set, ensuring the same ratio between classes. The GBTSVM_MC model fine-tuning process was performed through a 5-fold cross-validation method on the training set. The main hyperparameters tuned are listed in **Table 2**:

Table 2. List of Parameters, Symbols, and Tuning Intervals

Parameters	Symbols	Tuning
Regularization coefficients	c	0.01, 0.025, 0.05, 0.075, 0.1, 0.25, 0.5
Margin tolerance limit	ϵ	0.01, 0.025, 0.05, 0.075, 0.1
Granular ball size	num	1, 2, 3, 4, 5
Purity threshold	pur	0.85, 0.895, 0.925, 0.97, 0.985

4.2 Dataset

1. To evaluate the performance of GBTSVM_MC, experiments were conducted on 10 multi-class datasets sourced from the UC Irvine Machine Learning Repository[1] and Kaggle[2]. These datasets were carefully selected from various domains and exhibit diversity in sample size, feature dimensionality, and number of classes. Detailed information about each dataset is summarized in **Table 3**. This diverse selection enables a comprehensive assessment of GBTSVM_MC's capabilities across different levels of complexity, including variations in the number of samples, features, and classes.

Table 3. Overview of multi-class datasets utilized in these experiments

Dataset	Samples	Features	Classes
Hayes-roth	132	5	3
Iris	150	4	3
Teaching Evaluation	151	5	3
Image-segmentation	210	19	7
Seeds	210	7	3
Glass	214	9	6

[1] https://archive.ics.uci.edu
[2] https://www.kaggle.com

Ecoli	327	7	5
Dermatology	358	4	6
Balance	625	4	3
Global Cancer	50000	13	5

Specially, the Global Cancer dataset obtained from Kaggle is utilized to assess the scalability and computational efficiency of GBTSVM_MC. This dataset is partitioned into three different subset sizes—1,000, 10,000, and 50,000 samples—to systematically evaluate the algorithm's performance and computational scalability across varying data volumes.

4.3 Result

Table 4. Comparison of GBTSVM_MC, GB-TWKSVC, and Twin-KSVC on different datasets

Dataset		GBTSVM_MC(OvO) (c_1, c_2, ϵ_1, ϵ_2, num, pur)	GBTSVM_MC(OvR) (c_1, c_2, ϵ_1, ϵ_2, num, pur)	GB-TWKSVC (c_1, c_2, ϵ_1, ϵ_2, num, pur)	Twin-KSVC (c_1, c_2, ϵ_1, ϵ_2)
Hayes-roth	Accuracy (%)	74.19	**74.2**	53.57	57.06
	Time(s)	0.0352	0.0130	0.0732	0.0741
	Parameters	0.01, 0.01, 0.01, 0.01, 3, 0.895	0.01, 0.01, 0.01, 0.01, 3, 0.895	0.01, 0.01, 0.025, 0.025, 2, 0.97	0.01, 0.01, 0.075, 0.05
Iris	Accuracy (%)	86.67	93.33	**98.34**	95.32
	Time(s)	0.0189	0.0195	0.0332	0.0352
	Parameters	0.01, 0.01, 0.01, 0.01, 2, 0.985	0.01, 0.01, 0.01, 0.01, 2, 0.97	0.01, 0.01, 0.025, 0.025, 3, 0.985	0.01, 0.01, 0.025, 0.05
Teaching Evalua-tion	Accuracy (%)	62.5	68.75	**76.43**	69.35
	Time(s)	0.0261	0.0333	0.0525	0.0543
	Parameters	0.01, 0.01, 0.01, 0.01, 3, 0.97	0.01, 0.01, 0.01, 0.01, 3, 0.925	0.01, 0.01, 0.025, 0.025, 3, 0.97	0.01, 0.01, 0.025, 0.025
Image-segmen-tation	Accuracy (%)	**92.85**	90.47	90.43	89.65
	Time(s)	0.0840	0.0530	0.3967	0.5487
	Parameters	0.01, 0.01, 0.01, 0.01, 1, 0.97	0.01, 0.01, 0.01, 0.01, 2, 0.925	0.01, 0.01, 0.05, 0.01, 2, 0.925	0.01, 0.01, 0.025, 0.025
Seeds	Accuracy (%)	88.09	**100**	98.23	95.12
	Time(s)	0.0312	0.0266	0.0560	0.0624
	Parameters	0.01, 0.01, 0.01, 0.01, 2, 0.97	0.01, 0.01, 0.01, 0.01, 2, 0.97	0.01, 0.01, 0.025, 0.01, 3, 0.985	0.01, 0.01, 0.05, 0.025
Glass	Accuracy (%)	82.14	**84.52**	75.34	67.93
	Time(s)	0.1615	0.0525	0.3205	0.4710
	Parameters	0.01, 0.01, 0.01, 0.01, 2, 0.925	0.01, 0.01, 0.01, 0.01, 2, 0.985	0.01, 0.01, 0.01, 0.01, 3, 0.925	0.01, 0.01, 0.05, 0.025
Ecoli	Accuracy (%)	82.53	82.52	**92.54**	87.32
	Time(s)	0.2992	0.0849	0.8083	1.728
	Parameters	0.01, 0.01, 0.01, 0.01, 3, 0.85	0.01, 0.01, 0.01, 0.01, 3, 0.85	0.01, 0.01, 0.025, 0.025, 2, 0.085	0.01, 0.01, 0.01, 0.025

Derma-tology	Accuracy (%) Time(s) Parameters	**98.5** 0.1075 0.01, 0.01, 0.01, 0.01, 4 0.985	96.26 0.0708 0.01, 0.01, 0.01, 0.01, 4, 0.895	91,75 0.4103 0.01, 0.01, 0.01, 0.01, 2, 0.97	81.21 0.719 0.01, 0.01, 0.075, 0.05
Balance	Accuracy (%) Time(s) Parameters	**94.21** **0.0533** 0.01, 0.01, 0.01, 0.01, 4, 0.97	92.48 0.0733 0.01, 0.01, 0.01, 0.01, 1, 0.85	89.62 0.1405 0.01, 0.01, 0.01, 0.01, 2 0.97	87.52 0.2676 0.01, 0.01, 0.01, 0.01
Global Cancer – 1K	Accuracy (%) Time(s) Parameters	**63.07** 0.0395 0.01, 0.01, 0.01, 0.01, 4, 0.97	62.24 0.0505 0.01, 0.01, 0.01, 0.01, 4, 0.97	60.16 0.0518 0.01, 0.01, 0.01, 0.01, 4, 0.97	57.58 0.2628 0.01, 0.01, 0.01, 0.025
Global Cancer – 10K	Accuracy (%) Time(s) Parameters	**61.53** 0.8314 0.01, 0.01, 0.01, 0.01, 3, 0.985	60.85 1.5107 0.01, 0.01, 0.01, 0.01, 2, 0.97	57.38 1.6211 0.01, 0.01, 0.01, 0.01, 2, 0.925	54.65 3.2571 0.01, 0.01, 0.05, 0.05
Global Cancer – 50K	Accuracy (%) Time(s) Parameters	50.28 80.2156 0.01, 0.01, 0.01, 0.01, 1, 0.85	**50.71** 158.9295 0.01, 0.01, 0.01, 0.01, 2, 0.895	49.23 155.5041 0.01, 0.01, 0.01, 0.01, 2, 0.895	42.78 186.4782 0.01, 0.01, 0.1, 0.1

GBTSVM_MC shows competitive and sometimes superior performance compared to baseline and SOTA methods. Specifically, on the Seeds dataset, GBTSVM_MC(OvR) achieved perfect accuracy of 100%, outperforming GB-TWKSVC (98.23%) and Twin-KSVC (95.12%). On the Dermatology dataset, GBTSVM_MC(OvO) achieved 98.5%, significantly higher than GB-TWKSVC (91.75%) and Twin-KSVC (81.21%). However, GB-TWKSVC still maintained its superiority on certain datasets, such as Iris (98.34%) and Teaching Evaluation (76.43%), demonstrating the suitability of the OvOvR strategy for problems with specific characteristics.

On subsets of the Global Cancer dataset, GBTSVM_MC shows better scalability. When the dataset size increases from 1K to 50K samples, GBTSVM_MC still maintains relatively stable performance and acceptable training time (80.2156s for OvO and 158.9295s for OvR), while Twin-KSVC takes 186.4782s with lower accuracy (42.78%).

Above all, thanks to its lower complexity compared to the remaining models, GBTSVM_MC also shows a clear advantage in execution time. When looking at all the datasets used, it is easy to see that both OvO and OvR strategies of GBTSVM_MC have significantly faster training times than GB-TWKSVC and Twin-KSVC. For example, on the Image-segmentation dataset, GBTSVM_MC(OvR) only takes 0.0530s compared to 0.3967s of GB-TWKSVC and 0.5487s of Twin-KSVC. This is further demonstrated in **Fig. 6**.

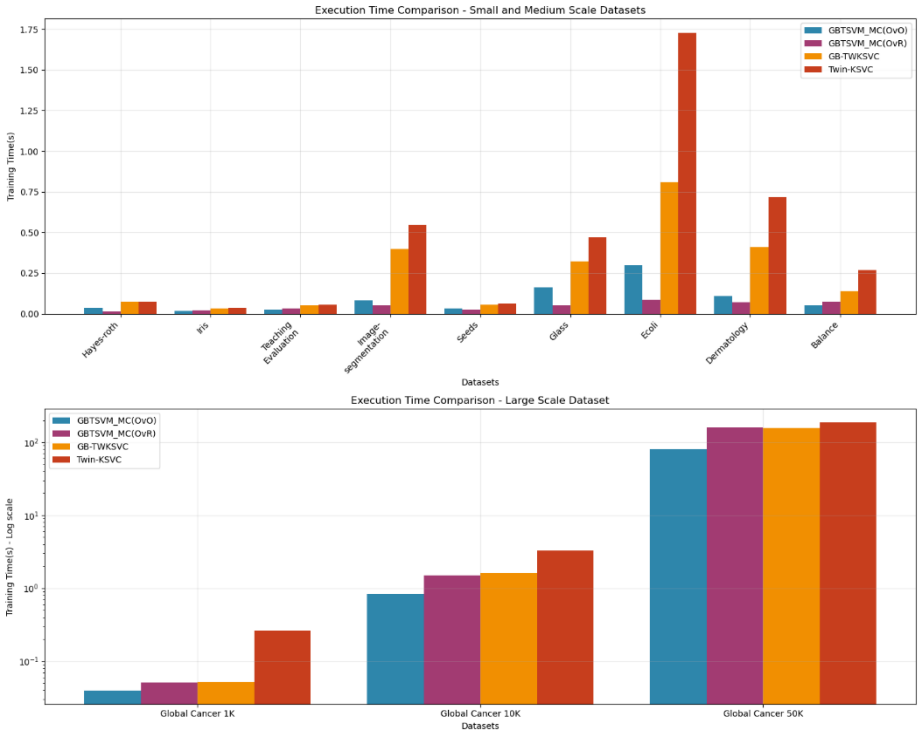

Fig. 6. Execution Time Comparison between GBTSVM_MC, GB-TWKSVC and Twin-KSVC Models on Different Datasets.

The experimental results confirm the effectiveness of GBTSVM_MC in balancing accuracy and computational efficiency. The proposed method not only achieves competitive performance in accuracy but also significantly outperforms in processing time, which is especially important in practical applications with real-time or large-scale data processing requirements.

4.4 Discussion

Experimental results show that GBTSVM_MC achieves competitive performance and often outperforms baseline and SOTA methods. On the Seeds dataset, GBTSVM_MC(OvR) achieves perfect accuracy of 100%, outperforming GB-TWKSVC (98.23%) and Twin-KSVC (95.12%). Similarly, on the Dermatology dataset, GBTSVM_MC(OvO) achieves 98.5%, significantly higher than GB-TWKSVC (91.75%) and Twin-KSVC (81.21%). In particular, GBTSVM_MC shows a clear advantage in processing time with computational complexity $O(K^2\overline{M}^3)$ for OvO and $O(KM^3)$ for OvR, significantly lower than GB-TWKSVC ($O(K^3M^3)$). For example, on the Image-segmentation dataset, GBTSVM_MC(OvR) takes only 0.0530s compared to 0.3967s of GB-TWKSVC and 0.5487s of Twin-KSVC.

Experiments on subsets of the Global Cancer dataset demonstrate the excellent scalability of GBTSVM_MC. When the dataset size increases from 1K to 50K samples, GBTSVM_MC still maintains relatively stable performance and acceptable training

time (80.2156s for OvO and 158.9295s for OvR), while Twin-KSVC takes 186.4782s with lower accuracy (42.78%). This demonstrates that using granular balls as the input unit not only improves the anti-noise ability but also greatly enhances the ability to handle large-scale data.

Although GBTSVM_MC shows many advantages, GB-TWKSVC still maintains its superiority on some datasets such as Iris (98.34%) and Teaching Evaluation (76.43%), demonstrating the suitability of the OvOvR strategy for problems with specific characteristics. The results show that no strategy is absolutely superior in all cases - OvR is generally better on small datasets such as Seeds and Iris, while OvO shows superiority on more complex datasets such as Dermatology. This shows that the choice of method still depends on the specific characteristics of the data and application requirements.

5 Conclusion

This paper proposes GBTSVM_MC, a new framework that efficiently combines Twin Support Vector Machines with Granular Ball Computing to solve multiclass classification problems. The proposed method uses two decomposition strategies OvO and OvR instead of the more complex OvOvR strategy, making three main contributions: improved computational efficiency with complexity $O(K^2\overline{M}^3)$ for OvO and $O(KM^3)$ for OvR, significantly lower than GB-TWKSVC ($O(K^3M^3)$); flexible multiclass handling with the ability to adapt to diverse dataset structures; and enhanced noise immunity through granular ball representation. Experiments on 10 multiclass datasets show that GBTSVM_MC achieves competitive and often superior performance, especially in terms of processing time with excellent scalability on datasets up to 50K samples.

Future research can focus on optimizing the granular ball generation algorithm to improve performance on datasets with special characteristics and studying hybrid strategies that combine the advantages of OvO, OvR, and OvOvR. In particular, it is necessary to develop techniques to improve the original GBTSVM model to optimize processing time for large-scale data, including applying parallel computing algorithms, optimizing memory, and developing granular ball generation methods that dynamically adapt to dataset size. Applying GBTSVM_MC to practical problems in medicine, computer vision, and natural language processing, along with studying the integration of deep learning to improve feature representation, are also potential research directions.

Acknowledgments. This research has received no external funding.

Disclosure of Interests. The authors confirm that no financial interests or personal relationships could have influenced the conduct, outcomes, or interpretation of this work.

References

1. Pisner DA, Schnyer DM (2020) Chapter 6 - Support vector machine. In: Mechelli A, Vieira S (eds) Machine Learning. Academic Press, pp 101–121

2. Shah SMS, Shah FA, Hussain SA, Batool S (2020) Support Vector Machines-based Heart Disease Diagnosis using Feature Subset, Wrapping Selection and Extraction Methods.

Computers & Electrical Engineering 84:106628. https://doi.org/10.1016/j.compeleceng.2020.106628

3. Hosseinzadeh M, Rahmani AM, Vo B, et al (2021) Improving security using SVM-based anomaly detection: issues and challenges. Soft Comput 25:3195–3223. https://doi.org/10.1007/s00500-020-05373-x

4. Chandra MA, Bedi SS (2021) Survey on SVM and their application in image classification. Int j inf tecnol 13:1–11. https://doi.org/10.1007/s41870-017-0080-1

5. Huang H, Wei X, Zhou Y (2018) Twin support vector machines: A survey. Neurocomputing 300:34–43. https://doi.org/10.1016/j.neucom.2018.01.093

6. A. Quadir, M. Sajid, M. Tanveer (2024) Granular Ball Twin Support Vector Machine. IEEE Transactions on Neural Networks and Learning Systems 1–10. https://doi.org/10.1109/TNNLS.2024.3476391

7. Psaltakis G, Rogdakis K, Loizos M, Kymakis E (2024) One-vs-One, One-vs-Rest, and a novel Outcome-Driven One-vs-One binary classifiers enabled by optoelectronic memristors towards overcoming hardware limitations in multiclass classification. Discov Mater 4:7. https://doi.org/10.1007/s43939-024-00077-7

8. Kumar Sahu S, Pandey M (2023) An optimal hybrid multiclass SVM for plant leaf disease detection using spatial Fuzzy C-Means model. Expert Systems with Applications 214:118989. https://doi.org/10.1016/j.eswa.2022.118989

9. Mehmood Z, Asghar S (2021) Customizing SVM as a base learner with AdaBoost ensemble to learn from multi-class problems: A hybrid approach AdaBoost-MSVM. Knowledge-Based Systems 217:106845. https://doi.org/10.1016/j.knosys.2021.106845

10. Don DR, Iacob IE (2020) DCSVM: fast multi-class classification using support vector machines. Int J Mach Learn & Cyber 11:433–447. https://doi.org/10.1007/s13042-019-00984-9

11. Ganaie MA, Ahire V, Girard A (2025) Granular Ball K-Class Twin Support Vector Classifier. Pattern Recognition 166:111636. https://doi.org/10.1016/j.patcog.2025.111636

12. Ciano T, Ferrara M (2024) Karush-Kuhn-Tucker conditions and Lagrangian approach for improving machine learning techniques: A survey and new developments. Atti della Accademia Peloritana dei Pericolanti - Classe di Scienze Fisiche, Matematiche e Naturali 102:1. https://doi.org/10.1478/AAPP.1021A1

13. Xu Y, Guo R, Wang L (2013) A Twin Multi-Class Classification Support Vector Machine. Cogn Comput 5:580–588. https://doi.org/10.1007/s12559-012-9179-7