Robust and General Sampling Method for Large-Scale Classification Problems

Abstract

This paper presents a general sampling method for large-scale classification problems based on granular computing, called as "Granular-Ball Sampling (GBS)". It is a first general method that is not designed for any specific classifiers and can ensure the consistency of the boundary before and after sampling. So, it has almost the same classification accuracy with the results on original datasets and obviously higher classification accuracy than random sampling. Besides, the proposed method has a time complexity close to O(N), so it can accelerate most classifiers. Moreover, the GBS can considerably enhanced the classification accuracy of various classifiers in processing the label noise-containing and imbalanced datasets because of the characteristics of granular computing.

1 Introduction

A Large-scale classification has always been a challenge in machine learning. Large-scale datasets denote those whose number is so large that it is difficult to be processed efficiently under the condition of a given hardware resource. Sampling is a general, scalable and efficient method to process large-scale classifications in comparison with revising a classifier's model to decrease its time complexity in a specific scenario.

Up to now, the commonly used sampling methods developed for classification problems mainly include the over-sampling, such as the SMOTE (synthetic minority over-sampling technique) Chawla *et al.* [2002], and under-sampling Liu *et al.* [2009]; Kang *et al.* [2017]. They are designed for imbalanced classifications to balance the number of points in different classes by generating more points in the minority classes or deleting data points in the majority classes. So, these sampling methods are not designed for large-scale classifications and not selected for comparison in this paper.

Some sampling methods for large-scale classifications are designed for specific classifiers or scenarios, such as Zhu *et al.* [2015] for large sample linear regression, Wang *et al.* [2018] for large sample logistic regression, Dasgupta *et al.* [2008] for Lp Regression, Drineas *et al.* [2006] for l2 regression, Dhillon *et al.* [2013] for least squares regression, Sevakula *et al.* [2015]; Sain and Purnami [2015] for support vector machine. In Yuan *et al.* [2012], the sampling methods are designed specially for ensemble learning classifier, and the sampling method in Dasgupta and Hsu [2008] for active learning where many samples are unlabeled. These algorithms are designed for some specific classifiers or scenarios and not general.

A general method is to extract the samples near the classification boundary based on the distances from a sample to the centers of other classes, such as Xia *et al.* [2015]. However, on the one hand, most of these methods are limited to linear datasets. Besides, some improvement versions based on kernel methods also rely on optimization of kernel parameters and have a high time complexity of no less than O(N^2). So, they are rarely used and not selected for comparison in this paper. In contrast, as a general method, although the random sampling may considerably lose the information of original data distribution because it cannot ensure the consistency before and after sampling, it

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

is the only relatively common method because it is simpler and faster than that based on boundary exaction. So, it is selected for comparison in this paper.

Overall, a general sampling method for the large-scale classification problems should meet two main requirements. First, it should have a low time complexity, lower than that of the common classifiers (i.e. O(N^2) or O($NlogN$)); otherwise, it cannot accelerate the learning process. Secondly, it should consider the consistency of boundary before and after sampling. Motivated by the mentioned challenge, we propose a general sampling method for dealing with the large-scale classification problems by introducing the granular computing Yao [2006]; Pawlak [1982].

The contributions of this paper are as follows.

(1) A general sampling method not designed for any specific classifiers, named as the granular-ball sampling (GBS), is proposed for large-scale classification problems. It can ensure the consistency before and after sampling. So, it has almost the same classification accuracy with the results on original datasets and obviously higher classification accuracy than the random sampling.

(2) The proposed method has a time complexity close to O(N). So, in the experiments, it can accelerate various classifiers, e.g. a state-of-the-art classifier XGBoost, from one days to half an hour on a large dataset while they have almost the same test accuracies.

(3) The GBS can considerably enhance the classification accuracy of various classifiers in processing the imbalanced and label noise-containing datasets due to the characteristics of granular computing.

2 Granular-ball Sampling Method

In this section, we will introduce the concept of GBS and analysis its performance theoretically.

2.1 The Idea of Granular-ball Sampling

Granular computing is a scalable, efficient and robust method that is very similar to the way the human brain thinks Yao [2006]; Pawlak [1982]. It uses simple, low-cost, satisfactory approximate solutions rather than the exact solutions to achieve the tractable, robust, low-cost intelligent systems that can describe the real-world better Ling and Bo [2003]. In Science Rodriguez and Laio [2014], Rodriguez pointed out that granular computing is an effective method for finding the knowledge in big data, and it has been combined with various learning methods, such as the rough sets model Pawlak [1982], the computing with words proposed by Zadeh [1997], and label noise detection Xia *et al.* [2018]. In this paper, we propose the GBS by introducing the idea of granular computing.

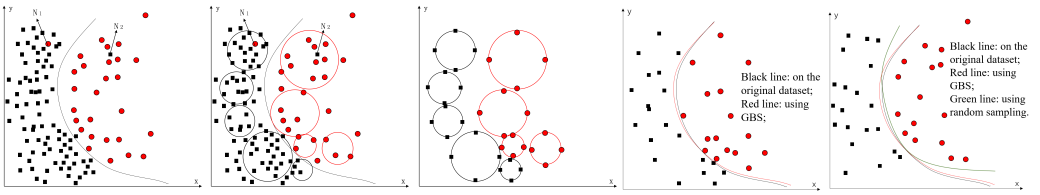
As a development of granular computing, "granular ball computing classifiers" is proposed by Xia *et al.* [2019]. It replaces the points input with granular balls in the mathmatical models of classifiers, resulting an improvement of performance in robustness, efficiency and scalability. In this paper, the concept of the granular ball is introduced to propose a novel general sampling method named as "granular ball sampling" (GBS) for large-scale classifications. A GB covering a point cluster consist of a center and a radius. It definition is as follows Xia *et al.* [2019].

Definition 1. *Given a dataset $D \in R^d$, a granular ball GB is generated on D, and its center, radius and correspoinding point cluster are respectively denoted as C, r and GB'. suppose $x_i \in GB'(i = 1, 2 \cdots, N)$, where N is the number of samples in GB', we define C and r as follows:*

$$C = \frac{1}{N}\sum_{i=1}^{N} x_i, \quad r = \frac{1}{N}\sum_{i=1}^{N} \|x_i - C\| \tag{1}$$

The idea of the GBS is presented in Fig. 1. In Fig. 1(a), the original dataset and its boundary between two classes of points are presented; in Fig. 1(b), the original dataset is covered by the balls under the condition that the boundary is almost consistent with the original dataset, i.e. that the original dataset is covered by GBs such that the original boundary is kept almost unchanged. In Fig. 1(c), it can be seen that the original dataset is sampled in the way that the points on the balls' edges replace the data points inside the balls. Furthermore, in this work, as shown in Fig. 1(c), we use the intersection points of a GB and the axes of the coordinate system whose origin is in the GB center to describe and replace the data points in the ball. The intersection points represent the sampled results as shown in

Fig. 1(d). Those intersection points whose number is equal to 2*d can describe the boundary of the point cluster corresponding to the ball. So, it can be observed that, the boundary in Fig. 1(d) is very consistent with the boundary in Fig. 1(a), and the samples in Fig. 1(d) is much sparser than those in Fig. 1(a). In contrast, the random sampling is likely to loss the boundary information. Therefore, as shown in Fig. 1(d) and (e), the boundary by using the GBS is more tighten and closer to that of the original dataset than the random sampling.



(a) (b) (c) (d) (e)

Figure 1: The concept of the GBS. (a)The original dataset and its boundary; (b)The generated GBs; (c)The sampling of the original data by using the GBs; (d)The final dataset and its boundary; (e)The sampling results by using the random sampling

The label of those intersection points is equal to the GB. As described in Xia *et al.* [2019], a GB's label is equal to the label of majority points inside it. Here, we formally defined it as follows.

Definition 2. *Given a granular ball GB containing k classes of points $(P_1, P_2, \ldots, P_k)$. The number of data points in P_i is the largest. For a point $x \in P_i$, we have:*

$$Lable(GB) = Lable(x) \tag{2}$$

In the GBS, as shown in Fig. 1(c), to describe the boundaries of the point cluster corresponding to a GB, each GB is replaced with those 2*d (i.e. 4 in Fig. 1) intersection points. The size of 2*d increases with its dimensionality. This indicates that, the GBS will not work when the dimensionality is close to the size of the dataset because the number of GBs wil be very small, such as 1 or 2. However, this is reasonable because it is consistent with a fact that, higher dimensionality will lead to higher sparsity, so the necessary of sampling is decreased.

2.2 GBS's Robustness to Label Noise and Its Performance in Processing Imbalanced Datasets

The label noise and imbalance are two important scenarios in classifications. Two common kinds of methods have been developed for label noise. The first is to detect and filter label noise, such as Xia *et al.* [2018]. The second is to improve the robustness of learning methods by revising the loss function, such as Liu and Tao [2016]. In the case of an imbalanced classification problem, imbalance can be alleviated by increasing the minority points or decreasing the majority points. Interestingly, the GBS not only has the good robustness to the label noise but a good performance in processing the imbalanced datasets. The dataset shown in Fig. 1 is an imbalanced dataset, wherein the number of positive samples (i.e. the white points in Fig. 1) is more than three times greater than the number of negative samples (i.e. the red points in Fig. 1). However, the number of GBs is not very related to the number of samples but more influenced by the data distribution characteristic. Therefore, as shown in Fig. 1(c), the number of positive GBs is similar to the number of negative GBs, and the sampled result in Fig.1(d) is much balanced than the data Fig. 1(a). Besides, the dataset shown in Fig. 1 contains two label noise points N_1 and N_2. So, since the label of a GB is equal to the label of its majority points, the noise points in a GB which is represented by the majority points will be neglected. This is very beneficial for improving the classifier performance in label noise-containing datasets.

3 Generation Methods of GBs and Termination Conditions

As an efficient method, the *k*-means clustering method is used as a GB generation method because the *k*-means clustering method not only has a low time complexity of O(Nkt) but also is likely to

generalize a spherical cluster, which is benificial to describe a ball. To make the generation process scalable, in this work, we iteratively use the 2-means clustering on the original dataset. Firstly, the 2-means clustering is used on the whole datasets, so two GBs are generated. Then, the quality of each GB is checked. If its quality is not good enough, it will be split iteratively by using the 2-means clustering until each GB meets the given quality requirements.

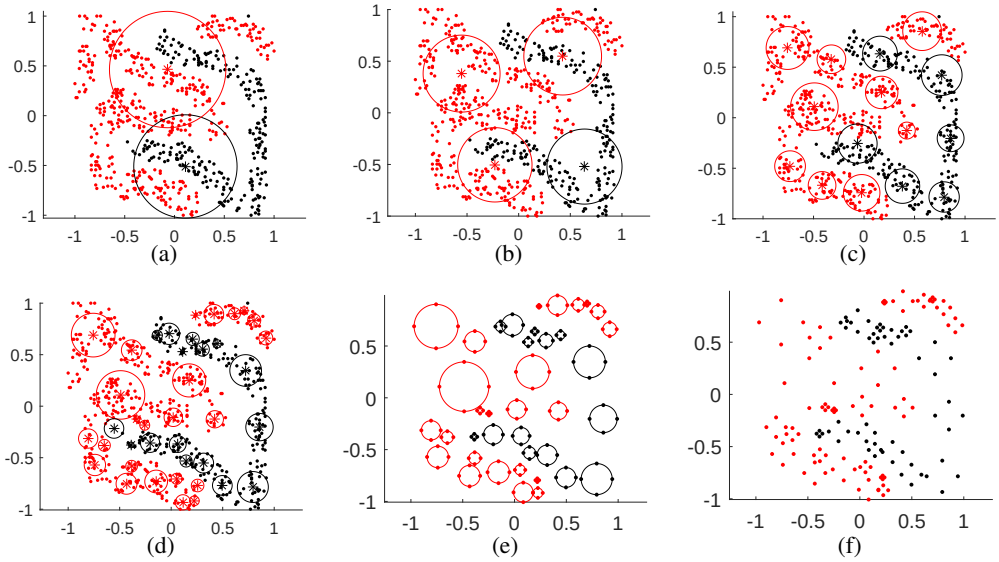


Figure 2: The generation process of GBs on the dataset fourclass when the purity threshold is set as 1. (a). The first iteration. (b). The second iteration. (c)An intermediate result. (d) the final GBs.(e) the final GBs without data points.(f) the sampled results by using the GBS. Both the red points and red GBs have the label of "+1", and both the black points and black GBs have the label of "-1".

As a simple and efficient measurement, the "purity" described in Xia *et al.* [2019] can be used to measure the quality of a GB. It is equal to the percentage of the majority points in it. The quality of a GB is better when its purity is higher. In this paper, we formally define it as follows.

Definition 3. *Given a granular ball GB and its corresponding points clusters GB'. GB' contains k classes of points $P_1, P_2...P_k$. $P_1 \cup P_2 \cup \ldots \cup P_k = GB'$, and $P_1 \cap P_2 \cap \ldots \cap P_k = \Lambda$. The number of data points in P_m is the largest among them and is denoted with p_m. We have:*

$$purity(GB) = \frac{p_m}{\sum p_i}, i = 1, 2, ..., k \tag{3}$$

The generation process can be shown in Fig. 2, in which the purity threshold is set to 1. When the split generation continues, the purities of most GBs increase, and the consistency of the boundary became better. In Fig.2(d), when the purity of each GB reaches to the purity threshold, i.e. 1, the area covered by the GBs had a very consistent boundary with the original dataset.The sampled points in Fig. 2(f) are much sparser than the orignial dataset. So, as described above, the first termination condition is that the purity of each GB should reach to a given threshold, which is defined in Termination 1:

Termination 1. *Given a dataset D and a purity threshold T. GB represents a granular ball generated on D. Its purity value should reaches to T, i.e. that: $purity(GB) \geq T$.*

Too small GBs, such the ones containing one point, will increase computation cost and not very useful for sampled result. Besides, a label noise point is likely to represent a very small granular ball when the purity threshold is set 1. These small granular balls consisting of label noise are harmful for classification. Therefore, That a granular ball has a certain size is necessary to be robust of label noise for the GBS when the purity threshold is set 1. So, another necessary condition to judge a GB whether should be split or not is that the number of points in a GB is large than 2*d. The another reason why the smallest size is set to 2*d is that, as each granular ball is replaced with 2*d points as described in Section 2.1, the splitting in the small granular balls whose size is smaller than 2*d is unnecessary. Therefore, the second termination condition is defined in Termination 2:

Algorithm 1 Granular-ball sampling

Input: Dataset D, the purity threshold T
Output: The resulting dataset D'

1: Implement the 2-means clustering algorithm on D and generate two GBs D_1^i and D_2^i, where i is initialized as 1, and it denotes the iteration number.
2: **for** each D_j^i **do**
3: Compute the center and the radius of D_j^i by using (1)
4: $PD_j^i = purity(D_j^i)$
5: **if** $(PD_j^i < T)$&(the number of points in $D_j^i > 2 * d$) **then**
6: split D_j^i by continuously applying the 2-means clustering until all sub-GBs meet the two termination conditions in Section 4.
7: **end if**
8: **end for**
9: D' is constituted of the points in small GBs (i.e. their size is lower than $2 * d$), the intersection points of GBs and axis, and the centers of all GBs.

Termination 2. *Given a dataset D with its dimensionality denoted as d. GB represents a granular ball generated on D. The number of samples in GB should be smaller than 2*d.*

Based on the above descriptions, the Algorithm 1 is designed.

4 The convergence of the GBS

As described in section 2.1, the GBS can describe the decision boundary more exactly than the random sampling. In the following analysis, we will show that the sampling process is convergent. Given a dataset $D \in R^n$. for a point $x_i \in D, i = 1, 2, ..., N$, where N denotes the number of data points in D, the 2-means clustering is iteratively implemented. The data distribution difference between the area covered by the GBs and the original dataset can be expressed by:

$$\sum_{k=1}^{K}\sum_{i=1}^{N}\left(x_i - c_k\right)^2 \tag{4}$$

where c_k represents the center of the k^{th} GB (i.e., the k^{th} cluster). When a GB is split into two GBs, the data distribution of a newly-formed area covered by the GBs will be more consistent with that of the original dataset because of the following fact:

$$\sum_{i=1}^{M_1}\left(x_i - \overline{x_1}\right)^2 + \sum_{i=M_1}^{M}\left(x_i - \overline{x_2}\right)^2 < \sum_{i=1}^{M}\left(x_i - \overline{x}\right)^2 \tag{5}$$

$$\Rightarrow min(\sum_{i=1}^{M_1}\left(x_i - \overline{x_1}\right)^2 + \sum_{i=M_1}^{M}\left(x_i - \overline{x_2}\right)^2) < min(\sum_{i=1}^{M}\left(x_i - \overline{x}\right)^2) \tag{6}$$

where $\overline{x_1}$ represents the center of the first GB, and x_i $(i = 1, 2, \ldots, M_1$, where M_1 denotes the number of the points in the first GB) denotes the point in the first GB; $\overline{x_2}$ represents the center of the second GB, and $x_i (i = 1, 2, \ldots, M_2)$ denotes the point in the second GB. Therefore, a larger number of GBs will decrease the data distribution difference between the area covered by the GBs and the original dataset. In other words, the data distribution difference is monotonically decreasing with the increase in the number of GBs. In addition, the data distribution difference will tend to zero when the number of GBs is equal to the number of data points in D. In that case, each GB is a point, and the radius of each GB is equal to zero. In other words, the lower bound of (3) is equal to zero. Hence, the data distribution difference is monotonically decreasing and has a lower bound, making the sampling process of the GBS convergent. Therefore, as a generation method of GBs, although *k*-mean clustering does not work in clustering many datasets, in the GBS it can guarantee the consistence of the boundary before and after the GBS is used if the number of GBs is enough large, i.e. that the quality of each GB is good enough.

5 Experiments

This section validates the advantages of the GBS on various widely used classifiers, including the BPNN, SVM, Logic Regression, Decision tree, Exact_kNNs, GBDT Ye *et al.* [2009], XGBoost Chen and Guestrin [2016] and lightGBM. The last three algorithms are state-of-the-art algorithms used in the Kaggle competition platform in recent years. The data sets and source codes are available in the following anonymous link: `https://yunpan.360.cn/surl_yLFDYGZS7IM`

We used the "scikit-learning" toolbox for the experiments on an average-performance laptop. Its detailed configuration is as follows: an Intel I7-6650 processor with 3.2 GHz frequency, 24 GB RAM, and Windows 10 operating system. The main parameters were left as the default values of the toolbox, and the default 5 fold cross validation is used. Each classifier was compared on 15 real world datasets randomly taken from the UCI Machine Learning Repository. Each dataset was randomly divided into two parts, 75 percent for training and 25 percent for testing. This process would be repeated 10 times to calculate the average accuracies and variances. The highest accuracy and other indicator are marked in black font. The purity in Section 5.2, 5.3 and 5.4 is optimized by being increased from 0.8 to 1 with an interval as 0.01. All tables supplementary are available in the anonymous link:`https://yunpan.360.cn/surl_yFNez2FreD4`. Table 1 Supplementary lists the datasets.

5.1 Analysis of Compression Performance

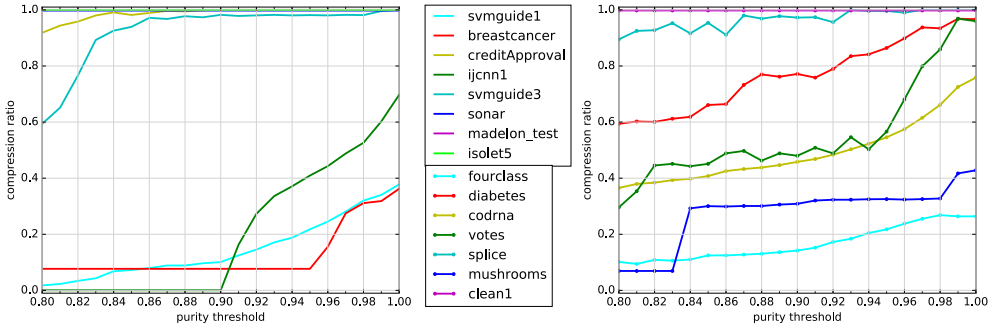


Figure 3: The ordinate represents the compression ratio (i.e. sampling rate), and the abscissa the purity when the purity increases from 0.8 to 1.

Fig. 3 shows the compression ratio, i.e. the percentage of data points after using the GBS with the increase in "purity" from 0.80 to 1.00. The compression ratio increases when the "purity" increases. When the "purity" is set as 0.80, the compression ratios of some datasets become very small, in which the ijcnn1 has the lowest compression ratio, i.e. about 0.03%. More than half of those datasets have a compression ratio lower than 50%. This indicates that, in some datasets whose distributions are appropriate for processing by the GBS, the effect of compression is very obvious. Even when the "purity" is equal to the largest value, i.e. 1, almost 30% of the datasets have a compression ratio lower than 50%, and the dataset fourclass has the lowest compression ratio, i.e. about 23%.

The impression ratios on the isolte5, madelon_test, sonar and clean1 are very close to 1. This is because the numbers of their points are close to their dimensionality; thus, their dataspace is very sparse. This sparse and possible mixed area of points in different classes will make the number of GBs close to the number of samples. Consequently, the compression ratio is close to 1. From a certain perspective, those samples in those four datasets are spare enough; therefore, they are not appropriate for sampling. In contrast, the dataset whose number of data points is large, such as ijcnn1, is more likely to have a low compression ratio. In Section 5.5, the GBS is still very effective when the compression ratio reaches to $10^{-6} \sim 10^{-5}$ on a dataset containing 100 million points.

5.2 Effectiveness

Tables 2-9 supplementary and Tables 1 show a comparison of test accuracies. It was observed that, the classifiers achieved similar average test accuracy before and after the GBS was used. Besides, the GBS-classifiers achieved higher accuracy on some datasets, such as on the creditApproval and svmguide3 as shown in the tables supplementary. Moreover, the GBS-BPNN and GBS-LR achieved

Table 1: Average Test Accuracy Comparison of Various Classifiers on Fifteen Benchmark Datasets

Model	Original	Random sampling	GBS	Model	Original	Random sampling	GBS
BPNN	0.8677	0.8587	**0.8715**	kNN	**0.8579**	0.8558	0.8576
SVM	**0.8136**	0.8112	0.8133	DT	**0.8582**	0.8464	0.8558
LR	0.8443	0.8428	**0.8455**	GBDT	**0.9012**	0.8979	0.8999
XGBoost	**0.9005**	0.8965	0.9003	lightGBM	**0.9084**	0.9055	0.9069

Table 2: Average Test Accuracy Comparison on datasets containing label noise

Noise ratio	5%		10%		15%		20%	
	Original	GBS	Original	GBS	Original	GBS	Original	GBS
BPNN	0.8071	**0.8346**	0.7632	**0.7943**	0.7258	**0.7546**	0.6881	**0.7194**
kNN	0.8087	**0.8104**	0.7628	**0.7655**	0.7196	**0.7248**	0.6706	**0.6769**
SVM	0.7746	**0.7775**	0.7419	**0.7450**	0.7079	**0.7104**	0.6743	**0.6760**
DT	0.7801	**0.8103**	0.7192	**0.7575**	0.6636	**0.7029**	0.6173	**0.6486**
LR	0.7983	**0.8040**	0.7637	**0.7691**	0.7230	**0.7287**	0.6870	**0.6914**
GBDT	0.8407	**0.8488**	0.7959	**0.8054**	0.7524	**0.7628**	0.6993	**0.7109**
XGBoost	0.8564	**0.8584**	0.7995	**0.8029**	0.7490	**0.7540**	0.7011	**0.7061**
LightGBM	0.8585	**0.8654**	0.7977	**0.8044**	0.7427	**0.7504**	0.6998	**0.7044**

higher average test accuracies than their conventional versions. The reason is that, as a sampling method, although the GBS may lose a slight distribution information, it can enhance the robustness and performance in processing imbalanced problems as described in Section 2.2.

Another phenomenon is that a GBS classifier achieving the highest accuracy is possible to not have the highest purity value, i.e. 1. This is because that a low purity can make the GBS classifiers robust for label noise. Besides, as shown in the tables, the random sampling always perform worse than the GBS, which will be more obvious in relatively large datasets in Section 5.5.

5.3 Experiments on Datasets Containing Label Noise

In this section, the datasets with four noise rates, i.e. 5%, 10%, 15% and 20%, are generated by fliping the labels of randomly selected samples on the datasets in Table 1 supplementary. Tables 10-41 supplementary provide the test accuracy of orignal and GBS-based classifiers on the datasets in different noise rates.

As shown in Tables 10-41 supplementary and Table 2, the GBS improves the test accuracy of various classifiers on almost all those datasets. The Table 2 show that, the DT and BPNN is more sensitive to label noise than other algorithms; therefore, their classification accuracy is improved by almost 4% after the GBS is used. Even on the GBDT, XGBoost and LightGBM, which are three state-of-the-art ensemble algorithms and robust to label noise to an extent, GBS-based classifiers also achieved a better performance in generalizability. On some datasets, such as the Votes, the improvement of test accuracy can reach to about 4%. The reason about these advantages is that, in the termination condition 2, the label of a granular ball is determined by the majority labels in it while the label noise points are the minority. So, the label noise samples contained in a small granular ball can be eliminated in the computation for the label of the granular ball. Therefore, the GBS classifiers can exhibit better robustness for label noise than the results on the orignial datasets.

5.4 Experiments on Imbalanced Datasets

In this section, the GBS is compared with conventional algorithm on imbalanced datasets. The datasets in Table 1 supplementary are sampled and the resulted imbalanced datasets are shown in Table 42 supplementary. The experimental results are shown in Tables 43-50 supplementary. Table 3 shows the average performance. F-measure and G-mean Weiss [2004]; Joshi *et al.* [2001], two popular indexes, were used as the evaluation indexes. The parameter β was set as 1 in the F-measure. This means that the precision and recall index are of the equal importance. It can be observed that

Table 3: The average performance of the comparison methods on imbalanced datasets

	F-measure		G-mean			F-measure		G-mean	
	Original	GBS	Original	GBS		Original	GBS	Original	GBS
BPNN	0.7420	**0.7767**	0.7627	**0.8210**	kNN	0.6977	**0.7112**	0.7579	**0.7740**
SVM	**0.4972**	0.4907	**0.5380**	0.5327	DT	0.7382	**0.7748**	**0.8075**	0.7825
LR	0.6931	**0.6938**	0.7536	**0.7563**	GBDT	0.7776	**0.7874**	0.8205	**0.8285**
XGB	0.7602	**0.7669**	0.8115	**0.8179**	LGB	0.7895	**0.7906**	0.8384	**0.8419**

most bold fonts appear most frequently in the GBS classifiers. This indicates that the GBS classifiers have better performance than the conventional classifiers in processing imbalanced datasets. The reason is that the balance of granualr balls is much better than that in the original dataset, so the sampled points are more balanced than the original dataset when the GBS is used. Conseqently, the classification performance is improved.

5.5 Efficiency Comparison on Large-scale Datasets

In this section, the large-scale artificial datasets were generated by using the gaussian distribution, and their numbers were increased from 100,000 to 100 million with an interval as 10 times.To produce as many samples as possible while the computer's memory capacity allows, the dimensionality was set as the smallest number, i.e. 2. The datasets can be downloaded in the following anonymous link:`https://yunpan.360.cn/surl_yL375qCcAAv`. The number of the sampled points in the random sampling is increased from 100 to 1000 with an interval as 100. To make the number of sampled points in the GBS almost equal to the number in the random sampling, the number of GBs in the GBS increases from 20 to 200. To check the performance of the compared algorithms in complex data environment, the generated datasets are imbalanced and contain label noise. The ratio of samples in positive and negative classes is 10:1. 5% of the points are randomly selected, and their labels are flipped to generate label noise. Since the experiments are time consuming, only the XGBoost is selected as the representative classifier. The experimental results and other parameters setting are shown in Tables 51-54 supplementary when the size of the dataset increases from 100,000 to 100 million with an interval as 10 times. It was observed that, as the GBS can guarantee the consistency of the boundary, it achieves much better generalizability than the random sampling method in processing large-scale datasets. Although it costs more time than the random sampling method, it is also much efficient. In processing 100 million points, it only cost about half an hour to sample one thousandth of the data points on the average-performance laptop while implementing the conventional XGBoost on the original datasets cost about one day. Besides, the GBS-XGBoost achieved a classification accuracy of 94.99%, which is much close to the classification accuracy of the original dataset (i.e. 95%); On the contrary, the random sampling only reached 92.46%. These indicate that the GBS can be very effective even when compression ratio reaches to $10^{-6} \sim 10^{-5}$. Moreover, when the number of points increases from 1000,000 to 100 million, the cost time of the GBS increases from about 20 seconds to about 2000 seconds. It indicates that the time complexity of the GBS is almost linear.

6 Conclusion

In this paper, we proposed a general sampling method named as the GBS to solve large-scale classification problems by introducing the granular computing. It is the first general sampling method for large-scale classifications that is not designed for any specific classifiers and can ensure the consistency before and after sampling. It has a low time complexity close to O(N). In processing 100 million points on an average-performance laptop, it only cost about half an hour to implement the GBS and achieved a classification accuracy of 94.99%, which was much close to the accuracy of the original dataset (i.e. 95%); In contrast, the random sampling only reached 92.46%, and the experiments on the original datasets cost about one day. In the large-scale datasets, the GBS can be very effective even when compression ratio reaches $10^{-6} \sim 10^{-5}$. Besides, the GBS-based classifiers exhibited better generalizability on most of the label noise containing or imbalanced datasets.

Despite of these advantages, the definition of the purity makes the GBS only work in supervised tasks. Therefore, to promote the GBS to semi-supervised learning will be researched in the future.

References

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

Sanjoy Dasgupta and Daniel Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th international conference on Machine learning*, pages 208–215. ACM, 2008.

Anirban Dasgupta, Petros Drineas, Boulos Harb, Ravi Kumar, and Michael W. Mahoney. Sampling algorithms and coresets for lp regression. *Siam Journal on Computing*, 38(5):2060–2078, 2008.

Paramveer Dhillon, Yichao Lu, Dean P Foster, and Lyle Ungar. New subsampling algorithms for fast least squares regression. In *Advances in neural information processing systems*, pages 360–368, 2013.

Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Sampling algorithms for l2 regression and applications. In *Seventeenth Acm-siam Symposium on Discrete Algorithm*, 2006.

Mahesh V Joshi, Vipin Kumar, and Ramesh C Agarwal. Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *Proceedings 2001 IEEE International Conference on Data Mining*, pages 257–264. IEEE, 2001.

Qi Kang, XiaoShuang Chen, SiSi Li, and MengChu Zhou. A noise-filtered under-sampling scheme for imbalanced classification. *IEEE transactions on cybernetics*, 47(12):4263–4274, 2017.

Zhang Ling and Zhang Bo. Theory of fuzzy quotient space (methods of fuzzy granular computing). 2003.

Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on pattern analysis and machine intelligence*, 38(3):447–461, 2016.

Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2009.

Zdzisław Pawlak. Rough sets. *International journal of computer & information sciences*, 11(5):341–356, 1982.

Alex Rodriguez and Alessandro Laio. Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496, 2014.

Hartayuni Sain and Santi Wulan Purnami. Combine sampling support vector machine for imbalanced data classification. *Procedia Computer Science*, 72:59–66, 2015.

Rahul K Sevakula, Mohammed Suhail, and Nishchal K Verma. Fast data sampling for large scale support vector machines. In *2015 IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions (WCI)*, pages 1–6. IEEE, 2015.

HaiYing Wang, Rong Zhu, and Ping Ma. Optimal subsampling for large sample logistic regression. *Journal of the American Statistical Association*, 113(522):829–844, 2018.

Gary M Weiss. Mining with rarity: a unifying framework. *ACM Sigkdd Explorations Newsletter*, 6(1):7–19, 2004.

Shu Yin Xia, Zhong Yang Xiong, Yue Guo Luo, and Li Mei Dong. A method to improve support vector machine based on distance to hyperplane. *Optik - International Journal for Light and Electron Optics*, 126(20):S0030402615004829, 2015.

Shuyin Xia, Guoyin Wang, Zizhong Chen, Yanling Duan, and Qun Liu. Complete random forest based class noise filtering learning for improving the generalizability of classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 2018.

Shuyin Xia, Yunsheng Liu, Xin Ding, Guoyin Wang, Hong Yu, and Yuoguo Luo. Granular ball computing classifiers for efficient, scalable and robust learning. *Information Sciences*, 483:136–152, 2019.

Yiyu Yao. Granular computing for data mining. In *Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2006*, volume 6241, page 624105. International Society for Optics and Photonics, 2006.

Jerry Ye, Jyh-Herng Chow, Jiang Chen, and Zhaohui Zheng. Stochastic gradient boosted distributed decision trees. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 2061–2064. ACM, 2009.

Hanning Yuan, Meng Fang, and Xingquan Zhu. Hierarchical sampling for multi-instance ensemble learning. *IEEE Transactions on Knowledge and Data Engineering*, 25(12):2900–2905, 2012.

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

Rong Zhu, Ping Ma, Michael W Mahoney, and Bin Yu. Optimal subsampling approaches for large sample linear regression. *arXiv preprint arXiv:1509.05111*, 2015.