分类问题中一种高效的、自适应的粒球生成方法

摘要

粒度球计算是一种高效、鲁棒且可扩展的粒度计算学习方法，粒球计算的基础是粒球生成方法。本文首先介绍了先前提出了一种使用 k-division 代替 k-means 来加速粒球生成的方法，它可以大大提高颗粒球的生成效率，同时保证与现有方法相似的精度。此外，这些通过考虑粒球的重叠消除等因素，提出了一种新的颗粒球生成自适应方法，这使得颗粒球生成过程无参数且完全具有真正意义上的自适应。本文首先对该方法进行详细的介绍，并利用该方法进行实验，与原文的效果进行对比，在一些真实数据集上的实验结果表明，所提出的两种粒球生成方法在实现自适应或加速的同时，与现有方法具有相似的准确性。同时，本文将进一步对以上工作尝试进一步改进，结果虽然没有显著提升，但优化了实验的时间消耗。

关键词：粒球；粒球计算；自适应粒球生成；粒球生成；分类

1 引言

认知计算与人类认知机制相结合，使决策过程更加可靠、高效和易于理解，是实现信息空间可靠治理的重要手段，也是人工智能发展的重要方向。陈院士 1982 年发表在《科学》杂志上的研究成果指出，人类认知的特点是大规模优先 [1]。如图 1 所示，首先看到大的轮廓字母，然后是轮廓字母中的特定小字母。基于这种认知特性，粒度计算可以实现高效、可扩展和健壮的学习过程。

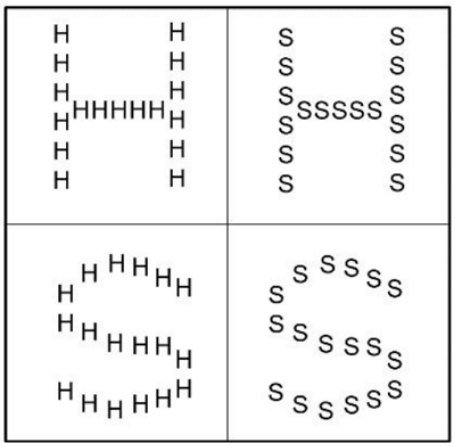

图 1. 人类认知——粗粒度大范围优先

在粒度计算中，粒度越大，效率更高并且噪音鲁棒性也更好；但它也更有可能导致忽视细节和失去准确性。粒度越小，对细节的关注就越多，但可能会降低效率并降低对噪声的鲁棒性。根据不同的场景选择不同的粒度可以更好地发挥多粒度学习方法的性能。虽然多粒度计算的研究历史悠久，但作为一门认知计算科学，它也面临着一些新的挑战，需要新的发展。例如，就人工智能使用最广泛的方法之一“分类器”而言，如图 2(a) 所示，大多数现有分类器的输入都是最细粒度的样本点或像素点 [4] [2] [3]，因此将缺乏粗粒度的表征。

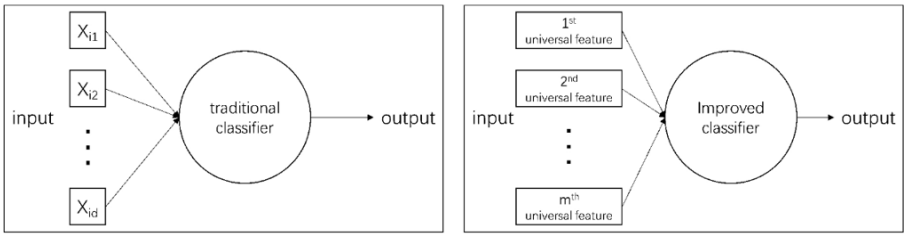

图 2. 多粒度分类器与传统分类器的比较

因此，如何实现图 2(b) 所示的多粒度分类器是一个重要的挑战。在图 2(b) 中的多粒度分类器中，输入不再是最细粒度的点，而是一个粒度可调的通用特征。这个通用特性的设计应该满足高维可扩展性，即在高维空间不需要复杂的计算，否则无法处理高维问题。为此，提出了用球作为“颗粒”来表示这一普遍特征，并提出了一种颗粒球计算方法 [5]。由于球的几何形状是完全对称的，只需要中心和半径两个数据就可以在任何维度上表征它，因此可以方便地应用于高维数据。同时还提出了一种高效且自适应的方法来生成颗粒球。本文的主要贡献如下：

1) 提出了使用 k-division 代替 k-means 的加速度粒球生成方法，它可以将颗粒球的产生速度提高数倍到数十倍，同时达到类似的精度。

2) 通过考虑粒球的重叠消除等因素，提出了一种新的颗粒球生成的自适应方法，使得粒球生成过程无参数且完全自适应。

3) 首先提供了颗粒球覆盖的数学模型。

2 相关工作

2.1 粒球计算

粒球计算的核心思想是用“粒球”来覆盖或部分覆盖样本空间 [5]。定义粒球 $GB = x_i \; i = 1...N$，其中 x_i 表示 GB 中的对象，N 为 GB 中对象的个数。GB 的圆心 C 和半径 r 分别表示如下：

$$C = \frac{1}{N}\sum_{i=1}^{N} x_i, \tag{1}$$

$$r = \frac{1}{N}\sum_{i=1}^{N} |x_i - C|. \tag{2}$$

这意味着半径等于 GB 中所有对象到其中心的平均距离，其中半径也可以设置为最大距离。粒球计算中分类问题的粒球生成的基本过程如图 3所示。

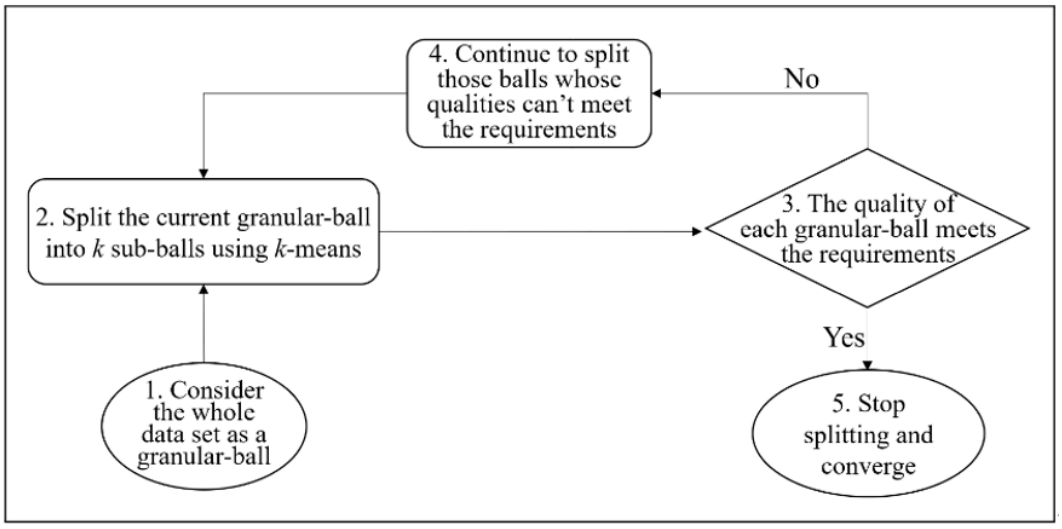

图 3. 粒球计算中现有的粒球生成过程

如图 3所示，在步骤 1 中，为了模拟算法开始时“人类认知的大尺度优先特性”，可以将整个数据集看作一个粒球，此时，颗粒球的纯度最差，无法描述数据的任何分布特征。在图 3的步骤 3 中，“纯度”被用来衡量一个粒状球的质量 [5]，它等于颗粒球中最多标签样本的比例。然后，统计粒球中不同类的个数，记为 k；在步骤 2 中，粒球被分裂成 k 个子粒球。在步骤 3 中，计算每个颗粒球的纯度，如果颗粒球未达到纯度阈值，则需要对其进行分裂。随着分裂过程的推进，颗粒球的纯度增加，决策边界越来越清晰；当所有颗粒球纯度符合要求时算法收敛，决策边界最清晰。对于一个数据集，无论它的数据具有什么分布，我们都可以用足够多的粒球来描述它的决策边界。

颗粒球计算已经发展了颗粒球分类器 [5]，颗粒球聚类 [6]，颗粒球邻域粗糙集 [7] 和颗粒球采样方法 [8]。

2.2 粒球 k 近邻算法

kNN 算法具有简单、自适应多个分类、独立于训练、可同时用于分类和回归、易于并行化和实现等特点，它是目前应用最广泛的人工智能算法之一。在 kNN 中，寻找查询点的 k 个最近邻居的基本原理是计算查询点到所有数据点的欧氏距离，并利用这些邻居的值 (或标签) 来预测或分类查询点，这种方法称为全搜索算法 (Full Search Algorithm，FSA)。FSA 存在以下共性问题：需要对 k 值进行优化；并且 k 值的优化需要二次时间复杂度，因此非常耗时。从多粒度的角度来看，k 优化问题的根源在于对细粒度的过度关注。为此，在 [5] 中将粒球引入到 kNN 中，提出了一种无需优化 k 值的高效最近邻算法——粒球 k 近邻方法 (GBkNN)。其基本思想是非常容易实现：基于粒球计算，单个查询样本点是一个半径非常小的粒球，其预测标签等于最近的粒球的标签，该标签由粒球中的多数类的标签决定。因此，GBkNN 与传统 kNN 的共同特点是查询点的标记由多个点决定。GBkNN 的第一个重要优点是不需要优化参数 k，通过自适应生成不同粗粒度的最近邻粒球来确定查询点的标签；GBkNN 的第二个优点是粒球数量远小于样本点，查询点查询到的最近粒球计算量远小于 kNN，因此 GBkNN 比传统 kNN 效率更高；第三个优点是传统 kNN 的决策会受到标签噪声的影响，而 GBkNN 由于其鲁棒性，特别是在噪声数据集上，可以获得更高的准确率。

3 本文方法

3.1 本文方法概述

现有的粒球生成方法采用 k-means 算法对粒球进行分割，粒球生成的效率并不比 k-means 高，k-means 在粒球生成的每一次迭代中都能产生稳定的分裂结果。然而中间过程的稳定性在过程中是不需要的；所需的仅仅是生成满足条件的粒球。由于不需要中间过程的稳定性，如图 3中的步骤 2 所示，我们使用一个分割，即 k-means 中的一个迭代过程来分割一个粒球，而不是整个 k-means 算法。此外，与图 3所示的现有方法不同的是，在最后增加了一个全局划分，以改善最终颗粒球的整体分布。在全局划分中，基于所有的划分点进行一次划分。具体过程如图 4所示。为了更清楚地描述加速粒子球的生成过程，我们首先给出了父球和子球的定义。

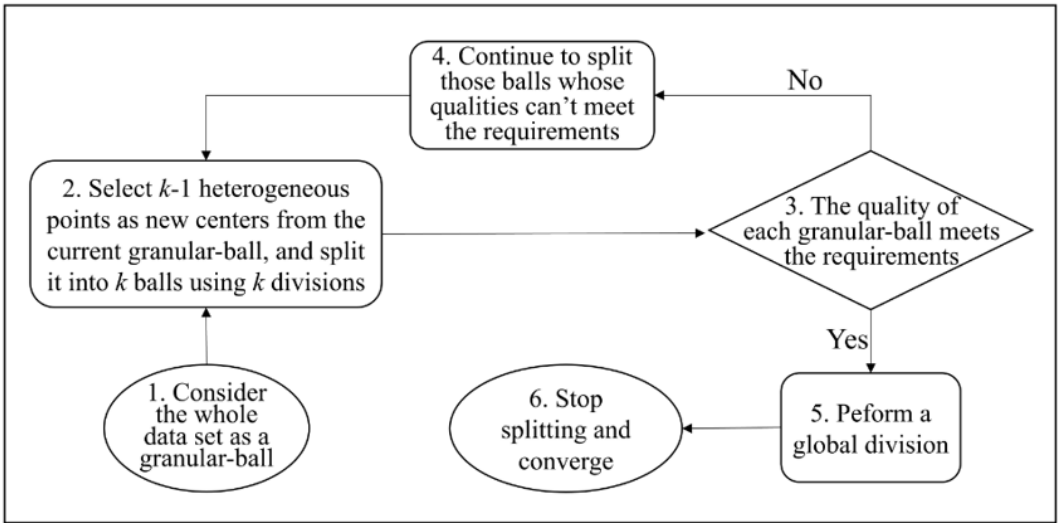

图 4. 粒球计算中的加速粒球生成过程

一个 k-means 算法由 t 次迭代组成，每次迭代是一个划分，其中所有的点根据到 k 个中心点的距离被划分成 k 个簇。在图 3 的步骤 2 中，用于划分粒球的 k-means 被替换为 k division，直接降低了计算代价。此外，如步骤 2 所示，以分裂粒球 A 为例，将粒球 A 的中心点 (记为) 保留为 A 的某一个子球的中心点，因此只选取 k-1 个点作为 A 的新的 k-1 个子球的中心点，所有 k 个子球中的样本点不需要计算与原中心点 a 的距离，因为它们之前已经被计算过。因此，计算成本进一步降低。k-means 的时间复杂度为 $O(Nkt)$ [9]，其中 k 表示聚类个数，t 表示迭代次数，k-means 的收敛速度快，可以近似认为是线性的。加速粒球生成方法在每次生成粒球时只需要计算该簇中的数据与新划分中心的距离。假设一个包含 m 类数据的数据集，在第一轮分裂中，计算时间为 mN。第二轮新生成 m 个 (m - 1) 新的划分中心，计算次数近似为

$$m(m-1) * \frac{N}{m} = (m-1)N$$

在第三轮中，新产生了 $m^2(m-1)$ 划分中心，计算时间约为

$$m^2(m-1)\frac{N}{m^2} = (m-1)N$$

在第四轮中，新产生 $m^3(m-1)$ 划分中心，计算时间约为

$$m^3(m-1)\frac{N}{m^3} = (m-1)N$$

...

假设总共迭代 n 次，最后一次全局划分的时间复杂度为 $O(kN)$，其中 k 为粒子球的个数，总的时间复杂度为 $O((mn-n+k+1)N)$。值得注意的是，当不满足分裂条件时，颗粒球将停止分裂, 大部分散粒体半途而废, 因此，加速粒球生成方法的实际时间复杂度远低于 $O((mn-n+k+1))N)$。加速方法的时间复杂度仍然是线性的，避免了减少不必要的计算。

3.2 自适应的粒球生成算法

通过引入纯度阈值参数，现有方法可以生成满足纯度阈值的颗粒球。现有方法的主要问题是纯度阈值参数无法适应每个数据集的数据分布，难以为每个数据集找到与数据分布相匹配的拆分标准。针对这一问题，本文在加速度粒子球生成方法的基础上，提出了一种纯度自适应的粒子球生成方法，因此粒球生成过程是完全无参数的，并开发了完全无参数的分类器GBkNN。我们提出了三个自适应条件来实现粒球的自适应生成：每个粒球的子球的加权纯度和是否增加；任意一对异构粒球之间是否存在重叠；每个粒球是否达到纯度的下界，即整个数据集的初始粒球的纯度。具体设计内容如下所述。

1）子球的纯度加权总和：纯度是为了衡量一个粒子球的质量而设计的，设计一个衡量子球纯度的指标是一个直接的思路。那么，一个颗粒状球体是否应该被拆分，取决于它的子球纯度是否大于它本身。考虑到粒球中样本越多，粒球越重要，因此我们设计了子球加权纯度和，用于衡量子球纯度。

使用条件 1 分裂后，异质颗粒球之间存在重叠。为此，我们引入了第二个条件，即异质颗粒球之间不能重叠。

2）异质粒球间的去重叠：针对粒球重叠问题，需要进一步检测是否存在异质粒球，并对重叠粒球进一步拆分细化，使决策边界更加清晰，为了提高效率，下一轮重叠检测只需要遍历已经重叠的粒球的子粒球即可。

3）一个自适应的纯度下界：另外，颗粒球的纯度应该有一个自适应的下界，下界为初始多数样本占总样本的比例，即初始颗粒球的纯度。对于少数类样本，可以考虑将错误分类样本的比例作为噪声率，也就是说，所有颗粒球的纯度必须大于初始颗粒球的纯度。

自适应粒球生成方法的基本思想如图 5所示。

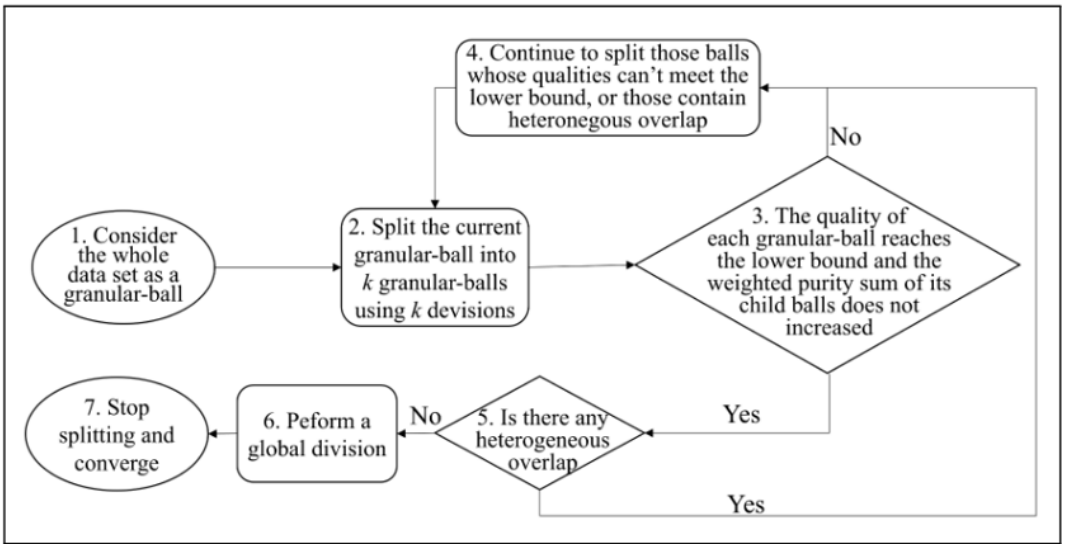

图 5. 自适应粒球生成方法的基本思想

在图 5的步骤 2 中，基于加速的粒球生成方法，使用 k division 对粒球进行拆分，其中 k 表示粒球中的类数，直接降低了计算代价。此外，如步骤 3 所示，当子球的加权纯度和大于其父球的纯度且粒球的纯度达到下界时，保留子球并检测异构粒球之间是否存在重叠。当算法收敛时，粒球的边界与数据集的边界非常一致。

4 复现细节

4.1 与已有开源代码对比

原文的代码已开源，源代码包含了“gb_origin.py”、“gb_knn.py”、“gb_accelerate_upload”、“gb_adaptive_upload.py” 四个代码文件，分别实现了已有粒球生成方法、knn 粒球生成方法、原文改进的快速粒球生成方法以及自适应的粒球生成方法。本文整个复现过程在此代码基础上进行了改进，在 “gb_adaptive_upload.py” 文件中，在 splits 与 get_label_and_purity 函数中实现。get_label_and_purity 里计算纯度同时，用 scipy.stats.entropy 算数据分布熵；splits 依纯度和熵变化决定是否分裂，进一步改进后的代码如 “gb_adaptive_entropy.py” 所示，具体如下图 6。

```
# 新增：获取分裂前的熵值
prev_entropy = entropy
# 计算分裂后子球的加权熵值和
total_entropy = 0
for key0 in gb_dict_re.keys():
    _, _, entropy_child = get_label_and_purity(gb_dict_re[key0][0])
    total_entropy += entropy_child * (len(gb_dict_re[key0][0]) / len(gb))
# The weighted purity sum of the child balls is greater than the purity of the parent ball,
# or the weighted purity sum of the child balls is less than the lower bound of purity
if p <= purity_init or weight_p > p and total_entropy <=  prev_entropy:
    gb_dict_new.update(gb_dict_re)
else:
    gb_dict_new.update(gb_single_temp)
```

图 6. 改进代码修改

4.2 实验环境搭建

实验硬件环境：配备 Intel(R) Core(TM) i5-95000 CPU @3.00 GHz 和 16 G RAM 的 PC。实验软件环境：Python 3.9。

4.3 创新点

在原文使用的自适应粒球生成方法中，纯度评估基于粒球内数据最大类别占比，若子球存在类别较多，即熵值过高，容易导致过度细分失去代表性，同时可能增加计算消耗以及过拟合风险。因此，可以增加评估条件，综合考量数据分布熵与类别纯度，高熵值表示数据分布离散混乱，低熵值表明数据集中有序。

分裂准则中，除纯度外，增加对分裂后子球数据分布熵变化的监测。若子球熵值过高或纯度提升不显著且熵增明显，暂停分裂，避免过度细分致粒球失去代表性、增加计算负担与模型过拟合风险，提升粒球生成合理性与稳定性。

5 实验结果分析

在这一部分，我将数据集拆分为十个部分，取其中一部分进行测试，以测试准确率为评价指标，验证加速粒球生成方法和自适应粒球生成方法的有效性。由于粒球生成仍然具有一定的随机性，我们对每种方法进行了十次实验，并取十次实验结果的平均分类准确率进行比较，另外 kNN 方法使用十折交叉验证结果。表 1展示了在无噪声条件下的实验平均精度。‘Acc+’、‘Adp’、‘Adp-ent’、‘Origin’ 和 kNN 分别表示本文提出加速粒球生成方法、自适应粒球生成方法、本文进行改进的方法以及已有粒球生成方法和 kNN 方法。

表 1. **平均测试精度**

Data	Acc+	Adp	Adp_ent	Origin	kNN
fourclass	**49**	103	874	293	952
svmguide1	1256	140495	**229**	3380	71149
diabetes	193	4936	**31**	1965	6485
breastcancer	35	2309	4122	**19**	344
creditApproval	**193**	3546	3758	1988	6049
votes	**51**	957	1070	365	786
svmguide3	**363**	12669	11449	3125	13497
sonar	**44**	448	490	404	726
splice	573	21319	**35**	2487	9686
mushrooms	757	530344	**351**	790	29219
Average	**351.4**	71712.6	2240.9	1481.6	13889.3

从表 1可以看出提出的两种方法，加速粒球生成方法和自适应粒球生成方法，与现有方法和 kNN 相比，在准确率上获得了相近甚至更好的性能。利用现有方法得到的决策边界不够清晰，因此，在测量 kNN 精度时，距离测试点最近的颗粒球很可能是不准确的。原文提出的

两种方法在颗粒球的分裂停止后进行全局划分，因此，在原始数据集上可以获得比现有方法更高的 kNN 准确率。本文进行的改进的方法虽然没有达到比原文的准确率更高的目标，但是也保持在一个相对较好的水平。

同样我实验验证加速粒子球生成方法的高效性，我们选择了现有的粒子球生成方法作为比较。表 2给出了两种方法在原始数据集上的运行时间，‘Acc+’、‘Adp’、‘Adp_ent’、‘Origin’和 kNN 分别表示本文提出加速粒球生成方法、自适应粒球生成方法、本文进行改进的方法以及已有粒球生成方法和 kNN 方法。

表 2. **运行时间比较**

Data	Acc+	Adp	Adp_ent	Origin	kNN
fourclass	0.994	0.996	0.996	**0.997**	0.996
svmguide1	0.959	0.956	0.956	0.960	**0.962**
diabetes	0.731	0.702	0.702	0.708	**0.736**
breastcancer	0.969	0.963	0.963	0.966	**0.970**
creditApproval	**0.669**	0.654	0.654	0.667	0.660
votes	**0.907**	0.904	0.904	0.895	0.897
svmguide3	**0.790**	0.776	0.776	0.771	0.783
sonar	**0.824**	0.820	0.820	0.818	0.814
splice	0.656	0.679	0.679	**0.688**	0.597
mushrooms	**1.000**	**1.000**	**1.000**	**1.000**	0.993
Average	**0.850**	0.845	0.845	0.847	0.841

与已有的粒球生成方法相比，原文加速方法在生成数量相近的粒球时，在大部分数据集上具有更高的准确率和效率。同时我们也可以看出自适应粒球方法在时间上消耗大幅增加，这是由于增加自适应条件后分裂过程需要更多的判断导致的时间消耗。本文对自适应粒球方法进行改进后，从表 2可以看出改进后的方法时间消耗有所减少。

6 总结与展望

本文的改进是针对自适应粒球生成方法，对纯度评估与分裂准则进行改进，虽然总体的准确率没有达到预期目标，但在一些数据集上运行时间取得下降，后续可以尝试对快速粒球生成方法进行改进。

可以进一步探究在粒球初始化时是否有更好的策略。例如，依据数据分布特性智能选择初始粒球中心，而非随机选取或简单基于距离阈值划分。在数据呈现聚类趋势的区域密集设置初始中心，于稀疏区域稀疏设置，提高生成效率与质量。通过密度估计技术，如核密度估计，精准定位高密度区域为粒球生成重点区域，减少不必要计算与不合理粒球划分，提升对复杂数据分布的适应性与生成效果。

参考文献

[1] L. Chen. Topological structure in visual perception. *Science*, 218(4573):699–700, 1982.

[2] Thomas M. Cover and Peter E. Hart. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory*, 13:21–27, 1967.

[3] Wei-Yin Loh. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1, 2011.

[4] Saber Salehi, Ali Selamat, M. Reza Mashinchi, and Hamido Fujita. The synergistic combination of particle swarm optimization and fuzzy sets to design granular classifier. *Knowledge-Based Systems*, 76:200–218, 2015.

[5] 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.

[6] Shuyin Xia, Daowan Peng, Deyu Meng, Changqing Zhang, Guoyin Wang, Elisabeth Giem, Wei Wei, and Zizhong Chen. Ball kk-means: Fast adaptive clustering with no bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):87–99, 2022.

[7] Shuyin Xia, Zhao Zhang, Wenhua Li, Guoyin Wang, Elisabeth Giem, and Zizhong Chen. Gbnrs: A novel rough set algorithm for fast adaptive attribute reduction in classification. *IEEE Transactions on Knowledge and Data Engineering*, 34:1231–1242, 2020.

[8] Shuyin Xia, Shaoyuan Zheng, Guoyin Wang, Xinbo Gao, and Binggui Wang. Granular ball sampling for noisy label classification or imbalanced classification. *IEEE Transactions on Neural Networks and Learning Systems*, 34(4):2144–2155, 2023.

[9] Yinghua Zhou, Hong Yu, and Xuemei Cai. A novel k-means algorithm for clustering and outlier detection. In *2009 Second International Conference on Future Information Technology and Management Engineering*, pages 476–480, 2009.