CHAOTIC-TO-FINE CLUSTERING FOR UNLABELED PLANT DISEASE IMAGES

A PREPRINT

Uno Fang
Deakin-SWU Joint Research Centre on Big Data
Deakin University
Burwood, VIC 3125, Australia
uno.fang@deakin.edu.au

Jianxin Li *
School of IT
Deakin University
Geelong, VIC 3216, Australia
jianxin.li@deakin.edu.au

Xuequan Lu
School of IT
Deakin University
Geelong, VIC 3216, Australia
xuequan.lu@deakin.edu.au

Mumtaz Ali
School of IT
Deakin University
Geelong, VIC 3216, Australia
mumtaz.ali@deakin.edu.au

Longxiang Gao
School of IT
Deakin University
Geelong, VIC 3216, Australia
longxiang.gao@deakin.edu.au

Yong Xiang
School of IT
Deakin University
Geelong, VIC 3216, Australia
yong.xiang@deakin.edu.au

January 19, 2021

ABSTRACT

Current annotation for plant disease images depends on manual sorting and handcrafted features by agricultural experts, which is time-consuming and labour-intensive. In this paper, we propose a self-supervised clustering framework for grouping plant disease images based on the vulnerability of Kernel K-means. The main idea is to establish a cross iterative under-clustering algorithm based on Kernel K-means to produce the pseudo-labeled training set and a chaotic cluster to be further classified by a deep learning module. In order to verify the effectiveness of our proposed framework, we conduct extensive experiments on three different plant disease datasets with five plants and 17 plant diseases. The experimental results show the high superiority of our method to do image-based plant disease classification over balanced and unbalanced datasets by comparing with five state-of-the-art existing works in terms of different metrics.

Keywords Under-clustering · Similarity Learning, Self-supervised Learning · Automated Annotation

1 Introduction

Plant disease detection, one of the most popular topics in Precision Agriculture (PA), has attracted remarkable attentions recently. A general procedure of plant disease detection system consists of five fundamental steps: image acquisition, dataset annotation, image processing, feature extraction, classification [1]. To our knowledge, the majority of studies of plant disease detection studied on improving image processing, feature extraction and image classification based on the well-labelled public datasets. For example, [2], [3], [4] and [5] introduced novel convolutional neural networks (CNNs) as image classifiers and/or proposed data segmentation and augmentation methods to increase the recognition and

*Corresponding author: E-mail: jianxin.li@deakin.edu.au
classification accuracy. By contrast, the dataset annotation, as a fundamental basis for other processing in plant disease detection, has been sparsely handled. Data annotation is often being conducted manually by agricultural experts, which consumes a lot of labour, material resources and time. Therefore, it expects an unsupervised learning method to enhance the efficiency and effectiveness of plant disease dataset annotation to minimise the manual cost from agricultural experts.

The essence of grouping plant disease images is to extract decent representations of input images and to develop metrics to measure the pairwise similarity for categorisation. Supervised learning can hardly deal with this problem, as it greatly depends on the labeled data and needs to learn all types of data to achieve decent classification outcomes. Therefore, the dataset annotation robustly relies on unsupervised learning and self-supervised learning (clustering). Previous clustering algorithms, such as K-means, Kernel K-means and Spectral clustering, can be utilised to cluster unlabeled plant disease images. However, these algorithms tend to work for balanced data, and greatly rely on the prior knowledge (e.g. ground-truth class number etc). Although there are rarely few clustering methods on plant disease dataset annotation, many state-of-the-art unsupervised methods were proposed for unlabeled face images, such as Approximate Rank Order Clustering (AROC) [6] and FINCH [7], both of which are graph-based clustering. These methods are developed to deal with various application scenarios from social media to law enforcement, where the number of faces in the collection can reach hundreds of millions. AROC is designed to cluster large-scale unbalanced data since it is developed based on rank order clustering which is often utilised to handle imbalanced data. However, it still faces difficulties in dealing with balanced data. In addition, FINCH is a “parameter-free” clustering algorithm, which, on the other hand, somewhat loses certain control during clustering. It essentially utilises the first neighbour of each point to produce the clustering hierarchy and iteratively merges clusters. In this work, a plant has different diseases which appear to be similar in images, such as potato early blight and potato late blight, if such images are clustered/merged together by FINCH, it would pose negative impact on the clustering accuracy. As for the self-supervised learning, [8], [9], [10] and [11] combined clustering methods (mostly K-means) and deep learning models where the clustering results are attached with pseudo-labels to train deep learning models, but they greatly depend on priori knowledge and the number of categories of the input dataset. They emphasise over-clustering principally as a reliable approach to allow CNNs to learn representations, but over-clustering would lead to the loss of focus and poor templates on representations.

To address the grouping of plant disease image datasets without knowing the data structure, we propose a novel framework in this paper. Our core observation is that Kernel K-means can easily produce some accurate clusters and one chaotic cluster (i.e., abnormal cluster) with tuning the threshold. Motivated by this, we attempt to first produce some accurate clusters and an abnormal cluster with a novel cross iterative clustering algorithm. These clusters are then merged, and fed into a simple image classifier for Top-µ results and a dual-component similarity measurements module including a ResNet50 [12] based similarity finder and a Siamese similarity network. The final result is computed as the average of the two similarity scores in the collaborative module. Our approach is denoted as CIKICS (Cross Iterative Kernel K-means enhanced with Image Classification and Similarity Measurements). We perform exhaustive evaluation on the proposed CIKICS by conducting comparative experiments to demonstrate its outstanding performance comparing to existing methods on image clustering, in which CIKICS has great potential to advance the existing system of plant disease detection. The contributions of this paper are as follows.

- We develop a under-clustering algorithm (cross iterative Kernel K-means), by leveraging the sensitivity of the parameter in the Radial Basis Function (RBF) Kernel. It produces many small but highly accurate clusters, which are merged and pseudo-labeled as the training set for follow-up deep learning module.

- To classify the non-clustered data, we introduce the collaboration of a CNN-based image classifier and two similarity measurements (i.e., a pre-trained CNN-based similarity finder and a trained Siamese network).

- We design the per-cluster quality measure to evaluate the performance of clustering algorithms, as a complement of existing measures.

- We conduct extensive experiments on three differently structured datasets to verify the performance of the proposed clustering framework using a set of metrics.

The remaining of this paper is structured as follows. In Section 2, we review the related work. Section 3 elaborates our proposed framework. Experimental settings and results analysis are presented in Section 4. Lastly, Section 5 provides the conclusion.
2 Related Work

2.1 Unsupervised Clustering Algorithms

2.1.1 Partitioning Clustering

For center-based clustering, K-means is a widely used but fragile algorithm which is sensitive to the selection of the initial K centroids. Fortunately, K-means++ [13] was proposed to address the initial centroids selection issue. In addition, instead of using Euclidean distance, Kernel K-means utilises kernel to calculate the distance for clustering, which can cluster non-linearly separable data [14]. A Kernel function utilises a decent nonlinear mapping from the original space to a higher-dimensional feature space. If we directly adopt a Kernel function to the original data, it would lead to a high cost of time and the low accuracy, from our observation. As [15] illustrated, three assumptions were expected to be fulfilled so that the use of K-means (including K-means++) would succeed in researching, where (i) the variance of the distribution of each attribute (variable) is assumed to be spherical in K-means; (ii) the variances of all variables are the same and (iii) and observations are clustered into each cluster approximately equally. Additionally, K-medoid (such as PAM, CLARA and CLARANS) [16] has similar issues with K-means [17]. Therefore, to reach these three assumptions would restrain the applicability of these algorithms on different datasets. Most of spectral clustering algorithms [18] need to calculate the complete similarity graph Laplacian matrix and even have secondary complexity, which severely limits their scalability to large data sets. Same as Kernel K-means, spectral clustering is used to obtain non-linear separable clusters in the input space. Although their performances are comparable, there is limitation of these algorithms, where they tend to group the data evenly rather than clustering it according to similarity for matching the ground truth.

2.1.2 Agglomerative/Divisive Hierarchical Clustering

Density-based clustering algorithms, such as DBSCAN, OPTICS and DENCLUE, have the capability of clustering non-spherical shaped clusters [19]. Although DBSCAN is the basis of all density-based clustering algorithms, it is sensitive to outliers and inflexible to varying density data since it is not developed to cluster every data point [19]. To achieve our research goal, all data points need to be clustered. However, these algorithms cannot be applied. There are many Hierarchical Agglomerative Clustering (HAC) algorithms [20, 21, 22], but these algorithms usually have the weak capability of clustering large-scale data due to the computation complexity. To deal with large-scale clustering efficiently, AROC was proposed by [6], which is a method based on rank order clustering with Approximate Nearest Neighbour (ANN) targeting to estimate whether there shall be an edge between a node and its k Nearest Neighbours (kNN). [7] introduced a parameter-free clustering algorithm, FINCH, on the basis of the hierarchical clustering algorithm, adopting the notion of the first neighbour relationship. The above mentioned algorithms have been evaluated with good performance on unconstrained face images grouping. However, these methods are not transferable as they are application oriented. Most importantly, some of them [22, 6] were density-aware and were devised to process data with unbalanced density.

2.2 Learning-based Self-supervised Clustering Methods

Recently, besides the above state-of-the-art unsupervised clustering algorithms, there is an increasing number of self-supervised clustering methods based on deep learning. For example, using an auto-encoder to optimise and learn features with existing clustering methods [8, 10, 23, 11]. [8] proposed a recurrent framework which contains a forward pass of agglomerative clustering and a backward pass of CNN-based representation learning. [10] proposed an Improved Deep Embedded Clustering (IDEC) algorithm to address the problem of data structure preservation. These deep learning-based methods mainly use existing clustering methods as a feature representation learning scheme for training deep learning models [8, 9, 10, 23] or as a means of generating pseudo-labels [11]. However, These methods have a higher potential to fail when clusters with similar features exist, which limits their application scenarios, especially when dealing with plant disease images, i.e. a plant has several diseases, and different lesions may not be identified correctly. Moreover, it is brilliant to include pseudo-labels in the self-supervised framework [11], but the framework design is not effective for high-accurate clusters.

3 Overview of Proposed Approach

As demonstrated in Figure 1, the framework consists of four functional parts, which are feature extraction, cross iterative clustering, image classification and similarity measurements. In the feature extraction step, the n-dimensional representations are extracted by leveraging the ResNet50, and are reduced into 2-dimensional vectors with t-SNE [24]. In the cross iterative clustering step, the 2-dimensional features are first transformed into kernelised vectors which
can easily enable a chaotic abnormal cluster and some small-sized highly accurate normal clusters. These clusters are merged and pseudo-labeled as the training set. In step 3 and Step 4, we introduce the collaboration of a CNN-based image classifier, a ResNet50-based similarity finder and a Siamese network, which are pre-trained or trained over the above obtained training set to predict the groups of the images in the abnormal cluster.

3.1 Plant Disease Image Representation

We utilise pre-trained CNN architectures, ResNet50, VGG16, VGG19 [25], Xception [26], Inception V3 [27] and MobileNet [28] as options, to extract image features and to calculate the Euclidean distance representing similarity for each pair of images in the abnormal cluster. We compare these CNN architectures, and utilise K-means as the baseline to evaluate their performance on the dataset PV AG (Table 3). Their performances are shown in Table 1 where we can obviously observe a prominent capability of ResNet50. Thus we adopt ResNet50 in our experiments.

Table 1

| CNN Model  | PRE | Rec | F1  | NMI | PUR | Run (sec) |
|------------|-----|-----|-----|-----|-----|-----------|
| ResNet50   | 0.958 | 0.349 | 0.272 | 0.390 | 171 |
| Xception   | 0.460 | 0.099 | 0.163 | 0.088 | 313 | 462 |
| Inception V3 | 0.349 | 0.075 | 0.123 | 0.040 | 267 | 547 |
| MobileNet  | 0.811 | 0.177 | 0.290 | 0.248 | 0.396 | 283 |
| VGG16      | 0.836 | 0.198 | 0.324 | 0.259 | 0.375 | 174 |
| VGG19      | 0.916 | 0.204 | 0.334 | 0.267 | 0.377 | 183 |

Briefly in our case, denote a feature vector set by $X = \{x_1, \ldots, x_N\}, x_i \in \mathbb{R}^{512}$, each feature vector is extracted from each image in the dataset by ResNet50. Then we use t-SNE to calculate the similarity probability $p_{ij}$ of point $x_i$ and $x_j$ in the feature vector set $X$ in the 512-dimensional space, and map the similarity probability $q_{ij}$ of point $x_i'$ and $x_j'$ in the corresponding feature vector $X'$ ($X' = \{x_1', \ldots, x_N'\}, (x_i' \in \mathbb{R}^2)$) in the 2-dimensional space. This also allows a dimensionality reduction.
3.2 Cross Iterative Kernel K-means Clustering

Given that we need to cluster unlabeled data into multiple classes, clustering algorithms are essential in this work. Considering that the clustering data is non-linearly separable, we adopt the Kernel K-means [14], which was developed based on K-means clustering algorithm with leveraging a kernel function.

Radial Basis Function (RBF) Kernel. The kernelisation targets to increase the dimensionality of vectors by comparing vectors pairwise. This way facilitates higher clustering accuracy, according to our empirical observation. We select the RBF kernel [30] which is a widely used kernel function in diverse kernelised learning algorithms. Given that we have obtained the feature vector set $X'$ from t-SNE, the RBF kernel [30] on two points $x'_i$ and $x'_j$, demonstrated as feature vectors in higher dimensional input space, is defined as Eq. 1.

$$K(x'_i, x'_j) = \exp \left( -\frac{\|x'_i - x'_j\|^2}{2\sigma^2} \right),$$

(1)

where $K(x'_i, x'_j) = K(x'_j, x'_i)$, and $\|x'_i - x'_j\|^2$ refers to the squared Euclidean distance between two feature vectors. Due to the fact that the value of the RBF kernel reduces with increasing distance, and it varies between 0 (in the limit) and 1 (when $x'_i = x'_j$). The kernelised feature vector induce a symmetric matrix in $N$-dimensional space as $Z (Z = \{z_1, z_2, \ldots, z_N\}, (z_i \in \mathbb{R}^N))$.

K-means++ Algorithm. In the framework, we apply K-means++ algorithm [13] instead of standard K-means. K-means++ can ensure more intelligent initialisation of the centroids and improve the quality of clustering. Given $Z (Z = \{z_1, z_2, \ldots, z_N\}, (z_i \in \mathbb{R}^N))$ obtained from the RBF kernel function, we define an integer $k$ (the value of $k$ can be defined around $2\sqrt{N}$) for K-means++ to choose $k$ centers $C$. $D(z)$ represents the shortest distance from a data point to the closest center in $C$.

Cross Iterative Clustering Algorithm. As known, $\sigma$ in Eq. 1 is a parameter that is set prior to the kernel computation. $\sigma$ is utilised to configure the sensitivity to differences in feature vectors, which conversely relies on various characteristics, such as input space dimensionality, data size and feature normalisation. According to Eq. 1 we can obtain:

$$\sigma = -\frac{\|x'_i - x'_j\|}{\sqrt{2 \ln K(x'_i, x'_j)}}$$

(2)

Breakthrough. In the case when $\sigma \to 0$, $K(x'_i, x'_j) \to \infty$, which means even similar elements will be pushed apart in the kernel space. Otherwise, when $\sigma \to \infty$, $K(x'_i, x'_j) \to 0$, which means even dissimilar elements will be gathered in the same cluster. We empirically found that appropriately considering more attributes of the applied dataset as variables in manipulating the value reduction of $\sigma$ could lead to a higher accuracy of clusters. However, one of these clusters (the abnormal cluster), which contains kernelising-pushed-away elements, is chaotic and of dramatically large size. This is our key observation and the foundation of our framework, which will be demonstrated in details below. This abnormal cluster enables us to build the cross iterative clustering algorithm, which will filter out images that are hardest to cluster.

![Figure 2: Structure of image classification based on CNN.](image)

Parameters. To obtain the “optimal” value of $\sigma$ to kernelise time series data, we refer to a Global Alignment Kernel (GAK) method proposed by [31]. The bandwidth $\sigma$ is set as a multiple of the median of Euclidean distance of pairwise
points observed in different time series of the training set, scaled by the square root of the maximum number of time points for each data point (the input space data dimensionality) \( N \). It is defined as:

\[
\sigma = \text{median}((x'_i - x'_j)^2) \times \sqrt{N}
\]  

(3)

As shown in Eq. 3 only data points and their space dimensionality are considered as parameters. To establish the cross iterative clustering, we need to involve additional parameters in defining \( \sigma \). There are few parameters to define, which are the iteration time \( \beta \), the data length \( N \) and the pre-defined cluster number \( k \). We re-define their relations based on Eq. 3 as:

\[
\sigma = \text{median}((x'_i - x'_j)^2) \times \frac{\ln k}{\ln N \times \beta}
\]  

(4)

We exclude the factor of the input space data dimensionality \( d \) because it is an invariant in our framework.

**Algorithm**  Given above, we develop the Cross Iterative Kernel K-means as shown in Algorithm 1 and we expect to annotate the merged normal clusters \( M \) with pseudo-labels, e.g. first cluster will be labeled as “\( M_1 \)”, second as “\( M_2 \)”, etc. The abnormal cluster and incorporated normal clusters with pseudo-labels produced by the Cross Iterative Kernel K-means provide the flexibility to the clustering framework for deep learning models.

**Algorithm 1:** Cross Iterative Clustering

**Input:** t-SNE feature vector \( X' \), RBF Kernel \( K(\cdot) \), K-means++ \( F(\cdot) \)

\( K(X') = Z = \{z_1, z_2, \ldots, z_N\} \); Build an empty array \( G_{nm} \) for collecting normal clusters;

Set the initial abnormal cluster \( A = Z \);

for \( i = 1, 2, \ldots, \beta \) do

\( F(A) = C = \{C_1, C_2, \ldots, C_k\} \);

for \( j = 1, 2, \ldots, k \) do

Find the cluster with the max size as the abnormal cluster \( C_j \);

Set \( A = C_j \);

end

Insert the rest \( k - 1 \) normal clusters into \( G_{nm} \);

end

Obtain the final \( A \);

Obtain \( G_{nm} = \{G_1, G_2, \ldots, G_k\} \), \( h = (k - 1) \times \beta \);

Set an new array \( B \) for collecting the centroid of points in each normal cluster of \( G_{nm} \):

for \( i = 1, 2, \ldots, h \) do

Get all point data of \( G_i \) from \( X' \);

Calculate the centroid \( a_i \) of \( G_i \);

Insert \( a_i \) into \( B \);

end

Obtain the merged clusters \( M = F(B) \)

### 3.3 Image Classification with CNN

Given pseudo-labeled normal clusters \( M \) as the training set, and the final abnormal cluster \( A \) as predicting set, we involve CNN-based image classifier to learn features of images in the training set \( M \), then predict the top-\( \mu \) classes of each image \( x_i \) in \( A \). The detailed deployment of our image classifier is demonstrated in Figure 2. We conduct a comparative experiment on the dataset PVAG (Table 3) with different \( \mu \) in the range \((1, n)\), where \( n \) is the number of clusters in the training set, to compare the accuracy of the correct classes in the top-\( \mu \) list, and the runtime for further similarity measurements. We observe that the accuracy is \( 0.97 \pm 0.02 \) in the range, but the runtime differs greatly since more potential classes lead to higher computation complexity for deep learning models. Additionally, we observe that, when \( \mu = 3 \), the framework works efficiently and effectively.
3.4 Similarity Measurements

Although a simple image classifier is reliable to detect edges, textures and colours, it is not proficient in obtaining high-level features, such as lesion features. Also, the noise in the training set might have negative impact on the image classifier. As such, we include two more similarity measurements as weights to help classifying images in the abnormal cluster into the merged normal clusters $M$.

3.4.1 Similarity Finder with Pre-trained CNN

First of all, we utilise ResNet50 to calculate the similarity score of each image in the abnormal cluster to classes of the training set (normal clusters with pseudo-labels), by comparing the Euclidean distance. In particular, the similarity of two image feature vectors $x_i$ and $x_j$ is defined as their squared Euclidean distance [32]:

$$D(x_i, x_j) = \|x_i - x_j\|_2^2$$

(5)

The smaller the distance $D(x_i, x_j)$ is, the more similar the two images $x_i$ and $x_j$ are. We cast the similarity ranking problem as a neighbour search problem in Euclidean space. From this similarity finder, for each predicting image, we can get its similarity scores $S(x_i) \ (x_i \in A)$ towards the top-$\mu$ classes (i.e. pseudo labels) from $M$.

3.4.2 Siamese Network

We introduce a Siamese network as a second similarity measure to calculate the similarity scores. It can be finally decided what classes those images in the abnormal cluster shall be classified into. Similar to the image classifier (above section), we train the Siamese network on the training set, and utilise the trained model to calculate the similarity score of each image in the abnormal cluster to each class in the top-$\mu$ classes. Given the nature of Siamese network, we input two same-sized groups of images to compare the similarity between them.

In the Siamese network, images are compared in paired groups, the $sigmoid$ layer will output the similarity score. We have obtained a predicting image set $A$ (i.e. the final abnormal cluster) and a pseudo-labeled image set $M$, where $M = \{M_1, \ldots, M_n\}$ and $n$ indicates the number of pseudo-labeled clusters. We utilise Siamese Network to compute the similarity score between $x_i$ in $A$ and $M_j$ (a pseudo-labeled cluster) in $M$. For computation purpose, we need to expand $x_i$ (a feature vector of an image) to $A'_1$ (a feature vector set) with blank images, where the size of $A'_1$ is expected to be equal to the size of $M_j$. The similarity score between $A'_1$ and $M_j$ is $S'(x_i) = \frac{\sum_{i=1}^{n} s'_i}{n}$.

**Final class determination.** Through the Siamese network, we get a second similarity score $S'(x_i)$ for $x_i$ in $A$. Finally, we calculate the average of $S'(x_i)$ and $S(x_i)$ (i.e. the similarity score obtained in Similarity Finder), and the class with the highest average score will be the image’s final class.

4 Experimental Results

4.1 Dataset and Preprocessing

In this work, we use a well-labeled plant disease image dataset, PlantVillage [33], which contains 54,305 images from 14 kinds of plants and 38 plant diseases, for our experiments. Since unlabeled datasets are expected in our work, we de-label the PlantVillage dataset. Furthermore, to fit the computing, we first re-create two sub datasets from this dataset as shown in Table 3. The first one is denoted as PVAG which consists of 8 plant disease classes from apple and grape. PVAG is a relatively large-scale dataset with balanced data distribution. The second is made up of 4 plant disease classes of images from peach and strawberry. This dataset is abbreviated as PVPS with unbalanced data structure.

In addition, we utilise another public annotated plant disease image dataset, Citrus Disease Dataset (CDD) [34], to evaluate the clustering algorithm, and the details of CDD are shown in Table 5. CDD is a small-scale dataset with an unbalanced data structure. Given the above three datasets (the re-created two and CDD), we are able to cover cases for balanced and unbalanced data.
Table 3
DETAILS OF DATASETS.

| Dataset | Plant | Disease / Health | Image Amount |
|---------|-------|------------------|--------------|
| PVAG    | Apple | Scrab            | 630          |
|         |       | Black Rot        | 621          |
|         |       | Cedar Rust       | 275          |
|         |       | Healthy          | 1645         |
|         | Grape | Black Rot        | 1180         |
|         |       | Black Measles (Esca) | 1383      |
|         |       | Healthy          | 423          |
|         |       | Leaf Blight      | 1076         |
| PVPS    | Peach | Bacterial Spot   | 2298         |
|         |       | Healthy          | 361          |
|         | Strawberry | Healthy | 457          |
|         |       | Leaf Scorch      | 1110         |
| CDD     | Citrus | Black Spot      | 172          |
|         |       | Canker           | 164          |
|         |       | Greening         | 205          |
|         |       | Healthy          | 59           |
|         |       | Melanose         | 14           |

4.2 Evaluation Metrics

We use external pairwise evaluation metrics [6] to calculate Precision, Recall and F1 score. We also employ the purity (PUR) and normalised mutual information (NMI) [23] as evaluation metrics.

Since above metrics can only reveal the overall quality of the clustering results, we develop an additional accuracy evaluation metric on the quality of each individual cluster. Since we aim to address the annotation of an unlabeled dataset, the high accuracy of each cluster is also crucial in evaluation. Therefore, we design a Per-cluster Quality Measure. For simplicity, we calculate the percentages of all same-label elements in each cluster, and find the highest one as the cluster accuracy which is denoted as Acc. We can also leverage Acc to demonstrate the overall performance. The average accuracy of all clusters ($\text{AveAcc} = \frac{\sum_{i=1}^{h} \text{Acc}_i}{h}$), where $h$ is the number of clusters, and the lowest accuracy amongst all clusters ($\text{MinAcc} = \min(\text{Acc}_i)$) can be calculated. Also, in the resulting clusters, the percentage of Acc that is 100% is denoted as $\text{FPP} = \frac{\sum_{i=1}^{h} [\text{Acc}_i = 1]}{h}$. And the percentage of Acc that is lower than 80% is denoted as $\text{LPP} = \frac{\sum_{i=1}^{h} [\text{Acc}_i < 0.8]}{h}$.

4.3 Baselines

We compare our CIKICS to existing clustering methods (including both classic and recent ones). We select 5 baselines which are divided into 2 different types of clustering algorithms: (1) algorithms without requiring the pre-defined number of clusters (including those that require hyper-parameter/threshold settings) and (2) algorithms with requiring the number of clusters as input. The first category of algorithms are AROC [6] and FINCH [7], where AROC takes
thresholds as input, and FINCH requires no parameters at all. Both of them are recently proposed. The second type of algorithms include K-means [13], Kernel K-means [14] and Spectral clustering [18].

4.4 Results

4.4.1 Comparison on balanced dataset

In Table 4, we compare the performance of our CIKICS with the 5 clustering methods on the balanced dataset, PVAG.

| Algorithm       | Clusters | PRE  | REC  | F1   | NMI  | PUR  |
|-----------------|----------|------|------|------|------|------|
| K-means [13]    | 8        | 0.878| 0.770| 0.820| 0.618| 0.676|
| KKM [35]        | 8        | 0.887| 0.776| 0.828| 0.564| 0.638|
| Spectral [18]   | 8        | 0.789| 0.800| 0.795| 0.622| 0.662|
| AROC (0.4) [6]  | 8        | 0.605| 0.995| 0.753| 0.544| 0.617|
| FINCH [7]       |          |      |      |      |      |      |
| CIKICS          |          |      |      |      |      |      |

The proposed method (last row) generates the highest clustering Precision, F1 score, NMI and Purity. On the other hand, AROC has a highest recall value on PVAG. However, if we look into further the data distribution in the generated clusters in Table 5, apparently, AROC was particularly developed to cluster data with an unbalanced structure. Since precision and recall metrics involve errors [6], AROC clustering produces the highest recall.

From Table 5, K-means, Kernel K-means, Spectral clustering and CIKICS can push data balancedly into clusters. This leads them to comparable performance if they are parameterised similarly. By contrast, the data distribution of CIKICS is closer to the ground-truth data structure (Table 3). On the other hand, AROC has been proved with the capability of dealing with well unbalanced and large-scale data. It clusters data based on the pre-defined threshold. We test several thresholds to show how AROC performs on PVAG, which reveals the limitation and inflexibility of AROC on the balanced data. FINCH was proposed with the gimmick of a parameter-free clustering algorithm, but its performance on PVAG is less satisfactory, and we simply exclude it in further comparisons.

| Cluster # | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----------|---|---|---|---|---|---|---|---|
| K-means [13] | 751 | 1128 | 1075 | 640 | 614 | 1436 | 731 | 857 |
| KKM [35]   | 954 | 1398 | 786  | 548 | 1075 | 755  | 654 | 1062|
| Spectral [18] | 931 | 1075 | 894  | 964 | 1090 | 1008 | 690 | 580 |
| AROC [6]   | 631 | 2584 | 2280 | 279 | 421  | 1073 | 2   | 2   |
| CIKICS [7] | 697 | 1193 | 1383 | 1047| 1089 | 1086 | 423 | 314 |

The numbers of images in each cluster for compared methods.

Figure 4: Evaluation results with different pre-defined cluster numbers on PVPS.

In clustering, each sample is taken as an individual cluster, it will have a higher accuracy but a lower recall rate. On the contrary, all samples are clustered in the same cluster will have a higher F1 and recall rate, but a low precision.
Figure 5: Evaluation results with different pre-defined cluster numbers on CDD.

Table 6
CLUSTERING PERFORMANCE ON PVAG (pt.2).

| Algorithm       | Ave Acc | Min Acc | FPP    | LPP    |
|-----------------|---------|---------|--------|--------|
| K-means [13]    | 0.878   | 0.553   | 0.125  | 0.250  |
| KKM [35]        | 0.831   | 0.544   | 0.125  | 0.375  |
| Spectral [12]   | 0.868   | 0.558   | 0.125  | 0.250  |
| FINCH (22) [6]  | 0.881   | 0.550   | 0.020  | 0.163  |
| CIKICS          | 0.894   | 0.556   | 0.125  | 0.250  |

Table 7
CLUSTERING PERFORMANCE ON PVPS (pt.1).

| Algorithm       | Clusters | PRE   | Rec   | F1    | NMI   | PUR   |
|-----------------|----------|-------|-------|-------|-------|-------|
| K-means [13]    | 4        | 0.841 | 0.604 | 0.702 | 0.541 | 0.800 |
| KKM [35]        | 4        | 0.688 | 0.547 | 0.598 | 0.319 | 0.703 |
| Spectral [18]   | 4        | 0.835 | 0.594 | 0.694 | 0.493 | 0.773 |
| AROC (0.05) [6] | 4        | 0.705 | 0.999 | 0.827 | 0.245 | 0.533 |
| FINCH           | 4        | 0.923 | 0.006 | 0.012 | 0.013 | 0.544 |
| CIKICS          | 4        | 0.849 | 0.657 | 0.715 | 0.619 | 0.800 |

As for the per-cluster accuracy, we list them in Table 6. Interestingly, our proposed framework outperforms other methods. Figure 3 intuitively demonstrates that CIKICS has the outstanding performance with different cluster numbers (around the ground-truth class number). For K-means, Kernel K-means, Spectral clustering and CIKICS, the clustering performance is generally better when the cluster number is closer to the real number of classes. Nevertheless, the performance of AROC changes gently, except the increase when the cluster number approaches the ground-truth number of classes. Furthermore, we can tell that sometimes other clustering methods outperform ours in these line plots. For example, as shown in Figure 3a the F1 score of spectral clustering at 6 clusters is slightly higher than others because it divides the smallest sized class in the ground-truth 8 classes balancedly into other 5 clusters. At the same time, as shown in Figure 3b K-means demonstrates higher NMI measure at 6 clusters, which means the clustering result is highly correlated with the ground-truth. As known, there are 8 classes in the dataset. When we look into the result we found that images in two classes with the smallest size have been separated evenly into the other 6 clusters. This enables the other 6 classes dominate the resulted 6 clusters, which increases the mutual information between the result and the ground truth.

4.4.2 Comparison on unbalanced datasets

To further evaluate the capabilities of dealing with unbalanced data, we use the re-created PVPS from PlantVillage. The performance is shown in Table 7.

AROC is generally better in F1 score according to Table 6 and Figure 4. However, as shown in Table 8 the Min Acc is the lowest, which indicates there is still room for improvement on AROC. Its Ave Acc arrives at 0.928. Impressively, its FPP reaches 0.803, i.e. 4/5 clusters has 100% same-class identities. Also, the clustering result on PVPS produced by AROC induces the lowest LPP. Interestingly, our framework is still comparable to AROC over the dataset with an unbalanced structure with 0.619 NMI, 0.8 purity, 0.928 Ave Acc and 0.715 Min Acc. We observe that the proposed
method has a noticeable applicability and scalability for both unbalanced and balanced data, especially when comparing it with K-means, Kernel K-means and Spectral clustering.

Besides the above comparisons, the comparative results on small-scale balanced data is shown in Table 9 and 10. AROC has been removed from the comparative study on CDD, since we observe through experiments that AROC has a great limitation on handling small-scale data. Specifically, AROC relies on the pre-defined threshold, and we test the threshold from 1000 to 0.001 (with a step size of 50 from 1000 to 100, 10 from 100 to 10, 1 from 10 to 1, 0.1 from 1 to 0.1, then 0.01 and 0.001). Unfortunately, we always obtain a single cluster using AROC. This demonstrates that our method is more robust than AROC to small-scale datasets.

Regarding the clustering accuracy, CIKICS reaches 0.830 F1 score, 0.636 NMI, 0.683 purity and 0.891 Ave Acc on PVAG (7323 images, 8 plant disease classes) in Table 4 and 6, which obviously outperforms other clustering algorithms. CIKICS achieves 0.715 F1 score, 0.616 NMI, 0.800 purity and 0.928 Ave Acc on PVPS (4222 images, 4 plant disease classes) in Table 7 and 8. This is relatively inferior to AROC but still reveals the feasibility of CIKICS on unbalanced data. For the evaluation of clustering algorithms on CDD (604 images, 5 plant disease classes), CIKICS has shown the capability of dealing with the small-scale datasets with 0.466 F1 score, 0.331 NMI, 0.537 purity and 0.779 Ave Acc in Table 9 and 10. The performance is significantly higher than the other clustering algorithms on CDD. Moreover, according to Figure 5, we can see that different pre-defined cluster numbers lead to significant up-and-downs on F1 score, NMI and purity measures. Specifically, the outstanding part appears generally for 4 or 5 clusters. This is mainly due to the challenge of the small scale of the dataset. As the ground-truth class number of CDD is 5, there is an obvious limitation for CIKICS to produce highly accurate clusters as training set to learn deep models for further image classification. When the pre-defined cluster number is more than 5, we can tell that the performance of CIKICS is generally better than others because the larger pre-defined cluster number provides more potential for CIKICS to cluster every image into the correct group.

### Table 8

| Algorithm      | Ave Acc | Min Acc | FPP | LPP |
|----------------|---------|---------|-----|-----|
| K-means [13]  | 0.888   | 0.711   | 0.250| 0.250|
| KKM [15]      | 0.891   | 0.598   | 0.250| 0.250|
| Spectral [18] | 0.901   | 0.715   | 0.250| 0.250|
| FINCH (69) [6]| 0.917   | 0.471   | 0.455| 0.182|
| AROC (0.05) [7]| 0.920   | 0.333   | 0.803| 0.175|
| CIKICS        | 0.926   | 0.715   | 0.000| 0.250|

### Table 9

| Algorithm      | Clusters | PRE  | Rec  | F1   | NMI  | PUR  |
|----------------|----------|------|------|------|------|------|
| K-means [13]  | 5        | 0.487| 0.362| 0.415| 0.226| 0.558|
| KKM [15]      | 5        | 0.468| 0.373| 0.415| 0.102| 0.486|
| Spectral [18] | 5        | 0.486| 0.360| 0.414| 0.129| 0.463|
| FINCH [7]     | 201      | 0.599| 0.009| 0.018| 0.021| 0.353|
|               | 62       | 0.576| 0.036| 0.068| 0.081| 0.425|
|               | 20       | 0.551| 0.127| 0.206| 0.071| 0.412|
| CIKICS        | 5        | 0.521| 0.422| 0.466| 0.331| 0.537|

### Table 10

| Algorithm      | Ave Acc | Min Acc | FPP | LPP |
|----------------|---------|---------|-----|-----|
| K-means [13]  | 0.670   | 0.571   | 0.000| 0.800|
| KKM [15]      | 0.775   | 0.514   | 0.400| 0.600|
| Spectral [18] | 0.736   | 0.593   | 0.000| 0.800|
| FINCH (20) [7]| 0.685   | 0.366   | 0.000| 0.750|
| CIKICS        | 0.779   | 0.538   | 0.400| 0.600|
Table 11
EVALUATION OF DEEP LEARNING MODULE ON PVAG. IM: IMAGE CLASSIFIER. SF: PRE-TRAINED CNN-BASED SIMILARITY FINDER. SN: THE SIAMESE NETWORK.

| Models   | Clusters | PRE  | Rec  | F1   | NMI  | PUR  |
|----------|----------|------|------|------|------|------|
| IM       | 8        | 0.883| 0.736| 0.803| 0.622| 0.675|
| IM&SF    | 8        | 0.865| 0.751| 0.804| 0.606| 0.643|
| IM&SN    | 8        | 0.887| 0.735| 0.804| 0.615| 0.680|
| CIKICS   | 8        | 0.895| 0.797| 0.835| 0.636| 0.683|

4.4.3 Evaluation on deep learning module

To evaluate the performance with part or the whole deep learning module in CIKICS, we conduct an ablation study in Table 11. It is obvious that CIKICS involving all three deep learning models outperforms other combinations. The image classifier is always included, as it is fast to produce a potential class list for each non-clustered image.

Remark. CIKICS is devised on top of Kernel K-means which is generally used to process balanced dataset. With integrating image classification and similarity measurements in our framework, CIKICS has the ability to group the similar elements without considering the ground-truth data structure. It is also robust to different scales of datasets. In brief, the proposed CIKICS has been provided with a sound capability in handling unbalanced/balanced data and large/small-scale data.

5 Conclusion

In this paper, we have presented a novel clustering framework, CIKICS, for grouping unlabelled plant disease images without any supervision. This method is built on our key observation that Kernel K-means can produce some accurate clusters and one chaotic cluster (i.e., abnormal cluster). This leads us to devise a cross iterative clustering algorithm to generate pseudo-labeled clusters and a chaotic cluster to be classified by further deep learning. It is flexible and robust to data organisation structure (balanced or unbalanced) and scales of datasets. Extensive experiments demonstrate that the proposed CIKICS has an outstanding performance comparing to existing clustering methods, in terms of quantitative evaluations.

References

[1] U Shruthi, V Nagaveni, and BK Raghavendra. A review on machine learning classification techniques for plant disease detection. In 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS), pages 281–284. IEEE, 2019.
[2] Mohammed Brahimi, Marko Arsenovic, Sohaib Laraba, Srdjan Sladojevic, Kamel Boukhalfa, and Abdelouahab Moussaoui. Deep learning for plant diseases: detection and saliency map visualisation. In Human and Machine Learning, pages 93–117. Springer, 2018.
[3] Xihai Zhang, Yue Qiao, Fanfeng Meng, Chengguo Fan, and Mingming Zhang. Identification of maize leaf diseases using improved deep convolutional neural networks. IEEE Access, 6:30370–30377, 2018.
[4] Gensheng Hu, Haoyu Wu, Yan Zhang, and Mingzhu Wan. A low shot learning method for tea leaf’s disease identification. Computers and Electronics in Agriculture, 163:104852, 2019.
[5] Alvaro F Fuentes, Sook Yoon, Jaesu Lee, and Dong Sun Park. High-performance deep neural network-based tomato plant diseases and pests diagnosis system with refinement filter bank. Frontiers in plant science, 9, 2018.
[6] Charles Otto, Dayong Wang, and Anil K Jain. Clustering millions of faces by identity. IEEE transactions on pattern analysis and machine intelligence, 40(2):289–303, 2017.
[7] Saquib Sarfraz, Vivek Sharma, and Rainer Stiefelhagen. Efficient parameter-free clustering using first neighbor relations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8934–8943, 2019.
[8] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5147–5156, 2016.
[9] Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. Variational deep embedding: An unsupervised and generative approach to clustering. arXiv preprint arXiv:1611.05148, 2016.
[10] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. Improved deep embedded clustering with local structure preservation. In *IJCAI*, pages 1753–1759, 2017.

[11] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

[14] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. *A unified view of kernel k-means, spectral clustering and graph cuts*. Citeseer, 2004.

[15] V Santhi and Rini Jose. Performance analysis of parallel k-means with optimization algorithms for clustering on spark. In *International Conference on Distributed Computing and Internet Technology*, pages 158–162. Springer, 2018.

[16] Erich Schubert and Peter J Rousseeuw. Faster k-medoids clustering: improving the pam, clara, and clarans algorithms. In *International Conference on Similarity Search and Applications*, pages 171–187. Springer, 2019.

[17] Leonard Kaufman and Peter J Rousseeuw. Partitioning around medoids (program pam). *Finding groups in data: an introduction to cluster analysis*, 344:68–125, 1990.

[18] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.

[19] R Chauhan, P Butra, and S Chaudhary. A survey of density based clustering algorithms. *Int. J. Comput. Sci. Technol.*, 5(2):169–171, 2014.

[20] Chunhui Zhu, Fang Wen, and Jian Sun. A rank-order distance based clustering algorithm for face tagging. In *CVPR 2011*, pages 481–488. IEEE, 2011.

[21] Wei-An Lin, Jun-Cheng Chen, and Rama Chellappa. A proximity-aware hierarchical clustering of faces. In 2017 *12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 294–301. IEEE, 2017.

[22] Wei-An Lin, Jun-Cheng Chen, Carlos D Castillo, and Rama Chellappa. Deep density clustering of unconstrained faces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8128–8137, 2018.

[23] Kai Tian, Shuigeng Zhou, and Jihong Guan. Deepcluster: A general clustering framework based on deep learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 809–825. Springer, 2017.

[24] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[25] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[26] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[27] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[28] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[29] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[30] Jean-Philippe Vert, Koji Tsuda, and Bernhard Schölkopf. A primer on kernel methods. *Kernel methods in computational biology*, 47:35–70, 2004.

[31] Marco Cuturi. Fast global alignment kernels. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 929–936, 2011.
[32] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1386–1393, 2014.

[33] David Hughes, Marcel Salathé, et al. An open access repository of images on plant health to enable the development of mobile disease diagnostics. arXiv preprint arXiv:1511.08060, 2015.

[34] Hafiz Tayyab Rauf, Basharat Ali Saleem, M Ikram Ullah Lali, Muhammad Attique Khan, Muhammad Sharif, and Syed Ahmad Chan Bukhari. A citrus fruits and leaves dataset for detection and classification of citrus diseases through machine learning. Data in brief, 26:104340, 2019.

[35] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: spectral clustering and normalized cuts. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 551–556, 2004.