Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Cost-effective land cover classification for remote sensing images

Dongwei Li[1,2], Shuliang Wang[1*], Qiang He[2] and Yun Yang[2]

Abstract

Land cover maps are of vital importance to various fields such as land use policy development, ecosystem services, urban planning and agriculture monitoring, which are mainly generated from remote sensing image classification techniques. Traditional land cover classification usually needs tremendous computational resources, which often becomes a huge burden to the remote sensing community. Undoubtedly cloud computing is one of the best choices for land cover classification, however, if not managed properly, the computation cost on the cloud could be surprisingly high. Recently, cutting the unnecessary computation *long tail* has become a promising solution for saving cost in the cloud. For land cover classification, it is generally not necessary to achieve the best accuracy and 85% can be regarded as a reliable land cover classification. Therefore, in this paper, we propose a framework for cost-effective remote sensing classification. Given the desired accuracy, the clustering algorithm can stop early for cost-saving whilst achieving sufficient accuracy for land cover image classification. Experimental results show that achieving 85%-99.9% accuracy needs only 27.34%-60.83% of the total cloud computation cost for achieving a 100% accuracy. To put it into perspective, for the US land cover classification example, the proposed approach can save over $1,593,490.18 for the government in each single-use when the desired accuracy is 90%.

Keywords: Remote sensing, Land cover classification, Cloud computing, FCM algorithm

Introduction

Land cover maps represent the spatial information of different categories of physical coverage (e.g., forests, wetlands, grasslands, lakes.) on surfaces of the earth [1], where dynamic land cover maps may contain changes in land cover categories over time, thereby capturing the changes of land arrangements, human activities, and the inputs people make within a land cover type to produce, alter or maintain it. Frequently updated land cover map is essential for a variety of environmental and socioeconomic applications, including urban planning [2], agricultural monitoring [3], forestry [4], sustainable development [5] and so on.

Considering the large geographic area and high temporal frequency covered by remote sensing satellite imagery, it provides a unique opportunity to obtain land cover information through the image classification. Land cover classification is the grouping of pixels in the images into homogeneous regions, each of which corresponds to a specific land cover type, usually modelled as a clustering problem [6–8]. Generally, unsupervised clustering is widely used in the land cover classification [9] because remote sensing images are often not available with ground truth of labels.

To generate updated land cover information at different scales, a series of remote sensing image classification techniques have been proposed in recent years [10]. Most representative clustering algorithms (e.g., *k-means* [11], *ISODATA* [12], *Expectation-Maximum* [13], *Markov Random Field* [14]) consider the pixel as the basic analysis unit, with which each pixel is labeled as a single land cover type. However, these pixel-wise clustering

*Correspondence: slwang2011@bit.edu.cn

[1] School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
Full list of author information is available at the end of the article

approaches, when applied to heterogeneous regions, may have limitations as the size of an object may be much smaller than the size of a pixel. In particular, a pixel may not only contain a single land cover type, but a mixture of several land cover types. Therefore, fuzzy clustering approaches have been developed for unsupervised land type classification [15, 16].

The advancement of spatial, spectral, temporal, and angular data has facilitated the generation of petabytes of data every year [17–19]. Land cover classification usually needs tremendous computational resources and becomes a huge burden to the remote sensing companies and organisations. With the ever-increasing demand for storing and analyzing large volumes of remote sensing imagery, cloud computing offers a suitable solution for the remote sensing community [20]. By acting as a near-real-time insight platform, cloud computing can rapidly perform big data analysis. It is a mature platform that provides global users with high-end computing resources without a huge IT infrastructure investment budget, and provide efficient and low-cost solutions for remote sensing classification.

However, the cost of cloud computing environments for big data storage and analytics is drawing increasing attention from researchers, which becomes a bottleneck for land cover classification in the cloud. For example, running 100 m4-2xlarge EC2 virtual machines (VM) instances in Amazon Sydney datacenter costs up to $62,496 per month [21]. Li et al. [22] found that cutting the unnecessary *long tail* (see Fig. 1) in the clustering process is a promising solution for cost-effective cloud computing, which inspires us that we can explore achieving sufficiently satisfactory clustering accuracy with the lowest possible computation cost. In particular, this method could be effectively applied to the land cover classification.

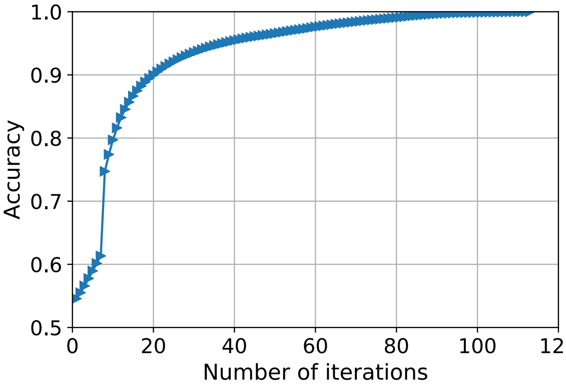

Fig. 1 Long tail phenomenon during the clustering process

In most clustering scenarios (e.g., spatial data analysis, weather forecast, marketing), we do not always need to have the optimal solution because users usually don't need 100% accuracy. Taking weather forecast as an example, clustering techniques have been used to predict weather conditions (e.g., rainy, snowy, sunny) based on various factors such as air temperature, air pressure, humidity, amount of cloud cover, and speed of the wind. In this case, a reasonable margin of inaccuracy is acceptable because users do not need to know 100% accurate weather information. As long as they have a general understanding of the weather, they will be able to make decisions about what to wear or whether to bring an umbrella when going out. In the real world, there will never be completely accurate for clustering, such as weather forecasting and land cover classification. It is critical to stop clustering at a reasonable point to save computation costs when achieving a sufficiently satisfactory accuracy at lower cost is preferable instead of achieving 100% accuracy at higher cost.

For the land cover classification problem, it is also not necessary to achieve the best accuracy all the time. Normally at least 85% accuracy can be considered a reliable land cover classification [23]. To achieve cost-effective land cover classification, a new framework needs to be explored to improve cost-effectiveness performance, rather than using the same methods in the general big data clustering scenarios. In general, there are three main challenges to be addressed for the design of the new framework: 1) unlike traditional pixel-wise clustering methods, we adopt fuzzy clustering methods (e.g., the FCM algorithm) to assign pixels to multiple land cover types; 2) before building the regression models between the change rate of objective function and accuracy, we first detect and remove the anomalies; 3) compared to the commonly used quadratic polynomial regression in previous literature [22], more regression models are explored for more cost-effective land cover classification.

In this research, we propose a generalized framework for cost-effective land cover classification with remote sensing images. We are the first to apply the FCM clustering algorithm to cost-effective land cover classification. *Rand Index* is used as the accuracy metric and *Local Outlier Factors* (LOF) [24] is employed to remove anomalies between the change rate of objective function and accuracy. *Support Vector Regressor* (SVR) [25] is applied to fit the relationship between the change rate of objective function and accuracy. Experimental results show that achieving 85%, 90%, 95%, 99%, 99.9% accuracy need only 27.34%, 29.33%, 33.25%, 55.93% and 60.83% of the computation cost required for achieving a 100% accuracy. Our contributions are as follows:

- We propose a generalized framework for the cost-effective land cover classification problem, with which the clustering algorithm can stop at an early point given the desired accuracy for cost-saving.
- We are the first to adopt the LOF algorithm to remove anomalies before fitting the relation between the change rate of objective function and accuracy, which improves the cost-effectiveness in the land cover classification.
- Experimental results show that the proposed framework can achieve sufficient accuracy and save much computation cost in the cloud.

The remainder of the paper is organised as follows. Section 2 discusses the current related works on remote sensing classification and the cost of cloud computing. In Section 3, we introduce the background knowledge used in our study and in Section 4 we demonstrate our generalized framework for land cover classification. Then, in Section 5, we conduct extensive experiments to illustrate the cost-effectiveness of the proposed framework. Section 6 gives conclusions and future work.

Related works

Remote sensing imagery classification

Fuzzy C-means (FCM) is first proposed by Dunn and improved by Bezdek [26], which is frequently used in the image segmentation field. Foody et al. [27] used the FCM algorithm for sub-urban land use mapping from remote sensing images. They found that the classification results can be improved significantly when using fuzzy clustering compared with hard clustering methods. Wang et al. [15] incorporated the spatial context to improve the robustness of the FCM algorithm in image segmentation. By combining these two concepts and modifying the objective function of the FCM algorithm, they solved the problems of sensitivity to noisy data and the lack of spatial information, and improved the image segmentation results. Sowmya et al. [16] proposed the reformed fuzzy C-means (RFCM) technique for land cover classification. Image quality metrics such as error image, peak signal to noise ratio (PSNR) and compression ratio were used to compare the segmented images.

Zhang et al. [28] compared six popular segmentation models for land cover classification of remote sensing images, and then proposed the model ASPP-U-Net which outperforms the other methods in both classification accuracy and time efficiency. De et al. [29] proposed to use pixel-based classification with random forest (RF) classifiers to generate high-quality land cover products. Their method achieved an overall accuracy of 83% and 81% for Liberia and Gabon and outperformed previous land cover products in these countries in terms of subject content and accuracy. Nguyen et al. [30] adopted object-based random forest classification to generate land cover information for the Vu Gia - Thu Bon river basin in 2010. They classified seven types of land cover (i.e., planted forest, natural forest, paddy field, urban residential, rural residential, bare land, and water surface), achieving the overall accuracy of 73.9% with kappa = 0.70.

Cost-effective cloud computing

With the development of the pay-as-you-go cost model, IT resources are often provided and utilized by cloud computing. Since most of the benefits offered by cloud computing are around the flexibility of the pay-as-you-go model, cost-effectiveness has become a key issue in the cloud computing area. With the continuous improvement of cloud services provided by cloud vendors, many scientists are beginning to pay attention to the performance and cost-effectiveness of public cloud services. In-depth research has been conducted on cost-effective computation in cloud environments.

Cui et al. [31] identified the high tail latency problem in cloud CDN via analyzing a large-scale dataset collected from 783,944 users in a major cloud CDN. A workload scheduling mechanism was presented aiming to optimize the tail latency while meeting the cost constraint given by application providers. A portfolio optimization approach was then proposed by [32] for cost-effective healthcare data provisioning. Li et al. [33] modelled the task scheduling on IoT-cloud infrastructure as bipartite graph matching, and proposed a resource estimating method.

A semi-elastic cluster computing model [34] was introduced for organizations to reserve and dynamically adjust the size of cloud-based virtual clusters. The experiment results indicated that such a model can save more than 60% cost for individual users acquiring and managing cloud resources without leading to longer average job wait times. The MapReduce cloud model Cura was proposed to offer a cost-effective solution to effectively deal with production resources, which implemented a globally effective resource allocation process that significantly reduces the cost of resource use in the cloud. Flutter [35] was designed and implemented as a task scheduler and reduced the completion time and the network cost for large-scale data processing tasks over data centres in different regions.

Berriman et al. [36] used Amazon EC2 to study the cost-effectiveness of cloud computing applications and Amazon EC2 was compared with the Abe high-performance cluster. They concluded that Amazon EC2 can provide better performance for memory- and processor-bound applications than I/O-bound applications. Similarly, Carlyle et al. [37] compared the computation cost of high-performance in Amazon EC2 environments and traditional HPC environments with Purdue University's HPC cluster program. Their research indicated that the in-house cluster can be more cost-effective while organizations take advantage of clusters or have IT departments that can maintain an IT infrastructure or prioritize cyber-enabled research. These features of in-house clusters actually demonstrated the cost-effectiveness and flexibility of running computation-intensive applications in the cloud.

A random multi-tenant framework was proposed by Wang et al. [38] for investigating the cloud services response time as an indicator with a universal probability distribution. Similarly, Hwang et al. [39] tested the performance of Amazon cloud services with 5 different benchmark applications and found it was more cost-effective in sustaining heavier workload, by comparing the scale-out strategies and the scale-up strategies. To explore the minimal cost of storing and regenerating datasets in multiple clouds, [40] proposed a novel algorithm that implements the best compromise among storage, bandwidth, and computation cost in the cloud. Jawad et al. [41] proposed an intelligent power management system in order to minimize data centre operating costs. The system can coordinate the workload of data centre, renewable power, battery bank, diesel generators, real-time transaction price for the purpose of reducing the cost of consumption. Aujla et al. [42] proposed an efficient workload slicing scheme for handling data-intensive applications in multi-edge cloud environments using software-defined networks (SDN) to reduce the migration delay and cost.

In order to save costs in the cloud, it is also important for algorithms to reduce processing time and improve their efficiency. Teerapittayanon et al. [43] proposed a deep network architecture *BranchyNet*, allowing predictions for most of the test samples to exit the network early through these branches, at which point the samples can already be inferred with high confidence. Combining the advantages of BranchyNet and the Edge-Cloud architecture, an open source framework based on Kafka-ML [44] was designed to support fault-tolerant and low-latency AI prediction. Their experimental results obtained a 45.34% improvement in response time compared to a pure cloud deployment. Passalis et al. [45] proposed a Bag-of-Features (BoF) based method that enables the construction of efficient hierarchical early exit layers with minimal computational overhead. They provided an adaptive inference method, allowing the inference process to be stopped early when the network has sufficient confidence in its output, resulting in significant performance benefits. Though above works explored the early stop schemes, they focused more on improved the architecture of deep learning networks which is not applicable for traditional clustering scenarios.

The current research on cloud computing indicates the prevalence of running computation-intensive applications in the cloud, which provides a general overview of the cost-effectiveness of big data analysis in the cloud by comparing traditional cluster environments and cloud environments. Li et al. [22] proposed a method for cutting the unnecessary long tail in the clustering process to achieve cost-effective big data clustering in the cloud. Sufficiently satisfactory accuracies can be achieved at the lowest possible costs by setting the desired accuracies, which presented an important step toward cost-effective big data clustering in the cloud. In this research, we adopt the approach proposed in [22] to a more specific field: remote sensing land cover classification, and explore more advanced and efficient ways to improve the performance of cost-effective clustering in the cloud.

In summary, different from previous studies, our work has several advantages: (1) we explore stopping traditional clustering algorithms early to save cloud computing costs, while others [43–45] typically focus on improving deep learning architectures that are neither intuitive nor easy to use for end users; (2) we remove anomalies to improve the cost-effectiveness in the cloud before fitting the relation between the change rate of objective function and accuracy while [22] does not consider the influence of anomalies; (3) to the best of our knowledge, we are the first to investigate the cost-effective land cover classification problem based on the remote sensing research while previous research usually focuses on general clustering problem with fixed algorithm, which is less scalable and less effective in practical applications.

Background

This section mainly introduces the background of the proposed cost-effective land cover classification method, including the fuzzy c-means clustering algorithm, accuracy calculation method, and the cloud cost computing model.

Fuzzy c-means clustering

As one of the most commonly used fuzzy clustering methods, the FCM algorithm [26, 46] is a clustering technique allowing each data point to belong to more than one cluster. Fuzzy logic principles are used to assign each point a membership in each cluster center from 0 to 1, which indicates the degree to which data points belong to each cluster. Therefore, the FCM algorithm can be very powerful compared to traditional hard clustering (i.e., K-means [47]) where every point can only belong to exactly one class. FCM clustering is based on minimizing the objective function as follows:

$$\mathcal{J}_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^{m} \left\| x_i - c_j \right\|^2, 1 \le m < \infty, m \in \mathbb{R} \quad (1)$$

where m is a real number larger than 1 and means the mth iteration during the clustering process. u_{ij} means the degree of membership of x_i in the cluster j, x_i indicates the ith d-dimensional measured data, c_j is the jth d-dimension center of the cluster, and $\|*\|$ is any norm expressing the similarity between any measured data and the center. Fuzzy partitioning is conducted through an iterative optimization of the objective function shown below, with the update of membership u_{ij} and the cluster centers c_j by:

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}} \quad (2)$$

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} \cdot x_i}{\sum_{i=1}^{N} u_{ij}^{m}} \quad (3)$$

This iteration will stop when

$$max_{ij} \left\{ \left| u_{ij}^{k+1} - u_{ij}^{k} \right| \right\} < \varepsilon, 0 < \varepsilon < 1, \varepsilon \in \mathbb{R} \quad (4)$$

where ε is a termination criterion between 0 and 1, whereas k is the iteration step. This procedure converges to a local minimum or a saddle point of $\mathcal{J}_m$. Overall, the algorithm is composed of the following steps:

1. Initialize matrix $U = [u_{ij}]$ as U^0.
2. In k step, calculate the centers vectors $C^k = [c_{ij}]$ with U^k based on Eq. (3).
3. Update U^k and U^{k+1}.
4. If $\left\| U^{k+1} - U^k \right\| < \varepsilon$, then stop; other wise, return to Step 2.

Rand index

Accuracy is a key metric for assessing the effectiveness of big data clustering. As suggested by [22], to demonstrate that the clustering accuracy gradually increases iteration by iteration, we adopt the final clustering partition P_f as a reference partition as 100% accuracy. By comparing the clustering results obtained in each iteration, we exhibit how the accuracy of the intermediate partition $P_i \in \{P_1, P_2, ..., P_f\}$ increases during the clustering process.

In our research, the accuracy of the clustering algorithm can be measured by the similarity between P_i and P_f. *Rand Index* [48] is adopted to evaluate the similarity between two clustering partitions, which is a popular method of accuracy calculation in the field of data clustering. Each partition is treated as a group of $(m-1) \times m/2$ pairs of data points, where m represents the size of the dataset. For each pair of data points, the partition either assign them to the same cluster or different clusters. Therefore, the similarity between the partitions P_1 and P_2 can be measured as follows:

$$Rand(P_1, P_2) = \frac{m_{00} + m_{11}}{m_{00} + m_{01} + m_{10} + m_{11}} = \frac{m_{00} + m_{11}}{\binom{m}{2}} \quad (5)$$

where:

m_{00} indicates the number of data point pairs located in the different clusters in both P_1 and P_2;

m_{11} indicates the number of data point pairs located in the same clusters in both P_1 and P_2;

m_{01} indicates the number of data point pairs located in the same clusters in P_1 but in different clusters in P_2;

m_{10} indicates the number of data point pairs located in different clusters in P_1, but in the same clusters in P_2.

With the *Rand Index* as the measure of similarity, the clustering accuracy can be calculated in each iteration of the clustering process. Take Fig. 2 for instance, the data point pairs located in the same cluster (indicated with same color) in P_1 and P_2 includes $(n1, n2)$, $(n3, n4)$, $(n5, n7)$, $(n5, n8)$, $(n7, n8)$. The pairs that are placed in different clusters in both P_1 and P_2 include $(n1, n3)$, $(n1, n4)$, $(n1, n5)$, $(n1, n6)$, $(n1, n7)$, $(n1, n8)$, $(n2, n3)$,$(n2, n4)$,

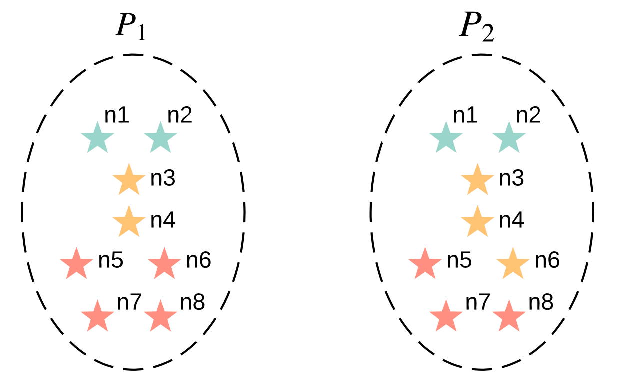

Fig. 2 An example of calculating *Rand Index* between P_1 and P_2

($n2$, $n5$), ($n2$, $n6$), ($n2$, $n7$), ($n2$, $n8$), ($n3$, $n5$), ($n3$, $n7$), ($n3$, $n8$), ($n4$, $n5$), ($n4$, $n7$), ($n4$, $n8$). Then, there is $Rand(P_1, P_2) = (5 + 18)/28 = 82.14\%$. Clearly, the value of *Rand Index* increases as the number of iterations increases. In the last iteration of clustering process where $P_i = P_f$, there is $Rand(P_i, P_f) = 1$, indicating that the clustering process completes with a 100% accuracy.

Cloud computing model

The computation cost for remote sensing image classification can be computed by the cost models provided by cloud vendors. Amazon EC2 web services are adopted in this research which usually have 4 different models: spot instances, on-demand, dedicated hosts and reserved instances. As the most basic cost model, on-demand model is paid by time and does not require upfront payments or long-term commitments. Therefore, the on-demand cost model is adopted in this research for calculating the computation cost in the cloud.

$$\text{Cost}_{\text{comp}} = \text{Price}_{\text{unit}} \times T_{\text{comp}} \tag{6}$$

Similar to [22], computation time T_{comp} is the CPU time (not the wall clock time) used by the cloud instances for the clustering processing. The unit price $\text{Price}_{\text{unit}}$ is defined by the computational resource used in running the algorithm. Take Amazon EC2 for instance, there are 7 major types of EC2 virtual machine instances: RHEL, SLES, Linux, windows, Windows with SQL Web, Windows with SQL Enterprise and windows with SQL Standard. Different types of EC2 VM instances have different unit prices. For example, in Windows type, 36 EC2 VM instances are displayed for 4 types: *Compute Optimized*, *General Purpose*, *Storage Optimized*, and *Memory Optimized*. Unit prices vary from region to region, ranging from \$0.0066 to \$38.054 per hour.

In this paper, for the sake of simplicity, the computation time is used as an indicator for calculating the computation cost. When we use a specific Amazon EC2 VM instance, we can see that the computation time and computation cost are positively correlated. The longer the computation time, the higher the computation cost. Some other costs may occur before running the algorithm, such as data transfer costs and storage costs for large datasets in the cloud. However, the cost of data storage and data transfer is independent of the clustering process. Therefore, in this study, we only focus on the computational cost of the land cover classification process and isolate it from other costs.

Approach

Figure 3 shows the proposed framework consisting of two phases: training phase and testing phase. For the training phase, we learn the relation between the accuracy and the change rate of objective function. Through the testing phase, we set the desired accuracy and stop the clustering algorithm at an early point by meeting sufficient accuracy. The detailed process is as follows:

Training phase

In the training phase, the FCM clustering algorithm is applied on RGB channels of training images. During the clustering process, we get $\mathcal{J}_m$ and $\mathcal{L}_m$, indicating the objective function and the predicted labels at the mth iteration of total n iterations, where the predicted labels

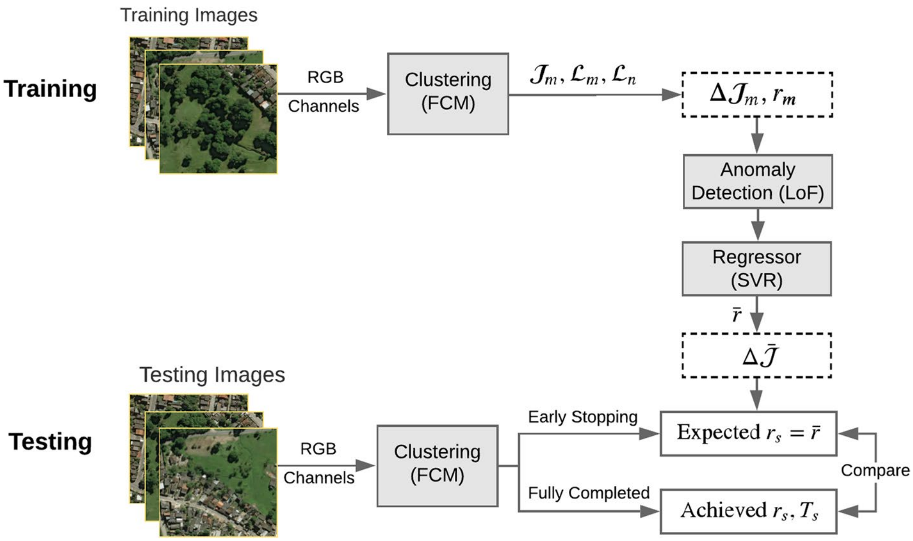

Fig. 3 Framework of cost-effective remote sensing image classification

at the last iteration are noted with $\mathcal{L}_n$. Once the clustering is finished, r_m (the accuracy at mth iteration) is calculated through the *Rand Index* between $\mathcal{L}_m$ and $\mathcal{L}_n$ based on Eq. (5).

The rate of change of objective function $\Delta\mathcal{J}_m$ is computed using the Eq. (7). For simplicity, we use 'change rate' instead of 'the rate of change' in this paper. The change rate is used to describe the percentage change in value over a specific period of time. In this research, we define the change rate of objective function as:

$$\Delta\mathcal{J}_m = \frac{\mathcal{J}_m}{\mathcal{J}_{m-1}} \times 100\%, 1 < m \leqslant n \tag{7}$$

where $\Delta\mathcal{J}_m$ indicates the change rate of objective function at the mth iteration of total n iterations.

For each training image, we can calculate the value of r_m and $\Delta\mathcal{J}_m, m \in \{2, ...n\}$. As a result, we get $n-1$ data points for each training image. To model the relationship between r_m and $\Delta\mathcal{J}_m$, anomaly points need to be mitigated first. As the most well-known anomaly detection algorithm, the LOF [24] is an unsupervised machine learning algorithm that finds anomalies by measuring the local deviation of a given data point based on its neighbors. In our research, the LoF algorithm is applied to mitigate the anomaly points.

With anomalies removed, we have tried several commonly used regression models to fit the relation between r_m and $\Delta\mathcal{J}_m$ in the remaining points, such as SVR, *Standard Linear Regressor* (LR) [49], *Gradient Booting Regressor*(GBR) [50], *Bayesian Ridge Regressor* [51] and *Random Forest Regressor* (RFR) [52]. *Support Vector Machine* (SVM) [25] in regression problems, commonly known as SVR, is one of the most widely used regression models. LR is a linear model which assumes the linear relationship between two variables. GBR is an ensemble method that combines a set of weak predictors to achieve reliable and accurate regression. *Bayesian Ridge Regressor* formulates linear regression by using probability distributions. RFR follows the idea of random forest in classification and can estimate the importance of different features.

After extensive experiments, we found that when SVR is applied, the experimental results usually show better performance. As a result, SVR is adopted as the regression model to fit the relationship between r_m and $\Delta\mathcal{J}_m$. Given the desired accuracy $\bar{r}$ (e.g., 85%, 90%, 95%, 99%, 99.9%), the predicted value of $\Delta\bar{\mathcal{J}}$ can be calculated from the trained regressor (see Fig. 4).

Testing phase

In the testing phase, we run the FCM clustering algorithm with the testing images. $\Delta\mathcal{J}$ at each iteration is calculated and compared with $\Delta\bar{\mathcal{J}}$. When $\Delta\mathcal{J} < \Delta\bar{\mathcal{J}}$, we record the early stopping point at this iteration, e.g., sth iteration. In the real scenario of remote sensing classification, we can stop the clustering algorithm at this point with the confidence of achieving the desired accuracy $\bar{r}$ at the sth iteration.

Evaluation method. To evaluate the performance of the proposed approach, we run the FCM algorithm until it is fully completed during training. Then we calculate the achieved accuracy r_s and computation time T_s at the sth iteration. Finally, we can evaluate the proposed method from two dimensions: the achieved accuracy (through comparing r_s and $\bar{r}$) and the percentage of saved time (T_s/T_n).

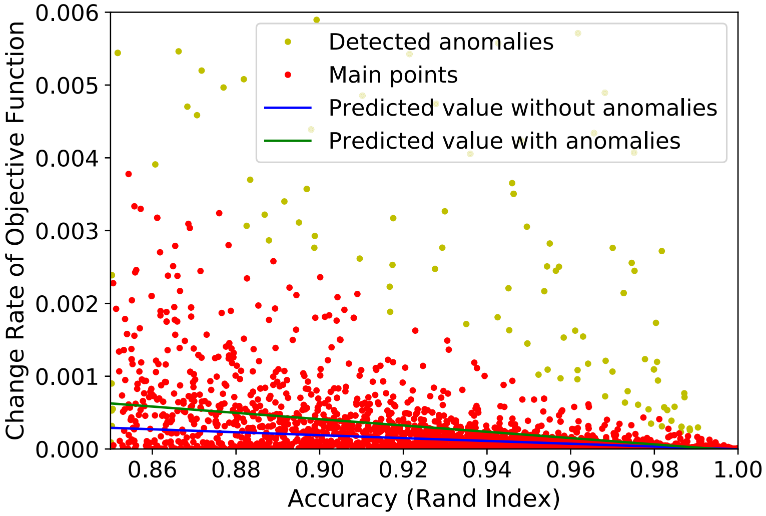

Fig. 4 Relation between change rate of objective function and accuracy

Cloud Computation Cost. Total computation time T_{comp} includes the overall clustering time in the training process T_{train}, and the actual clustering time T_{actual} (i.e., early-stop computation time) when clustering reaches the desired accuracy, which is computed as:

$$T_{\text{comp}} = T_{\text{train}} + T_{\text{actual}} \tag{8}$$

The training phase is carried out only once. Once it is completed, the regression model can be applied repeatedly to the remote sensing image classification in the future. Thus, T_{train} can be negligible compared with the overall cost in the long term. Since the computation time is the primary indicator of the cost in our research, the cost-effectiveness percentage $\text{Cost}_{\text{effective}}$ can be exhibited as follows:

$$\text{Cost}_{\text{effective}} \approx \frac{T_{\text{actual}}}{T_{\text{total}}} \tag{9}$$

where T_{total} represents the expected computation time in the clustering when 100% accuracy is achieved. The smaller the value of $Cost_{effective}$ is, the higher the cost-effectiveness of the clustering.

Experimental evaluation

In this section, we first introduce the experimental settings and the dataset. Then we conduct the experiments consisting of the training phase and testing phase. After that, we evaluate the proposed framework from two aspects: the achieved accuracy and the cost-effectiveness. Finally, we discuss the performance of the cost-effective land cover classification and real-world applications.

Experimental setup

The experiments were conducted on a laptop (Microsoft Corporation - Surface Laptop 4) with a 2.60 GHz Intel (R) Core (TM) i5 processor and 8G memory, and the operating system is 64-bit Windows 10 enterprise. The code was written in Python 3.6 and developed in PyCharm 4.5 IDE, making use of Scikit-learn, skfuzzy, Numpy, Pandas, SciPy and Matplotlib package for machine learning, mathematical, statistical operation and visualization.

We conduct experiments on the public satellite imagery dataset SpaceNet [53]. The dataset is released by Digital Globe, an American vendor of space imagery and geospatial content. The dataset includes a large amount of geospatial information related to various downstream use cases, e.g., infrastructure mapping and land cover classification. SpaceNet contains more than 17,533 high-resolution remote sensing images (438 × 406 pixels). SpaceNet is hosted as the Amazon Web Services public dataset, which contains approx. 67,000 square kilometers of high-resolution imagery in different cities (e.g., Las Vegas, Khartoum, Rio De Janeiro, Shanghai), more than 11 million building footprints, and approx. 20,000 kilometers of road labels, making it the most popular open-source dataset for geospatial machine learning research [22, 54]. Due to the huge size of the SpaceNet dataset, we randomly select 200 sample remote sensing images as the training dataset so that we could perform the clustering and simulate the regression process accurately.

In the experiment, the FCM clustering algorithm ($ncenters = 6$, $error = 0.005$, $m = 2$) was applied for cost-effective remote sensing image classification. Usually, finding the optimal number of clusters is crucial for the unsupervised clustering. For the SpaceNet dataset, through visual inspection, we find that the images can be generally divided into six different regions, i.e., forest, water, road, building, grassland, and wasteland. Therefore, we set the number of clusters $ncenters = 6$. m is an array exponentiation applied to the membership function at each iteration which is usually set to 2 for the FCM algorithm. The *error* indicates the stopping criterion and we use the default value $error = 0.005$ like previous studies [22].

After clustering, the LoF technique ($outliers\_fraction = 0.03$, $n\_neighbors = 40$) was applied to remove the anomalies. For the parameters *outliers_fraction* and *n_neighbors*, we experimented with different parameter settings, and we achieve the best performance of the proposed method using the above settings. Next, SVR (kernel='RBF') was used to predict the change rate of objective function with the desired accuracy (i.e., 85%, 90%, 95%, 99%, 99.9%). We choose the desired accuracy from 85% because it is generally regarded as a reliable accuracy for land cover classification [23]. Then, we evaluate the proposed approach from two dimensions: the achieved accuracy and the percentage of saved time.

Experiment results

Our experiment includes two phases: training phase and testing phase. For training remote sensing images, we first cluster the pixels in RGB channels with the FCM algorithm. We compute the objective function $\mathcal{J}_m$, predict label L_m at the mth iteration until the last iteration n. Then, in each iteration, the change rate of objective function $\Delta\mathcal{J}_m$ is computed based on from Eq. (7) and the accuracy r_m is computed from Eq. (5). Figure 4 shows the relation between $\Delta\mathcal{J}_m$ and r_m.

After that, the LoF technique is used to remove the anomalies. In Fig. 4, red points represent the normal ones and yellow dots mean the detected dots anomalies. SVR is then applied to fit the relation between r_m and $\Delta\mathcal{J}_m$. The green line represents the regression line with anomalies and blue line means the fitting line without anomalies. It can be observed that, given the same desired accuracy, the predicted value with anomalies (green line) is generally smaller than the predicted value without anomalies (blue line).

Given the desired accuracy $\bar{r}$, we can predict the corresponding $\Delta\bar{\mathcal{J}}$. Table 1 shows the different change rate of objective function with different required accuracies (e.g. 85%, 90%, 95%, 99%, 99.9%) for the FCM clustering algorithm with and without LOF anomaly detection. The results show that in the real-world scenario, if the desired accuracy is $\bar{r}$ (i.e., 99%), we can apply the FCM+ LOF algorithm on the remote sensing images, compute the change rate of objective function at each iteration and stop the algorithm when the change rate of objective function is below $\Delta\bar{\mathcal{J}}$ (e.g., 1.84e-6). However, when we make the FCM+LOF algorithm stop at an early point, is the achieved accuracy really up to 99%? How much time could we save by this approach? To evaluate the performance of our method, we propose two criteria: the achieved accuracy and cost-effectiveness (the percentage of saved time).

Achieved accuracy. To evaluate our proposed method, given the desired accuracy of $\bar{r}$, we first calculate the corresponding change rate of objective function (see Table 1). Then, we run the FCM+LOF algorithm on testing images, calculate the change rate of objective function until it reaches $\Delta\bar{J}$ at the sth iteration.

In this research, we complete the whole clustering process and calculate the achieved accuracy at the sth iteration. After that, the achieved accuracy r_s is compared with the desired accuracy $\bar{r}$. Table 2 shows the result of the average achieved accuracy (with standard deviation) of different desired accuracy for the FCM algorithm with and without LOF anomaly detection. We can see that the achieved accuracy is very close yet above the given desired accuracy and even higher than the desired accuracy for the FCM+LOF method. For example, on average, the achieved accuracy reaches 99.27% when the desired accuracy is 99%, and 99.92% when the desired accuracy is 99.9%. However, if we do not apply the LOF anomaly detection algorithm and adopt FCM algorithm only, the achieved accuracy may not reach to the desired accuracy in some cases. This illustrates that the proposed method has high accuracy on the FCM+LOF method and anomaly detection algorithm is necessary for the cost-effective clustering process.

Table 1 Change rate of objective function with different desired accuracies

Algorithm	Desired accuracy				
	85%	**90%**	**95%**	**99%**	**99.9%**
FCM+LOF	2.67e-4	1.76e-4	8.59e-5	1.61e-6	6.50e-7
FCM only	6.03e-4	3.42e-4	1.73e-4	1.58e-6	5.43e-7

Table 2 Average achieved accuracy over different desired accuracies

Algorithm	Desired accuracy				
	85%	**90%**	**95%**	**99%**	**99.9%**
FCM+LOF	89.15% (0.0594)	93.16% (0.0544)	95.07% (0.0370)	99.27% (0.0044)	99.92% (0.0018)
FCM only	83.03% (0.0786)	89.40% (0.0645)	94.49% (0.0409)	99.36% (0.0041)	99.94% (0.0021)

Cost-effectiveness. Table 3 shows the actual percentage of saved computation time with different desired accuracy for the FCM algorithm with and without LOF anomaly detection. For the FCM+LOF method, it can be found that we only use 27.34%, 29.33%, 33.25%, 55.93%, 60.83% computation time when the desired accuracies are 85%, 90%, 95%, 99% or 99.9% respectively. Since the cost of cloud computation is directly related to the actual computation time, the FCM algorithm can achieve high cost-effectiveness in the cloud with the proposed framework. It is worth noting that we do not show the result of the actual computation time, but only the actual time as a percentage of the expected time ($T_{\text{actual}}/T_{\text{total}}$), namely the $\text{Cost}_{\text{effective}}$ calculated with the Eq. 9. The reason is that the actual computation time may vary with different hardware resources or cloud computing platforms, and we aim to achieve high cost-effectiveness by stopping the clustering process at an early point, regardless of the platforms and hardware settings.

Discussion

Figure 4 shows the relation between the accuracy and the change rate of objective function. It can be seen that the predicted change rate of objective function without anomalies is generally lower than the predicted value with anomalies, indicating that the proposed methods can achieve higher accuracy compared to the previous methods without anomaly detection algorithms when given the same desired accuracy.

Table 3 Actual time (percentage) with different desired accuracies

Algorithm	Desired accuracy				
	85%	**90%**	**95%**	**99%**	**99.9%**
FCM+LOF	27.34%	29.33%	33.25%	55.93%	60.83%
FCM only	22.94%	27.80%	31.39%	56.05%	62.30%

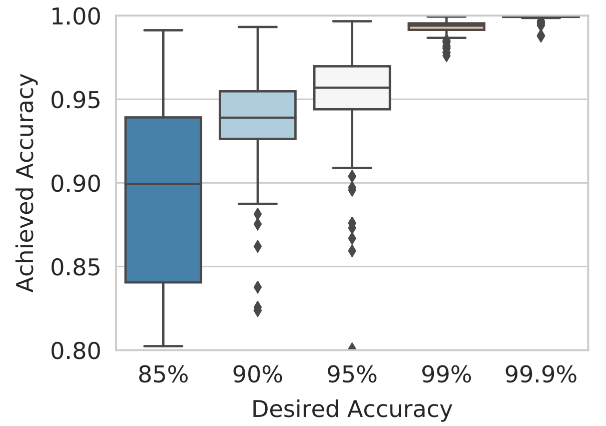

Fig. 5 Boxplot of achieved accuracy over different desired accuracy

Figure 5 shows the boxplot between the achieved accuracy and the desired accuracy for the proposed method. It can be seen that the achieved accuracy is very close to the desired accuracy in different settings. The variation of accuracy becomes smaller with the increase of the desired accuracy, which proves the high performance of the proposed cost-effective land cover classification method.

From the experiments, we have observed that the higher the desired accuracy, the longer the computation time and the less the time saved. by using the proposed approach, users can save more money with lower but sufficient accuracy (e.g., 90%). For example, achieving 90% accuracy needs only 29.33% computation cost of 100% accuracy. For the SpaceNet dataset, the training process is only computed once. The training process for 200 remote sensing images (using the FCM algorithm) took 6431.04 seconds and was only computed once. Taking the California land cover statistics as the instance for 423, 970 km^2 land, which needs around 2.567×10^7 partitioned remote sensing images (438×406 pixels) with each covering a $16,520.74m^2$ land. With the proposed approach, the saved computation time is approximately 162, 035.31 hours when the desired accuracy is 90%. Based on Amazon EC2 pricing [21], if we run m5.xlarge virtual machine instances ($0.424 per hour), the cloud computation cost saved can be up to $68,702.97 for California. Apparently, the cost in the training phase ($0.378) is negligible to the whole computation cost.

In real-world applications, the training phase is performed once and once completed, we can utilize the regression model many times. For instance, we can use the same regression model to carry out the whole United States land cover classification, which would save the computation cost up to $1,593,490.18 in each single-use for the case of the desired accuracy of 90%.

Conclusion

Traditional land cover classification usually requires huge computational resources, and how to save computation costs in the cloud has become an increasingly important issue. For land cover classification, it is often not necessary to achieve the best accuracy all the time, usually no less than 85% can be regarded as a reliable land cover classification.

In this research, we proposed a generalized framework for cost-effective remote sensing classification. FCM algorithm was applied for clustering remote sensing images, with *Rand Index* as the accuracy calculation method and *Local Outlier Factors* (LOF) as the anomaly detection algorithm. The *Support Vector Regressor* (SVR) was used to fit the relation between the change rate of objective function and accuracy. Extensive experimental results showed that given the desired accuracy (e.g., 85%, 90%, 95%,99%, 99.99%), we can make the FCM clustering process on remote sensing images stop earlier and therefore save a huge amount of computation time. Also, the achieved accuracy (i.e., 89.15%, 93.16%, 95.07%, 99.27%, 99.92%) are very close to yet above the desired accuracy. Especially, it is noteworthy that the main contribution of this research is not the widely used algorithms (e.g., FCM, LOF, SVR) but the proposed framework for cost-effective land cover classification. The biggest advantage of the proposed framework is that it is intuitive, flexible, and highly scalable, making it convenient for users to save a lot of cloud computing costs while achieving the desired accuracy.

However, there are some threats to the validity of this research. One main threat is the representativeness of the dataset used in the experiments. The real-world remote imagery dataset SpaceNet is used in our study, which covers 67000 square kilometers and is informative to be general enough for this study. Additionally, our framework is flexible and researchers can adjust the clustering algorithm, accuracy calculation method, anomaly detection algorithm, and regression model in different clustering scenarios (not limited to the SpaceNet datasets or even land cover classification problem) based on their own needs. They can also set the desired accuracy and then make the clustering algorithm stop early with sufficient accuracy to save much computation cost. Another threat is the representativeness of the experiment environment. We conduct experiments on the Microsoft Surface Laptop 4 with 64-bit Windows 10 enterprise, instead of using the EC2 VM instances on the Amazon cloud directly. The reason we do not compare the performance of the proposed framework is that we aim to achieve high cost-effectiveness by stopping the clustering process at an early stop point, and the saved time by reducing the number of iterations is independent of the platform. In the future, the proposed framework can be easily ported to different cloud platforms such as AWS Lambda, EC2, and Azure. Therefore, the threats to the validity are minimal in this research.

In future research, we will focus on several aspects to improve our proposed framework. Firstly, we will compare the performance of different clustering algorithms using the proposed framework. Secondly, more remote sensing datasets will be explored to verify the robustness and the generality of the framework. Additionally, we will investigate methods to bound the achieved accuracy within a given error range.

Appendix

The notations used in this research are shown in Table 4.

Table 4 Table of notations in this research

Notation	Defination
m	The *m*th iteration during the clustering process
x	A d-dimensional data point
x_i	The *i*th d-dimensional data point in cluster *x*
c_j	The d-dimensional center of the cluster *j*
u_{ij}	The degree of membership of x_j in the cluster *j*
$\mathcal{J}_m$	The objective function at the *m*th iteration
P_i	The *i*th partition during the clustering process
$Rand(P_1, P_2)$	The *Rand Index* of two partitions P_1 and P_2
$\mathcal{L}_m$	The predicted labels at the *m*th iteration
r_m	The accuracy at the *m* iteration
$\Delta\mathcal{J}_m$	The change rate of $\mathcal{J}_m$
$\bar{r}$	The desired accuracy (e.g., 85%, 90%, 95%, etc.)
$\Delta\bar{\mathcal{J}}$	The predicted objective function given the $\bar{r}$
$\mathrm{Cost}_{\mathrm{comp}}$	The computation cost in the cloud
$\mathrm{Price}_{\mathrm{unit}}$	The unit price
T_{comp}	Total computation time
T_{train}	The time taken in the training process
T_{actual}	The early-stop computation time in the clustering
T_{total}	The computation time for achieving 100% accuracy
$\mathrm{Cost}_{\mathrm{effective}}$	The cost-effectiveness percentage

Abbreviations

LOF: Local outlier factors; SVR: Support vector regressor; FCM: Fuzzy C-means; LR: Standard linear regressor; GBR: Gradient boosting regressor; RFR: Random forest regressor; SVM: Support vector machine.

Acknowledgements

This research is supported by the Beijing Institute of Technology and Swinburne University of Technology. We also sincerely thank the anonymous reviewers for this paper.

Authors' contributions

Dongwei Li designed the improved framework and conducted experiments under the supervision of Yun Yang, Qiang He and Shuliang Wang. All authors participated in the data analysis and manuscript writing. The authors read and approved the final manuscript.

Authors' information

Dongwei Li received his M.Sc. degree in software engineering from Wuhan University, China, in 2010. He is working toward his Ph.D. degree at Beijing Institute of Technology, Beijing, China. He is a currently visiting researcher at Swinburne University of Technology, Australia. His research interests include data mining and cloud computing.

Shuliang Wang received his Ph.D. degree from Wuhan University, China, in 2002. He is a full professor at Beijing Institute of Technology, China. His research interests include spatial data mining, data field, and big data.

Qiang He received his Ph.D. degree in information and communication technology from Swinburne University of Technology (SUT), Australia, in 2009. He is now a senior senior lecturer at Swinburne University of Technology. His research interests include software engineering, cloud computing and services computing.

Yun Yang received his Ph.D. degree from the University of Queensland, Australia in 1992. He is a full professor at Swinburne University of Technology, Australia. His research interests include distributed systems, cloud, and edge computing, software technologies, workflow systems and service-oriented computing.

Funding

This work is partly funded by National Key Research and Development Plan of China (2020YFC0832600), and the National Natural Science Fund of China (62076027).

Availability of data and materials

Not applicable.

Declarations

Ethical approval and consent to participate

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Consent for publication

Not applicable.

Author details

[1]School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China. [2]Department of Computing Technologies, Swinburne University of Technology, Melbourne, Australia.

Received: 31 July 2021 Accepted: 23 September 2022

Published online: 04 October 2022

References

1. Buchhorn M, Smets B, Bertels L, De Roo B, Lesiv M, Tsendbazar N-E, Li L, Tarko AJ (2021) Copernicus global land service. https://land.copernicus.eu/global/products/lc. Accessed 29 Sept 2022
2. Bechtel B, Conrad O, Tamminga M, Verdonck ML, Coillie V (2017) Beyond the urban mask. In: Joint Urban Remote Sensing Event (JURSE). IEEE, Manhattan, p 1–4
3. Alcantara C, Kuemmerle T, Prishchepov AV, Radeloff VC (2012) Mapping abandoned agriculture with multi-temporal modis satellite data. Remote Sens Environ 124:334–347
4. Asner GP, Broadbent EN, Oliveira PJ, Keller M, Knapp DE, Silva JN (2006) Condition and fate of logged forests in the brazilian amazon. Proc Natl Acad Sci 103(34):12947–12950
5. Glinskis EA, Gutiérrez-Vélez VH (2019) Quantifying and understanding land cover changes by large and small oil palm expansion regimes in the peruvian amazon. Land Use Policy 80:95–106
6. Bensaid AM, Hall LO, Bezdek JC, Clarke LP, Silbiger ML, Arrington JA, Murtagh RF (1996) Validity-guided (re) clustering with applications to image segmentation. IEEE Trans Fuzzy Syst 4(2):112–123

7. Zhang H, Zhai H, Zhang L, Li P (2016) Spectral-spatial sparse subspace clustering for hyperspectral remote sensing images. IEEE Trans Geosci Remote Sens 54(6):3672–3684
8. Wang S, Wang D, Li C, Li Y, Ding G (2016) Clustering by fast search and find of density peaks with data field. Chin J Electron 25(3):397–402
9. Lu D, Weng Q (2007) A survey of image classification methods and techniques for improving classification performance. Int J Remote Sens 28(5):823–870
10. Li M, Zang S, Zhang B, Li S, Wu C (2014) A review of remote sensing image classification techniques: the role of spatio-contextual information. Eur J Remote Sens 47(1):389–411
11. Celik T (2009) Unsupervised change detection in satellite images using principal component analysis and k-means clustering. IEEE Geosci Remote Sens Lett 6(4):772–776
12. Venkateswarlu N, Raju P (1992) Fast isodata clustering algorithms. Pattern Recog 25(3):335–342
13. Kersten PR, Lee JS, Ainsworth TL (2005) Unsupervised classification of polarimetric synthetic aperture radar images using fuzzy clustering and em clustering. IEEE Trans Geosci Remote Sens 43(3):519–527
14. Xu K, Yang W, Liu G, Sun H (2013) Unsupervised satellite image classification using markov field topic model. IEEE Geosci Remote Sens Lett 10(1):130–134
15. Wang Z, Song Q, Soh YC, Sim K (2013) An adaptive spatial information-theoretic fuzzy clustering algorithm for image segmentation. Comp Vision Image Underst 117(10):1412–1420
16. Sowmya B, Sheelarani B (2011) Land cover classification using reformed fuzzy c-means. Sadhana 36(2):153–165
17. Gao N, Xue H, Shao W, Zhao S, Qin KK, Prabowo A, Rahaman MS, Salim FD (2022) Generative adversarial networks for spatio-temporal data: A survey. ACM Trans Intell Syst Technol (TIST) 13(2):1–25
18. Kjærgaard MB, Ardakanian O, Carlucci S, Dong B, Firth SK, Gao N, Huebner GM, Mahdavi A, Rahaman MS, Salim FD et al (2020) Current practices and infrastructure for open data based research on occupant-centric design and operation of buildings. Building and Environment 177:106848
19. Gao N, Marschall M, Burry J, Watkins S, Salim FD (2022) Understanding occupants' behaviour, engagement, emotion, and comfort indoors with heterogeneous sensors and wearables. Sci Data 9(1):1–16
20. Fu JS, Liu Y, Chao HC, Bhargava BK, Zhang ZJ (2018) Secure data storage and searching for industrial iot by integrating fog computing and cloud computing. IEEE Trans Ind Inform 14(10):4519–4528
21. (2022) Amazon web services: Ec2 instance pricing. https://aws.amazon.com/ec2/pricing/on-demand/. Accessed 27 Feb 2022
22. Li D, Wang S, Gao N, He Q, Yang Y (2019) Cutting the unnecessary long tail: cost-effective big data clustering in the cloud. IEEE Trans Cloud Comput 10(1):292–303
23. Anderson JR (1976) A land use and land cover classification system for use with remote sensor data, vol 964. US Government Printing Office, US
24. Breunig MM, Kriegel HP, Ng RT, Sander J (2000) Lof: identifying density-based local outliers. In: ACM Sigmod Record, vol 29. ACM, p 93–104
25. Cawley GC, Talbot NL (2004) Fast exact leave-one-out cross-validation of sparse least-squares support vector machines. Neural Netw 17(10):1467–1475
26. Bezdek JC (2013) Pattern recognition with fuzzy objective function Algorithms. Springer Science & Business Media, Logan
27. Zhang J, Foody G (1998) A fuzzy classification of sub-urban land cover from remotely sensed imagery. Int J Remote Sens 19(14):2721–2738
28. Zhang W, Tang P, Zhao L (2021) Fast and accurate land-cover classification on medium-resolution remote-sensing images using segmentation models. Int J Remote Sens 42(9):3277–3301
29. de Sousa C, Fatoyinbo L, Neigh C, Boucka F, Angoue V, Larsen T (2020) Cloud-computing and machine learning in support of country-level land cover and ecosystem extent mapping in liberia and gabon. PLoS ONE 15(1):e0227438
30. Nguyen NV, Trinh THT, Pham HT, Tran TTT, Pham LT, Nguyen CT (2020) Land cover classification based on cloud computing platform. J Southwest Jiaotong Univ 55(2)
31. Cui Y, Dai N, Lai Z, Li M, Li Z, Hu Y, Ren K, Chen Y (2019) Tailcutter: wisely cutting tail latency in cloud cdns under cost constraints. IEEE/ACM Trans Netw 27(4):1612–1628
32. Alam MGR, Munir MS, Uddin MZ, Alam MS, Dang TN, Hong CS (2019) Edge-of-things computing framework for cost-effective provisioning of healthcare data. J Parallel Distrib Comput 123:54–60
33. Li W, Liao K, He Q, Xia Y (2019) Performance-aware cost-effective resource provisioning for future grid iot-cloud system. J Energy Eng 145(5):04019016
34. Niu S, Zhai J, Ma X, Tang X, Chen W, Zheng W (2016) Building semi-elastic virtual clusters for cost-effective hpc cloud resource provisioning. IEEE Trans Parallel Distrib Syst 27(7):1915–1928
35. Hu Z, Li B, Luo J (2018) Time- and cost-efficient task scheduling cross geo-distributed data centers. IEEE Trans Parallel Distrib Syst 29(3):705–718
36. Berriman GB, Juve G, Deelman E, Regelson M, Plavchan P (2010) The application of cloud computing to astronomy: a study of cost and performance. In: 6th IEEE International Conference on E-Science Workshops. IEEE, p 1–7
37. Carlyle AG, Harrell SL, Smith PM (2010) Cost-effective hpc: The community or the cloud? In: IEEE 2nd International Conference on Cloud Computing Technology and Science (CloudCom). IEEE, p 169–176
38. Wang Z, Hayat MM, Ghani N, Shaban KB (2017) Optimizing cloud-service performance: Efficient resource provisioning via optimal workload allocation. IEEE Trans Parallel Distrib Syst 28(6):1689–1702
39. Hwang K, Bai X, Shi Y, Li M, Chen WG, Wu Y (2016) Cloud performance modeling with benchmark evaluation of elastic scaling strategies. IEEE Trans Parallel Distrib Syst 27(1):130–143
40. Yuan D, Cui L, Li W, Liu X, Yang Y (2018) An algorithm for finding the minimum cost of storing and regenerating datasets in multiple clouds. IEEE Trans Cloud Comput 6(2):519–531
41. Jawad M, Qureshi MB, Khan U, Ali SM, Mehmood A, Khan B, Wang X, Khan SU (2018) A robust optimization technique for energy cost minimization of cloud data centers. IEEE Trans Cloud Comput
42. Aujla GS, Kumar N, Zomaya AY, Ranjan R (2017) Optimal decision making for big data processing at edge-cloud environment: An sdn perspective. IEEE Trans Ind Inform 14(2):778–789
43. Teerapittayanon S, McDanel B, Kung HT (2016) Branchynet: Fast inference via early exiting from deep neural networks. In: 2016 23rd International Conference on Pattern Recognition (ICPR). IEEE, pp 2464–2469
44. Torres DR, Martín C, Rubio B, Díaz M (2021) An open source framework based on kafka-ml for distributed dnn inference over the cloud-to-things continuum. J Syst Architect 118:102214
45. Passalis N, Raitoharju J, Tefas A, Gabbouj M (2020) Efficient adaptive inference for deep convolutional neural networks using hierarchical early exits. Pattern Recog 105:107346
46. Bezdek JC, Ehrlich R, Full W (1984) Fcm: The fuzzy c-means clustering algorithm. Comput Geosci 10(2–3):191–203
47. Hartigan JA, Wong MA (1979) Algorithm as 136: a k-means clustering algorithm. J R Stat Soc Ser C (Appl Stat) 28(1):100–108
48. Rand WM (1971) Objective criteria for the evaluation of clustering methods. J Am Stat Assoc 66(336):846–850
49. Uyanık GK, Güler N (2013) A study on multiple linear regression analysis. Procedia-Soc Behav Sci 106:234–240
50. Friedman JH (2001) Greedy function approximation: a gradient boosting machine. Ann Stat 29(5):1189–1232
51. Tipping ME (2001) Sparse bayesian learning and the relevance vector machine. J Mach Learn Research 1(Jun):211–244
52. Liaw A, Wiener M et al (2002) Classification and regression by randomforest. R News 2(3):18–22
53. Spacenet on amazon web services (aws) datasets (2019) https://spacenetchallenge.github.io/datasets/datasetHomePage.html. Accessed 28 July 2019
54. Van Etten A, Hogan D, Manso JM, Shermeyer J, Weir N, Lewis R (2021) The multi-temporal urban development spacenet dataset. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE, Manhattan, p 6398–6407

Publisher's Note