Incremental Land Cover Classification via Label Strategy and Adaptive Weights

Bo Ren, Zhao Wang, Biao Hou, Bo Liu, Zitong Wu, Jocelyn Chanussot, Licheng Jiao

Incremental Land Cover Classification Via Label Strategy and Adaptive Weights

Bo Ren, *Member, IEEE,* Zhao Wang, Biao Hou, *Member, IEEE,* Bo Liu, Zitong Wu, Jocelyn Chanussot, *Fellow, IEEE,* and Licheng Jiao, *Fellow, IEEE*

***Abstract*—During incremental learning tasks, catastrophic forgetting occurs when old models are updated with new information. To address this issue, we propose a novel method called Label Strategy and Adaptive Weights (LSAW) that improves the incremental learning process. The label strategy introduces the old classes and solves the problem of how to reasonably use the wrong samples predicted by the old model. In the cross-entropy loss, we apply a threshold to filter the pseudo labels predicted by the old model. Subsequently, we merge the pixel samples with high probability with the current label. The probability here refers to the probability that the pixel belongs to the true class. This process enables the introduction of information from old classes that are not directly accessible in the current stage. Moreover, this information is relatively reliable, and the model exhibits confidence in its accuracy. For the remaining pixels, we retain all classes' information through label smoothing. In the distillation function, the old class and background pixel samples are selected for distillation according to the prediction map of the old classes. The weights of the classes are adaptively updated and adjusted using specific label information from each batch and the different stages of incremental learning. As demonstrated by the results of our experiment, on three remote sensing image datasets: CCF, Potsdam, and Vaihingen, our method achieves the best results.**

***Index Terms*—Semantic segmentation, land cover classification, incremental learning.**

I. Introduction

In recent years, the development of satellite sensors has enabled us to capture large amounts of remote sensing data and rich spectral information with high spatial resolution. For a variety of applications, such as ocean monitoring, vegetation coverage analysis, urban planning, and disaster relief, it is essential to make full use of remote sensing image data. In order to properly analyze remote sensing image data and perform high-quality classifications, semantic segmentation can be extremely helpful. Therefore, it is imperative to use semantic segmentation technology when studying remote sensing images.

Segmenting an image semantically involves dividing it into regional blocks containing a specific semantic meaning and identifying the semantic class for each of these blocks. In computer vision, semantic segmentation is considered to be a fundamental and challenging problem. As neural networks have developed and large-scale datasets have become available for training, the accuracy of semantic segmentation has continued to improve. Using fully convolutional networks (FCN) [1], the current approaches extend the deep architecture from image-level classification to pixel-level classification. Semantic segmentation models based on FCNs have been improved in various ways over the years, including exploiting multi-scale representations [2], [3], modeling spatial dependencies and contextual cues [4], [5], and considering attention models [6].

Research on land cover classification tasks in remote sensing has been greatly improved by the development of semantic segmentation. By assigning corresponding surface category information labels to each pixel unit in the image, a large-scale classification map can be generated that is easy to observe and analyze.

While many developments have been made in existing semantic segmentation methods, many of the proposed methods do not take into account how to maintain the network's memory ability for old classes while learning new classes. The accuracy of the old classes will decline rapidly if no constraints are imposed on the network during the learning process. At the end of the training, the model is only able to achieve good results for the newly learned classes, while it performs poorly for the old classes. It is known as catastrophic forgetting [7].

Incremental learning is designed to solve the catastrophic forgetting problem. While the problem of incremental learning is solved through object recognition [8]–[10] and detection [11]–[13], there are relatively few studies on incremental learning in semantic segmentation. In incremental learning, information is continuously processed to cope with the nonstop flow of information that occurs in the real world. In addition to optimizing existing knowledge, it also seeks to acquire new information.

A major problem with remote sensing images is their large scale and class imbalance. Furthermore, the label will often change for the following reasons. Various remote sensing datasets are collected from around the world, so they are continuously expanding. A further problem with annotations is that they are usually retrieved from different sources and

B. Ren, Z. Wang, B. Hou, B. Liu, Z. Wu, and L. Jiao are with the Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education of China, Xidian University, Xi'an 710071, China (e-mail: asdwer2046@126.com; 21171213961@stu.xidian.edu.cn; avcodec@163.com; 22171214699@stu.xidian.edu.cn;wuzitong@stu.xidian.edu.cn;lchjiao@mail.-xidian.edu.cn).

J. Chanussot is with Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, Grenoble, 38000, France (email: jocelyn.chanussot@gipsa-lab.grenobleinp. fr).

This work was supported in part by the National Natural Science Foundation of China under Grant 62001355,61671350,61771379,61836009; the Foundation for Innovative Research Groups of the National Natural Science Foundation of China under Grant 61621005; the Key Research and Development Program in Shaanxi Province of China under Grant 2019ZDLGY03-05 and 2022GY-067.

often have varying classes, which makes it almost impossible to have many standard and unique annotations. There may be new classes in some regions that have been identified by remote sensing images of different regions, or the label maker may have divided the information in more detail, which will result in new classes being identified.

It is crucial in the field of remote sensing to design an incremental learning method for the reasons outlined above. When faced with additional remote sensing datasets from around the world, the model can learn new information about the new classes while maintaining the performance of the old classes without accessing the entire previous training data. In addition to reducing training time, it can also improve the model's ability to cope with a variety of data types. Usually, remote sensing image datasets are derived from large-scale images, which results in unbalanced classes. Additionally, because the model receives different classes at different stages of incremental learning, it will also cause an imbalance between old and new classes.

To solve the problems of large-scale remote sensing, class imbalances, and different numbers of classes in incremental learning, we devised adaptive weights. Adaptive weights can be applied depending on the number of classes in the image and the number of old and new classes. This method is called adaptive weights. Due to the lack of old classes' information in the training phase, it is necessary to obtain this information using the old model, and simultaneously, a strategy must be developed for filtering out the pixels with a high probability as part of the fusion label. This method is called label strategy.

In summary, the contributions of this paper are as follows:

1. To improve the performance of land cover classification tasks in incremental learning, we propose a new method. It has the capability of simultaneously addressing the class imbalance problem in remote sensing images at the current stage and at different stages in the incremental task, as well as making reasonable use of the incorrectly predicted classification information of land cover.

2. A novel label strategy is proposed aimed at classifying sample points into different classes based on their probabilities of belonging to the correct class. It is necessary to perform special processing for erroneous samples points since the current stage must be trained using the labels obtained from the model inference in the previous stage. Our label strategy can eliminate the possibility of erroneous sample points in remote sensing images that may adversely affect the analysis.

3. In order to address the issue of class imbalance in remote sensing images, novel adaptive weights are proposed. There is a greater problem of class imbalance in remote sensing images as compared with optical images. In this approach, the classes' weights can be adjusted adaptively based on the difference in the number of classes. This method effectively addresses the issue of class imbalance in remote sensing images.

II. Related Work

A. Semantic segmentation

Semantic segmentation methods can be classified into traditional methods and deep-learning-based methods. With traditional methods, different land covers can be distinguished by the use of feature extraction algorithms and classifier design techniques. As a result of the uneven types of objects in remote sensing images and the large differences in expressions of similar objects, it is difficult to obtain satisfactory results.

Since the development of deep learning, semantic segmentation has made significant progress. In 2014, Long *et al*. [1] proposed the concept of a fully convolutional network (FCN), which improved the structure of the original convolutional neural network (CNN). By using a deconvolution layer, it is possible to upsample the feature map generated by the last convolution layer from any size input image. During upsampling, the feature map is restored to the same size as the input image, allowing a prediction to be generated for each pixel while preserving the spatial information. The upsampled feature map is then classified pixel-by-pixel. The later improvements focus primarily on upsampling and skipping layers of FCN, such as SegNet, DeconvNet, and DeepLab(DL) [14]–[16]. U-net [17] is a symmetric semantic segmentation model. Following this, a number of multi-scale methods were developed, including DeepLabv3 [18] and PSPNet [19]. Due to different representations and sensor-induced scale transformations, these methods effectively address the issue of different scales of similar objects.

The use of these networks and their variants is widespread in the field of remote sensing. For example, [20] embedded an adversarial complementary learning strategy into a convolutional neural network, which is able to extract complementary information from multi-source data. In order to extract meaningful multiscale information and fuse features from multisource data, [21] developed an interactive multiscale information extraction block and a global dependence fusion module. In [22], a graph feature extraction module and a novel graph fusion strategy-graph dependence fusion is designed to extract topological structure information and combine with the rich spectral-spatial information and enhance the association and interaction between different graph features. The methods described above are various algorithms for analyzing remote sensing images and contributing to the land cover classification.

B. Incremental Learning

The purpose of incremental learning is to solve the catastrophic forgetting problem. The issue has been extensively studied in the context of image classification tasks. There are three categories of previous work: replay-based, parameter-isolation-based, and regularization-based. As a way of resolving catastrophic forgetting, the three types of work employ different approaches.

1) Replay-based: The replay-based approach involves storing some samples of previous training data or generating previous training data. As a result, the model can maintain memory for old data while dealing with new data. There are a number of methods that are representative of this concept, including Incremental Classifier and Representation Learning (iCaRL) [9], Deep Generative Replay (DGR) [23], and Memory Replay GANs (MRG) [24].

During training, iCaRL stores a small amount of data from the old classes and gradually adds new classes. It learns a

strong classifier and data representation simultaneously. As a novel framework, DGR consists of a collaborative dual model architecture that combines a deep generative model with a task-solving model. Training data for previous tasks can easily be sampled and interspersed with those for new tasks using only these two models. According to MRG, sequential fine-tuning prevented the network from correctly generating images based on previous classes. This issue is addressed by a conditional GAN framework that incorporates memory replay generators into joint training with replay and replay alignment.

2) Parameter-isolation-based: In general, a method based on parameter isolation extends the network. Its representative methods are PackNet [25], Piggyback [26] and progressive neural networks (PNNs) [27].

By performing iterative pruning and network retraining, PackNet sequentially incorporates multiple tasks into a single network while minimizing performance degradation and storage overhead. Piggyback is a novel method of obtaining good performance on new tasks by taking advantage of the fixed weights of the network. Through the use of a new sensitivity metric, PNNs can leverage prior knowledge by connecting laterally to previously learned features.

3) Regularization-based: Prior-centric representative methods in the regularization-based approach include path integral (PI) [28], elastic weight consolidation (EWC) [29], and Riemannian walks (RW) [30]. Knowledge is defined as parameter values that limit the learning of new tasks by penalizing important parameter changes from the old tasks.

The PI introduces intelligent synapses that accumulate task-relevant information over a period of time. In order to memorize learned tasks, the EWC slows down the learning of weights that are critical to those tasks. RW presents RWalk, a generalization of EWC and PI with a theoretically grounded Kullback-Leibler-divergence-based perspective.

Representative data-centric methods are learning without forgetting(LWF) [8], LwF multi-class(LWF-MC) [9], incremental learning techniques(ILT) [31], modeling the Background(MiB) [32], and Pseudolabeling and LOcal Pod(PLOP) [33]. To prevent catastrophic forgetting, these methods utilize distillation and the distance between the activations produced by the old and new networks as a regularization term.

The LWF can be viewed as a hybrid of knowledge distillation and fine-tuning. Based on the training data, the learned parameters are discriminative for the new task while maintaining the output of the original task. LWF-MC learns strong classifiers and a data representation simultaneously. ILT retains the knowledge of previously learned classes while updating the current model to learn new ones based on the distillation of the knowledge of the previous model. MiB introduces a new distillation-based framework and introduces a new method for initializing classifier parameters. PLOP proposes a multi-scale pooling distillation scheme that preserves long-range and short-range spatial relationships at the feature level. Additionally, a pseudo-label based on entropy is designed.

C. Incremental learning for land cover classification

During the past few years, the use of deep learning methods for classification has greatly improved the classification effect and the ability of this task to be applied. The model, however, will always suffer catastrophic forgetting in the face of updated data and new classes every day fetched from every corner of the globe. Compared to ordinary images, remote sensing images are larger. There is more information and classes contained in a remote sensing image. It is possible to include different classes in a single remote sensing image. In general, class imbalances pose a more serious problem than ordinary images. Due to the larger areas occupied by background classes in remote sensing images, the background offset problem is also more challenging. The Fig 1 illustrates the incremental learning process for the entire remote sensing image task.

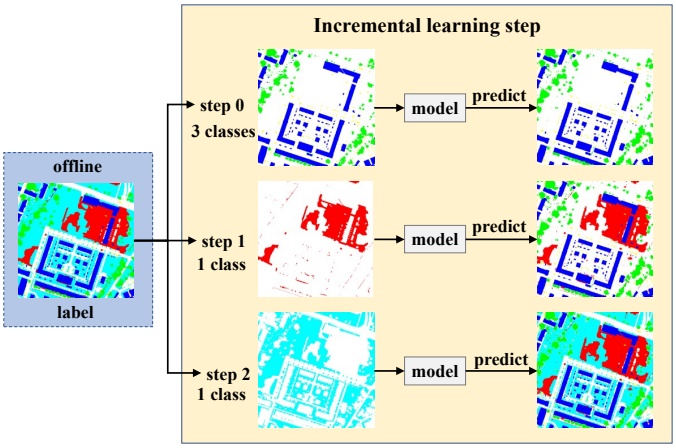


Fig. 1: Changes in labels during incremental learning.

There has also been research on incremental learning in remote sensing. A radial basis function (RBF) is proposed by Bruzzone L *et al.* [34] so that new information can be acquired periodically as new training sets become available while retaining the knowledge gained from previous training sets. During each retraining stage, the network architecture is automatically updated in order to accommodate new classes. A frozen copy of a previously trained network is maintained by Tasar O *et al.* [35] in order to update the network in the absence of pre-class annotations. An updated network balances the difference between the outputs from memory and previous classes. Rong X *et al.* [36] propose a feature global awareness module and a label reconstruction module. The former enables the current model to pay more attention to regions related to old classes identified by historical information when learning new classes. During this time, the latter retrieves pixels belonging to the learned class from the background to address the background shift problem and maintain the performance of the old classes.

It is evident that incremental learning in land cover classification is still relatively rare. Our research expands the research content of incremental learning and studies the problems existing in the process of incremental learning of remote sensing images. Detailed information about the method is provided in the following section.

III. METHODOLOGY

A. Problem definition and notation

For the convenience of readers, we have added a symbol table that contains the main symbols that appear in this article. Table I presents the main symbol table.

TABLE I: MAIN SYMBOL TABLE

Symbol of stage	
t	the current stage
$t-1$	the previous stage
t'	the subsequent stage
Symbol of classes	
C^b	the background class
C^{t-1}	the set of new classes in the $t-1$ training stage
C^t	the set of new classes in the t training stage
$C^{t'}$	the set of new classes in the t' training stage
C	the total set of classes
c_i	the i-th class
Other main symbol	
H	the height of the input image
W	the width of the input image
N	the total number of pixels of the input image
$\boldsymbol{X}$	the input image
$\boldsymbol{Y}$	the input label
$\boldsymbol{A}^t$	the output matrix of the t stage model
$\boldsymbol{G}^{t-1}$	the pseudo-label predicted according to the $t-1$ stage model
$\boldsymbol{L}_{c_i}$	the probability set of pixel belonging to the c_i in the pseudo-label
$\tilde{\boldsymbol{L}_{c_i}}$	the decreasing degree of confidence that the pixel belongs to the c_i.
$\boldsymbol{M}$	pseudo-label generated after the algorithm
ψ	The set of images
w_{c_i}	the weight of c_i
$len(C)$	the number of elements in the C
$p_x^{c_i}$	the probability that the x-th pixel of the c_i belongs to this class
η_{c_i}	the threshold of the class c_i
v_i	the pixel condition that participates in the calculation of the distillation function
n_{c_i}	the number of pixels belonging to c_i in each image
μ_{c_i}	the prediction accuracy of the previous stage model for the c_i
n_{t-1}	the sum of pixels in the merged label for the background and old classes
q_i	the probability that the pixel belongs to the most likely class inferred by the current model

In the incremental phase the current training stage is t, the previous stage is $t-1$, and subsequent stage is t'. The relationship between the number of classes at different stages is as follows:

$$C = C^b \cup C^{t-1} \cup C^t \cup C^{t'}, \tag{1}$$

in which C represents the total set of classes, and C^b stands for background class. C^{t-1}, C^t, and $C^{t'}$ are the set of new classes in the $t-1$, t, and t' training stage.

The set of images is ψ=($\boldsymbol{X}$, $\boldsymbol{Y}$). The input image is denoted as $\boldsymbol{X}$, $\boldsymbol{X}\in X^{\mathrm{H}\times\mathrm{W}\times 3}$, where $X \in \{i\}_{i=0}^{255}$, H and W represent the height and width of the input image. The total number of pixels of the input image is N, and the input label is denoted as $\boldsymbol{Y}$, $\boldsymbol{Y} \in (C^b \cup C^t)^{\mathrm{H}\times\mathrm{W}}$, where $(C^b \cup C^t)^{\mathrm{H}\times\mathrm{W}}$ represents a matrix with a length of H and a width of W, and the matrix elements belong to the $C^b \cup C^t$ set.

The overall frame diagram is shown in the Fig 2. GM stands for Generate Merge module. This module generates a trainable pseudo-label map from the merged label, which contains the old classes and the real label of the current stage. This algorithm filters the labels that contain old classes based on the predictions of the previous stage. The cross-entropy module uses the generated merged label ($\boldsymbol{M}$) and the output matrix($\boldsymbol{A}^t$) of the current stage model for calculation. To enhance the model's memory of old classes, the knowledge distillation module uses different model information from the current stage($\boldsymbol{A}^t$) and the previous stage($\boldsymbol{A}^{t-1}$).

B. Cross-entropy based on merged label and label smoothing

In the first stage of training, the distillation function is 0 due to $t-1<0$, and the loss function contains only cross-entropy. Background pixels received in the first stage contain potential new classes that may emerge in subsequent stages. The weight of the background is therefore reduced based on the proportion of each class to the overall classes.

In this study, the offset function of sigmod denoted as $g(x)$ is used as the mapping for many cases. The calculation formula is as follows:

$$g(x) = \frac{1}{1+e^{-(x-1)}}. \tag{2}$$

The function $g(x)$ is equivalent to a translation transformation of the sigmod function. When adaptive weights are calculated, the ratio of the number of classes is positive. Nevertheless, it may be extremely large or small, which will have an impact on the training effect. Using the function $g(\boldsymbol{x})$, the calculated value is transformed. This transformation allows the value to be distributed in a more reasonable manner when the weight is adaptively adjusted later, thereby improving the model's training. The calculation formula is as follows:

$$w_{c_i} = \begin{cases} g\left(\dfrac{\frac{\mathrm{N}}{n_{c_i}}}{len(C^{t'})+1}\right) & c_i \in C^b \\ g\left(\dfrac{\mathrm{N}}{n_{c_i}}\right) & c_i \in C^t \end{cases}, \tag{3}$$

where w_{c_i} represents the weight of c_i, c_i represents the i-th class, n_{c_i} represents the number of pixels belonging to c_i in each image, and $len(C^{t'})$ represents the number of elements in the $C^{t'}$, which represents the number of new classes that the model will learn in subsequent stages.

During the learning process, by reducing the weight of the background, the model is more likely to avoid misclassifying future new classes as background. As a result, the model is able to devote more attention to learning the correct classes. In future stages of training, the learning restrictions can be relaxed so that the model can better accommodate the new classes. As soon as $t>0$, the incremental phase of training begins. The prediction accuracy for the background and the old classes from the previous stage is stored in a list. The prediction accuracy of the previous stage model for

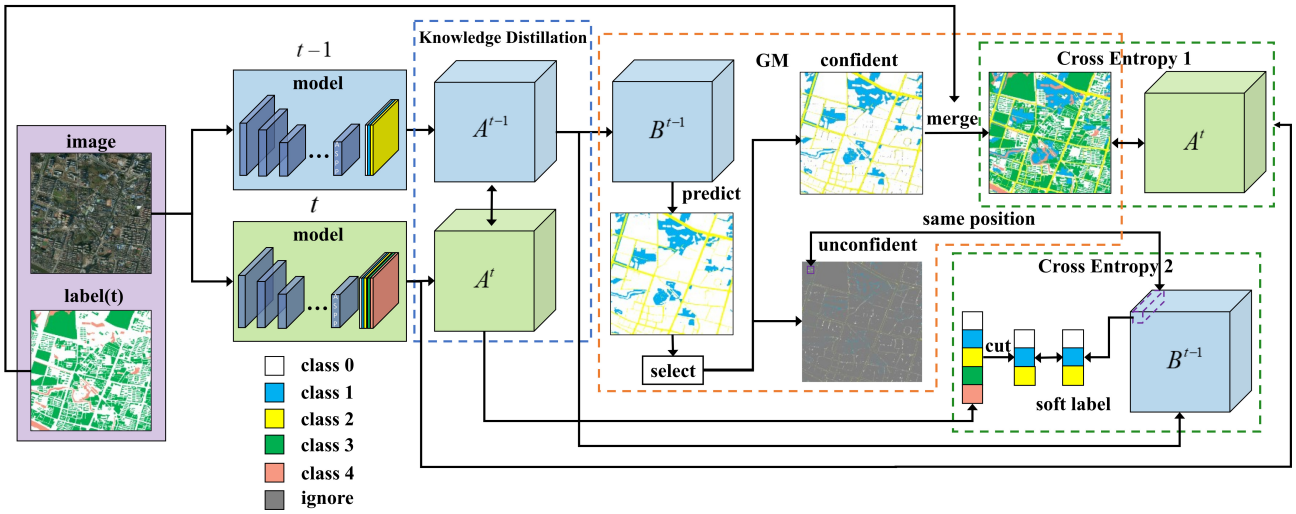


Fig. 2: Structure of the proposed method. GM stands for Generate Merge. The cross-entropy module uses the generated merged label ($\boldsymbol{M}$) and the output matrix($\boldsymbol{A}^t$) of the current stage model for calculation. To enhance the model's memory of old classes, the knowledge distillation module uses different model information from the current stage($\boldsymbol{A}^t$) and the previous stage($\boldsymbol{A}^{t-1}$).

the background and old classes is recorded as $[\mu_{c_i}]$, where $c_i \in C^b \cup C^{t-1}$.

1) Merged label: This part introduces the Generate Merge(GM) module in Fig 2.

Sort-Select: The sort-select module is shown in Fig 3. The pixels of each class are sorted according to probability and selected according to a calculated threshold in this part. The selected pixels are used as a component of the subsequent merged label.

The output matrix of the previous stage model and the current stage model are $\boldsymbol{A}^t$ and $\boldsymbol{A}^{t-1}$. $\boldsymbol{A}^{t-1} \in R^{\mathrm{H}\times\mathrm{W}\times len(C^b\cup C^{t-1})}$, $\boldsymbol{A}^t \in R^{\mathrm{H}\times\mathrm{W}\times len(C^b\cup C^{t-1}\cup C^t)}$, where $len(x)$ represents the number of distinct elements in the set x. According to the previous stage model output matrix $\boldsymbol{A}^{t-1}$, we perform a softmax operation on the category dimension to generate the matrix $\boldsymbol{B}^{t-1}$, where $\boldsymbol{B}^{t-1} \in R^{\mathrm{H}\times\mathrm{W}\times len(C^b\cup C^{t-1})}$. The pseudo-label predicted according to the previous stage model is $\boldsymbol{G}^{t-1}$, where $\boldsymbol{B}^{t-1} = \arg\max(\boldsymbol{a}_i^{t-1})[h,w,c]$, $\boldsymbol{G}^{t-1} \in (C^b \cup C^{t-1})^{\mathrm{H}\times\mathrm{W}}$.

The parts of the image pixels received in the current stage that are not of the new classes is integrated into a list $\boldsymbol{L}_{c_i}$. $\boldsymbol{L}_{c_i} = \left[p_1^{c_i}, p_2^{c_i}, p_3^{c_i}, \ldots, p_{n_{c_i}}^{c_i}\right]$, where $\boldsymbol{L}_{c_i}$ represents the probability set of pixel belonging to the c_i in the pseudo-label, and $p_x^{c_i}$ represents the probability that the x-th pixel of the c_i belongs to this class.

The elements in the list represent the probability that the x-th pixel in the pseudo-label $\boldsymbol{G}^{t-1}$ belongs to the c_i class. Sort the list in descending order, and denote the list in descending order as $\tilde{\boldsymbol{L}_{c_i}}$. $\tilde{\boldsymbol{L}_{c_i}}$ represents the decreasing degree of confidence that the pixel belongs to the c_i. The higher the probability of the pixel point belonging to c_i, the higher the degree of confidence and the greater the probability that it can be used as a pseudo-label. We can calculate the threshold:

$$\tilde{n}_{c_i} = len\left(\tilde{\boldsymbol{L}_{c_i}}\right), \tag{4}$$

where $\tilde{n}_{c_i}$ represents the elements' number of $\tilde{\boldsymbol{L}_{c_i}}$. The calculation formula is as follows:

$$\eta_{c_i} = \tilde{\boldsymbol{L}_{c_i}}\left[\tilde{n}_{c_i} \times \mu_{c_i}\right], \tag{5}$$

where η_{c_i} represents the threshold of the class c_i, and μ_{c_i} represents the prediction accuracy of the previous stage model for the c_i. Then we select the pixels larger than the threshold as the components of the confident pseudo-label.

Generate Merged Label: After obtaining the required threshold for each old class, the confident part of the pseudo-label is generated. $\boldsymbol{M}$ is the pseudo-label generated after the algorithm, where $\boldsymbol{M} \in (C^b \cup C^{t-1} \cup C^t)^{\mathrm{H}\times\mathrm{W}}$. The specific generation method is as follows:

$$m_i = \begin{cases} g_i^{t-1} & p_x^{c_j} \geq \eta_{c_j}, y_i \in C^b, g_i^{t-1} = c_j \in C^{t-1} \\ 255 & p_x^{c_j} < \eta_{c_j}, y_i \in C^b, g_i^{t-1} = c_j \in C^{t-1} \\ y_i & else \end{cases}, \tag{6}$$

where m_i is the i-th element of $\boldsymbol{M}$, g_i^{t-1} is the i-th element of $\boldsymbol{G}^{t-1}$, and y_i is the i-th element of $\boldsymbol{Y}$. For pixels belonging to the background class in the current stage label($\boldsymbol{Y}$), the real label may be the background class or the old classes. It is likely that a pixel belonging to the j-th class will be judged correctly if its probability exceeds or equals the threshold of the class. Therefore, it can be used for subsequent training. The pseudo-label value of a pixel is set to 255 when its probability of belonging to the j-th class is lower than the threshold for this class. This indicates that this pixel is ignored during this part. The pixels set to 255 are reserved for the calculation of the next part. At the current stage, if the pixel's label is a new class, then its specific new class is used as the category of the merged label.

The weights of the classes in this part are calculated after the required confident labels have been obtained. Datasets are obtained by cutting from large images, so they differ from the original images. Some images are likely to have only a few

classes, so when weighting the classes, it is necessary to soften this part, and not use the ratio of each class to the overall class directly. Each class' weight is updated using the formula. The formula is as follows:

$$n_{t-1} = \sum \tilde{n}_{c_i} \qquad c_i \in C^b \cup C^{t-1}, \tag{7}$$

in which n_{t-1} is the sum of pixels in the merged label for the background and old classes.

$$w_{c_i} = \begin{cases} g(\dfrac{n_{t-1}/n_{c_0}}{(\mathrm{N}/n_{t-1}) \times len(C^{t'})}) & m_i \in C^b \\ g(\dfrac{n_{t-1}/n_{c_i}}{\mathrm{N}/n_{t-1}}) & m_i \in C^{t-1} \\ g(\dfrac{n_{t-1}}{n_{c_i}}) & m_i \in C^t \end{cases}. \tag{8}$$

The equation above illustrates how the adaptive weights are calculated. After the model receives the image, it judges each pixel according to the merged label $\boldsymbol{M}$ generated in the previous part. It is likely that the pixel's class is the true old class if it appears as an old class in the merged label. The molecular part n_{t-1}/n_{c_0} represents the inverse of the proportion of the c_i vector in the old class. The denominator part N/n_{t-1} represents the inverse of the proportion of the old class in the total class. A small proportion of the c_i in the old class will lead to a larger molecule, which will allow the c_i to maintain its learning ability. Additionally, if the old classes constitute a small portion of the overall class, a limited number of old class pixels will remain in the pseudo-label after fusion. There may be a difficulty in learning the old class, resulting in a low level of model prediction accuracy, or there may be a limited number of pixels in the old class. By reducing the weight of the old class c_i, the model can reduce the influence of misclassified old classes' labels and focus on the new classes. If m_i is the background, the only difference from the old classes is that the denominator part has more $len(C^{t'})$, because the background pixels at the current stage may also be new classes in the next stage. So, it is necessary to further weaken the weight of the background according to the number of new classes. The labels are accurate if m_i is a new class. However, in order to learn about the new class, it is necessary to take into account the number of old classes in the image, so n_{t-1}/n_{c_i} is used. The weight will be greater if the number of pixels in the new class c_i is smaller than the number of pixels in the old class. Doing so is more conducive to learning new classes.

Based on the weight parameters of each class in the confidence label, the confidence part of the cross-entropy is calculated as follows:

$$l_{certain} = -\sum_{c_i} \frac{w_{c_i}}{\mathrm{N}} \sum_{j=1}^{\mathrm{N}} \log(q_j) \times q_j, c_i \in C^b \cup C^{t-1} \cup C^t \tag{9}$$

where q_i represents the probability that the pixel belongs to the most likely class inferred by the current model.

2) Label Smoothing: The model in the previous stage can't predict a particular class satisfactorily. Pseudo-labels will not be generated correctly when the threshold of some pixel points is lower than the threshold we calculate. So we use not only the values of pixels that are confident enough to calculate in the previous part, but also pixels with probabilities smaller than the threshold in this part. By including the probability of each class in the calculation, the risk of incorrect labels is reduced.

The second part of the loss function $l_{uncertain}$ generated by the uncertain pixels in the label is as follows:

$$l_{uncertain} = -\frac{1}{\hat{n}} \sum_{i=1}^{\hat{n}} \hat{a}_i^t \cdot a_i^{t-1}, \tag{10}$$

in which $\hat{a}_i^t$, a_i^{t-1} represent the element of $\hat{\boldsymbol{A}}^t$ and $\boldsymbol{A}^{t-1}$, where $\hat{\boldsymbol{A}}^t$ is the output matrix of the current stage model excluding the part of the new classes. $\hat{n}$ represents the number of pixels participating in the calculation of the formula. The condition is that the current label is C^b, the pseudo-label is C^{t-1}, and the probability of belonging to the pseudo-label is less than the threshold. The labels of the pixels that meet the requirements are all uncertain old classes.

As shown in the Fig 3, during the training, the image obtains the output matrix $\boldsymbol{A}^{t-1}$ through the old model, the label map $\boldsymbol{G}^{t-1}$ is predicted according to the old model to separate the pixels of different classes, and the pixels of each class are clustered together. Pixels within each class are arranged in descending order based on the size of the predicted value. The sample points with high confidence are selected as the pixels participating in the calculation of $l_{certain}$. The remaining sample points are used as pixels participating in the calculation of $l_{uncertain}$. The pixel label of the unconfident label is directly output. In the Generate Pseudo labels (GP) module, the label for the selected confident sample point is merged with the label for the current stage. The merged label map $\boldsymbol{M}$ is output, where $\boldsymbol{M}$ is the label of $l_{certain}$ and uncertain is the label of $l_{uncertain}$.

The overall cross-entropy function is as follows:

$$loss_{ce} = l_{certain} + \alpha \cdot l_{uncertain}. \tag{11}$$

in which α is a hyperparameter that can be used to adjust the balance between different loss functions.

C. Knowledge distillation based on class information reconstruction

It is added to the loss function as a regularization term, keeping the model connected to the old classes while learning new classes, effectively solving the problem of catastrophic forgetting. After obtaining the model background and the threshold of the old classes, the pseudo-labels used in the distillation function are assigned according to the following rules:

$$v_i = 1 \quad y_i \in C^b, g_i^{t-1} \in C^b \cup C^{t-1}, \tag{12}$$

in which v_i represents the pixel condition that participates in the calculation of the distillation function, g_i^{t-1} is the i-th element of $\boldsymbol{G}^{t-1}$, and y_i is the i-th element of $\boldsymbol{Y}$. When v_i is equal to 1, the corresponding pixel points participate in the calculation of the distillation function. That means v_i is equal to 1 only when the current stage label and the generated

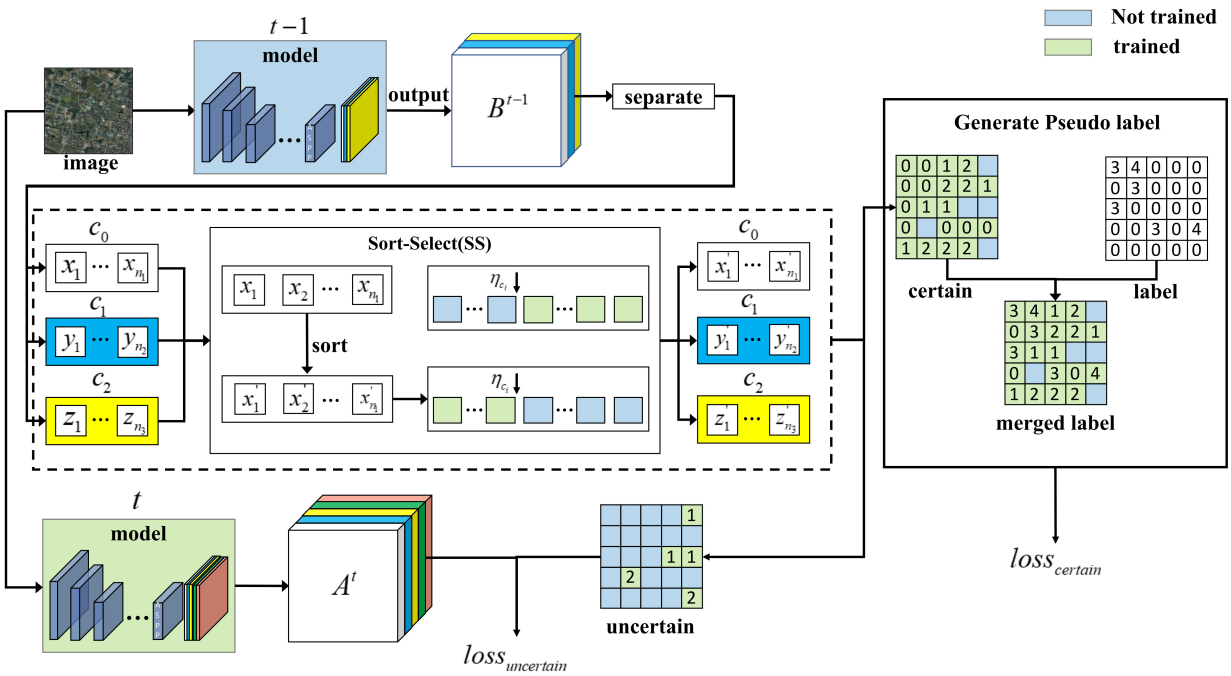


Fig. 3: A detailed illustration of the Generate Merge module. After the input image passes through the model of the previous stage, the output matrix($\boldsymbol{B}^{t-1}$) is generated through the softmax function mapping. In the Sort-Select (SS) module, the pixels are sorted in descending order according to the value of the class. Then the pixels' threshold greater than η are selected as components of the confident label. This part is merged with the real labels of the current stage to generate a merged label. The remaining pixels are used as components of unconfident label and participate in the calculation together.

previous stage predicted label are both background or the generated previous stage predicted label is the old class.

The weights of classes are calculated as follows:

$$w_{c_i} = g(\mathrm{N}/n_{c_i}). \tag{13}$$

To calculate the loss function, we count the number of different classes of pseudo-labels predicted by the old model and add them to the calculation. In this way, class imbalance can be effectively alleviated.

The i-th pixel vector of the output matrix $\boldsymbol{A}^{t-1}$ of the old model is $\boldsymbol{a}_i^{t-1}$, and the i-th pixel vector corresponding to the output matrix $\boldsymbol{A}^t$ of the current model is $\boldsymbol{a}_i^t$. The $\boldsymbol{a}_i^{t-1}$ vector is $[x_{c_i}^{t-1}], c_i \in C^b \cup C^{t-1}$, and the $\boldsymbol{a}_i^t$ vector is $[x_{c_i}^t], c_i \in C^b \cup C^{t-1} \cup C^t$. The following formula operations are performed:

$$\tilde{x}_{c_0}^t = x_{c_0}^t + \sum_{c_i} x_{c_i}^t \quad c_i \in C^t. \tag{14}$$

The transformed $\boldsymbol{a}_i^t$ vector is denoted as $\tilde{\boldsymbol{a}}_i^t$. This allows us to reconstruct the information about the classes. Based on the information provided by the old and new models about the input image, the following formula can be used to calculate the knowledge distillation.

The overall knowledge distillation function is:

$$loss_{kd} = -\sum_{c_k} \frac{f\left(w_{c_k}\right)}{\mathrm{N}} \sum_{i=1}^{\mathrm{N}} \log\left(f\left(\boldsymbol{a}_i^{t-1}\right)\right) * f\left(\tilde{\boldsymbol{a}}_i^t\right). \tag{15}$$

in which $c_k \in C^b \cup C^{t-1}$.

As shown in the Fig 4, during the training, the image obtains the output matrices $\boldsymbol{A}^{t-1}$ and $\boldsymbol{A}^t$ through the old model and the current model. They serve as component part of the computation of the distillation loss function. At this stage, the vector of the model output matrix $\boldsymbol{A}^t$ is numerically converted, as shown in the figure, and the values of the background and the new classes are added as the real output value of the background. Because the background and new classes' pixels of the current stage belong to the background pixels in the previous stage, the memory ability of the model can be better maintained by this processing, and the learning of the new classes will not be affected by the model. The pixel point vector $\boldsymbol{a}_i^{t-1}$ in the output matrix $\boldsymbol{A}^{t-1}$ of the old model (dimension is $(1, len(C^b \cup C^{t-1}))$) and the pixel vector $\tilde{\boldsymbol{a}}_i^t$ (dimension is $(1, len(C^b \cup C^{t-1}) \cup C^t)$) of the current model output matrix $\boldsymbol{A}^t$ are calculated for $loss_{kd}$. By reducing the difference between the two vectors in this way, the purpose of keeping the memory of the old classes is achieved.

D. Overall loss function

As the training time increases, the regularization coefficient in the distillation function increases slowly, which can effectively alleviate the forgetting of the old classes and enhance the model's memory and retention ability for the old classes:

$$loss_{total} = loss_{ce} + \beta \cdot f\left(\frac{e_t \cdot s_t}{e_n \cdot s_n}\right) \cdot loss_{kd}, \tag{16}$$

in which β is a hyperparameter that can be used to adjust the balance between different loss functions, e_t represents the epoch of the current stage, s_t represents the step in the current stage, and e_n and s_n represent the total epochs and

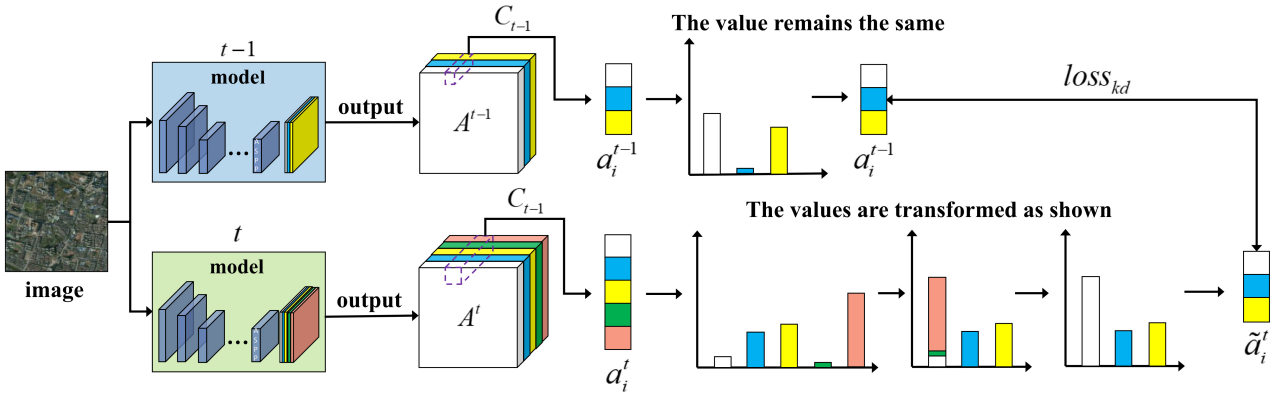


Fig. 4: Schematic diagram of distillation function calculation. By passing the input image through the previous stage model and the current stage model, two matrices are generated, $\boldsymbol{A}^{t-1}$ and $\boldsymbol{A}^t$. For matrix $\boldsymbol{A}^t$, the values of all new classes received at the current stage are added to the background classes' values. In this case, the dimension is changed to the number of the old classes. Following that, the pixels at the same position of both matrices are calculated.

total steps in the training respectively. The distillation loss will not be able to constrain the new model effectively if β is too small. An unbalanced loss function will prevent the model from learning features if β is too large. Accordingly, β serves to balance the values of the two loss functions.

IV. Experimental Analysis and Results

A. Experiment preparation

1) Dataset: Datasets Vaihingen and Potsdam are provided by ISPRS [37]. There are many detached buildings and small multi-story buildings in Vaihingen, which is a relatively small village. This dataset contains 33 images of an average size of 2494×2064, which have been extracted from a larger top-level orthophoto image. This process prevents the occurrence of a situation in which there is no data. In the dataset, we use images of RGB bands. The spatial resolution is 9 cm.

In contrast, Potsdam is a typical historic city with large building blocks, narrow streets, and dense settlements. A total of 28 images are contained in the dataset, with a size of 6000×6000. This dataset also consists of three-band remote sensing TIFF files as well as single-band DSM files, in the same way as the Vaihingen region. Moreover, we used images of RGB bands in the dataset as well. The spatial resolution is 5 cm.

Both datasets have been classified manually into six categories: impervious surfaces, building, low vegetation, high vegetation, car, and clutter.

China Computer Federation (CCF) [38] dataset is the third dataset from a competition supported by Jiage Data and China Computer Federation. Dataset includes a high-resolution remote sensing image of a region in southern China, as well as surface cover samples (pictures) that have been visually interpreted based on the remote sensing image. It has a spatial resolution of sub-meters, a spectrum in the visible light band (R, G, B), and the coordinate information has been removed. There are five categories of samples provided: vegetation, building, water, road, and others.

The distribution of each category of the above three datasets is shown in the Fig 5.

2) Implementation details: We follow the parameters set in MiB for all methods. Resnet101 is used as the backbone and deeplabv3 is used as the model. A pretrained model of ImageNet has been used to initialize the backbone. We use the SGD algorithm for the gradient descent, with a corresponding decay of the learning rate over training time, with a decay rate of 0.9. In the incremental stage setting, the initial learning rate is 10^{-2} for the first stage, and 10^{-3} for the subsequent stages. α is set at 1 and β is set at 30. Training epochs for the Potsdam and Vaihingen datasets are 10 and 20 for the CCF dataset, respectively. During the first stage of training, we use a batch size of 12. During the next incremental stage, we trained the model with a batch size of 10. We randomly cut the remote sensing image into 512×512 image patches. With the CCF dataset, since there are only five images, the first four images are randomly cropped by 2500 images as the training set, and the last image is randomly cropped by 1000 images as the validation set. Due to the large number of images in the Potsdam and Vaihingen datasets, 1000 images are cropped for each image. For the Potsdam dataset, the first 31 images are used as the training set, while the remaining 7 images are used as the validation set. In the Vaihingen dataset, the first 25 images are selected as the training set and the remaining 8 images are selected as the validation set.

B. Comparative methods

We choose Fine-tuning(FT), EWC, RW, PI, ILT, LWF, LWF-MC, MiB, and PLOP as the comparison algorithms for our experiments. We compare them with our proposed algorithm. Joint represents that the model is trained on all classes, and it represents the upper limit of incremental learning performance. Table II, Table III, and Table IV represent the results of the algorithm on the CCF, Potsdam, and Vaihingen datasets, respectively. Fig 6, Fig 7, Fig 8 represent the result graphs of the algorithm on the CCF, Potsdam and Vaihingen datasets, respectively. In Fig 9 and Fig 10, the experimental results for the Potsdam and Vaihingen datasets are presented. The evaluation indicators are Mean Acc (MA), Overall Acc

TABLE II: THE EXPERIMENTAL RESULTS OF THE CCF DATASET

Method	Category IoU	Category Acc	MA	OA	mIoU
	background, vegetation, building, water, road	background, vegetation, building, water, road			
		2-2			
FT	[0.6209 0.0000 0.0000 0.7040 0.6206]	[0.7995 0.0000 0.0000 0.8482 0.7940]	0.4884	0.7438	0.3891
EWC	[0.5658 0.0007 0.0000 0.5888 0.6381]	[0.8482 0.0002 0.0000 0.7181 0.6722]	0.6015	0.7204	0.3586
RW	[0.5034 0.0000 0.0000 0.5490 0.2997]	[0.7299 0.0000 0.0000 0.7905 0.4623]	0.4562	0.6114	0.2704
PI	[0.5317 0.0000 0.0000 0.6289 0.6734]	[0.7264 0.0000 0.0000 0.9147 0.8273]	0.7088	0.7916	0.3668
ILT	[0.5545 0.7912 0.7772 0.5389 0.5863]	[0.6253 0.9028 0.8957 0.8297 0.7736]	0.6293	0.7406	0.6496
LWF	[0.6696 0.7945 0.7822 0.6134 0.6565]	[0.7615 0.9048 0.8956 0.8136 0.7718]	0.6034	0.722	0.7032
LWF-MC	[0.6743 0.8145 0.7948 0.3776 0.6662]	[0.9470 0.9078 0.9045 0.3900 0.6874]	0.6019	0.7183	0.6655
MiB	[0.7033 0.7814 0.7750 0.5359 0.7138]	[0.8979 0.9110 0.8980 0.5890 0.7518]	0.8095	0.7185	0.7019
PLOP	[0.6250 0.7352 0.7879 0.6271 0.5581]	[0.7481 0.8304 0.8974 0.8070 0.7412]	0.8048	0.7574	0.6667
LSAW	[0.7543 0.8037 0.7486 0.7182 0.8331]	[0.8400 0.8907 0.9458 0.8416 0.9187]	0.8874	0.8541	**0.7716**
		2-1-1			
FT	[0.5747 0.0000 0.0000 0.6116 0.0000]	[0.8388 0.0000 0.0000 0.7755 0.0000]	0.3229	0.6786	0.2373
EWC	[0.5573 0.0000 0.0000 0.5965 0.0000]	[0.8089 0.0000 0.0000 0.7861 0.0000]	0.3190	0.6667	0.2308
RW	[0.5636 0.0000 0.0000 0.5776 0.0000]	[0.8372 0.0000 0.0000 0.7339 0.0000]	0.3142	0.6645	0.2282
PI	[0.5450 0.0000 0.0000 0.6279 0.0000]	[0.7554 0.0000 0.0000 0.8847 0.0000]	0.3280	0.6710	0.2346
ILT	[0.5801 0.7632 0.5448 0.5172 0.8282]	[0.7160 0.8854 0.6814 0.7120 0.9182]	0.7826	0.7239	0.6467
LWF	[0.6739 0.7679 0.6943 0.5803 0.8251]	[0.8211 0.9018 0.8120 0.7068 0.9119]	0.8307	0.7895	0.7083
LWF-MC	[0.6039 0.7809 0.0682 0.2994 0.8224]	[0.9746 0.8848 0.0682 0.3062 0.9133]	0.6294	0.6686	0.5150
MiB	[0.6306 0.7440 0.6991 0.2813 0.8108]	[0.9299 0.9112 0.8342 0.2889 0.9050]	0.7738	0.7137	0.6332
PLOP	[0.6539 0.6604 0.6305 0.5792 0.7801]	[0.8117 0.6854 0.8221 0.7120 0.8899]	0.7842	0.7716	0.6608
LSAW	[0.7353 0.7627 0.7103 0.7193 0.8329]	[0.8175 0.8296 0.9767 0.8488 0.9476]	0.8841	0.8434	**0.7521**
Joint	[0.7840 0.8234 0.7307 0.8452 0.7990]	[0.8752 0.9172 0.8396 0.9324 0.9094]	0.8948	0.8701	0.7946

TABLE III: THE EXPERIMENTAL RESULTS OF THE POTSDAM DATASET

Method	Category IoU	Category Acc	MA	OA	mIoU
	background, building, car, vegetation, tree, clutter	background, building, car, vegetation, tree, clutter			
		3-2			
FT	[0.4164 0.0000 0.0000 0.0000 0.5119 0.5703]	[0.8778 0.0000 0.0000 0.0000 0.7090 0.8702]	0.4095	0.4961	0.2498
EWC	[0.4149 0.0000 0.0003 0.0000 0.5061 0.5438]	[0.9031 0.0000 0.0003 0.0000 0.5965 0.8114]	0.3852	0.4890	0.2442
RW	[0.4113 0.8636 0.0051 0.0000 0.5165 0.5537]	[0.9095 0.0000 0.0052 0.0000 0.5961 0.7913]	0.3837	0.4882	0.2478
PI	[0.4188 0.0000 0.0000 0.0000 0.4336 0.5671]	[0.8811 0.0000 0.0000 0.0000 0.6348 0.8659]	0.3970	0.4923	0.2366
ILT	[0.7372 0.8266 0.7435 0.6827 0.4621 0.5232]	[0.8523 0.8853 0.8208 0.8089 0.4895 0.7669]	0.7706	0.8207	0.6626
LWF	[0.7701 0.8462 0.7517 0.6805 0.4716 0.5748]	[0.8581 0.8942 0.8397 0.8028 0.5640 0.8214]	0.8304	0.8414	0.6825
LWF-MC	[0.7143 0.8686 0.7486 0.6905 0.2982 0.5176]	[0.9405 0.9074 0.8217 0.8189 0.3062 0.6002]	0.7325	0.8220	0.6396
MiB	[0.7828 0.8491 0.7487 0.6810 0.5259 0.5977]	[0.8522 0.9295 0.8672 0.8657 0.6243 0.7665]	0.8176	0.8487	0.6975
PLOP	[0.7603 0.8520 0.7458 0.6099 0.4407 0.5342]	[0.7984 0.9010 0.8309 0.6682 0.6210 0.8989]	0.7864	0.8177	0.6571
LSAW	[0.7965 0.8679 0.7584 0.6905 0.5266 0.6167]	[0.8938 0.9126 0.8999 0.8544 0.6120 0.7735]	0.8297	0.8530	**0.7094**
		3-1-1			
FT	[0.4042 0.0000 0.0000 0.0000 0.0000 0.5908]	[0.9532 0.0000 0.0000 0.0000 0.0000 0.7684]	0.2869	0.4654	0.1658
EWC	[0.4005 0.0000 0.0000 0.0000 0.0000 0.5483]	[0.9348 0.0000 0.0000 0.0000 0.0000 0.7662]	0.2835	0.4585	0.1581
RW	[0.3946 0.0000 0.0000 0.0000 0.0000 0.5276]	[0.9433 0.0000 0.0000 0.0000 0.0000 0.6973]	0.2735	0.4504	0.1537
PI	[0.4056 0.0000 0.0000 0.0000 0.0000 0.5807]	[0.9385 0.0000 0.0000 0.0000 0.0000 0.8081]	0.2911	0.4666	0.1643
ILT	[0.7181 0.8322 0.7404 0.6902 0.4200 0.4842]	[0.8829 0.8871 0.8078 0.8169 0.4378 0.6625]	0.7491	0.8130	0.6475
LWF	[0.7596 0.8580 0.7483 0.6924 0.5297 0.5386]	[0.9096 0.8944 0.8248 0.8202 0.5943 0.6930]	0.8238	0.8370	0.6877
LWF-MC	[0.6783 0.8546 0.7299 0.6837 0.0080 0.4797]	[0.9449 0.8870 0.7879 0.8010 0.0080 0.5750]	0.6673	0.7940	0.5723
MiB	[0.7787 0.8475 0.7484 0.6672 0.5059 0.5423]	[0.8676 0.9281 0.8790 0.8994 0.6862 0.6297]	0.8150	0.7185	0.6816
PLOP	[0.7011 0.8355 0.7312 0.6264 0.3845 0.5150]	[0.7179 0.8659 0.8504 0.6930 0.7785 0.8846]	0.7984	0.7894	0.6323
LSAW	[0.7567 0.8519 0.7717 0.6992 0.5497 0.5628]	[0.9096 0.8810 0.8915 0.8431 0.6756 0.6640]	0.8191	0.8392	**0.6986**
		2-2-1			
FT	[0.4064 0.0000 0.0000 0.0000 0.0000 0.5877]	[0.9609 0.0000 0.0000 0.0000 0.0000 0.7544]	0.2859	0.4659	0.1657
EWC	[0.3996 0.0000 0.0000 0.0000 0.0000 0.5303]	[0.9558 0.0000 0.0000 0.0000 0.0000 0.6892]	0.2742	0.4535	0.1550
RW	[0.3929 0.0000 0.0000 0.0000 0.0000 0.5023]	[0.9495 0.0000 0.0000 0.0000 0.0000 0.6516]	0.2668	0.4452	0.1492
PI	[0.4049 0.0000 0.0000 0.0000 0.0000 0.5932]	[0.9562 0.0000 0.0000 0.0000 0.0000 0.7653]	0.2869	0.4660	0.1664
ILT	[0.6980 0.8679 0.7306 0.6006 0.4119 0.4471]	[0.9027 0.9025 0.7931 0.7160 0.4386 0.5961]	0.7248	0.8010	0.6260
LWF	[0.7498 0.8617 0.7383 0.6622 0.4978 0.5171]	[0.9125 0.8899 0.8010 0.7944 0.5299 0.6854]	0.7689	0.8305	0.6711
LWF-MC	[0.5513 0.8533 0.7139 0.0237 0.1021 0.3866]	[0.9705 0.8863 0.7591 0.0237 0.1023 0.4455]	0.5312	0.6911	0.4385
MiB	[0.7521 0.8505 0.7490 0.5720 0.5051 0.3073]	[0.8949 0.9299 0.8839 0.9243 0.6342 0.3277]	0.7658	0.8011	0.6227
PLOP	[0.6978 0.8101 0.6751 0.6052 0.3638 0.5287]	[0.7176 0.8403 0.8334 0.8095 0.6772 0.8507]	0.7881	0.7847	0.6134
LSAW	[0.7512 0.8467 0.7602 0.6837 0.5324 0.5835]	[0.8958 0.8847 0.9154 0.8088 0.6637 0.7187]	0.8145	0.8403	**0.6930**
Joint	[0.8109 0.8502 0.7738 0.6840 0.5273 0.6230]	[0.8758 0.9197 0.8600 0.8365 0.6447 0.8065]	0.8239	0.8583	0.7115

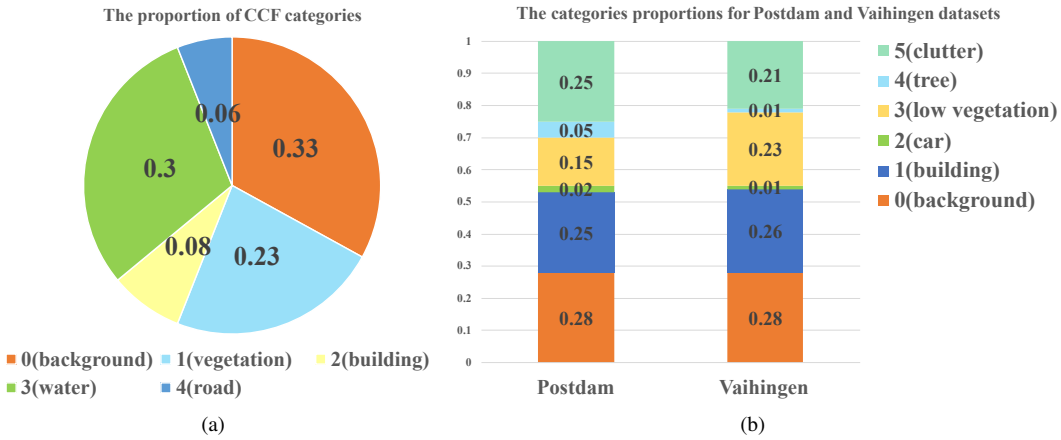


Fig. 5: The distribution of each category of the above three datasets.

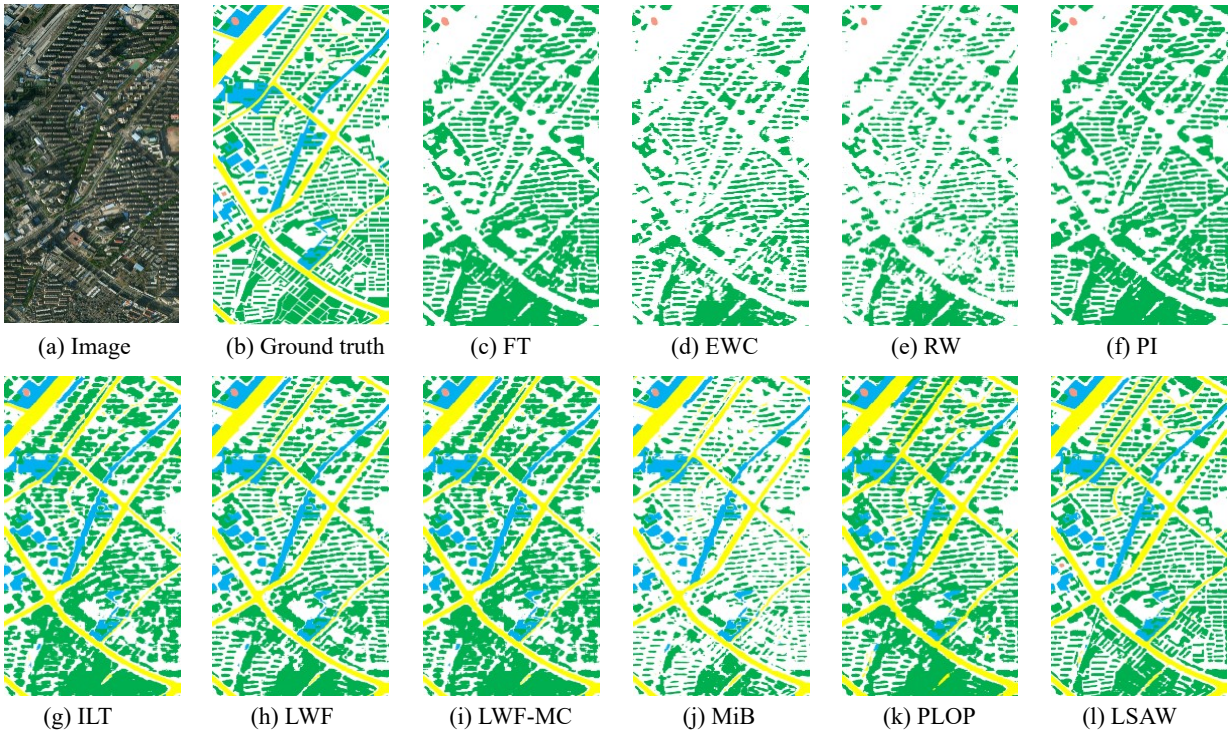
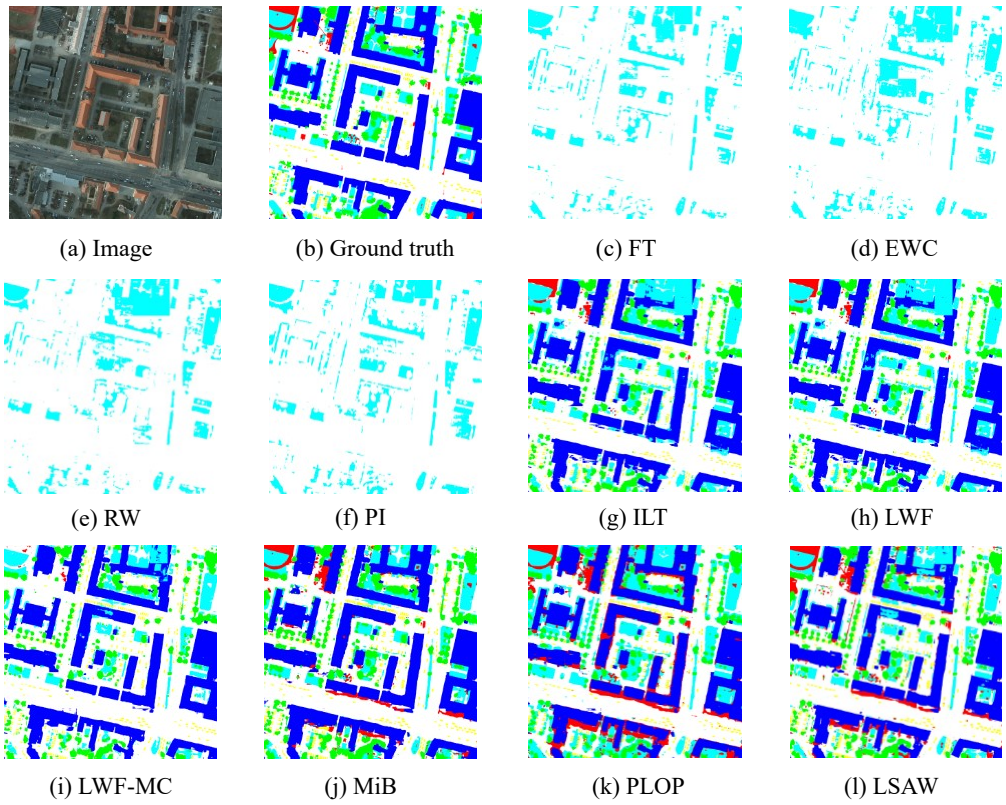


Fig. 6: Visualization of continual semantic segmentation results in the 2-2 (2 steps) setting of the CCF dataset. The picture's different columns of pictures indicate: (a) Image, (b) Ground truth, (c) FT, (d) EWC, (e) RW, (f) PI, (g) ILT, (h) LWF, (i) LWF-MC, (j)MiB, (k)PLOP, and (l) LSAW.

(OA), and mean Intersection over Union (mIoU). Our method achieves the best results.

1) Quantitative Analysis on the CCF Dataset: 1)Addition of Two Classes (2-2): In this experiment we perform two learning steps. In the first, we learn about the three classes of CCF, including the background. The second step involves learning the remaining two classes of CCF. A number of methods, including the FT, EWC, RW, and PI, perform poorly, and the mIoU of the previous old classes is close to 0. Thus, the old classes have almost completely been forgotten. In spite of this, prior-focused strategies are not competitive with data-focused ones. This confirms the effectiveness of this data-focused approach in preventing catastrophic forgetting, since ILT, LwF, LwF-MC, MiB, and PLOP substantially outperform them. While learning new classes, these methods preserve the ability to memorize the old ones. Their mIoU are all above 60%. Our method, however, maintains the memory ability of the old classes very well. Meanwhile, there is a significant improvement in the ability to learn new classes as well. Furthermore, our method achieves a mIoU of 77.11%, an improvement of 15% over the best-performing PLOP algorithm.

2)Addition of One Class in Two Steps (2-1-1): Three steps are involved in this experiment. In the first learning step, we learn the three classes of CCF, including the background. In the following steps, we will learn the remaining two classes



(a) Image (b) Ground truth (c) FT (d) EWC

(e) RW (f) PI (g) ILT (h) LWF

(i) LWF-MC (j) MiB (k) PLOP (l) LSAW

Fig. 7: Visualization of continual semantic segmentation results in the 3-1-1 (3 steps) setting of the Potsdam dataset. The picture's different columns of pictures indicate: (a) Image, (b) Ground truth, (c) FT, (d) EWC, (e) RW, (f) PI, (g) ILT, (h) LWF, (i) LWF-MC, (j)MiB, (k)PLOP, and (l) LSAW.

sequentially. FT, EWC, RW, and PI methods almost forget the previous classes, similarly to the above analysis. Other algorithms are still capable of remembering the old classes. In spite of this, due to the addition of new learning steps, the mIoU of all algorithms has been reduced, which is consistent with our theory. It is important to note that our algorithm still maintains a good performance, which is 6% higher than the best-performing LWF algorithm.

2) Quantitative Analysis on the Potsdam and Vaihingen Dataset: In the design of incremental learning tasks, Potsdam and Vaihingen are considered together since they have the same categories.

1)Addition of Two Classes (3-2): Two learning steps are performed in this experiment. In the first learning step, we learn the first four classes including the background of these two datasets. The second learning step involves learning the remaining two classes. Due to the fact that these two datasets have one more class than the CCF dataset and are composed of more complex data than the CCF dataset, the results of the experiment have declined. FT and PI methods have lost the ability to remember the third class, while EWC and RW methods still possess a weak memory for it. The ILT, LWF, LWF-MC, MiB, and PLOP methods still maintain good results. As the penultimate class occupies a small number of categories in the image, the algorithm has a poor learning effect on it. Additionally, our method achieves the highest mIoU for this class while still preserving the memory capacity of the older classes. Compared to the best performing MiB algorithm, our algorithm outperforms it by 1.7%.

2)Addition of One Class in Two Steps (3-1-1): Three learning steps are performed in this experiment. In the first learning step, we learn about the four classes of the two datasets, including the background. The remaining two classes are learned sequentially in the remaining two steps. It poses a challenge to the algorithm designers to maintain the memory ability of the penultimate classes during the final step of this experiment, due to the small number of penultimate classes. This class is not well memorized by algorithms. Despite this, we still achieve the highest mIoU value of 54.97% in this class. In terms of performance, our algorithm is superior to the best-performing LWF algorithm by 1.6%.

3) Addition of Two Classes in the First Step and Addition of One Class in the Second Step (2-2-1): Three learning steps are performed in this experiment. In the first learning step, we learn about the first three classes of the two datasets, including the background. The second learning step involves learning about the fourth and fifth classes. Lastly, we learn about the sixth class. It is important to note that since there is one fewer class available in the first stage, the model initially obtained less data information, which made all algorithms less accurate than previously experimented with. There is significant forgetting in the LWF-MC algorithm, which has performed relatively well in the past. This results in a mIoU value of 43.85%, which is significantly lower than those

TABLE IV: THE EXPERIMENTAL RESULTS OF THE VAIHINGEN DATASET

Method	Category IoU	Category Acc	MA	OA	mIoU
	background, building, car, vegetation, tree, clutter	background, building, car, vegetation, tree, clutter			
		3-2			
FT	[0.4742 0.0000 0.0000 0.0000 0.2551 0.5769]	[0.9228 0.0000 0.0000 0.0000 0.4346 0.9169]	0.3790	0.5234	0.2176
EWC	[0.4594 0.0003 0.0006 0.0000 0.1066 0.5995]	[0.9543 0.0003 0.0006 0.0000 0.1170 0.8219]	0.3157	0.5161	0.1943
RW	[0.4555 0.0029 0.0015 0.0001 0.1249 0.5979]	[0.9591 0.0029 0.0015 0.0000 0.1386 0.7918]	0.3157	0.5136	0.1971
PI	[0.4745 0.0000 0.0000 0.0000 0.2547 0.5785]	[0.9256 0.0000 0.0000 0.0000 0.4218 0.9134]	0.3768	0.5238	0.2179
ILT	[0.7738 0.8572 0.5603 0.6193 0.0802 0.5856]	[0.8826 0.9191 0.6319 0.6627 0.0895 0.8327]	0.6698	0.8380	0.5793
LWF	[0.7935 0.8586 0.5574 0.6137 0.2219 0.6020]	[0.8945 0.9176 0.6248 0.6533 0.2543 0.8532]	0.7253	0.8534	0.6078
LWF-MC	[0.7470 0.8630 0.5673 0.6366 0.0191 0.5480]	[0.9577 0.9140 0.6278 0.6909 0.0192 0.6315]	0.6402	0.8336	0.5635
MiB	[0.8110 0.8469 0.5546 0.6592 0.1881 0.6284]	[0.9104 0.9264 0.6363 0.7303 0.2222 0.8066]	0.7054	0.8574	0.6147
PLOP	[0.8110 0.8453 0.5811 0.5216 0.2038 0.5581]	[0.8724 0.9123 0.8186 0.5395 0.2211 0.8929]	0.7095	0.8296	0.5868
LSAW	[0.8386 0.8716 0.6148 0.6362 0.3343 0.6397]	[0.9150 0.9271 0.7619 0.6827 0.4722 0.8631]	0.7703	0.8670	**0.6559**
		3-1-1			
FT	[0.4659 0.0000 0.0000 0.0000 0.0000 0.6177]	[0.9557 0.0000 0.0000 0.0000 0.0000 0.8622]	0.3030	0.5228	0.1806
EWC	[0.4478 0.0000 0.0000 0.0000 0.0000 0.5887]	[0.9797 0.6174 0.0000 0.0000 0.0000 0.7069]	0.2811	0.5045	0.1727
RW	[0.4425 0.0000 0.0001 0.0001 0.0021 0.5582]	[0.9818 0.0000 0.0001 0.0001 0.0000 0.6504]	0.2732	0.4964	0.1671
PI	[0.4659 0.0000 0.0000 0.0000 0.0000 0.6187]	[0.9558 0.0000 0.0000 0.0000 0.0000 0.8631]	0.3031	0.5230	0.1807
ILT	[0.7395 0.8646 0.5501 0.6280 0.1974 0.5472]	[0.9412 0.9207 0.6096 0.6769 0.2729 0.6404]	0.6770	0.8302	0.5877
LWF	[0.7624 0.8651 0.5394 0.6231 0.2195 0.5937]	[0.9481 0.9137 0.6219 0.6666 0.2615 0.7037]	0.6714	0.8372	0.6005
LWF-MC	[0.7003 0.8539 0.5228 0.6040 0.0000 0.5051]	[0.9715 0.8912 0.5580 0.6406 0.4575 0.5556]	0.6028	0.8105	0.5310
MiB	[0.7945 0.8477 0.5473 0.6823 0.1199 0.5914]	[0.9255 0.9224 0.6302 0.7926 0.2464 0.6828]	0.7000	0.8497	0.5972
PLOP	[0.7938 0.8497 0.5772 0.5325 0.0038 0.5831]	[0.9146 0.9117 0.7910 0.5522 0.0040 0.8290]	0.6671	0.8339	0.5567
LSAW	[0.7957 0.8733 0.6172 0.6169 0.3398 0.6373]	[0.9384 0.9096 0.7441 0.6506 0.7341 0.7885]	0.7942	0.8554	**0.6467**
		2-2-1			
FT	[0.4640 0.0000 0.0000 0.0000 0.0000 0.6353]	[0.9716 0.0000 0.0000 0.0000 0.0000 0.6353]	0.2999	0.5228	0.1832
EWC	[0.4397 0.0000 0.0000 0.0000 0.0000 0.5182]	[0.9781 0.6174 0.0000 0.0000 0.0000 0.5182]	0.2671	0.4893	0.1597
RW	[0.4289 0.0000 0.0003 0.0009 0.0000 0.3514]	[0.9905 0.0000 0.0001 0.0001 0.0000 0.3845]	0.2294	0.4517	0.1302
PI	[0.4646 0.0000 0.0000 0.0000 0.0000 0.6345]	[0.9710 0.0000 0.0000 0.0000 0.0000 0.8313]	0.3004	0.5232	0.1832
ILT	[0.7049 0.8629 0.5633 0.5128 0.0018 0.4700]	[0.9305 0.9187 0.6382 0.5736 0.0018 0.5796]	0.6071	0.7976	0.5193
LWF	[0.7587 0.8682 0.5678 0.6068 0.1019 0.5694]	[0.9393 0.9154 0.6404 0.6636 0.1083 0.7002]	0.6589	0.8356	0.5788
LWF-MC	[0.5656 0.8524 0.5396 0.0836 0.0000 0.4152]	[0.9649 0.8966 0.5846 0.0838 0.0000 0.4316]	0.4936	0.7044	0.4094
MiB	[0.7408 0.8493 0.5608 0.6026 0.2565 0.2733]	[0.9325 0.9255 0.6725 0.8844 0.3148 0.2790]	0.6681	0.7969	0.5472
PLOP	[0.7363 0.8438 0.5075 0.5918 0.2801 0.5054]	[0.7710 0.8980 0.9029 0.7190 0.3976 0.8177]	0.7510	0.8036	0.5775
LSAW	[0.8098 0.8634 0.6334 0.6372 0.3800 0.6397]	[0.9167 0.9163 0.8364 0.6909 0.6198 0.8138]	0.7990	0.8602	**0.6606**
Joint	[0.8528 0.8742 0.6182 0.6717 0.2639 0.6635]	[0.9427 0.9233 0.7113 0.7397 0.3478 0.8344]	0.7499	0.8781	0.6522

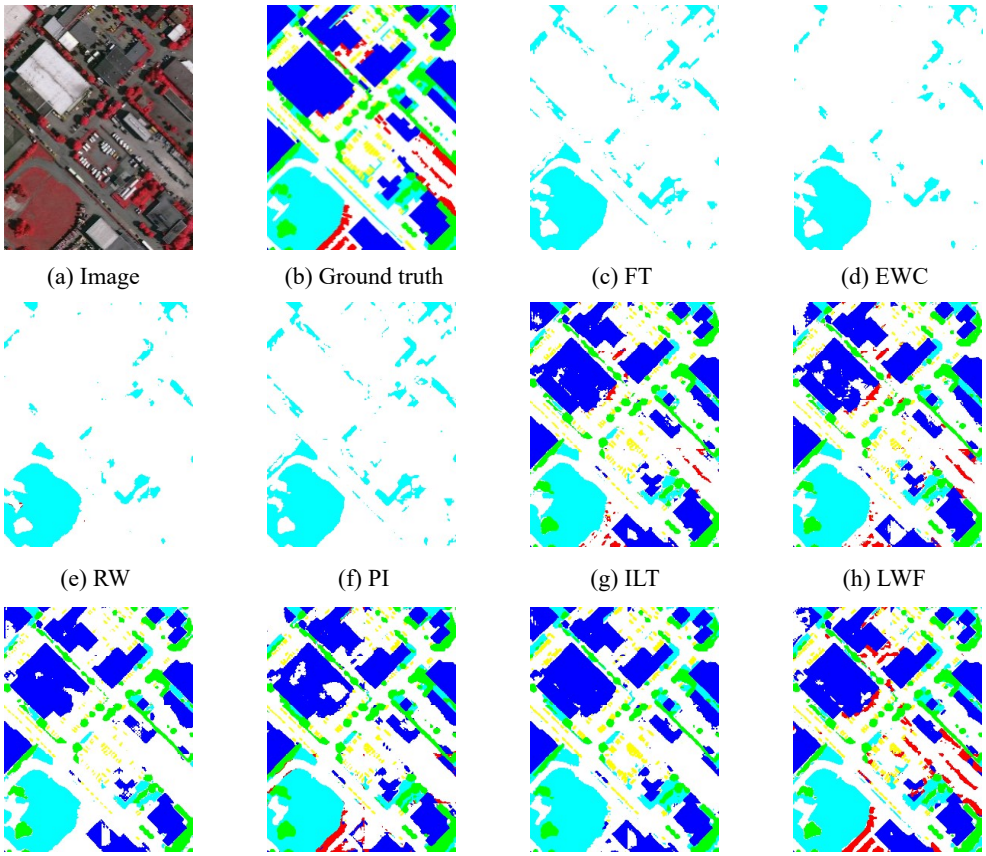
obtained by algorithms ILT, LWF, MiB, and PLOP. The results will still be significantly affected if the memory ability of an individual class is poor, even if the memory and learning abilities of other classes are good. Our algorithm achieves an mIoU value of 69.30%, which is 3.2% higher than the best-performing LWF algorithm.

C. Ablation experiment

LWF and our proposed method are selected as comparison algorithms and tested on the Vaihingen dataset. Table V represents the results of ablation experiments on the Vaihingen dataset. We use the LWF algorithm as a baseline for comparative experiments. A modified cross-entropy (CE) is first added to the baseline. By introducing pseudo-labels into the cross-entropy loss, which enhance the model's ability to distinguish between old and new classes, the model is more effective at learning new classes. Particularly, the mIoU of the fifth category has been improved significantly. Adding our self-designed cross-entropy to three different incremental learning tasks improved performance over the previous baseline. The most noticeable improvement is in the 2-2-1 task, and the mIoU value of the fifth category has increased from 9.77% to 34.68%. In addition, the overall mIoU value improved from 57.71% to 65.64%, an improvement of 13.7%. A distillation loss (KD) is added to the baseline. Due to the distillation loss, the model became more capable of remembering old classes, improving its performance as a whole. After adding KD to the 3-1-1 task, the overall mIoU value increased from 58.74% to 62.15%, an increase of 5.8%. After adding our cross-entropy and distillation loss at the same time, the results are the best in the 3-1-1 and 2-2-1 tasks. In the 3-2 task alone, the results are basically the same when the cross-entropy function is added. Therefore, the two losses are mutually beneficial. They can improve each other's outcomes, and work together on the task. Our proposed method has been demonstrated to be effective through ablation experiments.

V. CONCLUSION

In this paper, we propose an incremental learning algorithm based on adaptive weights and selection of merged labels for incremental learning tasks on remote sensing images. Cross-entropy and distillation functions have been redesigned. With the label strategy, the old classes are introduced and the problem of how to reasonably use incorrect samples predicted by the old model is solved. The weight of the remote sensing image class is dynamically adjusted based on the input image because of the imbalance of remote sensing image classes. The algorithm we propose solves the problem of catastrophic forgetting well, and it is capable of memorizing old classes as well as learning new ones. The results on three remote sensing datasets demonstrate the effectiveness of our method. In the future, we will continue to explore and mine information



(a) Image (b) Ground truth (c) FT (d) EWC

(e) RW (f) PI (g) ILT (h) LWF

(i) LWF-MC (j) MiB (k) PLOP (l) LSAW

Fig. 8: Visualization of continual semantic segmentation results in the 3-1-1 (3 steps) setting of the Vaihingen dataset. The picture's different columns of pictures indicate: (a) Image, (b) Ground truth, (c) FT, (d) EWC, (e) RW, (f) PI, (g) ILT, (h) LWF, (i) LWF-MC, (j)MiB, (k)PLOP, and (l) LSAW.

TABLE V: RESULTS OF ABLATION EXPERIMENTS ON THE VAIHINGEN DATASET

Method	Category IoU	Category Acc	mIoU
	background, building, car, vegetation, tree, clutter	background, building, car, vegetation, tree, clutter	
	3-2		
LWF	[0.8130 0.8626 0.6050 0.6338 0.1945 0.6067]	[0.8861 0.9259 0.7815 0.6845 0.2139 0.8600]	0.6193
LWF+CE	[0.8368 0.8713 0.6036 0.6390 0.3491 0.6393]	[0.9164 0.9288 0.7250 0.6859 0.5055 0.8567]	**0.6565**
LWF+KD	[0.7848 0.8473 0.5868 0.5921 0.1056 0.5771]	[0.8710 0.9075 0.6970 0.6284 0.1132 0.8739]	0.5823
LWF+CE+KD	[0.8386 0.8716 0.6148 0.6362 0.3343 0.6397]	[0.9150 0.9271 0.7619 0.6827 0.4722 0.8631]	0.6559
	3-1-1		
LWF	[0.7544 0.8544 0.5276 0.6220 0.1724 0.5936]	[0.9463 0.9046 0.5738 0.6617 0.2311 0.7108]	0.5874
LWF+CE	[0.7986 0.8311 0.5539 0.6192 0.2081 0.6377]	[0.9397 0.8651 0.6339 0.6505 0.7514 0.7967]	0.6081
LWF+KD	[0.7821 0.8700 0.6069 0.6392 0.2252 0.6053]	[0.9361 0.9258 0.7122 0.6836 0.3691 0.7339]	0.6215
LWF+CE+KD	[0.7957 0.8733 0.6172 0.6169 0.3398 0.6373]	[0.9384 0.9096 0.7441 0.6506 0.7341 0.7885]	**0.6467**
	2-2-1		
LWF	[0.7586 0.8671 0.5649 0.6050 0.0977 0.5689]	[0.9407 0.9128 0.6345 0.6609 0.1029 0.7014]	0.5771
LWF+CE	[0.8168 0.8745 0.6227 0.6412 0.3468 0.6363]	[0.9287 0.9218 0.7740 0.6988 0.4678 0.8076]	0.6564
LWF+KD	[0.7729 0.8659 0.6053 0.6078 0.2232 0.5700]	[0.9272 0.9212 0.7296 0.6707 0.2570 0.7160]	0.6075
LWF+CE+KD	[0.8098 0.8634 0.6334 0.6372 0.3800 0.6397]	[0.9167 0.9163 0.8364 0.6909 0.6198 0.8138]	**0.6606**

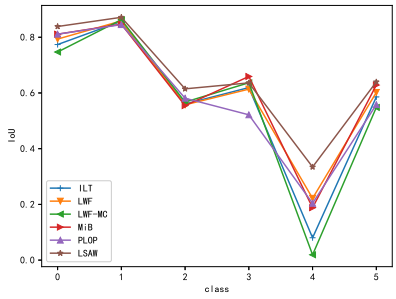


Fig. 9: Line chart of results on Potsdam datasets.

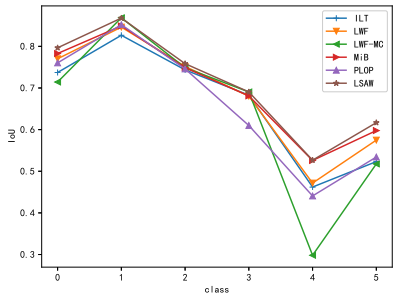


Fig. 10: Line chart of results on Vaihingen datasets.

from the previous model to improve the effect of incremental learning tasks on the segmentation of remote sensing images.

REFERENCES

[1] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3431–3440.

[2] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1925–1934.

[3] Z. Zhang, X. Zhang, C. Peng, X. Xue, and J. Sun, "Exfuse: Enhancing feature fusion for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 269–284.

[4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2017.

[5] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 801–818.

[6] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille, "Attention to scale: Scale-aware semantic image segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3640–3649.

[7] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of learning and motivation*. Elsevier, 1989, vol. 24, pp. 109–165.

[8] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2935–2947, 2017.

[9] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2001–2010.

[10] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Learning a unified classifier incrementally via rebalancing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 831–839.

[11] K. Shmelkov, C. Schmid, and K. Alahari, "Incremental learning of object detectors without catastrophic forgetting," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 3400–3409.

[12] J.-M. Perez-Rua, X. Zhu, T. M. Hospedales, and T. Xiang, "Incremental few-shot object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 13 846–13 855.

[13] K. Joseph, S. Khan, F. S. Khan, and V. N. Balasubramanian, "Towards open world object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 5830–5840.

[14] V. Badrinarayanan, A. Handa, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling," *arXiv preprint arXiv:1505.07293*, 2015.

[15] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1520–1528.

[16] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[17] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Lect. Notes Comput. Sci.* Springer, 2015, pp. 234–241.

[18] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[19] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2881–2890.

[20] Y. Gao, M. Zhang, W. Li, X. Song, X. Jiang, and Y. Ma, "Adversarial complementary learning for multisource remote sensing classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, pp. 1–13, 2023.

[21] J. Wang, W. Li, Y. Gao, M. Zhang, R. Tao, and Q. Du, "Hyperspectral and sar image classification via multiscale interactive fusion network," *IEEE Trans Neural Netw Learn Syst*, 2022.

[22] W. Li, J. Wang, Y. Gao, M. Zhang, R. Tao, and B. Zhang, "Graph-feature-enhanced selective assignment network for hyperspectral and multispectral data classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–14, 2022.

[23] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[24] C. Wu, L. Herranz, X. Liu, J. Van De Weijer, B. Raducanu *et al.*, "Memory replay gans: Learning to generate new categories without forgetting," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[25] A. Mallya and S. Lazebnik, "Packnet: Adding multiple tasks to a single network by iterative pruning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7765–7773.

[26] A. Mallya, D. Davis, and S. Lazebnik, "Piggyback: Adapting a single network to multiple tasks by learning to mask weights," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 67–82.

[27] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[28] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proc. Int. Conf. Mach. Learn.* PMLR, 2017, pp. 3987–3995.

[29] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proc. Natl. Acad. Sci.*, vol. 114, no. 13, pp. 3521–3526, 2017.

[30] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 532–547.

[31] U. Michieli and P. Zanuttigh, "Incremental learning techniques for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2019, pp. 0–0.

[32] F. Cermelli, M. Mancini, S. R. Bulo, E. Ricci, and B. Caputo, "Modeling the background for incremental learning in semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9233–9242.

[33] A. Douillard, Y. Chen, A. Dapogny, and M. Cord, "Plop: Learning without forgetting for continual semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 4040–4050.

[34] L. Bruzzone and D. F. Prieto, "An incremental-learning neural network for the classification of remote-sensing images," *Pattern Recong Lett*, vol. 20, no. 11-13, pp. 1241–1248, 1999.

[35] O. Tasar, Y. Tarabalka, and P. Alliez, "Incremental learning for semantic segmentation of large-scale remote sensing data," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 12, no. 9, pp. 3524–3537, 2019.

[36] X. Rong, X. Sun, W. Diao, P. Wang, Z. Yuan, and H. Wang, "Historical information-guided class-incremental semantic segmentation in remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–18, 2022.

[37] ISPRS, "Potsdam and vaihingen dataset," http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html.

[38] CCF, "Ccf dataset," https://www.datafountain.cn/competitions/270/datasets.

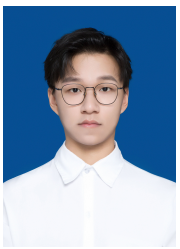
Bo Ren (M(M'18) received the B.S. degree in telecommunications engineering from Northwest University, Xian, China, in 2011, and the Ph.D. degree in circuits and systems from Xidian University, Xi'an, China, in 2017. Since 2022, he is an associate professor at the Key Laboratory of Intelligent Perception and Image Understanding of the Ministry of Education of China. In 2019, he is a visiting scholar in GIPSA-lab laboratory (Grenoble Images Parole Signal Automatique), Grenoble, France. His current research interests include data fusion, machine learning and incremental learning in remote sensing images.

Zhao Wang received the B.S. degree from Xidian University, Xi'an, China, in 2021, where he is pursuing the M.S. degree with the Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education, School of Artificial Intelligence.

His interests include deep learning, incremental learning and land cover classification.

Biao Hou (M(M'07) received the B.S. and M.S. degrees in mathematics from Northwest University, Xi'an, China, in 1996 and 1999, respectively, and the Ph.D. degree in circuits and systems from Xidian University, Xi'an, in 2003. Since 2003, he has been with the Key Laboratory of Intelligent Perception and Image Understanding of the Ministry of Education, School of Artificial Intelligence, Xidian University, where he is currently a Professor.

His research interests include deep learning and Synthetic Aperture Radar image interpretation.



Bo Liu received the B.S. degree from Hefei University of Technology, Hefei, China, in 2022, and he is pursuing the M.S. degree with the Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education, School of Artificial Intelligence in Xidian University, Xi'an, China.

His direction includes multi-modal remote sensing image fusion, contrast learning, and land cover classification.

Zitong Wu received the B.S. and M.S. degrees from Northwest University, Xi'an, China, in 2015 and 2018, respectively, where he is currently pursuing the Ph.D. degree with the Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education of China. His research interests include synthetic aperture radar image interpretation and deep learning in image processing.

Jocelyn Chanussot (M(M'04-SM'04-F'12) received the M.Sc. degree in electrical engineering from the Grenoble Institute of Technology (Grenoble INP), Grenoble, France, in 1995, and the Ph.D. degree from the University de Savoie, Annecy, France, in 1998. In 1999, he was with the Geography Imagery Perception Laboratory for the Delegation Generale de l'Armement (DGA - French National Defense Department). Since 1999, he has been with Grenoble INP, where he is currently a Professor of signal and image processing. He is conducting his research at the Grenoble Images Speech Signals and Automatics Laboratory (GIPSA-Lab). His research interests include image analysis, multicomponent image processing, nonlinear filtering, data fusion and machine learning in remote sensing. He has been a visiting scholar at Stanford University (USA), KTH (Sweden) and NUS (Singapore). Since 2013, he is an Adjunct Professor of the University of Iceland. In 2015-2017, he was a visiting professor at the University of California, Los Angeles (UCLA).

Dr. Chanussot is the founding President of IEEE Geoscience and Remote Sensing French chapter (2007-2010) which received the 2010 IEEE GRS-S Chapter Excellence Award. He was the co-recipient of the NORSIG 2006 Best Student Paper Award, the IEEE GRSS 2011 and 2015 Symposium Best Paper Award, the IEEE GRSS 2012 Transactions Prize Paper Award and the IEEE GRSS 2013 Highest Impact Paper Award. He is a member of the IEEE Geoscience and Remote Sensing Society AdCom. He was the General Chair of the first IEEE GRSS Workshop on Hyperspectral Image and Signal Processing, Evolution in Remote sensing (WHISPERS). He was the Chair (2009-2011) and Cochair of the GRS Data Fusion Technical Committee (2005-2008). He was a member of the Machine Learning for Signal Processing Technical Committee of the IEEE Signal Processing Society (2006-2008) and the Program Chair of the IEEE International Workshop on Machine Learning for Signal Processing, (2009). He is an Associate Editor for the IEEE Transactions on Geoscience and Remote Sensing and the IEEE Transactions on Image Processing. He was the Editor-in-Chief of the IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing (2011-2015). In 2013, he was a Guest Editor for the Proceedings of the IEEE and in 2014 a Guest Editor for the IEEE Signal Processing Magazine. He is a Fellow of the IEEE, a member of the Institut Universitaire de France (2012-2017) and a 2018 Highly Cited Researcher (Clarivate Analytics).

Licheng Jiao (M(SM'89-F'17) was born in Shaanxi, China, on October 15, 1959. He received the B.S. degree from Shanghai Jiaotong University, China, in 1982 and the M.S. and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 1984 and 1990, respectively. From 1984 to 1986, he was an Assistant Professor with the Civil Aviation Institute of China, Tianjin, China. During 1990 and 1991, he was a Postdoctoral Fellow with the Key Lab for Radar Signal Processing, Xidian University, Xi'an, China. Currently, he is the Director of the Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education of China.

His current research interests include signal and image processing, nonlinear circuits and systems theory, learning theory and algorithms, optimization problems, wavelet theory, machine learning.