Self-Supervised Pretraining of Transformers for Satellite Image Time Series Classification

Yuan Yuan, *Member, IEEE*, and Lei Lin

***Abstract*—Satellite image time series (SITS) classification is a major research topic in remote sensing and is relevant for a wide range of applications. Deep learning approaches have been commonly employed for the SITS classification and have provided state-of-the-art performance. However, deep learning methods suffer from overfitting when labeled data are scarce. To address this problem, we propose a novel self-supervised pretraining scheme to initialize a transformer-based network by utilizing large-scale unlabeled data. In detail, the model is asked to predict randomly contaminated observations given an entire time series of a pixel. The main idea of our proposal is to leverage the inherent temporal structure of satellite time series to learn general-purpose spectral-temporal representations related to land cover semantics. Once pretraining is completed, the pretrained network can be further adapted to various SITS classification tasks by fine-tuning all the model parameters on small-scale task-related labeled data. In this way, the general knowledge and representations about SITS can be transferred to a label-scarce task, thereby improving the generalization performance of the model as well as reducing the risk of overfitting. Comprehensive experiments have been carried out on three benchmark datasets over large study areas. Experimental results demonstrate the effectiveness of the proposed pretraining scheme, leading to substantial improvements in classification accuracy using transformer, 1-D convolutional neural network, and bidirectional long short-term memory network. The code and the pretrained model will be available at https://github.com/linlei1214/SITS-BERT upon publication.**

***Index Terms*—Bidirectional encoder representations from Transformers (BERT), classification, satellite image time series (SITS), self-supervised learning, transfer learning, unsupervised pretraining.**

I. Introduction

NOWADAYS, a huge volume of Earth observation (EO) data are being accumulated thanks to remarkable breakthroughs in the latest-generation of satellites. After the successive launch of the twin satellites (Sentinel-2A/B) of the Sentinel-2 mission, it is now possible to access large area, high-quality medium spatial resolution EO data with much higher frequency. Due to the high revisit time of Sentinel-2A/B satellites (five days from the two-satellite constellation), consecutively acquired images covering the same geographical area can be properly organized into satellite image time series (SITS) [1]. Such medium-resolution SITS data provide valuable information about the status and dynamics of the Earth surface, supporting analyses of the functional and structural characteristics of land covers as well as identifying change events [2], [3]. For this reason, SITS have been widely used in various application domains, such as ecology [4], agriculture [5], forest [6], land management [7], disaster monitoring [8] risk assessment [9], etc. In the meantime, new challenges are also being introduced by the question of how to extract valuable knowledge and meaningful information to exploit such abundant data.

SITS classification is one of the central problems in the SITS analysis, which is closely associated with many land applications, such as land cover mapping and change detection [10], vegetation species classification [11], and crop yields estimation [12]. SITS classification involves assigning every pixel in an image to a categorical label, primarily based on the pixel's spectral profile (trajectories of spectral variations over time) [3]. Machine learning algorithms provide effective tools to achieve automated SITS classification. Traditional algorithms such as support vector machine (SVM) and random forest (RF) classify SITS via handcrafted features, such as raw reflectances, spectral statistics, and phenological metrics [13], [14]. However, characterizing these features is rather difficult due to the strong interannual variations in seasonal patterns of the land surface reflectance, which can be caused by shifts in land cover or environmental conditions, management practices, and disturbance [15].

Lately, deep learning is gaining widespread popularity in the remote sensing community. Various deep neural network architectures have been introduced to advance the state of the art for many remote sensing classification problems [16]–[18]. The major advantage of deep learning methods is that they are capable of learning features from the input data optimized for a specific task without the need for manual feature engineering [1], [19]. Deep learning-based approaches are increasingly being applied to SITS classification. Among these methods, convolutional neural networks (CNNs) [20], [21] and recurrent neural networks [RNNs, including long short-term memory (LSTM) or gated recurrent units] [22], [23] have been most widely used to capture

Manuscript received September 15, 2020; revised October 22, 2020; accepted November 3, 2020. Date of publication November 9, 2020; date of current version January 6, 2021. This work was supported in part by the Research Project of Surveying Mapping and Geoinformation of Jiangsu Province under Project JSCHKY201905, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20170897, in part by the National Natural Science Foundation of China under Grant 41901356, and in part by the support of Environmental Protection Research Project of Jiangsu Province under Grant 2019010. *(Yuan Yuan and Lei Lin are co-first authors.) (Corresponding author: Yuan Yuan.)*

Yuan Yuan is with the School of Geographic and Biologic Information, Nanjing University of Posts and Telecommunications, Nanjing 210023, China (e-mail: yuanyuan@njupt.edu.cn).

Lei Lin is with the Beijing Qihoo Technology Company Ltd., Beijing 100015, China (e-mail: linlei1214@163.com).

This article has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/JSTARS.2020.3036602

temporal characteristics from spectral profiles. To exploit both spatial and temporal information of high-resolution SITS, hybrid architectures combining convolutional and recurrent layers [24], [25] and convolutional-recurrent neural networks [26] have been introduced and comprehensively compared [27]. Recently, a promising alternative to RNNs for sequence encoding, namely the transformer, has been proposed in the field of natural language processing (NLP) [28]. Transformers have been reported to have advantages over RNNs in terms of feature extraction and training efficiency. Garnot *et al.* first introduced transformers into SITS classification. They employed a transformer encoder to learn temporal correlations from a sequence of pixel-set embeddings [29]. A detailed comparison between transformers and other prevailing deep learning architectures for SITS classification can be found in [30], showing that transformers and RNNs outperform CNNs on processing raw satellite time series in terms of their architectures.

Nonetheless, the recent success of deep learning approaches has been primarily driven by deeper architectures and the availability of large amounts of labeled data. For instance, by training a deep CNN on millions of human-annotated photographs such as ImageNet, it can learn powerful visual features reusable in various image understanding tasks. When training samples are scarce, a common drawback of most deep learning architectures is that they are very prone to overfitting, since such models often have millions of parameters. However, there are rarely enough labeled data in remote sensing applications, since labeling is very time/labor-consuming and requires expertise. As a result, even though a huge supply of SITS data are available, the performance of deep learning approaches is restricted due to lack of labels.

To fully tap the potential of deep learning methods for the SITS classification, some studies have begun to explore how to mitigate the demand for labeled data. For example, Bazzi *et al.* proposed a transfer learning framework, which adapts a network trained on a label-rich dataset to a label-scarce task by means of knowledge distillation [31]. Ienco *et al.* developed a weakly supervised layerwise pretraining strategy to initialize parameters of RNNs by utilizing coarse granularity labels to provide supervision signals [32]. Although these approaches can partially alleviate the problem, their effectiveness is still limited by the quality and quantity of labeled samples. To the best of our knowledge, none of the existing work takes advantage of unlabeled data for SITS classification.

Self-supervised learning is a recently emerged unsupervised learning paradigm for tackling the challenge of insufficient labels [33]. In this paradigm, models are trained on unlabeled data by leveraging the structure present in the data itself to create supervised tasks (such tasks are often referred as "pretext tasks") [34]. The general pipeline of self-supervised learning is as follows: in the pretraining stage, models learn an initial set of representations by solving a predefined pretext task; in the fine-tuning stage, these representations are further adapted to a downstream task by continued training on task-related data in a supervised manner. In this way, deep learning models can transfer general knowledge learned from large-scale unlabeled data to a label-scarce downstream task, thereby enhancing the generalization capacity of deep learning models and preventing overfitting. Although a variety of self-supervised learning methods have emerged in recent years in computer vision [35]–[38] and NLP [39]–[41], we are not aware of any existing work that introduces the perspective of self-supervised learning into the SITS analysis to cope with the label scarcity problem.

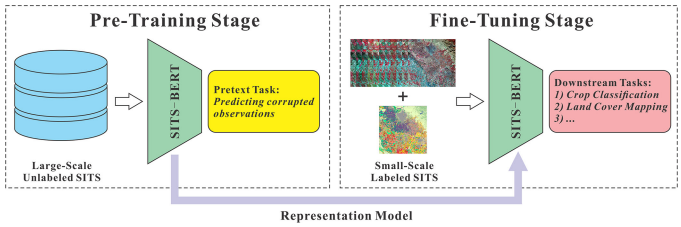

Fig. 1. Proposed pipeline for satellite image time series (SITS) classification: in the pretraining stage, a neural network (i.e., SITS-BERT, where BERT stands for bidirectional encoder representations from Transformers) is pretrained on massive unlabeled data to solve a pretext task; in the fine-tuning stage, the pretrained network serves as a representation model that is used in a specific classification task by fine-tuning all the parameters on task-related labeled samples.

In this article, we propose a novel self-supervised pretraining scheme for parameter initialization of deep neural networks on SITS classification tasks, which follows a standard unsupervised-pretraining/supervised-fine-tuning pipeline shown in Fig. 1. We design the following pretext task to train a transformer network: *the model is forced to predict corrupted observations with randomly added noise, given an entire annual time series of a pixel.* The central idea behind our pretext task is to leverage the inherent temporal structure of satellite time series to capture meaningful spectral-temporal characteristics from a large volume of SITS data, and that these characteristics are closely related to the natural changes at the Earth's surface. In this way, enormous amounts of background knowledge are accumulated in the network through pretraining, making the model "understand" what satellite time series should look like. Hence, it is more robust for neural networks to learn a mapping between spectral profiles and corresponding types given a small amount of labeled data, as the connection between temporal dynamics of satellite time series and land cover semantics is strong.

The key contributions of this article are threefold.

1) For the first time, we propose a self-supervised pretraining scheme to cope with the problem of insufficient labeled samples for the SITS analysis.
2) We introduce an end-to-end deep learning architecture as an effective alternative to convolutional and recurrent neural networks for the SITS classification.
3) We conduct comprehensive experiments on three large-scale datasets to validate the effectiveness of the proposed method.

The remainder of this article is organized as follows. Section II summarizes related work on self-supervised learning in remote sensing. Section III explains the motivation of the proposed method. Sections IV and V describe the proposed network architecture and pretraining scheme, respectively. Section VI provides the data information. Section VII reports the experimental results and discussions. Finally, Section VIII concludes this article.

II. Related Work

An effective pretext task is the key to successful self-supervised learning and should guarantee that the model learn meaningful representations instead of trivial solutions [34]. According to the strategies used to design pretext tasks, most of the available self-supervised learning approaches in remote sensing can be broadly classified into the following three categories.

Reconstruction-based methods include denoising autoencoders (DAEs) [42], [43] and deCNNs [44], [45]. These methods project data into a low-dimensional latent space and then reconstruct the input from the compressed features. However, exact input reconstruction is often not conducive to learning discriminant features.

Generation-based methods include variational autoencoders [46], [47] and generative adversarial networks [48], [49]. These methods aim to approximate the real data distribution and simultaneously learn a mapping from a latent space to the input space. However, the main goal of these methods is to generate more realistic samples rather than extracting meaningful features that facilitate downstream tasks.

Context-based methods leverage spatial, spectral, and temporal context similarity or correlation in the data to create supervision signals. These types of methods are generally heuristic and have no uniform theoretical patterns. For example, Wang *et al.* learned mapping functions between images and their transformed copies for the purpose of image registration [50]. Dong *et al.* sampled image patches from bitemporal images and then used the temporal-source of these patches as pseudolabels for change detection [51]. Vincenzi *et al.* attempted to predict visible bands of remote sensing images by other spectral bands [52]. At present, these kinds of methods are rarely applied to the remote sensing data.

In summary, the existing work mainly focuses on learning spatial and spectral features from single or bitemporal images, while very little effort has been devoted to satellite time-series analysis.

III. Motivation

BERT, which stands for bidirectional encoder representations from transformers [39], is undoubtedly the most remarkable self-supervised learning-based framework in the current NLP field. Following an unsupervised-pretraining/supervised-fine-tuning pipeline, BERT can learn general language representations that are reusable in various downstream tasks. In detail, BERT utilizes a "Masked Language Model" as its pretext task, in which some tokens in a sentence are randomly masked out and the model is forced to predict the missing tokens according to the context. This enables BERT to learn word correspondences from a plain text corpus. Since BERT was proposed, the concept of self-supervised pretraining has started dominating the state-of-the-art in NLP.

Similar to text, satellite time series also have strong temporal correlations. Likewise, the temporal dynamics of spectral profiles contain rich semantic information, which is closely associated with seasonal variations and plant phenology. However, while there are clear similarities between text and satellite time series, the insight of BERT has yet to be explored in the canon of existing work in the remote sensing community.

Inspired by BERT, we develop a context-based pretext task to capture meaningful spectral-temporal features from massive unlabeled SITS data. Specifically, the network is asked to recover contaminated observations by means of corresponding acquisition dates and clear observations. Our hypothesis is that noisy observations can be distinguished and reconstructed from dense satellite time series. The main idea to design this pretext task is based on the fact that noise caused by clouds and shadows is commonly found in optical satellite images. Human remote sensing experts can easily eliminate noise interference and distinguish different land cover types from a limited number of images. The absence of images or contaminated images does not necessarily result in a decrease in the interpretation accuracy because the missing information can be inferred from the remaining images. Intuitively, a good representation of SITS should be able to capture stable spectral-temporal patterns that are robust to noise.

IV. Model Architecture

A. Overall Network Architecture

In this article, we introduce a transformer-based network for the SITS classification. Since the model architecture is modified from BERT, we name it SITS-BERT.

We first introduce the symbols used in our description. Let $\mathrm{X} = \{\langle O_1, t_1\rangle, \ldots, \langle O_1, t_L\rangle\}$ be an annual time series of a pixel, where $O_i \in \mathcal{R}^D$ denotes a D-dimensional satellite observation vector whose elements correspond to each of the input spectral reflectances; t_i corresponds to the date O_i was captured, which is specified using day of year (DOY).

SITS-BERT's model architecture is comprised of two parts: an observation embedding layer and a standard transformer encoder (see Fig. 2). Specifically, all observation tuples $\langle O_1, t_1\rangle, \ldots, \langle O_1, t_L\rangle$ of a time series are first encoded into a sequence of observation embeddings, which are then fed into a multilayer bidirectional transformer network to produce their corresponding representations. The final observation representations can be aggregated into a single feature vector, which contains the global information of the entire time series and is used for classification. Here, SITS-BERT plays the role of a representation model. By adding an additional output layer, the whole network architecture of SITS-BERT can be further fine-tuned for a specific classification task.

B. Observation Embedding

The observation embedding layer projects an input tuple $\langle O_i, t_i\rangle$ into a higher-dimensional feature space. The reason for this is that the input dimension is tied with the hidden layer size of the transformer, and using larger embedding sizes gives better performance [41].

In this article, an observation embedding is a concatenation of two parts. Given an observation O_i, it is projected into a high-dimensional vector using a linear dense layer. In addition, the corresponding date t_i is also encoded into a vector of the same size using the positional encoding (PE) technique [28]. PE encodes the order information of a sequence with sine/cosine

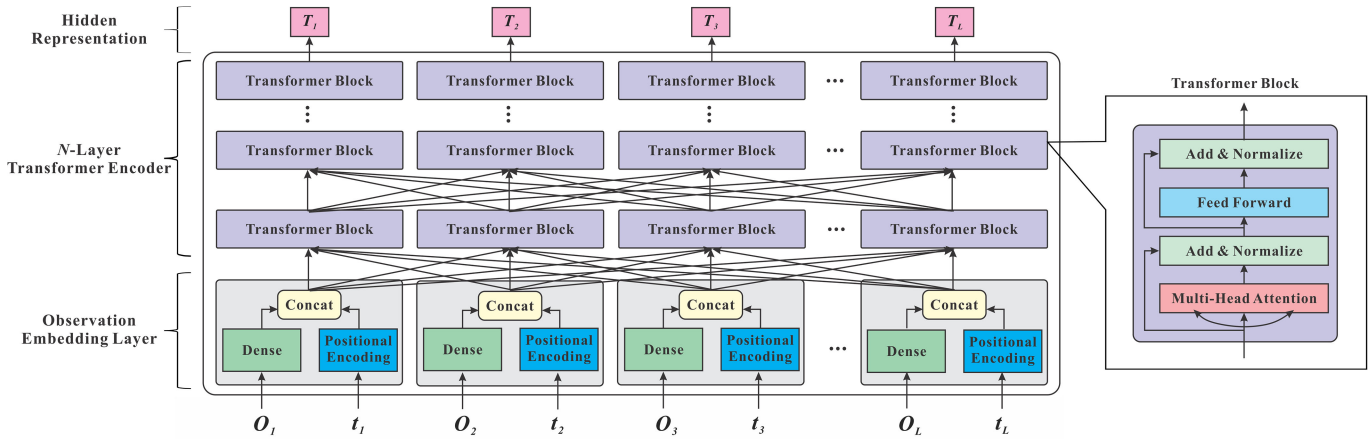

Fig. 2. Structure of the SITS-BERT architecture shown unfolded in time. Besides the output layer (not depicted in the figure), the same architecture is used in both the pretraining and fine-tuning stages. The input tuple $\langle O_i, t_i \rangle$ at each timestep is a pair of observation and the corresponding acquisition date (day of year, DOY). The final hidden representation is denoted as T_i. T_i encodes the information about the entire time series.

functions of different frequencies

$$\text{Embed}\,(O_i, t_i) = \text{Concat}\,(O_i W_e,\ \text{PE}(t_i)) \tag{1}$$

$$PE(t_i)_p = \begin{cases} \sin\left(t_i / 10000^{2k/d_m}\right), & \text{if } p = 2k \\ \cos\left(t_i / 10000^{2k/d_m}\right), & \text{if } p = 2k+1 \end{cases} \tag{2}$$

where $\text{Embed}(O_i, t_i)$ represents the observation embedding of $\langle O_i, t_i \rangle$; $d_m = 256$ is the embedding size, which is the same of the hidden layer size of transformer; Concat denotes a concatenation between two vectors; $W_e \in \mathcal{R}^{D \times \frac{d_m}{2}}$ is the weight matrix of the dense layer; $PE(t_i)_p$ represents the pth element of the PE vector. It should be noted that in BERT, the PE vector is directly added to a token embedding. However, we suggest concatenating the two parts in case the model cannot distinguish between observations and time. Experiments also indicate that the model converges faster using concatenation than summation.

The main purpose of utilizing time information (i.e., DOY rather than the order of an image in a sequence) is to make the model learn meaningful temporal variation patterns that are closely related to seasonal cycles and vegetation phenology. The benefits of this are twofold. On one hand, the model can overcome the limitations associated with different sampling dates and sequence lengths, making the model trainable and transferable across different years. On the other hand, the model is able to deal with irregular sampling problems caused by the presence of noise in the time series. Accordingly, no smoothing, gap filling, or compositing method is needed to reconstruct evenly spaced time series.

C. Transformer Encoder

Transformer was first introduced in NLP as an efficient alternative to RNNs [28], which has been introduced to some remote sensing tasks, such as hyperspectral image classification [53], image captioning [54], and SITS classification [29].

A transformer encoder is a stack of multiple transformer blocks. The first transformer block takes a collection of observation embeddings of a time series as its input and generates corresponding hidden representations. Then, these representations are passed to the next transformer block as its input to iteratively

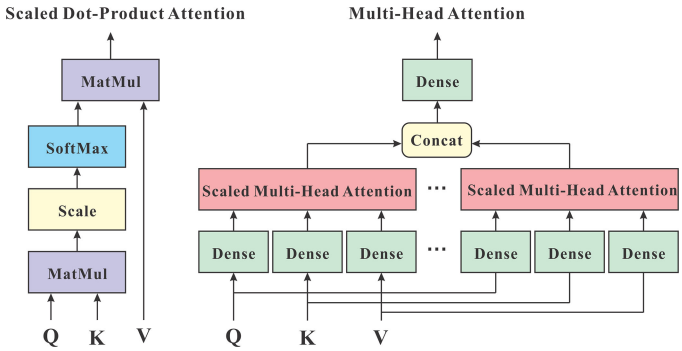

Fig. 3. Illustration of the scaled dot-product attention (left) and multihead attention (right). Q stands for queries' matrix; K stands for keys' matrix; V stands for values' matrix.

generate higher level representations. A single transformer block (depicted in Fig. 2) contains two major sublayers: a multihead attention layer and a positionwise fully connected feed-forward network (FFN). In addition, a residual connection [55] and layer normalization [56] are utilized for both sublayers. The two sublayers are described in detail below.

A multihead attention layer consists of H parallel scaled dot-product attention layers, each called a head (see Fig. 3) [28]. A scaled dot-product attention is a function that maps a query vector and a set of key-value pairs into an output vector

$$\text{Attention}\,(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3}$$

where Q, K, V represent the matrices stacked by multiple query, key, and value vectors as rows, respectively; d_k is the dimensionality of the query/key vectors. Multihead attention takes it a step further by first mapping Q, K, and V into different lower-dimensional feature subspaces via different linear dense layers, and then using the results to calculate attention. Finally, the outputs produced by H heads are concatenated and projected into a final hidden representation using another dense layer

$$\begin{aligned} \text{MultiHead}\,(Q, K, V) &= \text{Concat}\,(head_1, \ldots, head_H)\, W^o \\ \text{where head}_i &= \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right) \end{aligned} \tag{4}$$

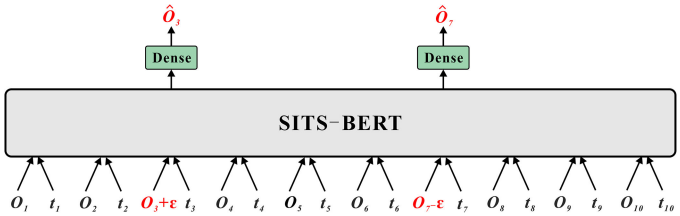

Fig. 4. Illustration of the proposed pretraining strategy. Some observations in a satellite time series are randomly selected and added with positive or negative noise, and then the model is trained to predict these contaminated observations using the rest of the observations. In the figure, the input observations O_3 and O_7 are added with noise $\pm\varepsilon$, where $\varepsilon \geq 0$, and the prediction loss at these two positions is used to optimize the model.

where W_i^Q, W_i^K, W_i^V are weight matrices of the inner dense layers of each head; W^o is the weight matrix of the top dense layer. In this article, W_i^Q, W_i^K, $W_i^V \in \mathcal{R}^{d_m \times d_v}$, $W^o \in \mathcal{R}^{Hd_v \times d_m}$, where we use $H = 8$ as the number of heads, and use $d_v = \frac{d_m}{H} = 32$ as the output dimensionality of each head.

In a transformer, a self-attention mechanism is used to learn the internal relationship of an input sequence [28]. In the self-attention mechanism, the query, key, and value vectors are identical. That is, attention is calculated between each position in a sequence and every other position (including itself). As a result, the hidden representation of each observation is a weighted sum of all positions in a time series, thereby capturing the global sequence information and highlights the part around the *i*th observation.

A positionwise FFN is then applied to the hidden state of each position independently and identically. FFN is made up of two linear transformations with a ReLU activation function in between

$$\text{FFN}\ (x_i) = \max\left(0, x_i W_1 + b_1\right) W_2 + b_2 \tag{5}$$

where x_i is the *i*th hidden state produced by the multihead attention layer, W_1, W_2 are weight matrices, and b_1, b_2 are bias terms of the inner and output dense layers, respectively. In this article, we use $d_{ff} = 4\, d_m = 1024$ as the dimensionality of the inner dense layer, i.e., $W_1 \in \mathcal{R}^{d_m \times d_{ff}}$, $b_1 \in \mathcal{R}^{d_{ff}}$, $W_2 \in \mathcal{R}^{d_{ff} \times d_m}$, $b_2 \in \mathcal{R}^{d_m}$.

V. Proposed Self-Supervised Pretraining Scheme

A. Pretraining SITS-BERT

In analogy to BERT, we train our model on a predesigned pretext task: some of the input observations are randomly chosen and added with noise, and then the model is forced to predict these contaminated observations (see Fig. 4). By solving this pretext task, the model can learn temporal correspondences between observations.

In the experiments, 15% of the observations in a time series are selected at random and added with noise. When an observation O_i is selected, there is a 50% chance that all elements of the vector is added a positive noise to simulate abnormal reflectance increases caused by clouds and snow/ice; and a 50% chance that all element of the vector is subtracted from a positive noise to simulate abnormal reflectance decreases caused by shadows. The noise is generated from a uniform distribution in the interval [0, 0.5]. In the meantime, the corresponding PE vector remains unchanged.

The model is then forced to predict the original values of these contaminated observations according to their acquisition dates and contextual information. In detail, the final hidden representations corresponding to the contaminated observations are fed into a linear dense layer for prediction. The mean-square error (MSE) between original observations and predictions is used as the optimization function for model training

$$\text{MSE}_{\text{Loss}} = \frac{\sum_{O_j \in \Psi} \|O_j - \hat{O}_j\|^2}{N} \tag{6}$$

where Ψ denotes the set of contaminated observations; $\hat{O}_j$ is the predicted value of O_j; and N is the number of contaminated observations. In this article, we pretrain SITS-BERT for 100 epochs with a batch size of 256 sequences. We utilize Adam optimizer with a learning rate of 1e$-$4, learning rate warmup over the first 30 epochs, and exponentially decay of the learning rate. Dropout is implemented on all transformer layers with a dropout rate of 0.1.

As a side benefit, the proposed pretext task overcomes the pretrain-fine-tune discrepancy suffered by the original BERT [15]. Specifically, BERT uses a special token [MASK] to tell the model which inputs are missing; this token does not exist in the downstream tasks. In contrast, our model does not introduce artificial tokens. Our assumption is that the model can learn how to distinguish noise from normal observations from the context without explicit annotations.

It is noteworthy that the proposed pretext task is different from DAEs [57], because our aim is to predict contaminated observations instead of reconstructing the entire time series.

B. Fine-Tuning SITS-BERT

The pretrained SITS-BERT can be easily adjusted to a specific SITS classification task by adding an additional output layer. In this article, we fine-tune SITS-BERT on the following two representative tasks, i.e., crop classification and land cover mapping. For both tasks, we simply feed the input–output pairs to the model and fine-tune all the parameters end-to-end on task-related data. Specifically, the input of the model is an annual time series, and the output is the corresponding category label.

A common method to address time series classification is to map time series into a feature space, such that the values of features should be close for pixels belonging to the same class. In general, there are two strategies for aggregating individual observation representations produced by SITS-BERT into a single sequence-level representation. These two approaches are 1) the use of a [CLS] token, and 2) the pooling method (see Fig. 5).

1) *SITS-BERT with a [CLS] token*: The first method is to insert a special token [CLS] (an abbreviation for "classification") at the front of every input time series and then use the output of [CLS] as a global representation of the whole time series [39]. In our implementation, we set [CLS] to be an all-ones vector with the same dimension as the input

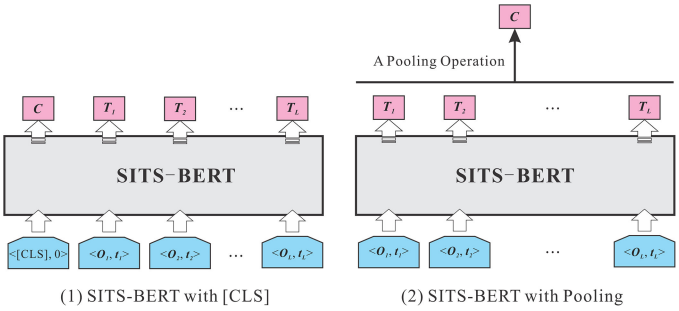

Fig. 5. Two strategies to generate a sequence-level representation (denoted as *C*) from individual observation representations of a time series: (1) SITS-BERT with a [CLS] token, (2) SITS-BERT with pooling.

observations, and set the corresponding time to be zero. The [CLS] token also needs to be added to the input time series at the pretraining stage.

2) *SITS-BERT with pooling:* The second method is to apply a pooling operation to the output vectors of BERT by either averaging the output (average pooling) or computing max-over-time of the output (max pooling) to aggregate them into a single fixed-size vector.

Once the global representation of a time series is obtained, it is then fed into a softmax layer to be classified. The cross-entropy loss is used for the model optimization in the fine-tuning stage

$$CrossEntropy_{Loss} = -\sum_{j=1}^{\mathrm{P}} y_j log\left(\hat{y}_j\right) \tag{7}$$

where $\tilde{y}_j$ is the probability score inferred by the softmax function for class j, and y_j is the ground truth, and P is the number of classes.

In this article, we fine-tune SITS-BERT on task-specific data in a supervised manner for 100 epochs. Adam optimizer is used with a learning rate of 2e-4 and a batch size of 128.

VI. Study Areas and Datasets

In this article, we applied the proposed method on Sentinel-2 image time series. The data were organized into four datasets: one was used for model pretraining, the other three (including two crop classification datasets and a land cover mapping dataset) were used for model evaluation.

We investigated the model performance on two kinds of SITS data. In the first type of data, the pretraining and fine-tuning data belonged to the same area, serving the purpose of verifying whether the pretrained model can extract informative and discriminant features from homologous unlabeled data. In the second type of data, the pretraining and fine-tuning data came from different areas, with the purpose of exploring whether the pretrained model can be transferred to unseen data.

A. Pretraining Dataset

We chose three Sentinel-2 tiles (T10SEJ, T10SFH, and T11SKA, each covering a region of about 100 × 100 km) located in the Central Valley, CA, USA to create a pretraining dataset. The location of the study area is depicted in Fig. 6. For each tile, all the available Sentinel-2 A/B images with $<10\%$ cloud coverage, taken between January 2018 and December 2019, were used. There were a total of 219 images to construct the pretraining dataset. Samples were collected from each tile at a regular sampling interval of 10 rows/columns. Sequences shorter than 10 were discarded. Finally, the pretraining dataset was composed of about 6.5 million sequences.

We chose California as the pretraining area for the following reasons. First, California has extremely diverse natural landscapes, representing most of the major biomes in North America, including grasslands, shrublands, deciduous forests, coniferous forests, tundra, mountains, deserts, rainforest, marine, estuarine, and freshwater habitats. The state's varied topography and climate have given rise to a remarkable plant diversity that cannot be found in any other state of the United States [58]. Moreover, the Central Valley where the study area is located is California's most productive agricultural region as well as one of the most productive worldwide. The agricultural systems in this study area are rather complicated, characterized by small parcels and a vast variety of crop types [59]. For all those reasons, this area can provide abundant and diverse training samples for learning general-purpose SITS representations. Finally, this area has a Mediterranean climate with no significant precipitation during the summer season. Therefore, the spectral trajectory of major crops (such as corn, cotton, rice, and soybeans) throughout the entire growth cycle can be characterized, making it feasible to inferring the corrupted part of a time series from the remainder.

B. Two Crop Classification Datasets

We applied the proposed method to two crop classification datasets.

The first dataset came from the pretraining area. A total of 45 images captured in 2019 from Sentinel-2 tile T11SKA in California (see Fig. 6 Tile ③) were used. Ten major crop types covering most of the study area were targeted in this study: corn, cotton, winter wheat, alfalfa, tomatoes, grapes, citrus, almonds, walnuts, and pistachios. Three nonagricultural categories were also included: developed, evergreen forest, and grass/pasture. It should be noted that the fine-tuning data were not used for pretraining.

The second dataset came from the area at the border between Missouri and Arkansas, USA (see Fig. 7), and was composed of 26 images acquired in 2019 from Sentinel-2 tile 15SYA. Ten agricultural and nonagricultural categories were considered in this study: corn, cotton, rice, soybeans, fallow/idle cropland, open water, developed, forest, grass/pasture, and woody wetlands.

For both datasets, the 2019 Cropland Data Layer (CDL) [60] and the corresponding CDL confidence layer [61] provided by the United States Department of Agriculture National Agricultural Statistics Service (USDA NASS) were used as reference data to collect samples. The original spatial resolution of CDL and confidence layer products was 30 m. They were upsampled to 10 m to be consistent with Sentinel-2 images. Then, we extracted samples from the whole scene at a regular interval of 10 rows/columns. To increase the accuracy of the samples collected

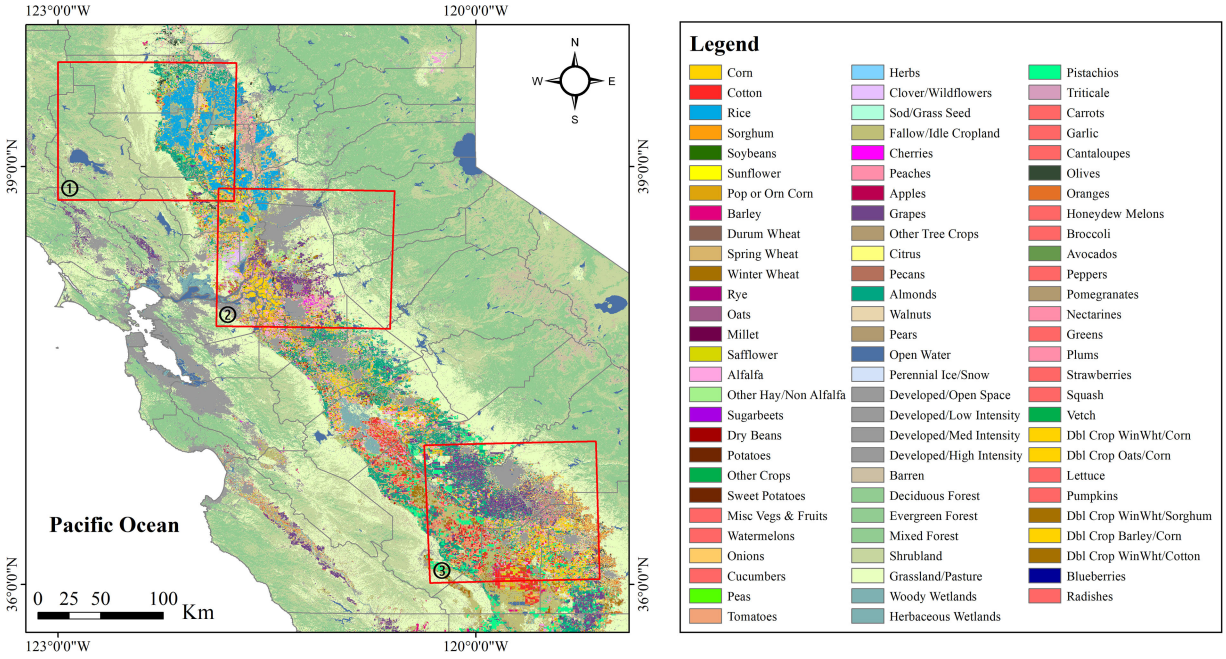

Fig. 6. 2019 cropland data layer (CDL) product of California, USA, which is colored in the same way as the CDL website (https://nassgeodata.gmu.edu/CropScape/). Three Sentinel tiles were used for model pre-training: ① T10SEJ, ② T10SFH, ③ T11SKA. The coverage of each Sentinel-2 tile is marked with a red rectangle.

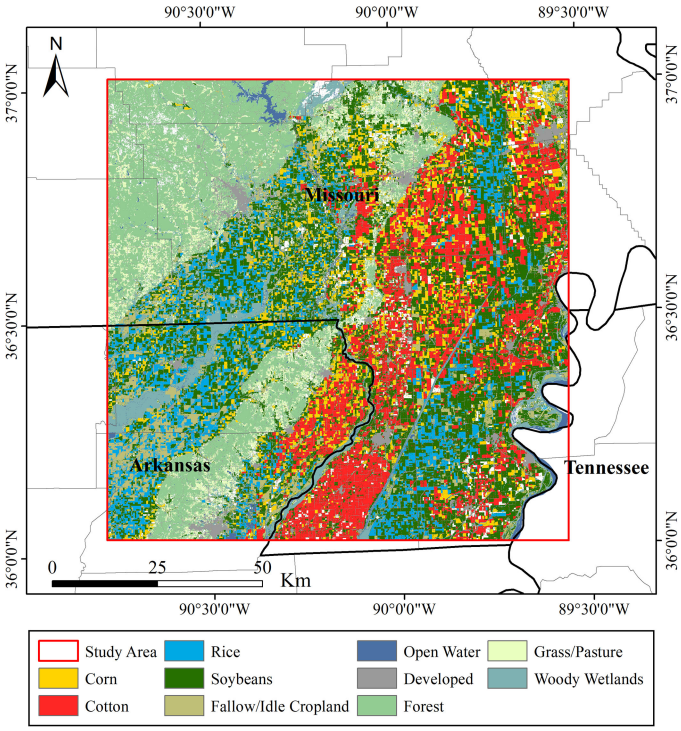

Fig. 7. Location of the second study area for crop classification, showing the 2019 CDL product.

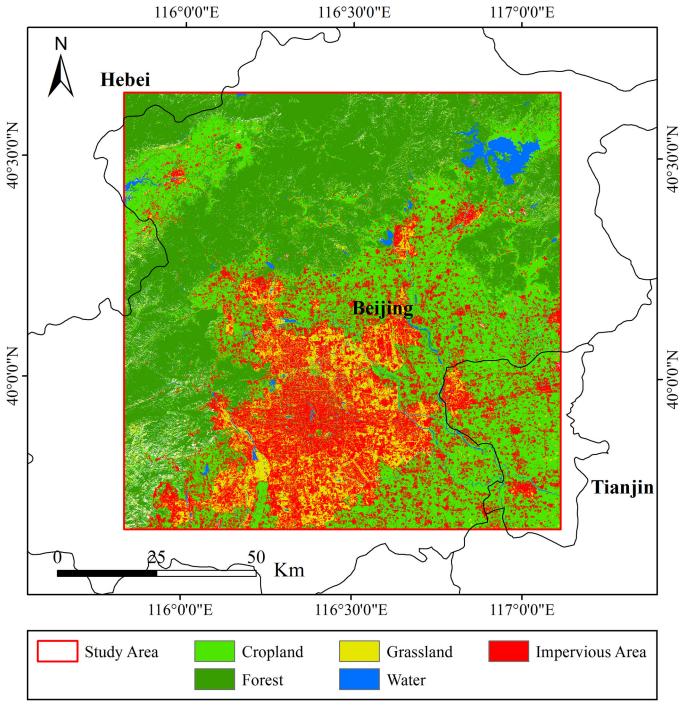

Fig. 8. Location of the study area for land cover mapping, showing the FROM_GLC10 product.

from the CDL, we employed the following two criteria. First, all the neighboring pixels along with the center pixel in a 7×7 window should belong to the same category, to avoid sampling from the boundary of land parcels. Second, the classification confidence of all neighboring pixels in the window should be higher than 80%.

C. Land Cover Mapping Dataset

We also applied the proposed method to a land cover mapping dataset, which was built using 20 images taken in Beijing, China, in 2017 from Sentinel-2 tile 50TMK (see Fig. 8). Beijing is characterized by a monsoonal humid continental climate with hot humid summers and cold dry winters. The natural vegetation

is deciduous broadleaf forest, and the dominant crop types are winter wheat, corn, and soybean. There is a large divergence between the study area and the pretraining area in terms of both geographical conditions and vegetation species. Therefore, this dataset provides a good example to explore whether the proposed pretraining scheme is effective when the target domain and the pretraining domain have distinct data distributions.

The reference data we used was a public global land cover product named FROM_GLC10 (v0.1.3), which was built on 10 m resolution Sentinel-2 data acquired in 2017 [62]. We adopted the same classification systems of FROM_GLC10. Five majority land cover types were considered: cropland, forest, grassland, water, and impervious area.

D. Data Collection and Preprocessing

All the Sentinel-2 images (Level-1C) we used were downloaded from the United States Geological Survey (USGS) EarthExplorer website and preprocessed to Bottom-Of-Atmosphere (BOA) reflectance Level-2A using the Sen2Cor plugin v2.8 and the Sentinel Application Platform (SNAP 7.0). The Multispectral Instrument (MSI) sensor provides 13 spectral bands, i.e., four bands at 10 m (Blue, Green, Red, NIR), six bands at 20 m (Vegetation Red Edge 1–3, Narrow NIR, SWIR 1–2), and three atmospheric bands at 60 m spatial resolution. With the exception of the atmospheric bands, all bands were used in this article. Bands at 20 m resolution were resampled to 10 m via nearest sampling. A scene classification map was generated for each image along with the Level-2A processing, which assigned pixels to clouds, cloud shadows, vegetation, soils/deserts, water, snow, etc. According to the scene classification map, low-quality observations belonging to clouds (including cirrus), cloud-shadows, and snow were discarded when extracting the annual time series of each pixel. Each selected sample should include at least three clear observations. The total number of samples in each evaluation dataset is displayed in Table I.

TABLE I
NUMBERS OF SAMPLES IN THE THREE FINE-TUNING DATASETS

Study Area	Class Name	Number of Samples	Total Number of Samples
California	Corn	4004	321188
	Cotton	24501	
	Winter Wheat	2380	
	Alfalfa	14823	
	Tomatoes	13392	
	Grapes	66792	
	Citrus	11941	
	Almonds	53961	
	Walnuts	14552	
	Pistachios	34690	
	Developed	3168	
	Evergreen Forest	75118	
	Grass/Pasture	1866	
Missouri	Corn	45329	388935
	Cotton	83737	
	Rice	56576	
	Soybeans	116317	
	Fallow/Idle Cropland	14061	
	Open Water	5781	
	Developed	1311	
	Forest	42047	
	Grass/Pasture	2146	
	Woody Wetlands	21630	
Beijing	Cropland	166327	510173
	Forest	286798	
	Grassland	22080	
	Water	13494	
	Impervious Area	21474	

VII. EXPERIMENTAL RESULTS

A. Evaluation Criteria and Methods

We compared the performance of different evaluated algorithms with the following metrics derived from the confusion matrix [18]:

1) *Overall Accuracy (OA):* This metric represents the proportion of correctly classified samples in all tested samples, and is computed by dividing the number of correctly classified samples by the total number of test samples.
2) *Kappa Coefficient:* This metric is considered to be a consistency measure between the classification result and the ground truth. Kappa coefficient is generally considered to be more robust than OA as it takes into account the possibility of agreement occurred solely by chance.
3) *Average Accuracy (AA):* This metric is an objective indicator of classification accuracy on unbalanced datasets. AA is calculated by dividing the sum of the accuracy for all classes by the number of classes, where the accuracy (i.e., the producer's accuracy) is the number of correctly classified samples divided by the number of total samples of each class.

To assess the effectiveness of the proposed network, we compared it with five methods that are widely employed in SITS classification. For traditional machine learning algorithms, we selected SVM and RF classifiers. SVM is widely considered as a powerful technique for classification tasks, while RF has some advantages such as short training time, easy parameterization, and high robustness to high-dimensional input features [13], [63]. Concerning deep learning models, we selected three approaches that performed well in previous SITS classification studies: CNN-1D (1-D CNN) [21], LSTM [22], [23], and bidirectional LSTM (Bi-LSTM) [64], [65]. These architectures have been proven to have advantages to capture temporal dependencies in sequential data.

1) *SVM:* We used a SVM classifier with radial basis function (RBF) kernel. The hyperplane parameters C and γ were optimized in a logarithmic grid from 10^{-2} to 10^{2}.
2) *RF:* We optimized a RF classifier using a varied number of trees in the forest. The optimal number of trees was selected in the set of $\{100,\ 200,\ 300,\ 400,\ 500\}$.
3) *CNN-1D:* A three-layer CNN-1D network was used in the experiment [21], which was formed by three convolutional layers (128 units), one dense layer (256 units), and one softmax layer. The kernel size of all convolutional layers was set to 5. Dropout was used after each convolutional layer with a dropout rate of 0.5.
4) *LSTM:* A three-layer stacked LSTM network was used, which was formed by three LSTM layer (256 units) and

one softmax layer. Dropout was used after each LSTM layer with a dropout rate of 0.5.

5) *Bi-LSTM:* A three-layer stacked Bi-LSTM network was used [66], which was formed by three Bi-LSTM layers (128 units) and one softmax layer. The outputs at each timestep from both forward and backward directions were concatenated and used the input of the next layer. Dropout was used after each layer with a dropout rate of 0.5.

SVM and RF handle a multivariate time series as a flat vector information, where the dimensionality of the input features should be identical. Hence, we padded the time series to the maximum length by filling in the missing observations with zeros. For deep learning models, we used the same observation embedding layer to encode the input observations. SVM and RF were implemented using the Python Scikit-Learn library, while all deep learning methods were implemented using the Python PyTorch library. Experiments were carried out on a workstation with an Intel Xeon Silver CPU 4210R 20-Core (2.40GHz) with 128 GB of RAM and a NVIDIA TITAN RTX 24G GPU.

In order to simulate real-world scenarios, we randomly selected 100 samples per category from each dataset to create the training and validation sets, respectively, and used all the residual samples for testing. The training set was used for training or fine-tuning pretrained models, and the validation set was used for choosing hyper-parameters (e.g., training epochs). It is worth mentioning that we did not follow the standard paradigm for method evaluation, which splits all data into training-validation-test set at certain rates. In our setting, the data distributions of training samples and test samples may be different. The main purpose of this is to verify the effectiveness of the proposed pretraining scheme under small-sample conditions.

B. Model Configuration

In this section, we evaluate the performance of three SITS-BERT variants to derive sequence-level representations. To verify whether the pretraining scheme is effective, we also trained each SITS-BERT variant from scratch using only labeled data (we refer to such models as "non-pre-trained" models). The three variants are described as follows.

1) *SITS-BERT using the [CLS] token*: In this case, the fine-tuned hidden representation of the [CLS] token was regarded as a representation of the entire input time series. We refer to such competitor as "SITS-BERT$_{\mathrm{CLS}}$."
2) *SITS-BERT using average pooling*: In this case, the hidden representations of all the observations in a time series were averaged and used for classification. We refer to such competitor as "SITS-BERT$_{\mathrm{AVG}}$."
3) *SITS-BERT using max pooling*: In this case, the maximum value for the hidden representations along the time dimension was computed and used for classification. We refer to such competitor as "SITS-BERT$_{\mathrm{MAX}}$."

The classification results are shown in Table II, and the confusion matrices are given in the Supplementary Materials. We observe that for all variants, the pretrained models remarkably outperform their randomly initialized versions (except for SITS-BERT$_{\mathrm{AVG}}$ in the third dataset). Specifically, the proposed pretraining scheme leads to an averaged accuracy increment of 2.97%, 1.88%, 2.58% for OA, 0.034, 0.023, 0.045 for Kappa coefficient, and 0.030, 0.029, 0.068 for AA on each dataset, respectively. The accuracy improvement on the latter two datasets indicates that unsupervised pretraining does produce positive transfer, that is, the representations learned from large-scale unlabeled data can be transferred to new context, thereby improves the performance of the model on label-scarce downstream tasks. We also find that SITS-BERT$_{\mathrm{MAX}}$ exceeds other variants in term of OA and Kappa coefficient, while SITS-BERT$_{\mathrm{CLS}}$ yields higher AA than SITS-BERT$_{\mathrm{MAX}}$ in the latter two datasets, showing that SITS-BERT$_{\mathrm{MAX}}$ is more advantageous for handling imbalanced classification problems.

TABLE II
CLASSIFICATION ACCURACY COMPARISON OF DIFFERENT SITS-BERT VARIANTS ON THE THREE DATASETS

Study Area	Method	OA	Kappa	AA
California	SITS-BERT$_{\mathrm{CLS}}$ (non-pre-trained)	91.16	0.897	0.922
	SITS-BERT$_{\mathrm{CLS}}$ (pre-trained)	93.82	0.927	0.942
	SITS-BERT$_{\mathrm{AVG}}$ (non-pre-trained)	90.56	0.889	0.917
	SITS-BERT$_{\mathrm{AVG}}$ (pre-trained)	93.35	0.922	0.947
	SITS-BERT$_{\mathrm{MAX}}$ (non-pre-trained)	90.76	0.892	0.912
	SITS-BERT$_{\mathrm{MAX}}$ (pre-trained)	**94.21**	**0.932**	**0.953**
Missouri	SITS-BERT$_{\mathrm{CLS}}$ (non-pre-trained)	96.20	0.953	0.939
	SITS-BERT$_{\mathrm{CLS}}$ (pre-trained)	98.57	0.982	**0.970**
	SITS-BERT$_{\mathrm{AVG}}$ (non-pre-trained)	96.68	0.959	0.938
	SITS-BERT$_{\mathrm{AVG}}$ (pre-trained)	98.43	0.981	0.969
	SITS-BERT$_{\mathrm{MAX}}$ (non-pre-trained)	97.23	0.966	0.939
	SITS-BERT$_{\mathrm{MAX}}$ (pre-trained)	**98.76**	**0.985**	0.964
Beijing	SITS-BERT$_{\mathrm{CLS}}$ (non-pre-trained)	85.81	0.762	0.838
	SITS-BERT$_{\mathrm{CLS}}$ (pre-trained)	91.32	**0.853**	**0.908**
	SITS-BERT$_{\mathrm{AVG}}$ (non-pre-trained)	89.14	0.813	0.837
	SITS-BERT$_{\mathrm{AVG}}$ (pre-trained)	88.44	0.806	0.896
	SITS-BERT$_{\mathrm{MAX}}$ (non-pre-trained)	88.41	0.802	0.821
	SITS-BERT$_{\mathrm{MAX}}$ (pre-trained)	**91.33**	**0.853**	0.896

C. Method Comparison

In this section, we compare the performance of SITS-BERT with other algorithms for the SITS classification. We used SITS-BERT$_{\mathrm{MAX}}$ as the default model configuration. The term "SITS-BERT" hereinafter refers to the pretrained SITS-BERT$_{\mathrm{MAX}}$. All other deep learning models were trained from scratch for 300 epochs with a learning rate of 2e$-$4 and a batch size of 128. The classification accuracy assessment for the three datasets are displayed in Table III, and the confusion matrices are given in the Supplementary Materials.

A major finding is that both CNN and RNN networks yield comparable or even worse results than traditional methods (SVM and RF), while the non-pre-trained SITS-BERT performs slightly better than traditional methods on the latter two datasets. Apart from the proposed model, RF performs best overall, ranking first on the latter two datasets. This suggests that for small datasets, sophisticated deep learning models have no substantial advantages over traditional machine learning methods in SITS classification tasks. This is likely due to the serious overfitting problem that occurs when training deep neural networks on insufficient labeled samples, even though transformer and

TABLE III
CLASSIFICATION ACCURACY COMPARISON OF DIFFERENT METHODS ON THE THREE DATASETS

Study Area	Method	OA	Kappa	AA
California	SVM	91.72	0.903	0.915
	RF	88.94	0.871	0.897
	CNN-1D	90.16	0.885	0.914
	LSTM	86.21	0.840	0.887
	Bi-LSTM	87.23	0.851	0.894
	SITS-BERT (non-pre-trained)	90.76	0.892	0.912
	SITS-BERT (pre-trained)	**94.21**	**0.932**	**0.953**
Missouri	SVM	95.82	0.949	0.943
	RF	96.38	0.956	0.946
	CNN-1D	95.90	0.950	0.933
	LSTM	90.22	0.881	0.905
	Bi-LSTM	92.14	0.904	0.913
	SITS-BERT (non-pre-trained)	97.23	0.966	0.939
	SITS-BERT (pre-trained)	**98.76**	**0.985**	**0.964**
Beijing	SVM	87.90	0.797	0.866
	RF	87.98	0.799	0.870
	CNN-1D	83.46	0.726	0.828
	LSTM	83.74	0.733	0.832
	Bi-LSTM	83.90	0.734	0.830
	SITS-BERT (non-pre-trained)	88.41	0.802	0.821
	SITS-BERT (pre-trained)	**91.33**	**0.852**	**0.896**

CNN-1D seem to be more robust to overfitting compared to LSTM-based networks.

In general, SITS-BERT outperforms other competitors remarkably in terms of all indicators. Specifically, compared with the RF baseline, SITS-BERT increases OA by 5.27%, 2.38%, 3.35%, Kappa coefficient by 0.061, 0.029, 0.053, and AA by 0.056, 0.018, 0.026 on each dataset, respectively. The results highlight that the risk of overfitting utilizing deep neural networks can be largely reduced by starting from a pretrained model. Moreover, according to the confusion matrices, SITS-BERT exhibits a notably lower rate of misclassifications for complicated and heterogeneous categories compared to other methods. For instance, in the first dataset, pistachios is easily confused with evergreen forest; the proposed method achieves a 20 points gain in the producer's accuracy with respect to the second best method (SVM). In the third dataset, while cropland has extremely high intraclass variability, the proposed method also achieves nine points of gain with respect to the second best method (RF).

To better visualize the classification results of different methods, we selected a 3000 $\times$ 3000 area of interest (ROI) in each study area for testing. The classification results are shown in Figs. 9–11. It can be seen that for crop classification tasks, the results obtained by SITS-BERT are more consistent with the reference data and the salt and pepper effect is negligible. For the land cover mapping task, apart from SITS-BERT, all deep learning methods exhibit a high confusion between cropland and grassland. The large misclassifications of the two categories (i.e., cropland and grassland) may come from errors in the reference data, which underlines the fact that the pretrained model exhibits relatively higher robustness to noisy labels.

D. Effect of the Pretraining Scheme on Other Models

The proposed pretraining scheme may be widely effective for a number of deep learning models. In this section, we validate its ability on CNN-1D and Bi-LSTM. For CNN-1D, we utilized zero-padding to keep the sequence length unchanged. Both models were pretrained on the same pretraining dataset and then fine-tuned on each target dataset. The same pretraining/fine-tuning configurations used for SITS-BERT were adopted. The accuracy assessment is shown in Table IV, and the confusion matrices are given in the Supplementary Materials.

TABLE IV
CLASSIFICATION ACCURACY OF PRETRAINED AND NON-PRE-TRAINED BI-LSTM NETWORKS ON THE THREE DATASETS

Study Area	Methods	OA	Kappa	AA
California	CNN-1D (non-pre-trained)	90.16	0.885	0.914
	CNN-1D (pre-trained)	92.41	0.911	0.938
	Bi-LSTM (non-pre-trained)	87.23	0.851	0.894
	Bi-LSTM (pre-trained)	**92.63**	**0.914**	**0.939**
Missouri	CNN-1D (non-pre-trained)	95.90	0.950	0.933
	CNN-1D (pre-trained)	97.22	0.966	**0.960**
	Bi-LSTM (non-pre-trained)	92.14	0.904	0.913
	Bi-LSTM (pre-trained)	**97.91**	**0.974**	**0.960**
Beijing	CNN-1D (non-pre-trained)	83.46	0.726	0.828
	CNN-1D (pre-trained)	87.63	0.793	0.869
	Bi-LSTM (non-pre-trained)	83.90	0.734	0.830
	Bi-LSTM (pre-trained)	**88.99**	**0.816**	**0.892**

We observe that for all datasets, the classification accuracy of pretrained models is improved from their non-pre-trained versions. In particular, the pretraining scheme brings an averaged increment of 2.58% OA and 5.42% OA to CNN-1D and Bi-LSTM, respectively. The results demonstrate that the proposed pretraining scheme also works for CNNs and RNNs for SITS classification.

In addition, we observe that the performance gain of Bi-LSTM is larger than that of CNN-1D on all datasets, indicating that the proposed pretraining scheme seems to be more effective for transformers and LSTM-based networks than for CNNs. This may be attributed to transformer and LSTM's ability in capturing long-term dependencies between observations. In detail, LSTM encodes the whole time series by receiving each observation once a timestep, and utilizes internal gates to control the update of the memory content. Transformer reads the entire time series at once and calculates dependencies between each observation and all other observations through the self-attention mechanism. In contrast, CNN can only capture observation dependencies within the width of its filters. In other words, it can only model contextual information within a local neighborhood. This character makes it inferior to transformers and LSTM-based networks for temporal feature extraction.

E. Influence of the Number of Labeled Samples

It is well known that the number of training samples can affect classification performance, but its influence on pretrained models is unknown. In this section, we fine-tuned the pretrained CNN-1D, Bi-LSTM, and SITS-BERT with different numbers of labeled samples, varied from 50 to 500 per category, and took RF (300 trees) as a baseline for comparison. The results are depicted in Fig. 12.

We observe that all the pretrained deep learning models significantly outperform RF and the non-pre-trained model, among

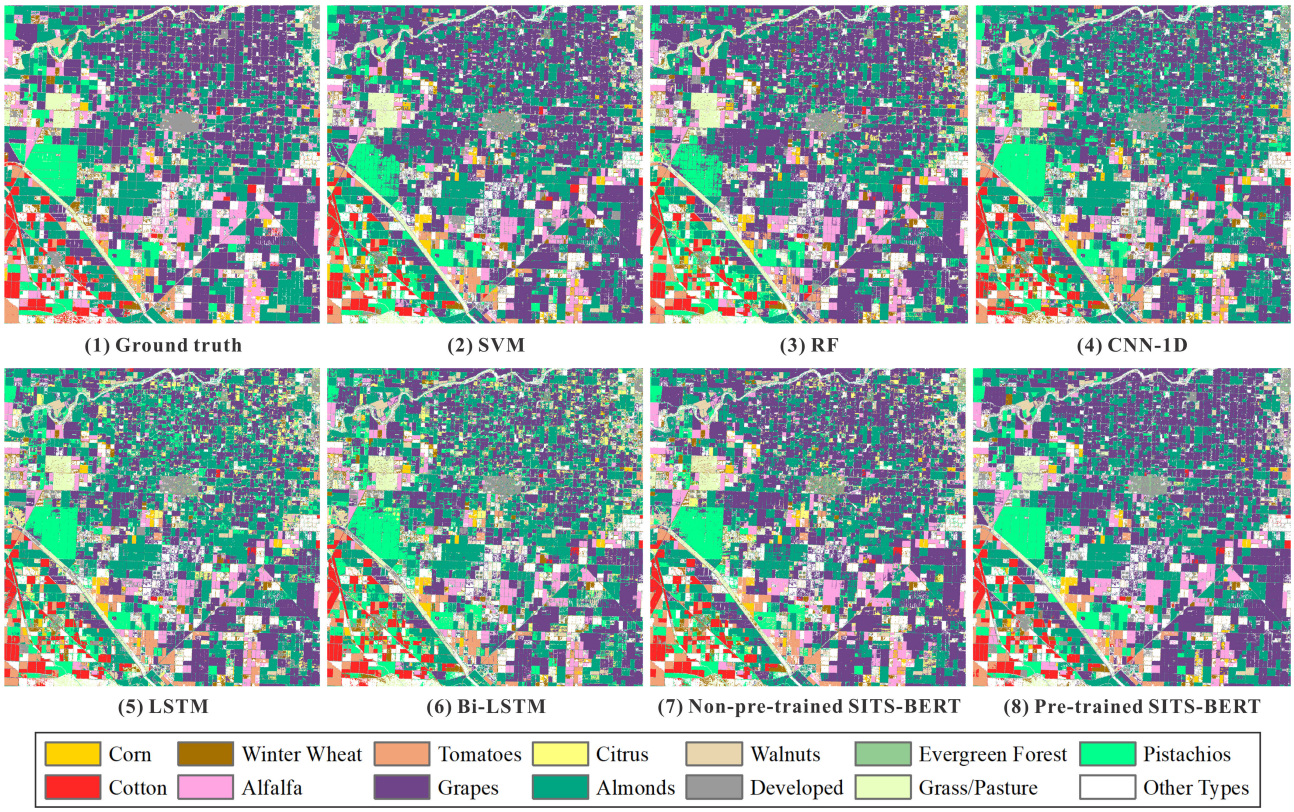

Fig. 9. Crop classification results of different methods in the first study area (California). (1) Ground truth. (2) SVM. (3) Random forest (RF). (4) Long-short term memory (LSTM). (5) Bidirectional LSTM (Bi-LSTM). (6) Non-pre-trained SITS-BERT. (7) SITS-BERT.

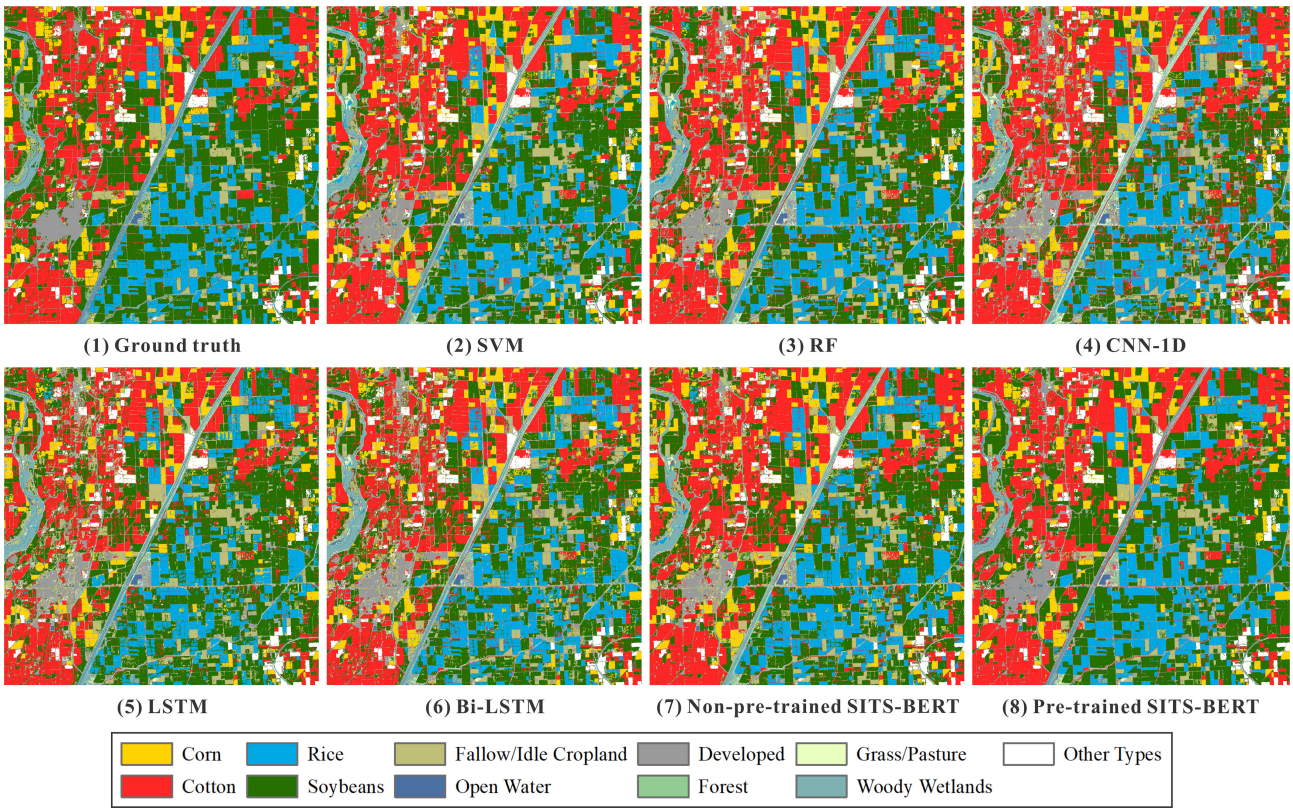

Fig. 10. Crop classification results of different methods in the second study area (Missouri). (1) Ground truth. (2) SVM. (3) RF. (4) LSTM. (5) Bi-LSTM. (6) Non-pre-trained SITS-BERT. (7) SITS-BERT.

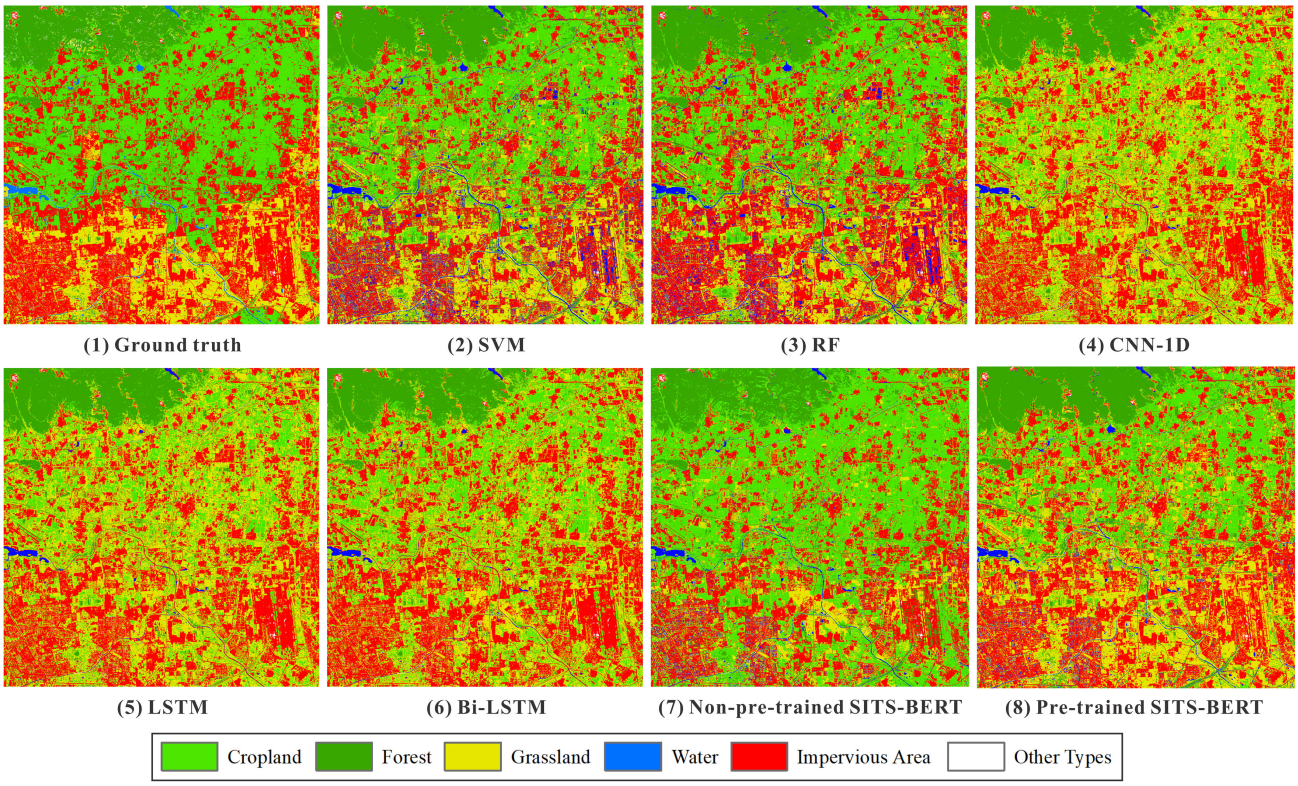

Fig. 11. Land cover classification results of different methods in the third study area (Beijing). (1) Ground truth. (2) SVM. (3) RF. (4) LSTM. (5) Bi-LSTM. (6) Non-pre-trained SITS-BERT. (7) SITS-BERT.

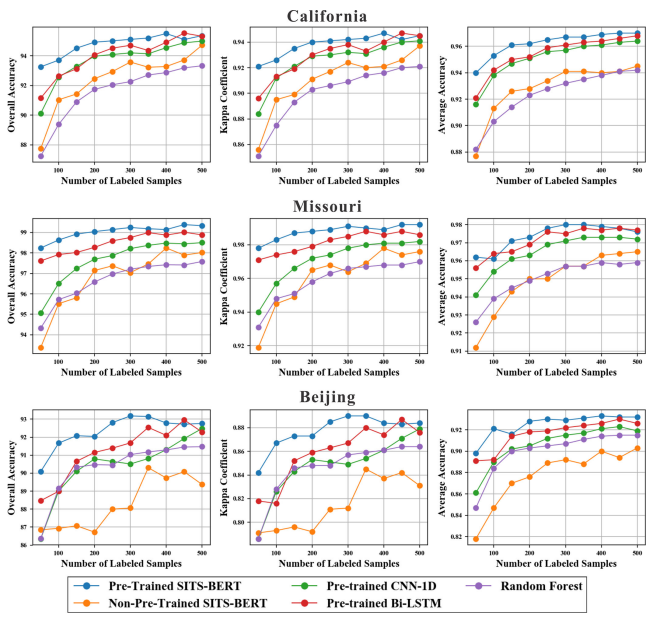

Fig. 12. Classification accuracy of SITS-BERT and the RF baseline using varied training samples. Evaluation metrics in each row from left to right: Overall accuracy (OA), Kappa coefficient, and average accuracy (AA).

which SITS-BERT surpasses all other competitors in all datasets. It is worth noting that the accuracy of SITS-BERT fine-tuned on only 50 labeled samples per category is comparable with the accuracy of RF trained on ten times more samples for the first and second datasets. Specifically, the best OA yielded by RF are 93.33% and 97.58% on the first and second dataset, respectively, while the worst OA yielded by SITS-BERT are 93.26% and 98.24%. Two main conclusions can be drawn: first, the experiments confirm and clarify the effectiveness of self-supervised pretraining using varied numbers of labeled samples; and second, the transformer architecture has advantages over CNNs and RNNs for satellite time series classification.

In contrast, although the non-pre-trained SITS-BERT performs slightly better than RF on the first two datasets, it performs much worse on the third dataset. In addition, it is not guaranteed that the accuracy of non-pre-trained SITS-BERT will improve as the number of training samples increases, indicating that randomly initialized deep neural networks give much more unstable results.

F. Computational Efficiency

In practice, SITS classification methods need to be applied to millions of satellite time series. Therefore, the computational efficiency of deep learning models should be taken into consideration. In this section, we compare the computation speeds of several deep learning models. The results are given in Table V.

According to the results, CNN-1D is the fastest, followed by LSTM-based models, and SITS-BERT is the slowest on a single GPU. Specifically, the computation speed of CNN-1D is about 19% faster than SITS-BERT. For Bi-LSTM, which has shown competitive performance in our previous experiments, it is also 5% faster than SITS-BERT. Fortunately, an advantage of transformer is that the computation can be parallelized

TABLE V
COMPUTATION SPEED (TIME SERIES PER SECOND) OF DIFFERENT METHODS

Method	Samples/second
CNN-1D	1820
LSTM	1680
Bi-LSTM	1598
SITS-BERT	1524

on multiple GPUs. Thanks to the self-attention mechanism in transformers, the attention weights for all positions in a sequence can be calculated simultaneously, whereas a recurrent layer should be computed successively after the output of the previous layer is obtained [28]. Therefore, SITS-BERT is better suited for massive parallel computations on modern machine learning acceleration hardware.

VIII. CONCLUSION

In this article, we proposed a simple but powerful self-supervised pretraining scheme for the SITS classification. The aim was to leverage large volumes of unlabeled SITS data to learn robust and transferable spectral-temporal features to facilitate label-scarce downstream tasks. We also introduced a transformer-based neural network architecture, named SITS-BERT, for the SITS classification.

The evaluation on three benchmark datasets have revealed that the proposed pretraining scheme is generally effective for several deep learning models, i.e., CNN, Bi-LSTM, and transformer. The experiments support our hypothesis that the representations learned from large-scale unlabeled data can produce positive transfer, thereby improves the performance and generalization ability of deep learning models on a specific SITS classification task. In addition, the results also demonstrate that the transformer network exceeds CNNs and RNNs for SITS classification.

We would like to emphasize that the pretraining data we used in this article are far from enough to fully exploit the potential of transformers. Nevertheless, the proposed approach is easily scaled to far larger datasets with very low costs, since no human-annotated labels are required during the pretraining stage.

Since the main purpose of this article was to validate the effectiveness of self-supervised learning for SITS, we put emphasis on the characteristics of spectral profiles of individual pixels while ignored the spatial correlation information between pixels. In the future, it will be of interest to incorporate spatial information into our framework to learn more discriminant features.

REFERENCES

[1] D. Ienco, R. Interdonato, R. Gaetano, and M. D. H.. Tong, "Combining Sentinel-1 and Sentinel-2 satellite image time series for land cover mapping via a multi-source deep learning architecture," *ISPRS J. Photogramm. Remote Sens.*, vol. 158, pp. 11–22, Dec. 2019.

[2] P. Jonsson and L. Eklundh, "Seasonality extraction by function fitting to time-series of satellite sensor data," *IEEE Trans. Geosci. Remote Sens.*, vol. 40, no. 8, pp. 1824–1832, Aug. 2002.

[3] C. Gómez, J. C. White, and M. A. Wulder, "Optical remotely sensed time series data for land cover classification: A review," *ISPRS J. Photogramm. Remote Sens.*, vol. 116, pp. 55–72, 2016.

[4] S. Rapinel, C. Mony, L. Lecoq, B. Clement, A. Thomas, and L. Hubert-Moy, "Evaluation of Sentinel-2 time-series for mapping floodplain grassland plant communities," *Remote Sens. Environ.*, vol. 223, pp. 115–129, Mar. 15 2019.

[5] M. J. Lambert, P. C. S. Traore, X. Blaes, P. Baret, and P. Defourny, "Estimating smallholder crops production at village level from Sentinel-2 time series in Mali's cotton belt," *Remote Sens. Environ.*, vol. 216, pp. 647–657, Oct. 2018.

[6] E. Grabska, P. Hostert, D. Pflugmacher, and K. Ostapowicz, "Forest stand species mapping using the Sentinel-2 time series," *Remote Sens.*, vol. 11, no. 10, May 2019, Art. no. 1197.

[7] J. Inglada, A. Vincent, M. Arias, B. Tardy, D. Morin, and I. Rodes, "Operational high resolution land cover map production at the country scale using satellite image time series," *Remote Sens.*, vol. 9, no. 1, Jan. 2017, Art. no. 95.

[8] W. T. Yang, Y. Q. Wang, S. Sun, Y. J. Wang, and C. Ma, "Using Sentinel-2 time series to detect slope movement before the Jinsha river landslide," *Landslides*, vol. 16, no. 7, pp. 1313–1324, Jul. 2019.

[9] M. Sudmanns, D. Tiede, L. Wendt, and A. Baraldi, "Automatic ex-post flood assessment using long time series of optical earth observation images," *GI_Forum.*, vol. 1, pp. 217–227, 2017.

[10] Y. Yuan *et al.*, "A new framework for modelling and monitoring the conversion of cultivated land to built-up land based on a hierarchical hidden semi-Markov model using satellite image time series," *Remote Sens.*, vol. 11, no. 2, Jan. 2019, Art. no. 210.

[11] M. Immitzer, F. Vuolo, and C. Atzberger, "First experience with Sentinel-2 data for crop and tree species classifications in central Europe," *Remote Sens.*, vol. 8, no. 3, Mar. 2016, Art. no. 166.

[12] E. Kamir, F. Waldner, and Z. Hochman, "Estimating wheat yields in Australia using climate records, satellite image time series and machine learning methods," *ISPRS J. Photogramm. Remote Sens.*, vol. 160, pp. 124–135, Feb. 2020.

[13] C. Pelletier, S. Valero, J. Inglada, N. Champion, and G. Dedieu, "Assessing the robustness of random forests to map land cover with high resolution satellite image time series over large areas," *Remote Sens. Environ.*, vol. 187, pp. 156–168, 2016.

[14] Q. Hu *et al.*, "How do temporal and spectral features matter in crop classification in Heilongjiang province, China?," *J. Integrative Agriculture*, vol. 16, no. 2, pp. 324–336, 2017.

[15] L. H. Nguyen, D. R. Joshi, D. E. Clay, and G. M. Henebry, "Characterizing land cover/land use from multiple years of landsat and MODIS time series: A novel approach using land surface phenology modeling and random forest classifier," *Remote Sens. Environ.*, vol. 238, Mar. 2020, Art. no. 111017.

[16] R. Hang, Q. Liu, D. Hong, and P. Ghamisi, "Cascaded recurrent neural networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 8, pp. 5384–5394, 2019.

[17] D. Hong, L. Gao, J. Yao, B. Zhang, A. Plaza, and J. Chanussot, "Graph convolutional networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, to be published, doi: 10.1109/TGRS.2020.3015157.

[18] D. Hong *et al.*, "More diverse means better: Multimodal deep learning meets remote-sensing imagery classification," *IEEE Trans. Geosci. Remote Sens.*, to be published, doi: 10.1109/TGRS.2020.3016820.

[19] L. H. Zhong, L. N. Hu, and H. Zhou, "Deep learning based multi-temporal crop classification," *Remote Sens. Environ.*, vol. 221, pp. 430–443, Feb. 2019.

[20] N. Kussul, M. Lavreniuk, S. Skakun, and A. Shelestov, "Deep learning classification of land cover and crop types using remote sensing data," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 5, pp. 778–782, May 2017.

[21] C. Pelletier, G. I. Webb, and F. Petitjean, "Temporal convolutional neural network for the classification of satellite image time series," *Remote Sens.*, vol. 11, no. 5, Mar. 2019, Art. no. 523.

[22] D. Ienco, R. Gaetano, C. Dupaquier, and P. Maurel, "Land cover classification via multitemporal spatial data by deep recurrent neural networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 10, pp. 1685–1689, Oct. 2017.

[23] M. Rußwurm and M. Korner, "Temporal vegetation modelling using long short-term memory networks for crop identification from medium-resolution multi-spectral satellite images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul 2017, pp. 11–19.

[24] P. Benedetti, D. Ienco, R. Gaetano, K. Ose, R. G. Pensa, and S. Dupuy, "M3Fusion: A deep learning architecture for multiscale multimodal multitemporal satellite data fusion," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 12, pp. 4939–4949, 2018.

[25] R. Interdonato, D. Ienco, R. Gaetano, and K. Ose, "DuPLO: A DUal view point deep learning architecture for time series classification," *ISPRS J. Photogramm. Remote Sens.*, vol. 149, pp. 91–104, Mar. 2019.

[26] M. Rußwurm and M. Körner, "Multi-temporal land cover classification with sequential recurrent encoders," *ISPRS Int. J. Geoinf.*, vol. 7, no. 4, 2018, Art. no. 129.
[27] V. S. Garnot, L. Landrieu, S. Giordano, and N. Chehata, "Time-space tradeoff in deep learning models for crop classification on satellite multi-spectral image time series," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2019, pp. 6247–6250.
[28] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Neural Inf. Process. Syst.*, Dec. 2017, pp. 5998–6008.
[29] V. S. F. Garnot, L. Landrieu, S. Giordano, and N. Chehata, "Satellite image time series classification with pixel-set encoders and temporal self-attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun 2020, pp. 12325–12334.
[30] M. Rußwurm and M. Körner, "Self-attention for raw optical satellite time series classification," 2019. [Online]. Available: https://arxiv.org/abs/1910.10536
[31] H. Bazzi, D. Ienco, N. Baghdadi, M. Zribi, and V. Demarez, "Distilling before refine: Spatio-temporal transfer learning for mapping irrigated areas using Sentinel-1 time series," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 11, pp. 1909–1913, Nov. 2020.
[32] D. Ienco, Y. J. Eudes Gbodjo, R. Interdonato, and R. Gaetano, "Attentive weakly supervised land cover mapping for object-based satellite image time series data with spatial interpretation," 2020. [Online]. Available: https://arxiv.org/abs/2004.14672
[33] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proc. IEEE Int. Conf. Mach. Learn.*, Jun. 2007, pp. 759–766.
[34] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published, doi: 10.1109/TPAMI.2020.2992393.
[35] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. Eur. Conf. Comput. Vis.*, Oct 2016, pp. 649–666.
[36] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2016, pp. 69–84.
[37] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 132–149.
[38] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020. [Online]. Available: https://arxiv.org/abs/2002.05709
[39] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018. [Online]. Available: https://arxiv.org/abs/1810.04805
[40] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized autoregressive pretraining for language understanding," in *Proc. Neural Inf. Process. Syst.*, Dec. 2019, pp. 5753–5763.
[41] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A lite BERT for self-supervised learning of language representations," 2019. [Online]. Available: https://arxiv.org/abs/1909.11942
[42] A. Elshamli, G. W. Taylor, A. Berg, and S. Areibi, "Domain adaptation using representation learning for the classification of remote sensing images," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.*, vol. 10, no. 9, pp. 4198–4209, Sep. 2017.
[43] X. Tang, X. Zhang, F. Liu, and L. Jiao, "Unsupervised deep feature learning for remote sensing image retrieval," *Remote Sens.*, vol. 10, no. 8, Aug. 2018, Art. no. 1243.
[44] Y. T. Tao, M. Z. Xu, F. Zhang, B. Du, and L. P. Zhang, "Unsupervised-restricted deconvolutional neural network for very high resolution remote-sensing image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 12, pp. 6805–6823, Dec. 2017.
[45] X. Q. Lu, X. T. Zheng, and Y. Yuan, "Remote sensing scene classification by unsupervised representation learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 9, pp. 5148–5157, Sep. 2017.
[46] X. Wang, K. Tan, Q. Du, Y. Chen, and P. J. Du, "CVA2E: A conditional variational autoencoder with an adversarial training process for hyperspectral imagery classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 8, pp. 5676–5692, Aug. 2020.
[47] X. Ma, Y. Lin, Z. Nie, and H. Ma, "Structural damage identification based on unsupervised feature-extraction via variational Auto-encoder," *Measurement*, vol. 160, Aug 2020, Art. no. 107811.
[48] D. Y. Lin, K. Fu, Y. Wang, G. L. Xu, and X. Sun, "MARTA GANs: Unsupervised representation learning for remote sensing image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 11, pp. 2092–2096, Nov. 2017.
[49] R. Hang, F. Zhou, Q. Liu, and P. Ghamisi, "Classification of hyperspectral images via multitask generative adversarial networks," *IEEE Trans. Geosci. Remote Sens.*, to be published.
[50] S. Wang, D. Quan, X. Liang, M. Ning, Y. Guo, and L. Jiao, "A deep learning framework for remote sensing image registration," *ISPRS J. Photogramm. Remote Sens.*, vol. 145, pp. 148–164, 2018.
[51] H. Dong, W. Ma, Y. Wu, J. Zhang, and L. Jiao, "Self-supervised representation learning for remote sensing image change detection based on temporal prediction," *Remote Sens.*, vol. 12, no. 11, Jun. 2020, Art. no. 1868.
[52] S. Vincenzi *et al.*, "The color out of space: Learning self-supervised representations for earth observation imagery," 2020. [Online]. Available: https://arxiv.org/abs/2006.12119
[53] J. He, L. Zhao, H. Yang, M. Zhang, and W. Li, "HSI-BERT: Hyperspectral image classification using the bidirectional encoder representation from transformers," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 1, pp. 165–178, Jan. 2020.
[54] X. Shen, B. Liu, Y. Zhou, J. Zhao, and M. Liu, "Remote sensing image captioning via variational autoencoder and reinforcement learning," *Knowl. Based Syst.*, vol. 203, 2020, Art. no. 105920.
[55] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," 2016. [Online]. Available: https://arxiv.org/abs/1603.05027
[56] J. Lei Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016. [Online]. Available: https://arxiv.org/abs/1607.06450
[57] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. IEEE Int. Conf. Mach. Learn.*, Jul. 2008, pp. 1096–1103.
[58] B. G. Baldwin, "Origins of plant diversity in the California floristic province," *Annu. Rev. Ecol. Evol. Syst.*, vol. 45, no. 1, pp. 347–369, 2014.
[59] L. Zhong, T. Hawkins, G. Biging, and P. Gong, "A phenology-based approach to map crop types in the San Joaquin Valley, California," *Int. J. Remote Sens.*, vol. 32, no. 22, pp. 7777–7804, Nov 2011.
[60] USDA National Agricultural Statistics Service Cropland Data Layer. 2019. Published crop-specific data layer. USDA-NASS, Washington, DC, USA. Accessed May 2020. [Online]. Available: https://nassgeodata.gmu.edu/CropScape/
[61] W. Liu, S. Gopal, and E. F. Wood, "Uncertainty and confidence in land cover classification using a hybrid classifier approach," *Photogramm. Eng. Remote Sens.*, vol. 70, no. 8, pp. 963–971, Aug. 2004.
[62] P. Gong *et al.*, "Stable classification with limited sample: Transferring a 30-m resolution sample set collected in 2015 to mapping 10-m resolution global land cover in 2017," *Sci. Bull.*, vol. 64, no. 6, pp. 370–373, 2019.
[63] A. Chakhar, D. Ortega-Terol, D. Hernandez-Lopez, R. Ballesteros, J. E. Ortega, and M. A. Moreno, "Assessing the accuracy of multiple classification algorithms for crop classification using Landsat-8 and Sentinel-2 data," *Remote Sens.*, vol. 12, no. 11, Jun. 2020, Art. no. 1735.
[64] L. Lin, "Satellite image time series classification and change detection based on recurrent neural network model," *Doctor Eng.* China, Univ. Chinese Acad. Sci., 2018.
[65] H. Wang, X. Zhao, X. Zhang, D. Wu, and X. Du, "Long time series land cover classification in China from 1982 to 2015 based on Bi-LSTM deep learning," *Remote Sens.*, vol. 11, no. 14, 2019, Art. no. 1639.
[66] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov 1997.

Yuan Yuan (Member, IEEE) received the B.S. degree in geography from Nanjing University, Nanjing, China, in 2011 and the Ph.D. degree in signal and information processing from Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing, China, in 2016.

She is currently working at Nanjing University of Posts and Telecommunications, Department of Surveying and Geoinformatics. Her research interests include remote sensing time series analysis, change detection, and deep learning.

Lei Lin received the B.S. degree in geographic information system from Shandong Agricultural University, Taian, China, in 2013, the Ph.D. degree in signal and information processing from Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing, China, in 2018.

He is currently working at Qihoo Technology Corporation. His research interests include time series analysis and natural language processing.