Reconstruct missing pixels of Landsat land surface temperature product using a CNN with partial convolution

Maosi Chen*[a], Benjamin H. Newell[b], Zhibin Sun[a], Chelsea A. Corr[a], Wei Gao[a,c]
[a] United States Department of Agriculture UV-B Monitoring and Research Program, Natural Resource Ecology Laboratory, Colorado State University, Fort Collins, CO 80521, USA; [b] Geospatial Centroid, Colorado State University, Fort Collins, CO 80523; [c] Department of Ecosystem Science and Sustainability, Colorado State University, Fort Collins, CO 80523, USA

ABASTRACT

U.S. Landsat Analysis Ready Data (ARD) recently included the Land Surface Temperature (LST) product, which contains widespread and irregularly-shaped missing pixels due to cloud contamination or incomplete satellite coverage. Many analyses rely on complete LST images therefore techniques that accurately fill data gaps are needed. Here, the development of a partial-convolution based model with the U-Net like architecture to reconstruct the missing pixels in the ARD LST images is discussed. The original partial convolution layer is modified to consider both the convolution kernel weights and the number of valid pixels in the calculation of the mask correction ratio. In addition, the new partial merge layer is developed to merge feature maps according to their masks. Pixel reconstruction using this model was conducted using Landsat 8 ARD LST images in Colorado between 2014 and 2018. Complete LST patches (64x64) for two identical scenes acquired at different dates (up to 48 days apart) were randomly paired with ARD cloud masks to generate the model inputs. The model was trained for 10 epochs and the validation results show that the average RMSE values for a restored LST image in the unmasked, masked, and whole region are 0.29K, 1.00K, and 0.62K, respectively. In general, the model is capable of capturing the high-level semantics from the inputs and bridging the difference in acquisition dates for gap filling. The transition between the masked and unmasked regions (including the edge area of the image) in restored images is smooth and reflects realistic features (e.g., LST gradients). For large masked areas, the reference provides semantics at both low and high levels.

Keywords: partial convolution layer; partial merge layer; satellite image inpainting; Landsat Analysis Ready Data (ARD); Land Surface Temperature (LST)

1. INTRODUCTION

Land Surface Temperature (LST) is an important environmental variable necessary to monitor Earth's surface energy and water balance related landscape processes and responses [1]. For example, LST is the key variable in evapotranspiration retrievals, a parameter that has wide environmental monitoring applications including water resources management, precision agriculture, vegetation analysis, and climate dynamics studies [2]. LST is also an important variable for urban heat island and land-cover change studies [3] [4]and may be used to validate and improve the global meteorological model prediction [5].

Satellite-based thermal infrared observations are ideal for deriving spatial distribution of LST at various (e.g., regional, and global) scales [5] [6]. For example, Landsat, Moderate Resolution Imaging Spectroradiometer (MODIS), and Geostationary Operation Environmental Satellite (GOES) all carry thermal infrared bands suitable for LST retrievals [7].

Retrieved LST maps inherit missing pixel issues due to defective satellite sensors (e.g., Landsat ETM+ SLC failure) and/or cloud contamination (i.e., thick clouds and cloud shadows), reducing the usability of the LST maps and complicating image interpretation [8]. The global average cloud cover is approximately 35% with high regional variability [8], suggesting a large amount of LST information is missing and needs to be reconstructed. Based on complementary information available for satellite missing pixel reconstruction, Shen et al. 2015 [8] identified four categories of traditional

* maosi.chen@colostate.edu; +1-970-491-3604

methods that can be used to reconstruct missing pixels in satellite images: (1) use unobscured spatial information by assuming that the statistical or geometrical structures are transferable (e.g., interpolation [9], propagated diffusion [10], variation regularization [11], and texture synthesis [12]); (2) use the spectral dependency information (e.g., [13]); (3) use the temporal dependency information (e.g., temporal replacement [14], temporal filtering [15], and dictionary learning [16]); and (4) use information from the combination of the other three categories (e.g. [17]).

Convolutional Neural Networks (CNNs) are capable of capturing high-level features (semantics of visual structures) of input images by optimizing spatially-shared trainable parameters with large datasets. CNNs have achieved better performance than traditional approaches in many computer visual tasks such as image classification [18] [19], image segmentation [20], image denoising [21].

CNNs have also been applied for image inpainting (filling gaps in an image, e.g., [22]), a technique necessary for satellite missing pixel reconstruction.

When only the damaged image is available, CNNs are usually capable of finding high-level semantics and using them to fill missing parts if the holes are small. For example, denoising auto-encoders [23] or blind inpainting [24] are capable of reconstructing input images from local corruptions. However, if the holes are large (e.g., missing of entire objects) or if scenes are complex, it is harder to extract sufficient high-level context from the damaged image alone for recovering the image [25]. In addition, the initialization values of large holes can introduce various types of visual artifacts that necessitate expansive post-processing [26].

One solution is to assume the dataset (e.g., CelebA [27]) is composed of images with similar high-level semantics and borrow the shared features from complete images within the dataset to recover the damaged image. For example, Pathak et al. 2016 [22] proposed the context encoder to fill larger missing area of an image. The context encoder is capable of capturing the semantics of visual structures and generating sharp results by considering both the per-pixel reconstruction loss and the adversarial loss. To utilize the shared high-level semantics both during training and inference, Yeh et al. 2017 [25] trained a Generative Adversarial Networks (GANs, e.g. [28]) with uncorrupted image dataset, applied the trained generator on the corrupted image, searched for the closest encoding features from the entire image dataset, and generated the missing content using the trained generator. Yu et al. 2018 [29] used the contextual attention layers as part of the generative network to refine the inpainting results.

Another solution is to find image(s) of the same object taken at another time. This is nearly impossible for natural images (e.g., ImageNet [30]), however, easy to obtain for satellite images. Unlike natural images, satellite images are usually acquired periodically over the identical scene (e.g., Landsat 8 has a revisit frequency of 16 days). There have been CNNs using such reference images to provide high-quality semantics for missing pixel reconstruction or super resolution on satellite images (e.g., [31] [32]).

In this study, we use the reference image to provide high-level semantics. To reduce the land-use-change induced mismatch, we limit the difference of acquisition dates between the reference (i.e., complete) and the target (i.e., damaged) images to be less than or equal to 48 days. We adopt the U-Net architecture [20] as the base model to fill the missing pixels in the Landsat LST images. The typical convolution in the encoders are replaced with the partial convolution and the corresponding mask-update steps as proposed by Liu et al. 2018 [26] (which are independent of the initialization values in the holes). The partial convolution results from the reference and target images are combined into a complete image as the U-Net skip link at each encoder level by the proposed partial merge layer.

In summary, the key contribution of this work is the development of missing pixel reconstruction methodology for satellite images (i.e., Landsat LST). Key features of this methodology include:

(1) Consideration of both the convolution kernel weights and the number of valid pixels in the partial convolution layer to calculate the correction ratio instead of only accounting for the number of valid pixels in the local sliding window.

(2) Utilization of the partial merge layer to merge spatially-matched incomplete and complete images through an input-weighted convolution. The input-weights are dynamically assigned by the relative pointwise (across-feature) importance. The relative importance of a feature is determined based on its mask value and the total number of valid features at the pixel. Similar to the partial convolution layer, the convolution results are corrected by considering the convolution kernel weights, the number of valid pixels, and the relative feature importance. The merge layer is the special case of the partial merge layer where inputs are complete other than in the padded area.

2. METHODOLOGY AND DATA

2.1 Detailed layer description

2.1.1 Partial Convolution Layer (original and modified)

The original Partial Convolution layer (*pconv*) developed by Liu et al. 2018 [26] extends the classical 2D convolution with the mask ratio correction and the mask updating step as expressed in the following equations:

$$x_j' = \begin{cases} \text{sum}\left(\mathbf{W}_j \odot \left(\mathbf{X} \odot \mathbf{M}\right)\right) \cdot ratio_j^{pconv} + b_j & , \quad \text{sum}(\mathbf{M}) > 0 \\ 0 & , \quad otherwise \end{cases}, \tag{1}$$

$$m_j' = m' = \begin{cases} 1, & \text{sum}(\mathbf{M}) > 0 \\ 0, & otherwise \end{cases}. \tag{2}$$

Where,

$$ratio_j^{pconv} = ratio = \frac{\text{sum}\left(\mathbf{1}\right)}{\text{sum}\left(\mathbf{M}\right)}, \tag{3}$$

X (H,W,F_{in}) is the input image(s) (with missing values) for the current convolution (sliding) window. H and W represent the height and width of the convolution window. F_{in} denotes the number of feature layers in **X**. **M** is the corresponding binary mask(s) (1: good pixel; 0: missing pixel). $\mathbf{W}_j$ (H,W,F_{in}) are the weights of the convolution kernel for the j^{th} output image and *b* is the corresponding bias. The symbol $\odot$ indicates the element-wise multiplication. The term $ratio_j^{pconv}$ (Eq. (3)) represents the mask correction ratio (>=1) that serves to scale the convolution results to those as if there are no missing values (or pixels) in **X** by accounting for the number of unmasked pixels. The matrix **1** has the same shape (H,W,F_{in}) as **M** but all of its elements have the value 1. x_j' is the j^{th} output feature of the convolution window (scalar). To have a bounded and meaningful x_j', the sum(**M**) must be greater than 0; otherwise, a fill value 0 for x_j' will be set. Similarly, the output mask is generated with Eq. (2). Both **X** and **M** are zero-padded to allow the convolution operation around the image edge. The new images populated by x_j' and m_j' will have the matching shape for the potential subsequent partial convolution layer.

Since there is missing information in the **X**, it is expected that the mask correction ratio may not perfectly restore the convolution results in all cases. $\mathbf{W}_j$, on the other hand, can be considered as a complete matrix if trained properly. Since **X** and $\mathbf{W}_j$ both contribute to the partial convolution results, we modified the original mask correction ratio with the absolute-weighted mask correction ratio to better utilize the known information about $\mathbf{W}_j$ in restoring the convolution results. The modified $ratio_j^{pconv}$ is expressed in Eq. (4):

$$ratio_j^{pconv} = \frac{\text{sum}\left(\left|\mathbf{W}_j\right| \odot \mathbf{1}\right)}{\text{sum}\left(\left|\mathbf{W}_j\right| \odot \mathbf{M}\right)} \tag{4}$$

Where, the symbol |.| indicates getting the absolute values of the input matrix.

For example, if we let $\mathbf{W}_j$ be a 3x3x1 kernel with values [[[1],[2],[1]], [[2],[5],[2]], [[1],[2],[1]]], **M** equal to [[[0],[0],[0]], [[0],[1],[1]], [[0],[0],[0]]], **X** with values of [[[7],[6],[5]], [[6],[5],[4]], [[4],[3],[2]]], and *b* being a zero, the ideal convolution result is sum($\mathbf{W}_j \odot (\mathbf{X} \odot \mathbf{1})$) = 81. For the original partial convolution, the mask correction ratio ($ratio_j$) equals to 9/2=4.50, and x_j' equals to 33*4.5=148.50. For the modified partial convolution, the absolute-weighted mask correction ratio equals to 17/7=2.43, and x_j' equals to 80.14, which is considerably closer to the idea convolution result.

Because the output of the partial convolution may still contain missing/filled values, the output may need to be fixed before passing to Batch Normalization (BN, [33]). If the output mask shows no zero values, then BN is applied directly on the partial convolution output; otherwise, the mean and standard deviation of the partial convolution output on good (i.e.,

unmasked) pixels are calculated, the missing (i.e., masked) pixels are replaced with random numbers drawn from a normal distribution with the calculated mean and standard deviation, and the replaced image is passed through BN.

2.1.2 Partial Merge Layer

The Partial Merge Layer (*pmerge*) merges an incomplete (target) image and its mask image with a spatially-matching and complete image (reference) through an adjusted 2D convolution (Eq. (5)):

$$x_j' = \mathrm{sum}\left(\mathbf{W}_j \odot \left(\mathbf{X} \odot \mathbf{M} \odot \mathbf{t}_\mathbf{M}\right)\right) ratio_j^{pmerge} + b \quad . \tag{5}$$

Where,

$$\mathbf{t}_\mathbf{M} = \mathbf{t}(\mathbf{M}) = \mathbf{M} / \sum_{f \in F} \mathbf{M}_f \tag{6}$$

$$ratio_j^{pmerge} = \frac{\mathrm{sum}\left(\left|\mathbf{W}_j\right| \odot \left(\mathbf{t}_\mathbf{1} \odot \mathbf{1}\right)\right)}{\mathrm{sum}\left(\left|\mathbf{W}_j\right| \odot \left(\mathbf{t}_\mathbf{M} \odot \mathbf{M}\right)\right)} \tag{7}$$

X (H,W,F_{ref}+F_{tgt}) represents the input image in the convolution (sliding) window that is created by concatenating the reference and target images along the feature dimension. H and W represent the height and width of the convolution window. F_{ref} and F_{tgt} denote the numbers of feature layers in the reference and target images, respectively. **M** is the matching masks for **X**. The mask layers corresponding to the reference images have a constant value of 1. $\mathbf{W}_j$ (H,W,F_{ref}+F_{tgt}) are the weights of the convolution kernel for the j^{th} output image and *b* is the corresponding bias. $\mathbf{t}_\mathbf{M}$ depicts the relative across-feature importance of the inputs during merge. *F* represents the collection of all input features (F_{ref}+F_{tgt}). The term $ratio_j^{pmerge}$ (Eq. (7)) represents the *pmerge* correction ratio that scales the convolution results to those as if there are no missing values (or pixels) in **X** by accounting for the absolute kernel weights, the mask, and the relative across-feature importance. Unlike $ratio_j^{pconv}$, the denominator of $ratio_j^{pmerge}$ is guaranteed to be non-zero (positive) because the $\mathbf{t}_\mathbf{M}$ and **M** for the reference layer(s) are always greater than zero and $\mathbf{W}_j$ is not a zero matrix. As such, the conditional branch for sum(**M**)=0 is not necessary. x_j' is the merged feature for the j^{th} output image. If the mask(s) for the target image(s) are zero, the *pmerge* will only utilize the reference information to derive x_j'.

2.1.3 Merge Layer

The Merge layer (*merge*) is the special case of the Partial Merge layer, where the two input sources (**X**) are both complete. Nevertheless, we still include the mask term (**M**) in its equation to account for convolution over zero-padded edge regions. The relative across-feature importance term is ignored because, even in the edge area, each feature has equal relative contribution (i.e., $1/F_{in}$) to the merge result. Eqs. (8) and (9) shows the calculation of *merge* in the convolution (sliding) window:

$$x_j' = \mathrm{sum}\left(\mathbf{W}_j \odot \left(\mathbf{X} \odot \mathbf{M}\right)\right) ratio_j^{merge} + b \tag{8}$$

$$ratio_j^{merge} = \frac{\mathrm{sum}\left(\left|\mathbf{W}_j\right| \odot \mathbf{1}\right)}{\mathrm{sum}\left(\left|\mathbf{W}_j\right| \odot \mathbf{M}\right)} \tag{9}$$

Where, **X** is the stack of two complete input images. **M** are the matching masks for **X**. In the edge area, both **X** and **M** are padded with zeros.

2.1.4 Encoder Layer

The Encoder Layer (*ec*) is based on the *pconv* and *pmerge* layers. It takes a complete image (reference) and a spatially-matched incomplete image (masked target) along with its mask as the inputs, pass them through the same *pconv* layer (i.e., shared weights) to get the two convoluted images ($\mathbf{I}_{ref}^{pconv}$ and $\mathbf{I}_{tgt,msk}^{pconv}$) and the updated mask ($\mathbf{I}_{mask,updt}^{pconv}$). The masked values in $\mathbf{I}_{tgt,msk}^{pconv}$ are replaced with random numbers drawn from the distribution of the unmasked part of $\mathbf{I}_{tgt,msk}^{pconv}$, BN and ReLU activation are performed on $\mathbf{I}_{ref}^{pconv}$ and the replaced $\mathbf{I}_{tgt,msk}^{pconv}$ to get $\mathbf{I}_{ref,act}^{pconv}$, $\mathbf{I}_{tgt,msk,act}^{pconv}$, and then $\mathbf{I}_{ref,act}^{pconv}$, $\mathbf{I}_{tgt,msk,act}^{pconv}$ are exported

as the Layer's two outputs. In addition, the *ec* layer merges the original $\mathbf{I}_{ref}^{pconv}$ and $\mathbf{I}_{tgt,msk}^{pconv}$ with the *pmerge* layer. The merged image ($\mathbf{I}^{pmerge}$) is used to gradually restore the spatial resolution in the corresponding decoder.

2.1.5 Decoder Layer

The Decoder Layer (*dc*) is based on the *merge* layer. It takes a lower-resolution image and a spatially-matched higher-resolution image as the inputs, up-samples the lower-resolution image to match the higher resolution, passes them through the *merge* layer, performs BN, activates the output image with the LeakyReLU function (α=0.3), and exports the activated image ($\mathbf{I}_{act}^{merge}$) as the layer's output. The lower-resolution image comes from either the last *ec* layer's target-side activated image (i.e., $\mathbf{I}_{tgt,msk,act}^{pconv4}$) or from the previous *dc* layer's activated output ($\mathbf{I}_{act}^{merge}$). The spatially-matched higher-resolution image comes from either the partial merged image ($\mathbf{I}^{pmerge}$) of the matching *ec* layer or the original reference image.

2.2 Complete model architecture

We adopt a U-net like (encoder-decoder) architecture similar to Liu et al. 2018 [26] and others [34] [20] for the missing pixel reconstruct model. The detailed design of the model is summarized in Table 1.

Inputs: Two 3-channel images ($\mathbf{I}_{ref}^{input}$, $\mathbf{I}_{tgt,msk}^{input}$) and one mask image ($\mathbf{I}_{mask}^{input}$) are the inputs to the model. One of the 3-channel image ($\mathbf{I}_{ref}^{input}$) contains the reference image, the acquisition day-of-year (DOY) of the reference (DOY_{ref}), and the DOY difference between the target and the reference images ($DOY_{tgt} - DOY_{ref}$). The two DOY related channel have constant values with their sizes set to match the size of the reference image. The other 3-channel image ($\mathbf{I}_{tgt,msk}^{input}$) contains the masked target image, DOY_{tgt}, and $DOY_{ref} - DOY_{tgt}$.

Encoder path: The model passes the inputs through 5 *ec* layers sequentially to gradually fill the missing pixels, reduce spatial size, capture higher semantic information, and create complete, merged image pyramids for the corresponding decoders. The first *ec* layer outputs 64 features. The numbers of output features are twice as large as the number of input features in the subsequent *ec* layers and plateaus after reaching 512. The first two *ec* layers use the 7x7 and 5x5 kernels respectively and the rest *ec* layers use the 3x3 kernels.

Decoder path: The output of last *ec* layer ($\mathbf{I}_{tgt,msk,act}^{pconv4}$) as well as the merged image from the second to the last *ec* layer ($\mathbf{I}^{pmerge3}$) are passed into the top decoder layer (*dc4*) to increase the spatial resolution through the up-sampling and *merge* operations. In the lower level decoder layers (except for the level 0), the output from the decoder one level above and the merged image from the encoder one level below are processed the in the same manner. In *dc0*, $\mathbf{I}_{ref}^{input}$ serves as the merged image to provide the complete higher resolution information. The merged images from the encoders and $\mathbf{I}_{ref}^{input}$ allow the model to recover sharp object boundaries gradually in the decoder path [34].

In both the encoder and decoder paths, no dilation is used in their convolutions and separable (depthwise and pointwise) convolution is used in place of full convolution to reduce model complexity and accelerate the computation.

Linear_conv: The output of the decoder path is activated by the LeakyReLU function. This linear 1x1 convolution slightly scales the distribution of the final decoder output.

Msdcorrect: This layer performs a per-sample linear correction so that the distribution of $\mathbf{I}^{linear\_conv}$ can better match that of the masked target image. Specifically, the following steps are performed: 1) means and standard deviations of the masked target image ($\mathbf{I}_{tgt,msk}^{input}[:,:,:,0]$) in the unmasked region ($\mathbf{I}_{mask}^{input}$ ==1) are calculated; 2) means and standard deviations of the linear_conv output ($\mathbf{I}^{linear\_conv}$) both in the entire image and in the unmasked region are calculated; 3) the mean of $\mathbf{I}^{linear\_conv}$ in the entire image is temporarily removed from $\mathbf{I}^{linear\_conv}$; 4) the zero-mean $\mathbf{I}^{linear\_conv}$ (entire image) is multiplied by the ratio between the standard deviations of $\mathbf{I}_{tgt,msk}^{input}[:,:,:,0]$ and the original $\mathbf{I}^{linear\_conv}$ (both in the unmasked region); 5) the means of $\mathbf{I}^{linear\_conv}$ (for entire original image), as well as the difference between the means of $\mathbf{I}_{tgt,msk}^{input}[:,:,:,0]$ and the original $\mathbf{I}^{linear\_conv}$ (in the unmasked region), are added back to generate the final model output.

Output: The model forward path generates the restored image $\mathbf{I}_{out}$ for the masked target with the same spatial dimensions.

2.3 Loss functions:

The goal of the model is to restore the masked target image with high accuracy. Therefore, we first define the two per-pixel losses, one for the unmasked region (valid, $L_{unmasked}$ or L_{valid}) and the other for the masked region (hole, L_{masked} or L_{hole}), as:

$$L_{unmasked} = L_{valid} = \frac{\left\| \mathbf{M} \odot \left(\mathbf{I}_{out} - \mathbf{I}_{gt} \right) \right\|_2^2}{\sum \mathbf{M}} \tag{10}$$

$$L_{masked} = L_{hole} = \frac{\left\| (\mathbf{1} - \mathbf{M}) \odot \left(\mathbf{I}_{out} - \mathbf{I}_{gt} \right) \right\|_2^2}{\sum (\mathbf{1} - \mathbf{M})} \tag{11}$$

Where, $\mathbf{I}_{gt}$ is ground truth image, which is the complete target image. $\mathbf{M}$ is the corresponding binary mask image. $\mathbf{I}_{out}$ is the model prediction (the restored target image).

Furthermore, to force the model to preserve the sharpness of the image, particularly near the transition area, we introduce the losses on the Sobel filtered images for the two regions:

$$L_{edge,unmasked} = L_{edge,valid} = \frac{\left\| \mathbf{M} \odot \left(Sobel\left(\mathbf{I}_{out} \right) - Sobel\left(\mathbf{I}_{gt} \right) \right) \right\|_2^2}{2 \sum \mathbf{M}} \tag{12}$$

$$L_{edge,masked} = L_{edge,hole} = \frac{\left\| (\mathbf{1} - \mathbf{M}) \odot \left(Sobel\left(\mathbf{I}_{comp} \right) - Sobel\left(\mathbf{I}_{gt} \right) \right) \right\|_2^2}{2 \sum (\mathbf{1} - \mathbf{M})} \cdot \left| R(\mathbf{I}_{ref}, \mathbf{I}_{gt}) \right| \tag{13}$$

Where, *Sobel*() is the function that returns the two Sobel edge maps. *R*() is function that calculates the correlation coefficient between the two given images. $\mathbf{I}_{ref}$ is the input reference image that matches the $\mathbf{I}_{gt}$. $\mathbf{I}_{comp}$ is the combination of the unmasked part of $\mathbf{I}_{gt}$ and the masked part of $\mathbf{I}_{out}$. This is performed to relax the requirement on the model performance if the reference image is not well correlated with the ground truth image.

In addition, all kernels and bias in the encoder and decoder paths are also subject to the l2 regularization loss $L_{regularizations,l2}$.

Finally, the total loss is calculated as the weighted sum of the losses functions defined above.

$$L_{total} = 1.0 L_{unmasked} + 1.0 L_{masked} + 0.25 L_{edge,unmasked} + 0.25 L_{edge,masked} + 2e^{-7} L_{regularizations,l2} \tag{14}$$

Where, the weights were determined by hyperparameter searching.

2.4 Dataset preparation

Data source: The original Landsat 8 Analysis Ready Data (ARD) Land Surface Temperature tiles [35] overlapped with Colorado between 2014 and 2018 with cloud cover < 80% and cloud shadow < 80% are downloaded from EarthExplorer (https://earthexplorer.usgs.gov/) using its Bulk Download Application (BDA).

Preprocessing: The Surface Temperature (ST) and two associated Quality Assessment (QA) layers were extracted from the downloaded tiles, converted to the same Int16 type, and stacked by acquisition date using the GDAL python API. The preprocessed images (.tif) were first uploaded to Google Cloud Storage buckets using gsutil commands. These .tif files along with 52 tile-based and layer-based metadata (such as acquisition date, fill value, scale factor, and data type) were eventually transferred to a Google Earth Engine (GEE) asset (ImageCollection ID: projects/ColoradoView/Landsat/ARD/8) using the GEE command-line utilities.

Reference and target LST image pairs: In this step, pairs of spatially-matched cloud-free images were sampled from Landsat 8 ARD LST image using GEE Python API. Note that in this study pixels with QA label 'fill' or 'cloud' are both considered as cloudy pixels (i.e., bad/missing pixels). A buffered mask image for every LST image was first created with the mask pixel value equal to 1 indicating no cloudy pixels are within the given Chebyshev distance (i.e., 32 pixels) from the current pixel or 0 indicating there is at least one cloudy pixels within that distance. For any two dates that are less than 48 days apart, the two LST images as well as their buffered mask images were gathered, pixels with the mask value 1 on both images were determined, a small fraction (i.e., $\sim 1/65^2$) of these pixels were randomly chosen as central pixels, LST patches (65x65 array images) surrounding these central pixels from the two days were exported as pairs of reference and target LST image pairs. Key GEE (Python) APIs used in this step include: ee.Image methods (e.g., addBands, updateMask, bitwiseAnd, And, reduceRegion, fastDistanceTransform, stratifiedSample, neighborhoodToArray), ee.ImageCollection methods (e.g., filterDate, map, mosaic), and ee.batch.Export.table.toAsset.

Cloud mask images: Raw cloud mask images were first sampled from Landsat 8 ARD LST QA images using similar GEE APIs. To improve computation speed, the raw cloud masks were generated to ensure they contained at least one good pixel and one bad pixel but without the information of the precise cloud percentage. The precise cloud percentage in each raw mask was determined after the GEE export. The masks with cloud percentage between 10% and 60% were retained. They were augmented by rotations, flips, and their combinations.

Assembled dataset: The reference and target LST images are randomly paired (zipped) with the cloud mask images. The masked target images were created by the element-wise multiplication (between mask and target image). There are a total of 5.6 million pairs of zipped data generated for this study. They are sharded once into 1000 files to reduce the necessity of large buffer size for shuffling during the training. 95% of the files are used for training and the rest 5% are reserved for validation. The data are standardized using tensorflow_transform and apache_beam functions on Google Cloud Platform (GCP) DataFlow (worker machine type: n1-highmem-16, processing time: ~3 hours).

2.5 Implementation

The model is implemented in TensorFlow [36] using its high-end Keras API (v1.14). Custom layers *pconv*, *pmerge*, *merge*, *ec*, and *dc* are sub-classed from tf.keras.layers.Layer. Layer *linear_conv* is an instance of tf.keras.layers.Conv2D. Layer msdcorrect is implemented in a function wrapped by tf.keras.layers.Lambda. Inputs are wired through multiple tf.keras.layers.Input. Lambda layers with single tf.identity function are used extensively for compartmentation between the custom layers. The complete model is composed of these Layers using the Keras functional model API (tf.keras.Model). The total number of parameters is approximately 1.97 million.

Each of the custom layers is tested individually in Google Colab with sample data before assembled.

The training routine follows the standard Keras procedure: (1) model creation, (2) model compilation, and (3) model fitting. The first two steps are optionally conducted within the tf.distribute.MirroredStrategy scope to utilize multiple GPUs during training.

The model is trained on Google Cloud Platform (GCP) (AI Platform) using the base machine n1-standard-8 (8 vCPU, 30 GB memory) plus two NVIDIA Tesla V100 GPUs. The training time per epoch (including validation) is around 1.5 hours. Key hyperparameters include global_batch_size (512), initial_learning_rate ($8e^{-5}$), regularization_factor ($2e^{-7}$), mask_weight (1.0), edge_weight (0.25). The decay of the learning rate is scheduled as follows: the first 3 epochs uses the initial learning rate; epochs 4-7 uses the halved rate; epochs 8-10 uses the one-fifth rate; and the further epochs uses the one-tenth rate. Hyperparameter searching is conducted on 10% of the training dataset for 1 epoch.

Table 1. Details of the missing pixel reconstruct model. BN stands for Batch Normalization. 'Both' means the sub-layer provides outputs both before and after BN. BN for all encoders (*ec0-4*) requires the special treatment to replace any missing values with random numbers (drawn from the normal distribution with mean and standard deviation matching the unmasked part). The upsample method is 'Nearest'. Alpha is 0.3 for all LeakyReLU. The model inputs are highlighted in blue and the output is highlighted in red.

Module Name	Sub-Layer	Filter Size	Out filters	Stride/ UpFactor	BN?	Act. Func.	Inputs	Outputs
ec0	pconv0	7x7	64	2	Both	ReLU	$\mathbf{I}_{ref}^{input}$	$\mathbf{I}_{ref}^{pconv0}$ $\mathbf{I}_{ref,act}^{pconv0}$
	pconv0	7x7	64	2	Both	ReLU	$\mathbf{I}_{tgt,msk}^{input}$ $\mathbf{I}_{mask}^{input}$	$\mathbf{I}_{tgt,msk}^{pconv0}$ $\mathbf{I}_{tgt,msk,act}^{pconv0}$ $\mathbf{I}_{mask,updt}^{pconv0}$
	pmerge0	3x3	64	1	N	-	$\mathbf{I}_{ref}^{pconv0}$ $\mathbf{I}_{tgt,msk}^{pconv0}$	$\mathbf{I}^{pmerge0}$
ec1	pconv1	5x5	128	2	Both	ReLU	$\mathbf{I}_{ref,act}^{pconv0}$	$\mathbf{I}_{ref}^{pconv1}$ $\mathbf{I}_{ref,act}^{pconv1}$
	pconv1	5x5	128	2	Both	ReLU	$\mathbf{I}_{tgt,msk,act}^{pconv0}$ $\mathbf{I}_{mask,updt}^{pconv0}$	$\mathbf{I}_{tgt,msk}^{pconv1}$ $\mathbf{I}_{tgt,msk,act}^{pconv1}$ $\mathbf{I}_{mask,updt}^{pconv1}$
	pmerge1	3x3	128	1	N	-	$\mathbf{I}_{ref}^{pconv1}$ $\mathbf{I}_{tgt,msk}^{pconv1}$	$\mathbf{I}^{pmerge1}$
ec2	pconv2	3x3	256	2	Both	ReLU	$\mathbf{I}_{ref,act}^{pconv1}$	$\mathbf{I}_{ref}^{pconv2}$ $\mathbf{I}_{ref,act}^{pconv2}$
	pconv2	3x3	256	2	Both	ReLU	$\mathbf{I}_{tgt,msk,act}^{pconv1}$ $\mathbf{I}_{mask,updt}^{pconv1}$	$\mathbf{I}_{tgt,msk}^{pconv2}$ $\mathbf{I}_{tgt,msk,act}^{pconv2}$ $\mathbf{I}_{mask,updt}^{pconv2}$
	pmerge2	3x3	256	1	N	-	$\mathbf{I}_{ref}^{pconv2}$ $\mathbf{I}_{tgt,msk}^{pconv2}$	$\mathbf{I}^{pmerge2}$
ec3	pconv3	3x3	512	2	Both	ReLU	$\mathbf{I}_{ref,act}^{pconv2}$	$\mathbf{I}_{ref}^{pconv3}$ $\mathbf{I}_{ref,act}^{pconv3}$
	pconv3	3x3	512	2	Both	ReLU	$\mathbf{I}_{tgt,msk,act}^{pconv2}$ $\mathbf{I}_{mask,updt}^{pconv2}$	$\mathbf{I}_{tgt,msk}^{pconv3}$ $\mathbf{I}_{tgt,msk,act}^{pconv3}$ $\mathbf{I}_{mask,updt}^{pconv3}$
	pmerge3	3x3	512	1	N	-	$\mathbf{I}_{ref}^{pconv3}$ $\mathbf{I}_{tgt,msk}^{pconv3}$	$\mathbf{I}^{pmerge3}$
ec4	pconv4	3x3	512	2	Y	ReLU	$\mathbf{I}_{tgt,msk,act}^{pconv3}$ $\mathbf{I}_{mask,updt}^{pconv3}$	$\mathbf{I}_{tgt,msk,act}^{pconv4}$
dc4	upsample4	-	512	2	-	-	$\mathbf{I}_{tgt,msk,act}^{pconv4}$	$\mathbf{I}^{upsample4}$
	merge4	3x3	512	1	Y	Leaky ReLU	$\mathbf{I}^{pmerge3}$ $\mathbf{I}^{upsample4}$	$\mathbf{I}_{act}^{merge4}$
dc3	upsample3	-	512	2	-	-	$\mathbf{I}_{act}^{merge4}$	$\mathbf{I}^{upsample3}$
	merge3	3x3	256	1	Y	Leaky ReLU	$\mathbf{I}^{pmerge2}$ $\mathbf{I}^{upsample3}$	$\mathbf{I}_{act}^{merge3}$
dc2	upsample2	-	256	2	-	-	$\mathbf{I}_{act}^{merge3}$	$\mathbf{I}^{upsample2}$
	merge2	3x3	128	1	Y	Leaky ReLU	$\mathbf{I}^{pmerge1}$ $\mathbf{I}^{upsample2}$	$\mathbf{I}_{act}^{merge2}$
dc1	upsample1	-	128	2	-	-	$\mathbf{I}_{act}^{merge2}$	$\mathbf{I}^{upsample1}$
	merge1	3x3	64	1	Y	Leaky ReLU	$\mathbf{I}^{pmerge0}$ $\mathbf{I}^{upsample1}$	$\mathbf{I}_{act}^{merge1}$
dc0	upsample0	-	64	2	-	-	$\mathbf{I}_{act}^{merge1}$	$\mathbf{I}^{upsample0}$
	merge0	3x3	1	1	Y	Leaky ReLU	$\mathbf{I}_{ref}^{input}$ $\mathbf{I}^{upsample0}$	$\mathbf{I}_{act}^{merge0}$
	linear_conv	1x1	1	1	N	-	$\mathbf{I}_{act}^{merge0}$	$\mathbf{I}^{linear\_conv}$
	msdcorrect	-	1	-	-	-	$\mathbf{I}_{tgt,msk}^{input}$ $\mathbf{I}_{mask}^{input}$ $\mathbf{I}^{linear\_conv}$	$\mathbf{I}_{out}$

3. RESULTS AND DISCUSSION

Validation metrics (the total loss, all components in the total loss, and the unweighted whole-image MSE all decreased monotonically in the 10 epochs. However, as the learning rate drops with more epochs, the improvement of these metrics slowed. The RMSEs in the original Kelvin scale are also estimated for the applicable MSE metrics. The weighted regularization loss at the end of epoch 10 is approximately $4.708e^{-4}$, which accounts for 10.8% of the total validation loss or 11.1% of the total training loss.

The validation of the model shows that the per-image RMSE values for the restored LST image in the unmasked, masked, and whole region are 0.29K, 1.00K, and 0.62K, respectively. The edge losses are also higher in the masked region (i.e., 0.0111, unscaled from the model output) than in the unmasked region (i.e., 0.0033, unscaled as well). The validation dataset has an average cloud fraction of 0.27 and an average correlation coefficient of 0.67 between the reference and target images. The per-image variance of the target image is 4.48 (or 2.12 K for the per-image standard deviation).

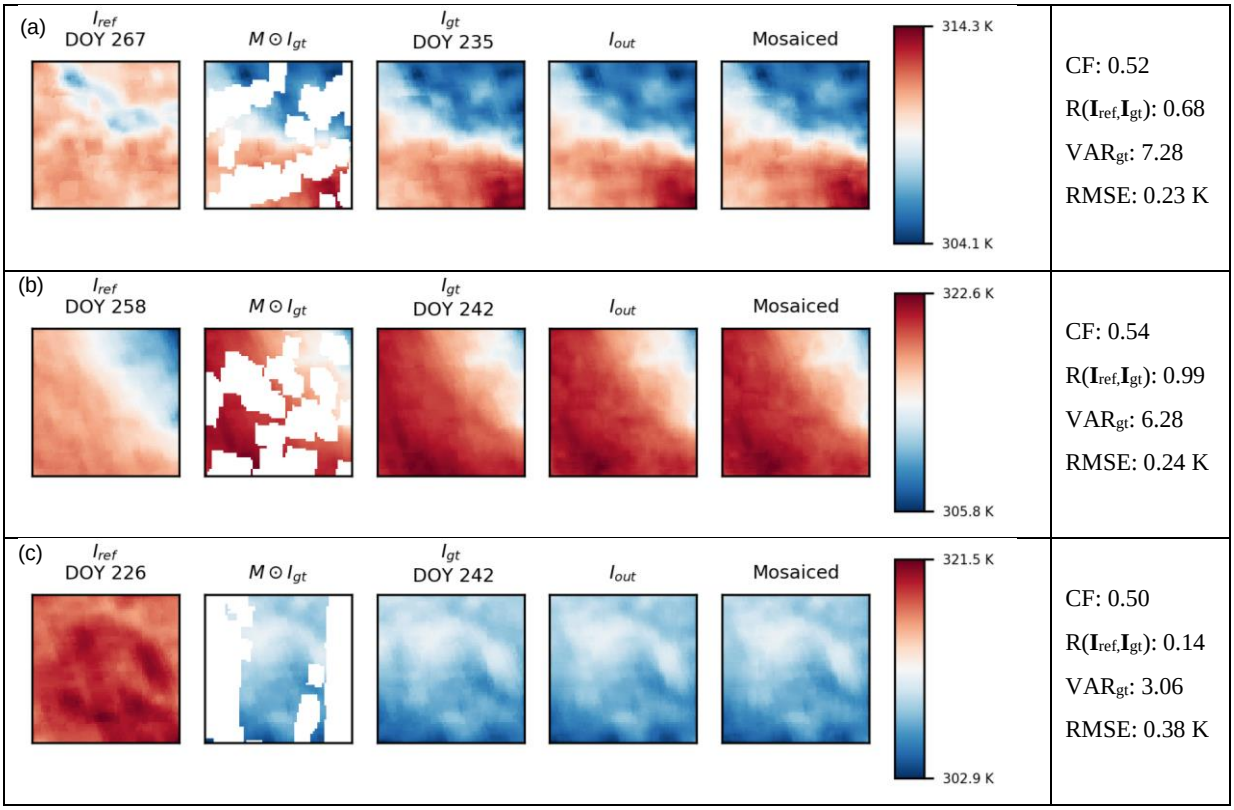

Figure 1. Model validation examples from the cloud fraction group 0.498-0.595 (quantile range: 0.95-1.00). The subplots in each example from left to right refer to (1) the reference image ($\mathbf{I}_{ref}$), (2) the to-be-recovered masked target image ($\mathbf{M}\odot\mathbf{I}_{gt}$), (3) the complete target image (ground truth, $\mathbf{I}_{gt}$), (4) the model prediction ($\mathbf{I}_{out}$), and (5) $\mathbf{I}_{comp}$, the mosaic of the unmasked part of $\mathbf{I}_{gt}$ and the masked part of $\mathbf{I}_{out}$. The metrics on the right refer to: (1) CF: the cloud (missing) fraction of the mask, (2) R($\mathbf{I}_{ref}$,$\mathbf{I}_{gt}$): the correlation coefficient between the $\mathbf{I}_{ref}$ and $\mathbf{I}_{gt}$, (3) the variance of $\mathbf{I}_{gt}$, and (4) the root mean square error (RMSE) between $\mathbf{I}_{out}$ and $\mathbf{I}_{gt}$ (unit: K).

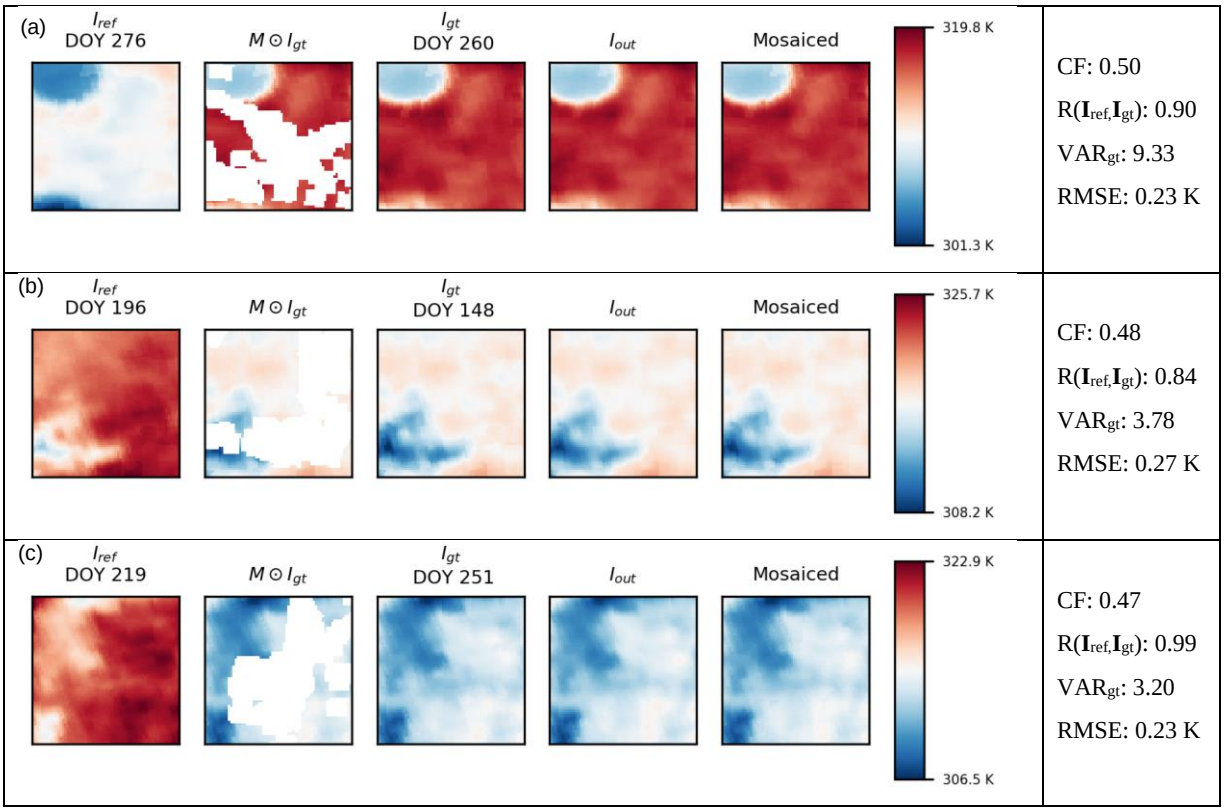

Figure 2. Model validation examples from the cloud fraction group 0.468-0.498 (quantile range: 0.90-0.95). The notations are the same as in Figure 1.

Since it is more challenging to restore images that are heavily masked and/or have high LST variance, we choose test cases with the target variance in the higher half from the validation dataset and the cloud fraction in the five medium to high quantiles (i.e., quantiles 0.45-0.5, 0.70-0.75, 0.85-0.90, 0.90-0.95, and 0.95-1.00). Figure 1 to Figure 5 show the model performance on 16 such cases. They are separated into five figures according to their cloud fractions.

For all 16 cases, the transition between the masked and unmasked as well as in the image boundary are smooth and unbiased particularly in the model prediction ($\mathbf{I}_{out}$). This indicates that the proposed mask correction ratios in the *pconv* and *pmerge* layers scale the partially convoluted values correctly.

For cases with high (e.g., >0.7) correlation between the reference and target images (e.g. Figure 1(b)), both sharp local patterns (e.g., Figure 2(a)) and high-level semantics (e.g., Figure 2(b) and (c), Figure 3(a)) are recovered by merging information from the reference counterparts. In contrast, for cases with low correlation coefficients (e.g., < 0.3), the model relies more on gradually expanding the unmasked information in the target to fill the holes (e.g., Figure 4(a), Figure 5(a)). As a result, the restored images for these cases may be blurrier (e.g., Figure 4(a), Figure 5(b)).

For partially correlated cases (e.g., Figure 3(b)), the unmatched high-level pattern (e.g., the bottom crop circle) from the reference image is ignored. However, the unmatched low-level structure from the reference image may be "copied" into the recovered image (e.g., the vertical stripes surrounding the bottom crop circle).

Many of these cases show large mean LST difference between the reference and target images (e.g., Figure 2(a)) resulting from differences in image collection dates. Despite this, the model still restores the masked features reasonably well, suggesting that the non-linear LST-date relationship has been taken into account to adjust the LST scales.

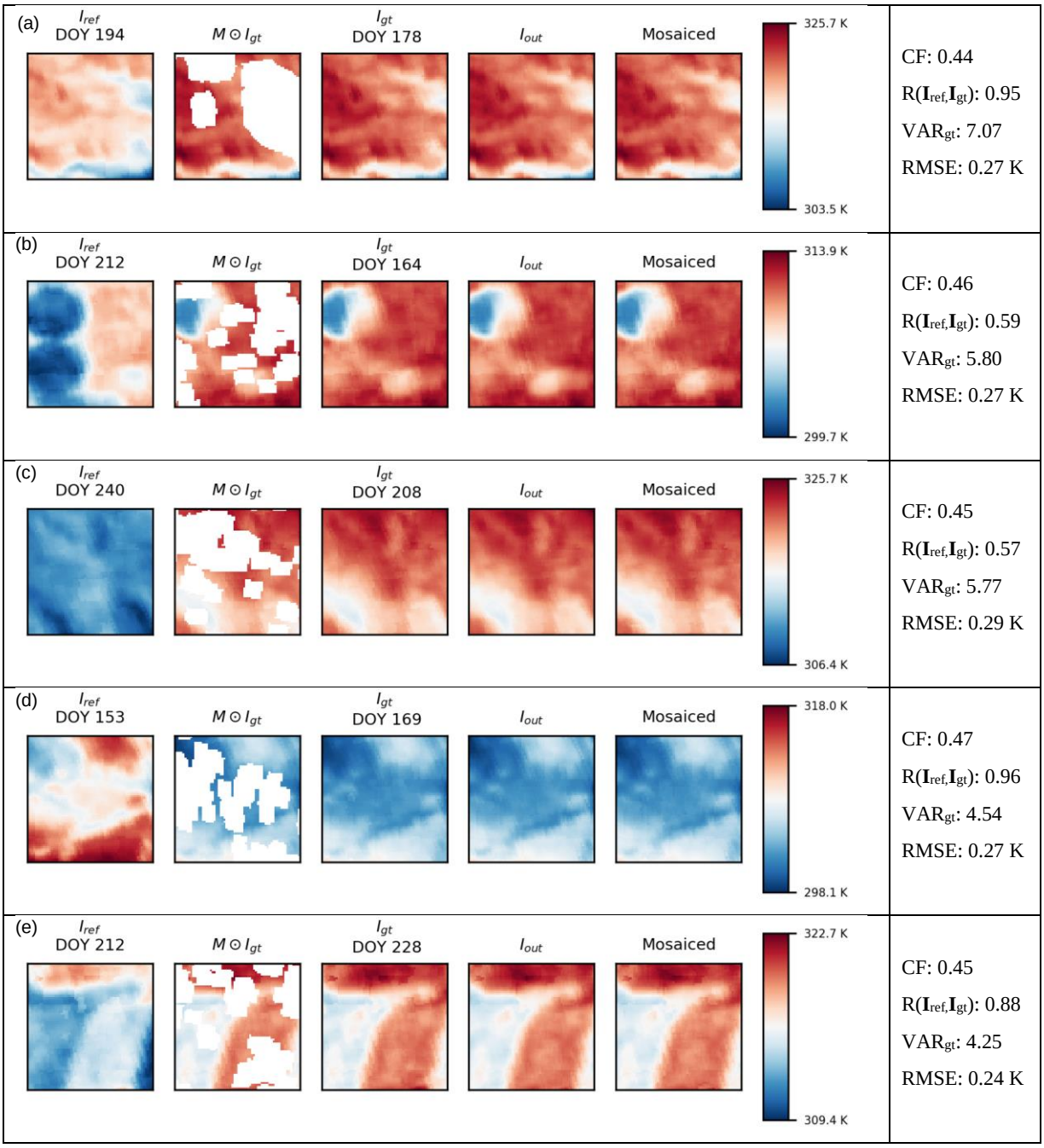

Figure 3. Model validation examples from the cloud fraction group 0.442-0.468 (quantile range: 0.85-0.90). The notations are the same as in Figure 1.

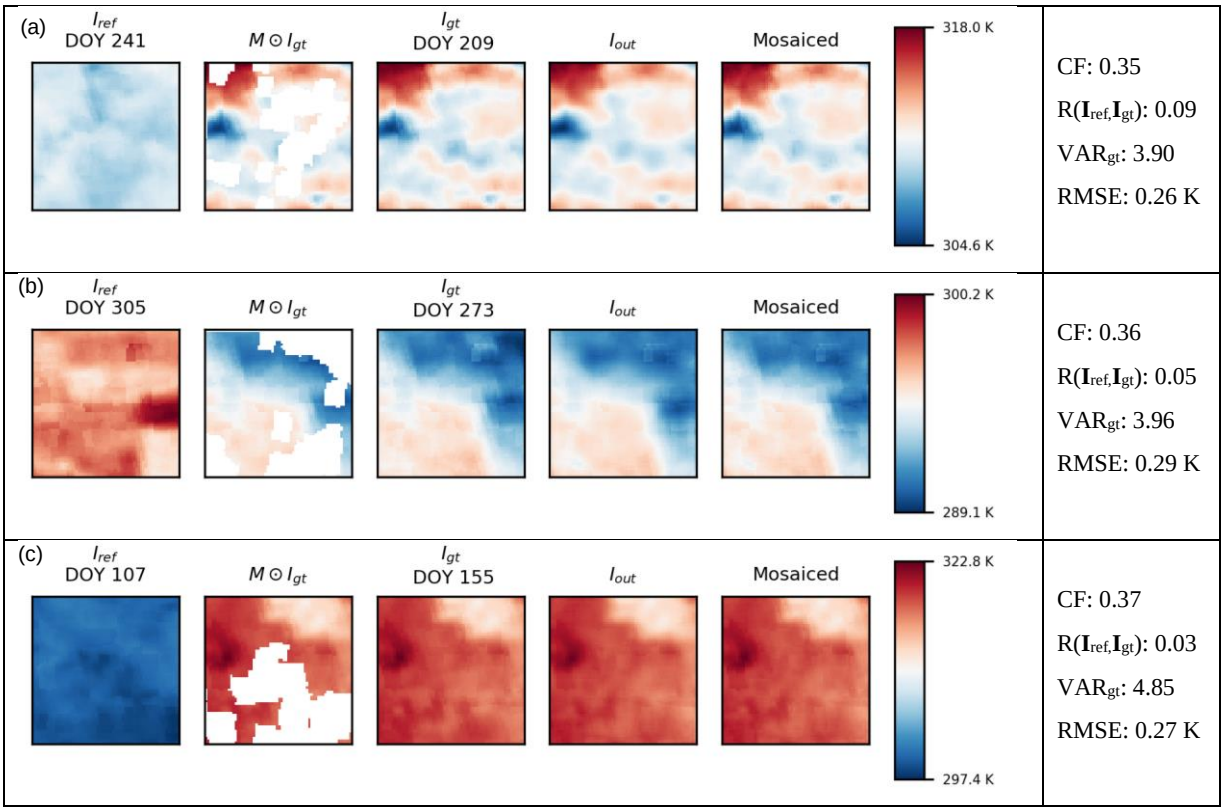

Figure 4. Model validation examples from the cloud fraction group 0.335-0.374 (quantile range: 0.70-0.75). The notations are the same as in Figure 1.

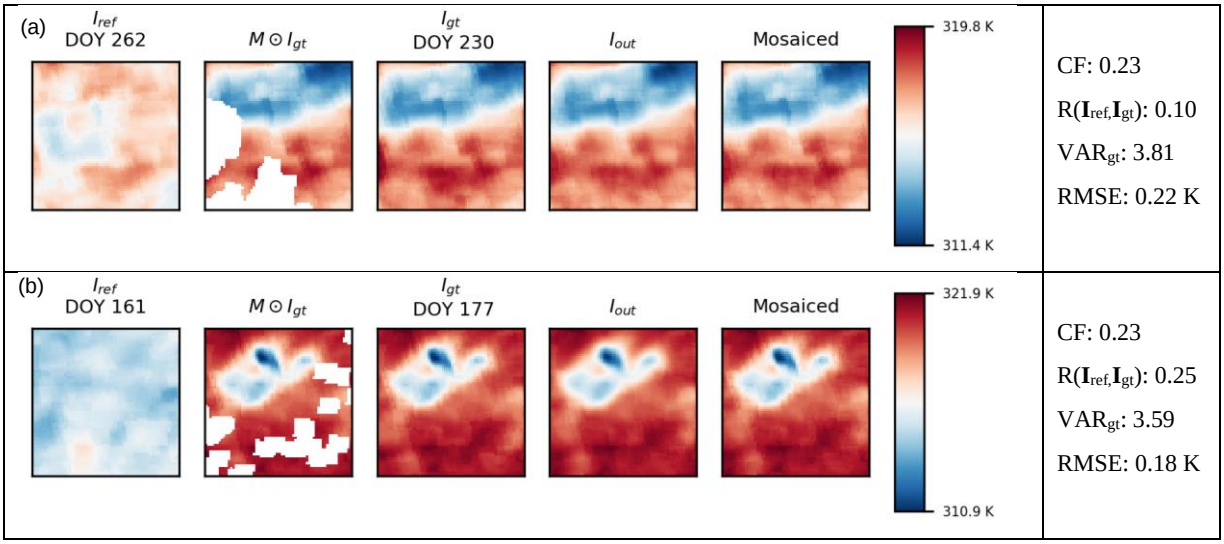

Figure 5. Model validation examples from the cloud fraction group 0.226-0.250 (quantile range: 0.45-0.50). The notations are the same as in Figure 1.

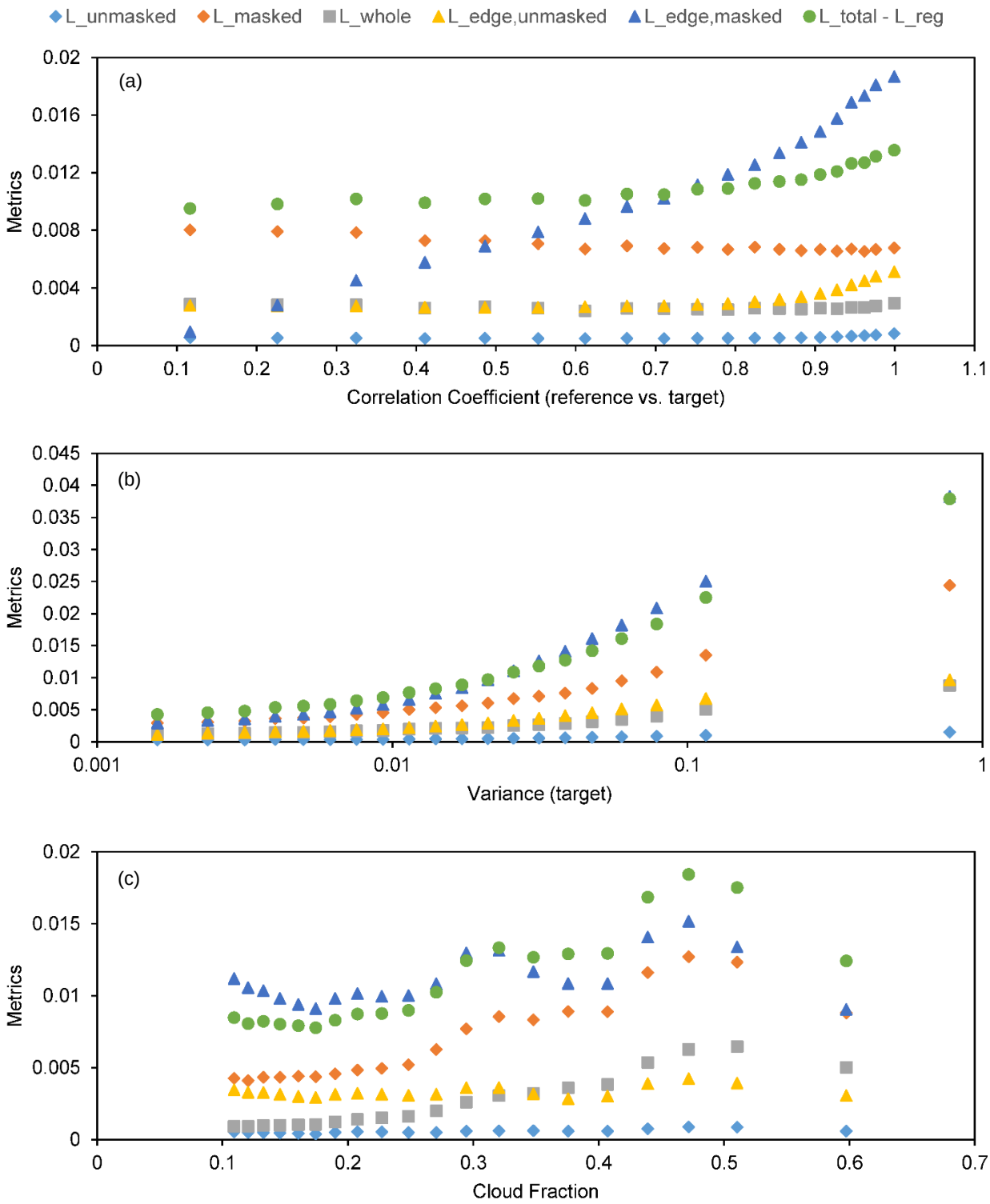

Figure 6. The mean metrics of the 20 quantile groups for the three aspects: (a) the correlation coefficient between the reference and the target images, (b) the variance of the target image, and (c) the cloud fraction. Note that not all metrics in this figure are comparable or having particular physical meanings.

Three aspects of the inputs are thought to affect model performance: (1) the strength of the (linear) correlation between the complete reference and target (ground truth) images $R(\mathbf{I}_{ref}, \mathbf{I}_{gt})$, (2) the variation of the target image, and (3) the fraction of the missing part (i.e., the cloud fraction).

The six metrics of the entire validation dataset (283,000 records) were aggregated by 20 quantiles for each of the three aspects (Figure 6).

For the correlation coefficient $R(\mathbf{I}_{ref}, \mathbf{I}_{gt})$ (r for short), the model's performance is generally independent of r. For example, the three MSEs are either invariant with r (i.e., $L_{unmasked}$ and L_{whole}) or slightly negatively correlated with r (i.e., L_{mask}). The masked edge loss (i.e., $L_{edge,masked}$, dark blue triangles in Figure 6(a)) show a clear positive correlation with r (up to 0.8) because the loss contains the absolute r as a multiplier. Both the masked and unmasked edge losses show faster increase rates as r approaches one (i.e., between 0.8 and 1.0). Similarly, the total loss (excluding the regularization loss) shows a flat pattern until r=0.8 and slightly increases as r goes higher. The explanation for these (unmasked edge and total losses) patterns is unclear and warrants further investigation. However, they may suggest that either 1) for the 64x64 LST dataset the model only needs the semantics in the unmasked part of the target to reconstruct the whole image and ignores the information from the reference image (i.e., the current model has not fully utilized the information in the reference image); or 2) the model found an imperfect balance between information extracted from the reference and the unmasked target depending on r.

All metrics show consistently exponential increases as the target's variance increases. The pattern is expected as larger variance usually indicating more complex spatial context and more missing information for the same mask. Increasing the model complexity and the size of the training samples may alleviate this problem.

In general, most of the metrics are positively correlated with the cloud fraction as expected. Although there are no consistent monotonic patterns, the variances of these metrics are increasing consistently as the cloud fraction increases. This may be partly explained by the distribution of the training samples in terms of cloud fraction where the number of samples is decreasing with the cloud fraction (data not shown). To reduce the high variance for the higher cloud fractions, a more balanced training dataset across all cloud fractions may be necessary.

4. CONCLUSIONS

We developed the missing pixel reconstruct model in a U-net like architecture that includes an improved mask correction ratio for the partial convolution layer and newly developed partial merge layer. The model was applied on the Landsat 8 ARD LST product to restore large holes due to cloud contamination. A spatially-matched complete reference image with an adjacent acquisition date (up to 48 days) was found for each of the incomplete target image to supplement the missing semantics in holes. The validation results (after the 10-epoch training) show that the average RMSE values for a restored LST image in the unmasked, masked, and whole region are 0.29K, 1.00K, and 0.62K, respectively. In general, the model is capable of capturing the high-level semantics from the inputs and bridging the difference in acquisition dates for gap filling. The transition between the masked and unmasked regions (including the edge area of the image) is smooth and reflected realistic features (e.g., LST gradients). For large masked areas, the reference provides semantics at both low and high levels. Interestingly, higher correlation between the reference and target images was not positively related to the model performance. To further improve of the model performance, future investigations may focus on 1) preparing more balanced training data in terms of cloud fraction, and 2) introducing mechanisms (such as gating) to merge the reference information according to its correlation to the unmasked target.

ACKNOWLEDGEMENTS

This work is supported by the US Department of Agriculture (USDA) UV-B Monitoring and Research Program, Colorado State University, under USDA National Institute of Food and Agriculture Grant 2016-34263-25763. This work is also supported by the U.S. Geological Survey under Grant/Cooperative Agreement No. G18AP00077. We thank Google Cloud Platform Education Grants (especially Kara Capozzi) for their research credits support (EDU Credit kcapozzi 35804832). Landsat Level 2 Surface Temperature Science Product courtesy of the U.S. Geological Survey. We thank Google Earth Engine (especially Tyler Erickson) for their support. We also thank Geospatial Centroid, Colorado State University (especially Sophia Linn and Melinda Laituri) for their support.

REFERENCES

[1] Quattrochi, D. A., and Luvall, J. C., “Thermal infrared remote sensing for analysis of landscape ecological processes: methods and applications,” Landscape Ecology, 14(6), 577-598 (1999).

[2] Batra, N., Islam, S., Venturini, V., Bisht, G., and Jiang, L., “Estimation and comparison of evapotranspiration from MODIS and AVHRR sensors for clear sky days over the Southern Great Plains,” Remote Sensing of Environment, 103(1), 1-15 (2006).

[3] Weng, Q., Lu, D., and Schubring, J., “Estimation of land surface temperature–vegetation abundance relationship for urban heat island studies,” Remote Sensing of Environment, 89(4), 467-483 (2004).

[4] Lambin, E. F., and Ehrlich, D., “Land-cover changes in sub-saharan Africa (1982–1991): Application of a change index based on remotely sensed surface temperature and vegetation indices at a continental scale,” Remote Sensing of Environment, 61(2), 181-200 (1997).

[5] Wan, Z., Zhang, Y., Zhang, Q., and Li, Z. L., “Quality assessment and validation of the MODIS global land surface temperature,” International Journal of Remote Sensing, 25(1), 261-274 (2004).

[6] Price, J. C., “Using spatial context in satellite data to infer regional scale evapotranspiration,” IEEE Transactions on Geoscience and Remote Sensing, 28(5), 940-948 (1990).

[7] Li, Z.-L., Tang, B.-H., Wu, H., Ren, H., Yan, G., Wan, Z., Trigo, I. F., and Sobrino, J. A., “Satellite-derived land surface temperature: Current status and perspectives,” Remote Sensing of Environment, 131, 14-37 (2013).

[8] Brooks, E. B., Thomas, V. A., Wynne, R. H., and Coulston, J. W., “Fitting the Multitemporal Curve: A Fourier Series Approach to the Missing Data Problem in Remote Sensing Analysis,” IEEE Transactions on Geoscience and Remote Sensing, 50(9), 3340-3353 (2012).

[9] Yu, C., Chen, L., Su, L., Fan, M., and Li, S., "Kriging interpolation method and its application in retrieval of MODIS aerosol optical depth," 2011 19th International Conference on Geoinformatics, 1-6 (2011).

[10] Barcelos, C. a. Z., and Batista, M. A., "Image inpainting and denoising by nonlinear partial differential equations," 16th Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2003), 287-293 (2003).

[11] Chan, T., Marquina, A., and Mulet, P., “High-Order Total Variation-Based Image Restoration,” SIAM Journal on Scientific Computing, 22(2), 503-516 (2000).

[12] Criminisi, A., Perez, P., and Toyama, K., “Region filling and object removal by exemplar-based image inpainting,” IEEE Transactions on Image Processing, 13(9), 1200-1212 (2004).

[13] Lingli, W., Qu, J. J., Xiaoxiong, X., Xianjun, H., Yong, X., and Nianzeng, C., “A new method for retrieving band 6 of aqua MODIS,” IEEE Geoscience and Remote Sensing Letters, 3(2), 267-270 (2006).

[14] Lin, C., Tsai, P., Lai, K., and Chen, J., “Cloud Removal From Multitemporal Satellite Images Using Information Cloning,” IEEE Transactions on Geoscience and Remote Sensing, 51(1), 232-241 (2013).

[15] Chen, M., Gao, Z., and Gao, W., "Crop classification using MODIS EVI series in North China," Proc. SPIE 7454, Remote Sensing and Modeling of Ecosystems for Sustainability VI 7454, 74541D (2009).

[16] Li, X., Shen, H., Zhang, L., Zhang, H., Yuan, Q., and Yang, G., “Recovering Quantitative Remote Sensing Products Contaminated by Thick Clouds and Shadows Using Multitemporal Dictionary Learning,” IEEE Transactions on Geoscience and Remote Sensing, 52(11), 7086-7098 (2014).

[17] Benabdelkader, S., and Melgani, F., “Contextual Spatiospectral Postreconstruction of Cloud-Contaminated Images,” IEEE Geoscience and Remote Sensing Letters, 5(2), 204-208 (2008).

[18] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "Imagenet classification with deep convolutional neural networks," Advances in neural information processing systems, 1097-1105 (2012).

[19] Lee, H., and Kwon, H., “Going Deeper With Contextual CNN for Hyperspectral Image Classification,” IEEE Transactions on Image Processing, 26(10), 4843-4855 (2017).

[20] Ronneberger, O., Fischer, P., and Brox, T., "U-net: Convolutional networks for biomedical image segmentation," International Conference on Medical image computing and computer-assisted intervention, 234-241 (2015).

[21] Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L., “Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising,” IEEE Transactions on Image Processing, 26(7), 3142-3155 (2017).

[22] Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., and Efros, A. A., "Context Encoders: Feature Learning by Inpainting," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2536-2544 (2016).

[23] Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A., [Extracting and composing robust features with denoising autoencoders] ACM, Helsinki, Finland(2008).

[24] Cai, N., Su, Z., Lin, Z., Wang, H., Yang, Z., and Ling, B. W.-K. J. T. V. C., “Blind inpainting using the fully convolutional neural network,” The Visual Computer, 33(2), 249-261 (2017).

[25] Yeh, R. A., Chen, C., Yian Lim, T., Schwing, A. G., Hasegawa-Johnson, M., and Do, M. N., "Semantic Image Inpainting With Deep Generative Models," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5485-5493 (2017).
[26] Liu, G., Reda, F. A., Shih, K. J., Wang, T.-C., Tao, A., and Catanzaro, B., "Image Inpainting for Irregular Holes Using Partial Convolutions," European Conference on Computer Vision (ECCV) 11215, 89-105 (2018).
[27] Liu, Z., Luo, P., Wang, X., and Tang, X., "Deep Learning Face Attributes in the Wild," IEEE International Conference on Computer Vision, 3730-3738 (2015).
[28] Goodfellow, I., Pouget-Abadie, J., Mirza, M., B. Xu, Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., "Generative adversarial nets," Advances in neural information processing systems, 2672-2680 (2014).
[29] Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., and Huang, T. S., "Generative Image Inpainting With Contextual Attention," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5505-5514 (2018).
[30] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L., "ImageNet Large Scale Visual Recognition Challenge," International Journal of Computer Vision, 115(3), 211-252 (2015).
[31] Zhang, Q., Yuan, Q., Zeng, C., Li, X., and Wei, Y., "Missing Data Reconstruction in Remote Sensing Image With a Unified Spatial–Temporal–Spectral Deep Convolutional Neural Network," IEEE Transactions on Geoscience and Remote Sensing, 56(8), 4274-4288 (2018).
[32] Yuan, Q., Wei, Y., Meng, X., Shen, H., and Zhang, L., "A Multiscale and Multidepth Convolutional Neural Network for Remote Sensing Imagery Pan-Sharpening," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 11(3), 978-989 (2018).
[33] Ioffe, S., and Szegedy, C., "Batch normalization: Accelerating deep network training by reducing internal covariate shift," International Conference on Machine Learning, 448-456 (2015).
[34] Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H., "Encoder-decoder with atrous separable convolution for semantic image segmentation," Proceedings of the European conference on computer vision (ECCV), 801-818 (2018).
[35] Cook, M., Schott, J. R., Mandel, J., and Raqueno, N., "Development of an Operational Calibration Methodology for the Landsat Thermal Data Archive and Initial Testing of the Atmospheric Compensation Component of a Land Surface Temperature (LST) Product from the Archive," Remote Sensing, 6(11), 11244-11266 (2014).
[36] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., [TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems], (2015).