

# Creating Cloud-Free Satellite Imagery from Image Time Series with Deep Learning

Stefan Oehmcke  
stefan.oehmcke@gmail.com  
University of Copenhagen  
Copenhagen, Denmark

Alexander V. Prishchepov  
alpr@ign.ku.dk  
University of Copenhagen  
Copenhagen, Denmark

Tzu-Hsin Karen Chen  
thc@envs.au.dk  
Aarhus University  
Roskilde, Denmark

Fabian Gieseke  
fabian.gieseke@uni-muenster.de  
University of Münster  
Münster, Germany

## ABSTRACT

Optical satellite images are important for environmental monitoring. Unfortunately, such images are often affected by distortions, such as clouds, shadows, or missing data. This work proposes a deep learning approach for cleaning and imputing satellite images, which could serve as a reliable preprocessing step for spatial and spatio-temporal analyzes. More specifically, a coherent and cloud-free image for a specific target date and region is created based on a sequence of images of that region obtained at previous dates. Our model first extracts information from the previous time steps via a special gating function and then resorts to a modified version of the well-known U-Net architecture to obtain the desired output image. The model uses supplementary data, namely the approximate cloud coverage of input images, the temporal distance to the target time, and a missing data mask for each input time step. During the training phase we condition our model with the targets cloud coverage and missing values (disabled in production), which allows us to use data afflicted by distortion during training and thus does not require pre-selection of distortion-free data. Our experimental evaluation, conducted on data of the Landsat missions, shows that our approach outperforms the commonly utilized approach that resorts to taking the median of cloud-free pixels for a given position. This is especially the case when the quality of the data for the considered period is poor (e.g., lack of cloud free-images during the winter/fall periods). Our deep learning approach allows to improve the utility of the entire Landsat archive, the only existing global medium-resolution free-access satellite archive dating back to the 1970s. It therefore holds scientific and societal potential for future analyses conducted on data from this and other satellite imagery repositories.

## CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; • **Applied computing** → **Earth and atmospheric sciences**.

## KEYWORDS

remote sensing, satellite imagery, image reconstruction

## ACM Reference Format:

Stefan Oehmcke, Tzu-Hsin Karen Chen, Alexander V. Prishchepov, and Fabian Gieseke. 2020. Creating Cloud-Free Satellite Imagery from Image Time Series with Deep Learning. In *9th ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data (BIGSPATIAL '20)*, November 3–6, 2020, Seattle, WA, USA. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3423336.3429345>

## 1 INTRODUCTION

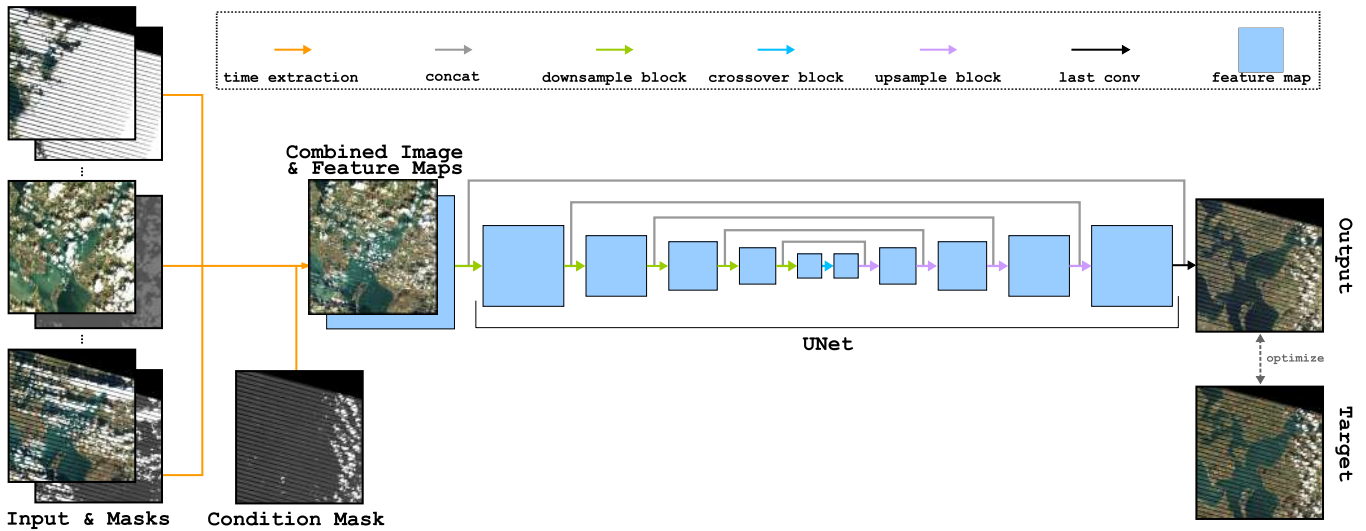
Earth observations serve as an important source for global land-cover and land-use monitoring as well as for studying trends and dynamics of phenomena on the surface of the Earth [1]. In particular, machine learning models extract high-level information from satellite images, which, in turn, allows for detailed analyses of landscape transformations [22]. In the past, satellite remote sensing has already supported various activities for sustainable land-use development, food security, hazard mitigation (e.g. tracing the patterns of landslides [6]), and monitoring urban sprawl [31]. Unlike aerial imagery, which requires actively launching a drone or flying a plane, satellites orbit Earth for a longer time span and collect usable and comparable data for any location. As such, satellite data can be harnessed to reliably monitor global environmental changes.

Unfortunately, cloud contamination and other distortions, such as shadows and missing data, greatly compromise the usability of satellite imagery for environmental monitoring. For example, clouds can partly obscure or even fully cover the land surface, which renders this part of the image difficult to be used by data analysis techniques commonly applied in the field of remote sensing. For instance, in the case of Landsat imagery, which depicts one of the key data resources for monitoring and assessing land-cover and land-use from 1972 to date [35], barely more than half of the terrestrial world holds a cloud-free observation per season [17]. The presence of the highly variable amount of missing data due to such distortions in different parts of space and time can present notable difficulties in the analysis of the data.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

*BIGSPATIAL '20*, November 3–6, 2020, Seattle, WA, USA

© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
ACM ISBN 978-1-4503-8162-8/20/11...\$15.00  
<https://doi.org/10.1145/3423336.3429345>



**Figure 1:** Our deep learning approach for the generation of cloud/distortion-free images based on time series imagery for specific target dates and locations. First, temporal features are extracted from the input sequence, while taking into account the missing data as well as the cloud cover masks produced by the FMASK algorithm. In addition, the target’s cloud cover mask is used to condition the temporal features (such a mask is only used during training; no mask is needed during testing). Afterwards, a fully-convolutional model (U-Net) is applied, which produces a composite image that is compared with the target image. Note that the target has distortions and our model tries to learn those with help of the condition mask. At test time, the condition mask is set to distortion free.

The need of the remote sensing community to detect distortions resulted in the development of the so-called FMASK [39] approach, an operational algorithm adopted by the United States Geological Survey (USGS). However, FMASK is still far from generating perfect masks and some clouds, including thin cirrus clouds, and the edges of clouds often remain in the resulting masks/satellite imagery.<sup>1</sup> In the literature, various approaches have been proposed to not only detect but also remove gaps caused by cloud contamination and distortions. For instance, by having access to nearly-anniversary image dates from preceding years, one can resort to mean and median values to fill data gaps [8]. More advanced techniques were proposed to assign higher weights to image dates close to the desired target date while resorting to lower weights when moving away from the target date [8, 9]. Nevertheless, such approaches generally do not fully eliminate clouds and other distortions due to the complex relationship of the reflectances, particularly due to thin clouds and because various important parameters have to be taken into account to obtain satisfactory results (such as complex distributions of satellite imagery and thus reflectances across the years). This all suggests that more advanced approaches have to be used to account for the non-linearity and the complex patterns of satellite data distortions.

**Contribution:** While current techniques can already detect and address the distortion of satellite imagery, these approaches usually suffer from a variety of shortcomings (see Section 2.3 and 3.1). To harness the open-access Landsat satellite data and other repositories of adequate coverage and sufficient historical sampling frequency, a

method for creating cloud- and distortion-free image mosaics at any point in time and that does not rely on pre-selected distortion-free training data is highly desirable. In this work, we propose such an approach based on deep learning. Our model can efficiently handle large amounts of satellite time series data instances and exhibits a superior performance compared to the commonly used approaches. A special property of our model is that is able to handle time series data of varying input, which permits to utilize more data in a simple and effective manner. This is in contrast to other deep learning methods that often rely on time series of fixed lengths (if they consider time series data at all). The overall model architecture and data flow are shown in Figure 1: In the first step, the time series data are aggregated via a special gating function (time extraction, yellow arrow) to obtain an aggregated representation of the data. In the second step, the well-known U-Net architecture [23] is used to generate the desired cloud- and distortion-free image. An important ingredient for the overall approach is the way the input time series are aggregated, which is inspired by the workflow of state-of-the-art median/mean-based approaches. In particular, the aggregation is conditioned on cloud masks that are typically available with the raw input data (or which could be obtained via standard segmentation models). Our experiments show that our approach clearly outperforms the commonly used baseline approach.<sup>2</sup> Our approach can easily be integrated into existing GPU-clusters for spatial and spatio-temporal analyzes, which could increase the performance of any subsequent models and inquiries.

<sup>1</sup>Another problem is the distortion of satellite images and loss of signal due to the malfunction of sensors [19], such as Landsat-7 ETM+ Scan-Line Correction (SLC). While there have been attempts to fill SLC-off gaps [5], artifacts may still exist.

<sup>2</sup>We also provide a novel benchmark dataset based on data from the Landsat program, which will facilitate the development and comparison of future methods. This dataset and our source code will be made publicly available in case of acceptance.

## 2 BACKGROUND

We start by outlining some background related to the satellite imagery considered and by providing the related work in the context of cloud- and distortion-free image generation.

### 2.1 Satellite Imagery and Distortions

In this work, we consider data from the USGS Landsat program, which allows to assess and to monitor land-cover and land-use changes since 1972. It is worth stressing, however, that the approach provided in this work is generally applicable to any kind of satellite time series data. Each Landsat sensor has a 16-day revisit time and acquires 30-meter resolution images for most of the bands. In general, the sensors on-board of the Landsat satellite family were designed to conduct continued Earth observation. Therefore, the radiometric resolution of the sensors of the different Landsat satellites matches reasonably well, particularly for Landsat-4 and 5 (Thematic Mapper, TM), Landsat-7 (Enhanced Thematic Mapper Plus, ETM+), and Landsat-8 (Operational Land Imager, OLI). The sensors have an overlapping period of observation and, thus, increase the probability of monthly and annually cloud-free land surface observation. Landsat TM covers the period of Jan 1, 1984 till May 5, 2012, Landsat ETM+ covers the period of Jan 1, 1999 till Apr 30, 2017, and Landsat OLI was launched on Feb 11, 2013 [24].

Kovalskyy and Roy [17] provide a world-wide statistical analysis of the occurrence of cloud and missing data in the Landsat archive in different seasons and lengths of time frame. Among the six continents, the probabilities of cloud-free observations were highest in North America, followed by Australia in both 2000 and 2010. At the same time, the lowest probability of cloud-free observations was in Africa. The seasonal variation of cloud-free Landsat acquisitions is evident, with relatively fewer acquisitions in the northern hemisphere in winter months (lowest in February with a mean of two images in 2010) than in the summer months (the highest in August with the mean for three images in 2010). However, considering the areas covered by clouds, only 56.4 and 48.1 percent of the terrestrial world have been clearly observed in each season in the northern hemisphere in 2000 and 2010, respectively. Although the advent of Landsat OLI from 2013 [24] and the combined sensors in specific years may provide a higher density of cloud-free observations from satellite images, challenges still remain in the successful reconstruction of cloud-free images that can be representative to a given target period. This is particularly relevant when the temporal sequence of satellite images, which is used for filling gaps in distorted satellite images, extends for a longer period.

### 2.2 Image Compositing

Several temporal compositing algorithms have been proposed that can be used to fill the cloud gaps and potentially other artefacts such as the Landsat ETM+ 2003 scan line failure. Such approaches generally aim at deriving complete cloud-free image data that are close to the ones given for a specific target date and location (potentially containing many gaps). These approaches first remove noise using information about the pixel quality band produced by the aforementioned FMASK algorithm [39]. Afterwards, the layers which are close to the target date are utilized for image composition. Other approaches are based on the maximum value of the

so-called Normalized Difference Vegetation Index (NDVI) derived from satellite bands [11], on the so-called medoid, which represents a multi-dimensional median in the time series [8], or on three-year moving window median values in order to fill data gaps, particularly in the cases of varying reflectance signals, due to crop rotation, urbanization, changing coast lines [26, 31].

Although such compositing methods can fill most of the gaps, it generally remains challenging to produce harmonized spatial patterns in the resulting composite image. For instance, the medoid approach was designed to select the values from all bands for a particular date and location. Yet, the spectral differences caused by intra-annual variation in phenology and sun-angle differences affect the harmonization of the temporal compositing image [3]. Similarly, the median compositing may occur across varying periods, therefore making the compositing product not entirely consistent in its appearance. An inconsistent time series may also lead to abnormal drops and peaks when monitoring land-cover changes such as deforestation, fire disturbances, and urbanization. Because image compositing approaches are based on the availability of comprehensive multi-temporal images, they generally have difficulties to accurately reconstruct pixels in case fewer observations are given.

### 2.3 Deep Learning and Cloud Removal

Deep learning models have also been applied for the task of removing clouds from satellite imagery. So far, however, such schemes have mostly focused on single image processing and semi-transparent clouds. For instance, Singh *et al.* [29] remove clouds from aerial imagery with a cyclic generative adversarial network (GAN), which works well for single images with translucent cloud coverage. Another GAN approach has been proposed by Grohnfeldt *et al.* [10], who add artificial clouds to the images and use this information to condition their GAN. Again, this approach is based on processing a single time step only. Malek *et al.* [18] have proposed an image reconstruction approach based on auto-encoders. They use entirely cloud-free images as target and cloud-obscured images as inputs, whereas our approach learns a meaningful reconstruction *without* relying on complete cloud-free targets. These approaches only work well for thin cloud coverage as they can only “guess” outputs for parts covered by clouds. In contrast, our approach resorts to different steps in order to extract existing information in a temporal coherent way.

Temporal information has successfully been integrated into deep learning models for land-cover classification. For instance, the work of Pelletier *et al.* [21] uses sequences of satellite imagery and resorts to convolutional networks, but only on the spectral (channel) and temporal dimensions, for land-cover segmentation. Similarly, Rußwurm and Körner [25] use recurrent convolutional networks for land-cover classification that learn to ignore cloud cover. For cloud removal tasks, only few works consider the temporal aspect, e.g. Sarukkai *et al.* [27], who use three time steps to remove clouds. Hence, current methods are generally restricted to a static number of time steps, while our method can utilize varying amount of steps, which is more practical since the availability of time steps varies per location and time frame.

### 3 FRAMEWORK

The goal is to derive a model that can generate a distortion-/cloud-free image given a sequence of partially clouded/occluded images, see again Figure 1. The model's input for a single instance is a series of images  $T \in \mathbb{R}^{t \times (c_{in} \times h \times w)}$  with associated ground truth/label information  $Y \in \mathbb{R}^{c_{in} \times h \times w}$  (i.e., images partly *without* clouds), where  $c_{in}$  depicts the number of input channels,  $h$  the height of an image,  $w$  the width of an image, and  $t$  the length of the sequence (i.e., the number of previous images). Additionally, we also have access to a missingness indicator  $M^x \in \{0, 1\}^{t \times (1 \times h \times w)}$  (0 indicates missing values) and a cloud coverage indicator  $C^x \in [0, 1]^{t \times (2 \times h \times w)}$  (1 indicates fully visibility) for the input series  $T$  as well as the temporal distance  $\delta^y \in \mathbb{R}^{t \times (1 \times 1 \times 1)}$  of each step to the target image. Note that the missingness is valid for all  $c_{in}$  channel and thus only requires one channel, while for the cloud coverage, we have cloud and cloud shadow data available and thus require two channels. To train with distorted and partly missing target images, we also use the missing indicator  $M^y \in \{0, 1\}^{1 \times (1 \times h \times w)}$  and cloud coverage  $C^y \in [0, 1]^{1 \times (2 \times h \times w)}$  of the target patch as conditional input during training (such indicators are *not* needed during the test phase).

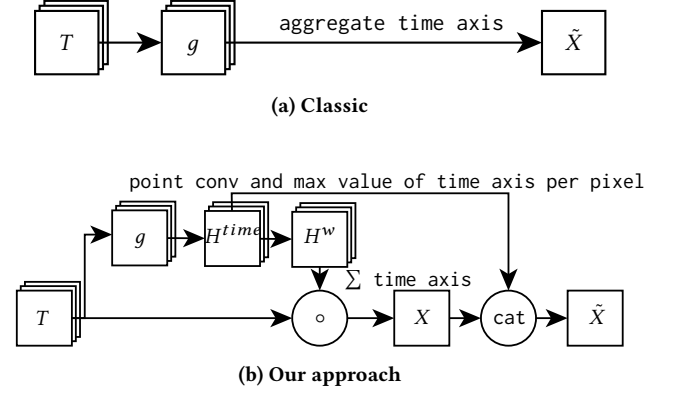
#### 3.1 Median Compositing

As baseline, we use median compositing, a typical approach for cloud removal [26]. This approach first selects only observations of sufficient quality, where pixels covered by clouds, cloud shadows at a high confidence level, and missing data (e.g., scan-line corrector (SLC)-off gaps) are removed according to the confidence mask provided by the FMASK algorithm [39]. Second, the images at the designated time frame are stacked into composites for each tile, except for the target image that is located in the middle month of each season. Third, the median value at each pixel is extracted to represent the reflectance of the target image. The main shortcomings of this method are that it does not take any spatial correlation into account and also does not account for the temporal distance to the target date.

#### 3.2 Deep Image Reconstruction

Most convolutional neural network architectures for image processing, such as the U-Net [23], do not consider temporal relations. If such architectures shall be applied efficiently, one has to reduce the time series of images  $T \in \mathbb{R}^{t \times (c_{in} \times h \times w)}$  in an appropriate way to a single image or to a set of feature maps  $\tilde{X} \in \mathbb{R}^{(c_{fuse} \times h \times w)}$ , where  $c_{fuse}$  denotes the number of input channels for the subsequent application of the fusion model. Our approach to remove distortions in satellite images is therefore threefold. The first part processes the time series and extracts abstract features, the second aggregates the series into a single step feature map, and the third one fuses this feature map to one coherent output image. We call the overall approach TimeGate.

**3.2.1 Processing Time Series.** There are many ways to process time series data. In our case, we are looking for the most recent and valid information that is not afflicted by distortions. This assumption allows us to process time steps independently. Particularly, our model can assign an importance value for each step without seeing



**Figure 2: Computational graphs for the time layers:** The classic approach (top) usually transforms the original input with some model  $g$  and aggregates over the time axis. Our proposed time layer (bottom) learns a normalized weighting for each time step per pixel, combines it to a composite, and finally concatenates the higher level information with the weighted average input image.

**Algorithm 1** Creating temporally aware feature maps with model  $g$  with  $\circ$  being the element-wise product and  $\text{emax}$  being the element-wise maximum value compared to the second argument.

```

1: function  $g(T, M^x, C^x, M^y, C^y, \delta^y)$ 
2:    $\text{decay} \leftarrow \exp \left( -\text{emax} \left( \text{point\_conv}_{\text{decay}}(\delta^y, 0) \right) \right)$ 
3:    $X \leftarrow \text{concat}(T, M^x \circ \text{decay}, C^x, \text{dim}=1)$ 
4:    $X \leftarrow \text{point\_conv}_{\text{combine}}(X)$ 
5:    $\text{cond} \leftarrow \text{point\_conv}_{\text{cond}}(\text{concat}(C^y, M^y, \text{dim}=1))$ 
6:    $X \leftarrow X + \text{repeat}(\text{cond}, t)$ 
7:    $X \leftarrow \text{point\_conv}_{\text{out}}(\text{conv}(X))$ 
8:   return  $X$ 

```

the other steps. In contrast, most tasks assume there is an importance w.r.t. the order of the time steps, for instance in forecasting tasks, where techniques such as 3d-convolutions [14] or recurrent network [7, 36] are employed.

We define model  $g$ , which extracts temporal features in Algorithm 1. First, the exponential function on the negated output of the rectified linear unit (ReLU) is applied to ensure a decay that is monotonically decreasing between 0 and 1 in Line 2 (other functions such as the Sigmoid activation function are also possible). The decay is inspired by Che *et al.* [4], but we decay all time steps w.r.t. the target instead of decaying the hidden state of the previous step. This decay indicates if an input time step should be weighted less based on its temporal distance to the target. Then, the image time series  $T$ , the missingness indicator  $M^x$ , and the cloud coverage indicator  $C^x$  of the input series are concatenated to obtain a single feature tensor  $X$  (Line 3 and Line 4). We also multiply the decay (scalar for every step) with the pixel-wise missing indicator to make this information available over the whole image. In Line 5, the conditional inputs  $C^y$  and  $M^y$  (during training: cloud and missingness indicators of the target) are combined and then added to the feature

**Algorithm 2** Our proposed algorithm to extract a single image plus time features from a series of images, where  $\circ$  being the element-wise product and  $\text{emax}$  being the element-wise maximum value compared to the second argument.

---

```

1: function TIME_GATE( $T, M^x, C^x, C^y, \delta^y$ )
2:    $H^{time} \leftarrow \exp(\text{emax}(g(T, M^x, C^x, C^y, \delta^y), 0)) - 1$ 
3:    $H^w \leftarrow H^{time} \circ \frac{1}{(\sum_{j=1}^t H_j^{time}) + \epsilon}$ 
4:    $X \leftarrow \sum_{i=1}^t H_i^w \circ T_i$ 
5:    $H^{agg} \leftarrow \max(\text{point\_conv}(H^{time}), \text{dim}=0)$ 
6:   return  $X, H^{agg}$ 

```

---

tensor  $X$  after being repeated for each of the  $t$  time steps (Line 6). Point-wise convolution layers (32 kernels of size  $1 \times 1$ ) are used to create linear combinations on a pixel-level and to keep the number of channels consistent. Afterwards, in Line 7, a convolution layer with 64 kernels of size  $3 \times 3$  is used to take into account the spatial relations followed by a another point convolution to reduce the number of kernels.<sup>3</sup> A non-linearity function (Mish [20]) and batch normalization are applied after each convolution operation.

**3.2.2 Aggregation to Single Image.** The fusion model requires an image or a set of fixed feature maps. One way to deal with this is to flatten the time steps into the channels, but this would not allow for varying lengths of time series. Another way would be to define a maximum length, which either restricts the available information or induces unnecessary computations if fewer steps are available; this renders this approach impractical in most scenarios, especially in our case where image availability is not always the same. The classical way is the use of temporally sensitive layers throughout the architecture, e.g. 3d convolutions, throughout the fusion model as proposed by Arbelle and Raviv [2], but this would impose a high computational cost that is avoidable in our scenario since our time steps can be processed individually. This is especially impractical if many time steps are available or bigger patches have to be considered. However, a straightforward approach to handle this is to run some model  $g$  to extract temporal features (see Figure 2a) and then aggregate the results (e.g., max pooling over the time axis or the final hidden state from GRU), which we call 2d- or 3d-classic. One problem with this approach, if used alone, it is unclear how much each time step contributed to the final output and that the original imagery could already be changed before it is given to the single image network architecture.

In our case, we assume an aggregation of all relevant steps should not only hold information that have been altered by the time information extraction layer (as the 2d/3d-classic method), but also an aggregate of the original images (e.g., weighted average). The overall approach is shown in Figure 2b. To achieve a good aggregation  $X \in \mathbb{R}^{(c_{in} \times h \times w)}$ , we create a tensor  $H^w \in \mathbb{R}^{t \times (c_{in} \times h \times w)}$  to weight the different time steps w.r.t. their relevance on a pixel-level. This time-gating mechanism is defined in Algorithm 2. First, by utilizing the element-wise max function, which is equal to the non-linearity function ReLU (Line 2), we can disable elements from

certain steps when they are set to zero in  $H^{time} \in \mathbb{R}^{t \times (c_{in} \times h \times w)}$ :

$$H^{time} := \exp(\text{emax}(g(T, M^x, C^x, C^y, \delta^y), 0)) - 1 \quad (1)$$

We resort to the element-wise max function followed by applying the exponential function to have the benefit of both, the exponential gaps between values as in softmax and completely deactivating values as in ReLU. Subtracting one makes sure that we retain zero values as such, but it can be set to  $1 - \epsilon$  during training to enable gradient flow, with a small positive constant  $\epsilon$ . The function  $g$  should be a temporal model, see again Algorithm 1. Then, we create a normalized aggregation weight (Line 2):

$$H^w := H^{time} \circ \frac{1}{(\sum_{j=1}^t H_j^{time}) + \epsilon}, \quad (2)$$

where  $\epsilon$  is again a small positive number for the case that all entries in an element of  $H^{time}$  are zero, and  $\circ$  denotes element-wise multiplication. Thereafter, our weight  $H^w$  is applied to the original input  $T$  and summed over the time axis to create a weighted average version of the input (Line 4):

$$X := \sum_{i=1}^t H_i^w \circ T_i \quad (3)$$

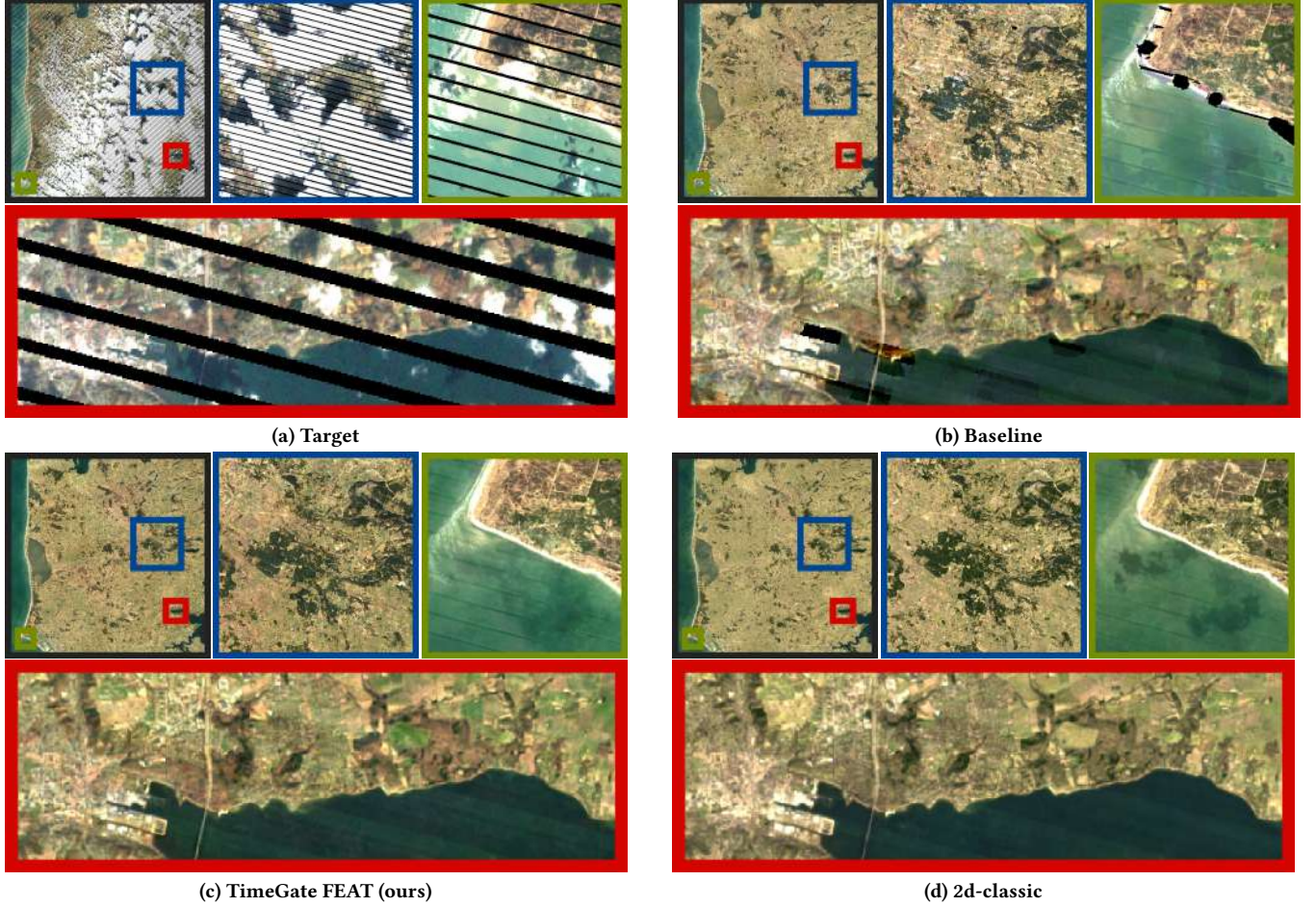
To extract abstract features, we apply a point-wise convolution and use the maximal value of the time axis per pixel to  $H^{time}$  to encode the temporal information while removing the temporal dimension in Line 5. Finally, we return the aggregated single step image  $X$  and the abstract features  $H^{time}$ .

The learning goal is to derive a distortion-/cloud-free image of the target image  $Y$ . During the training phase, the loss is evaluated on all parts of the target image that are not missing (yet, these parts could contain clouds). During testing/evaluation phase, the conditioning mask is set to zero to generate a completely distortion-/cloud free output.<sup>4</sup> For the evaluation, only non-missing and non-distorted/occluded pixels are considered. Our approach to handle the time series is comparable to the baseline algorithm: median compositing creates a single image by selecting the median value per pixel, but our algorithm learns how to weight the important time steps dynamically (and also appends abstract information) instead of only resorting to the median value. Another comparison can be drawn to gating mechanisms as in gated recurrent units (GRUs) or attention networks that are used to propagate certain information with the difference that we use our gating mechanism to reduce the temporal dimensionality.

**3.2.3 Fusion Model.** The resulting  $X$  is not coherent in intensity and spatial features since it consists of multiple images with different settings/from different time stamps. To fuse the information from the time layer, we use the popular U-Net architecture proposed by Ronneberger *et al.* [23] with some modifications. The architecture offers an encoding and a decoding part that first reduces the spatial resolution to encode abstract features and then upscales to the original size. Long skip connections, which concatenate previous feature maps to the current upscale block, ensure that previous features do not get lost. We apply MnasNet [32] (B1-version with  $\alpha = 1.0$ ) as encoder, where each block in MnasNet corresponds to a downsample block. The crossover block doubles the number

<sup>3</sup>We also tested larger kernel sizes (e.g.,  $5 \times 5$ ) and more layers, but did not see improvements from these.

<sup>4</sup>Note that the conditioning on the cloud mask is inspired by [30].



**Figure 3: Model output for one of the Denmark test tiles. It shows different sections, which are marked with different colored frames, of the complete tile for the different models in each subfigure.**

of neurons from the last downsample block ( $320 \rightarrow 640$ ) in one  $3 \times 3$ -convolutional layer and then reduces it again to 320 by a second  $3 \times 3$ -convolutional layer. Each upsample block consists of two  $3 \times 3$ -convolutional layers and utilizes skip connections, pixel-shuffle [28] for upscaling, and self-attention [33, 37] after the second block. The number of neurons in each upsample block is the same as the one from the corresponding downsample block.

The final output  $\tilde{Y}$  is:

$$\tilde{Y} = \gamma \cdot \text{U-Net}(\text{concat}(X, H^{agg}, \text{dim}=0)) + X, \quad (4)$$

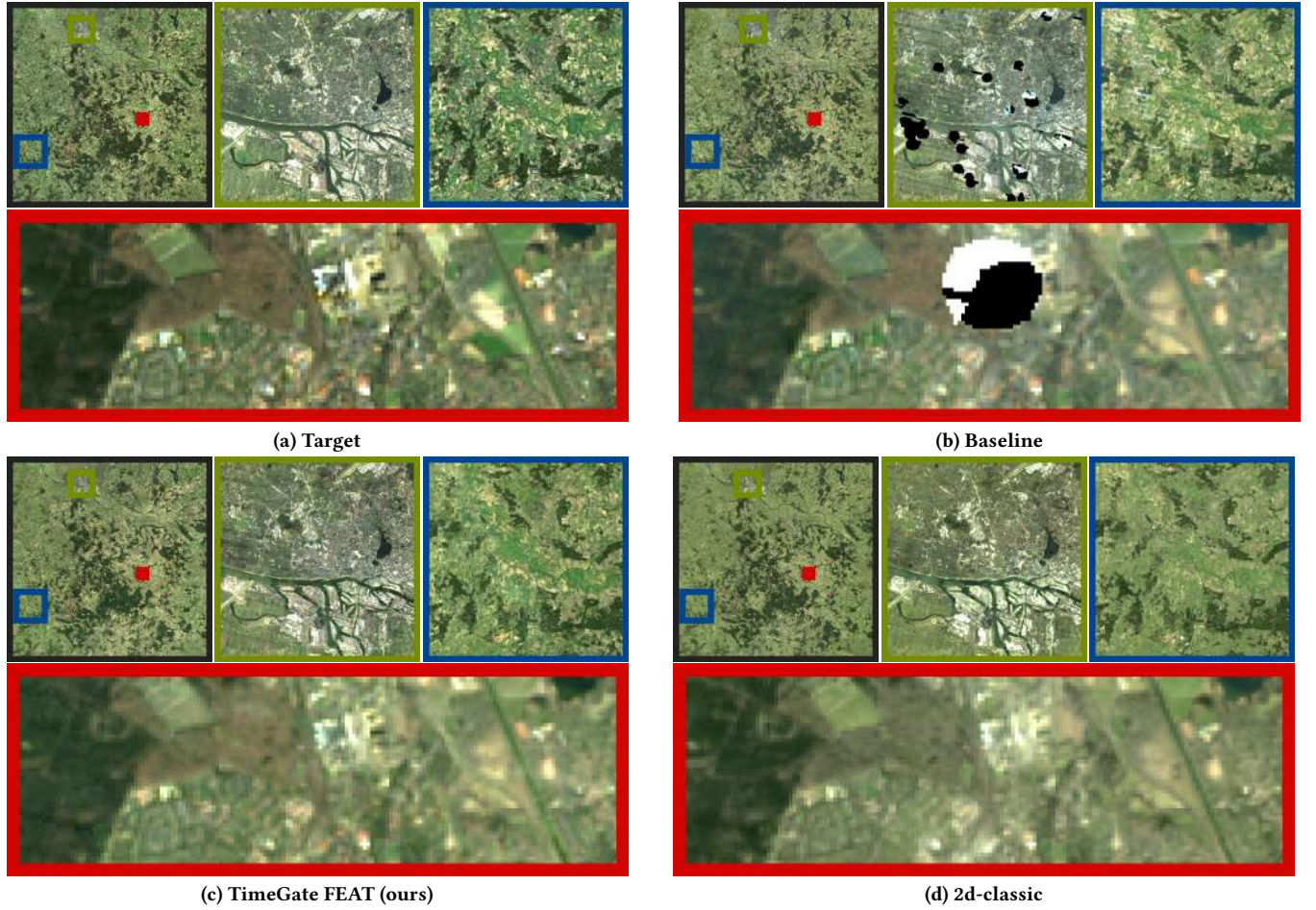
where  $\gamma$  is a learnable scalar value initialized with 0. This last skip connection allows the fusion model to only learn what to add or subtract from the aggregated image  $X$ . The output  $\tilde{Y}$  is then compared to the ground truth  $Y$  to obtain gradient information and finally adjust the parameters.

## 4 EXPERIMENTS

In this section, we report and compare the results obtained via the baseline and various instances of our deep learning approach.

### 4.1 Experimental Setup

For our experimental evaluation, we resorted to Landsat satellite imagery. Training a single model took about 10-16 hours using one GPU per model (Volta-100 cluster). Once trained, the models can be applied efficiently to large amounts of test data. For instance, one tile ( $5000 \times 5000$ ) with 20 time steps (total of 9 billion input features) could be processed in about 2-5 minutes. Depending on available hardware (e.g., GPU-clusters) one could predict multiple tiles at the same time allowing for rapid creation of distortion free tiles for subsequent analyses. Further, we employed two evaluation metrics, the mean squared error (MSE) and the structural similarity index (SSIM) [34] with the algorithm's default window size of 8 (meaning it compares  $192\,000\text{ m}^2$  of terrain per pixel). The MSE is a pixel-wise metric that compares values without considering spatial correlations, while the SSIM compares images within small windows and thus considers spatial correlations. Note that the MSE might be high if the output image does not have the same lighting although the structure might be identical. In contrast, the SSIM is more resilient to these changes and therefore our preferred metric. In addition, the evaluation metrics are only computed for target pixels that were *not* occluded by clouds and *not* affected by missing data.



**Figure 4: Model output for one of the Hamburg test tiles. It shows different sections, which are marked with different colored frames, of the complete tile for the different models in each subfigure.**

We also excluded water pixels since its structure changes rapidly, which makes it unsuitable for our evaluation purposes.

**4.1.1 Data.** Our dataset consists of images and the baseline we obtained via the Google Earth Engine<sup>5</sup> framework. This framework provides online access to archived Landsat data as a collection of the USGS [12]. All available Level-2 Landsat ETM+ and Landsat OLI images were used for the corresponding regions. Level-2 products are surface reflectance data at a 30-m spatial resolution with systematic atmospheric correction and are provided with per-pixel quality information assessed via the FMASK algorithm [38]. In total, nine tile locations are used (each of size  $185 \times 185$  km), including 195/21, 195/22, 196/20, 196/21, 196/22, 197/20, 197/21, 195/23, 117/43 (path/row) of the Landsat worldwide reference system (WRS-2).

We employ two seasonal frames to reflect typical land-cover patterns with different difficulties (e.g., the winter in Denmark is generally more cloudy than the summer). This results in a winter frame (from day 32 to day 120 of the year, covering February, March, and April) and a summer frame (from day 152 to day 243 of the year, covering June, July, and August). Further, we utilize an annual frame of 3 years with data from 2016 to 2018. This 3-year frame

is composed of 13 to 25 time steps for each tile in the winter and of 20 to 27 images in the summer periods. The resulting dataset consists of 5 tile locations for training, 1 for validation (197/20), and 3 for testing. For testing, we chose one tile location ( $5000 \times 5000$  30-meter pixels) in Denmark (197/21), in Hamburg (195/23, a temperate European city), and in Taipei (117/43, a subtropical Asian city), respectively. For each location multiple time steps are used as test target: 3 for Denmark, 3 for Hamburg, and 5 for Taipei. Note that the testing tile locations are not present in the training data, meaning that we chose different test sites to show generalizability. The raw data size for these test tiles are 20 GB, 36 GB, and 35 GB, respectively.

A single training instance was acquired in the following way: First, we randomly selected a patch of size  $96 \times 96$  pixels that contained at most 50% clouds and 33% missing data as target patch  $Y_i$ . Then, the input image series  $T_i$  was selected at the same patch position as the target patch from all available time steps. Additionally, the cloud coverage mask of the input image series was added to the input, and the target image cloud cover was used as conditional input. This process was repeated several times for all steps and tiles.

<sup>5</sup><https://earthengine.google.com/>

In total, this yields 26.000 training (85 GB), and 7.000 validation (25 GB) instances.

For testing, we cut out patches with a size of ( $320 \times 320$ ) pixels and stitched the results together to obtain a complete tile ( $5000 \times 5000$ ). To eliminate border artifacts, we removed 15 (overlapping) border pixels on all sides during the testing phase.<sup>6</sup> During training, we randomly flipped all images in a series horizontally or vertically with a 50% probability and randomly removed entire time steps down to at least three steps also with a 50% probability. Another training augmentation was the random insertion of the target image in 5% of the cases to train the temporal distance.

**4.1.2 Models.** As main baseline model, we resorted to the median composition method described in Section 3.1, which is the most commonly used method in remote sensing for the task at hand. We implemented several deep learning models in Python (3.8) using PyTorch (1.6). The Mish activation function [20] was used for all networks. We trained our model with the described approach from Section 3.2. Our TimeGate model had 15 046 582 learnable parameters. If the Pix2Pix learning method was used, the discriminator added 1 064 916 learnable parameters. Further, we also trained models—called 2d-classic and 3d-classic—without our time-gating mechanism but only with the temporal features. Note that we only changed the time processing method (TimeGate, 2d-classic, 3d-classic); the U-Net architecture (see Section 3.2.3) stayed always the same. For 2d/3d-classic, we first used point-wise convolution, then a convolution operation over the spectral (2d) or the spectral and temporal (3d) channel(s) (kernel size 3) were applied. Finally, another point-wise convolution was used. The induced architecture is comparable to a block from the MnasNet [32], but with the additional temporal depth-wise convolution. The 2d/3d-classic models used 15 046 583 and 15 054 711 learnable parameters, respectively.

**4.1.3 Training.** We considered three different training loss functions for training: Either the mean squared error (MSE), a simple pixel-wise error, or the feature reconstruction loss (FEAT) adapted from Johnson *et al.* [15], which is a more complex feature loss. The FEAT loss compares the intermediate feature maps of the VGG16 network (pre-trained on the ImageNet dataset) at three depths (3rd, 4th, and 5th block) given the predicted and target patch as input. This layer-wise comparison is done at a pixel level and we chose a smooth L1 loss for that task. The intuition for using the FEAT loss is that the feature maps hold abstract information, such as structure and coherence. We had six instead of the typical three channels, so we applied this loss on channels 1-3 (visible red, green, and blue) and 4-6 (infrared) separately. The third training scheme is based on GAN training, the so-called Pix2Pix [13] training, where the discriminator uses both, the input series  $T$  and the target image  $Y$  to condition our model (inspired by [29], who do this with single time steps). We also tried to use negative SSIM as a training loss, but it was unstable (nan and infinite values) and did not converge, which was probably because the extreme differences between clouds and no clouds make it difficult to train (since SSIM relies on averages

<sup>6</sup>Note that during production, one would include all time steps and specify a single one as the target, which has a temporal distance of 0. This would improve the image quality as the models can utilize brightness options of the target image and only fill the missing/obscured parts.

of small windows). This was, however, not a problem during evaluation since we do not apply it to distorted pixel there.

As optimizer, we used Adam [16] with a learning rate of 0.001 and a weight decay of 0.0005. All models were trained for 200 epochs and the batch size was set to 32. After 100 epochs we linearly annealed the learning rate towards 0 over the remaining 100 epochs. The final model was chosen w.r.t. to the best validation MSE.

## 4.2 Comparing Loss Functions

First, we compared the performance of our TimeGate network trained either with MSE or FEAT loss as well as the Pix2Pix (GAN) training scheme. The results are presented in Table 1. Using the FEAT loss, we obtained a low MSE and, more importantly, the SSIM of the FEAT model was higher than for the other models, including the baseline. This was expected as the MSE loss only yields point-wise errors that do not offer any spatial comparison (e.g., if the structure is kept). Overall, we observed that the FEAT loss performed best and used it therefore to train our models.

**Table 1: Comparison of the different training loss functions (MSE and FEAT) for our TimeGate model using the MSE (lower is better) and SSIM (higher is better) as evaluation metrics on all three test tile locations.**

loss	MSE			SSIM		
	Denmark	Hamburg	Taipei	Denmark	Hamburg	Taipei
MSE	<b>0.0912</b>	0.1146	<b>21.685</b>	0.4652	0.4760	0.5761
Pix2Pix	0.0976	0.1116	21.856	0.4536	0.4771	0.5508
FEAT	0.0977	<b>0.0919</b>	22.019	<b>0.4658</b>	<b>0.5497</b>	<b>0.5897</b>

## 4.3 Comparing Number of Time Steps

Next, we analyzed the effect of varying the number of input time steps. We reduced the number of steps by removing the temporal distant steps until the desired amount was obtained. The results can be seen in Table 2. The MSE results exhibit a tendency that more steps are generally better, but when all steps are used, the MSE results are slightly worse. With an increasing number of available time steps, the SSIM performance increased, which suggests that for all three tile locations, more input data yielded better results. This makes sense since fewer time steps induce a higher chance of not covering missing or cloudy parts of the image. Further, this indicates that static approaches, such as [27], would miss crucial information to create distortion free imagery.

**Table 2: Analysis if using more time steps (# steps) results in lower a MSE and higher a SSIM for TimeGate.**

# steps	MSE			SSIM		
	Denmark	Hamburg	Taipei	Denmark	Hamburg	Taipei
1	0.4249	0.3227	21.961	0.1713	0.1460	0.1916
3	0.0997	0.2126	<b>21.859</b>	0.3772	0.3138	0.3820
6	0.0868	0.2055	22.061	0.3988	0.3185	0.4825
12	<b>0.0740</b>	0.1031	21.932	0.4369	0.5141	0.5601
all	0.0977	<b>0.0919</b>	22.019	<b>0.4658</b>	<b>0.5497</b>	<b>0.5897</b>

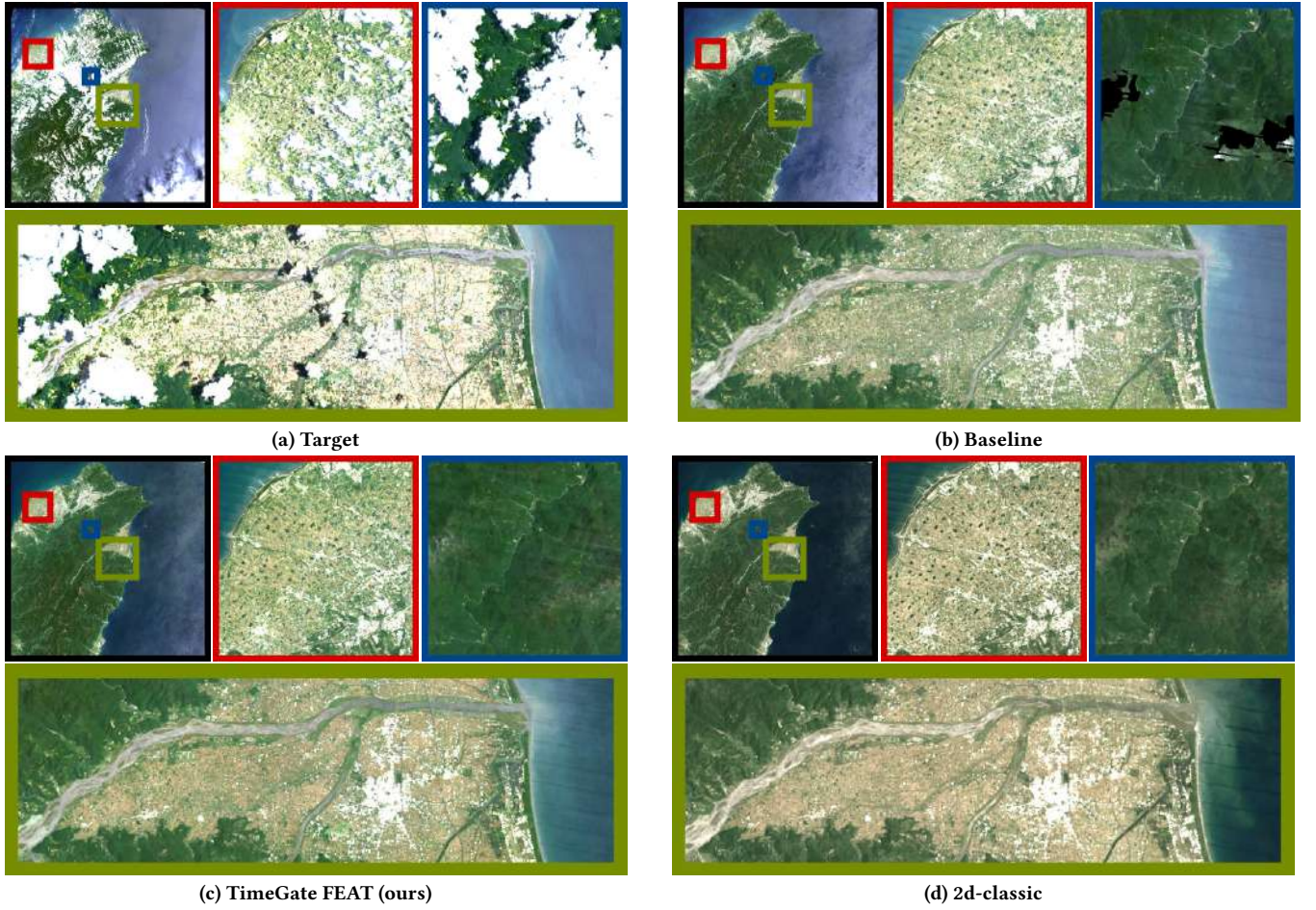


Figure 5: Model output for one of the Taipei test tiles. It shows different sections, which are marked with different colored frames, of the complete tile for the different models in each subfigure.

#### 4.4 Quantitative Model Comparisons

Next, we compared different models against the baseline in Table 3. Our proposed TimeGate model trained with FEAT outperformed the baseline w.r.t. the SSIM on the Denmark and the Hamburg tile locations. The magnitude of the MSE of all models for Taipei was significantly higher than for the other two tile locations for all models. We suspect that the bright city areas produce outliers (large values) and the magnitude of values fluctuates much between time steps. Our model generally performed better than the classic time layer approach (2d/3d-classic). It is noteworthy that due to the missclassified cloud coverage, it was not possible to quantitatively evaluate our model at interesting locations where the baseline could not predict at all or produced artifacts. This renders an additional qualitative comparison necessary.

#### 4.5 Qualitative Model Comparisons

Now, we discuss the qualitative results by presenting model outputs for three of the test tiles in Figure 3, 4, and 5. In general, the deep learning models were able to distinguish between real clouds and mistakes caused by the FMASK algorithm (see, e.g., the beach in the green boxed image in Figure 3, missing building in the red

Table 3: Comparison of the TimeGate model (ours) and traditional approaches (2d/3d-classic). Lower MSE and higher SSIM values are better, respectively. Evaluation was only possible on areas that were classified as cloud-free.

model	MSE			SSIM		
	Denmark	Hamburg	Taipei	Denmark	Hamburg	Taipei
Baseline	<b>0.0851</b>	0.1436	<b>21.883</b>	0.4507	0.4800	<b>0.6035</b>
2d-classic	0.1229	0.1302	21.979	0.4114	0.4648	0.5481
3d-classic	0.1228	0.2599	22.051	0.4098	0.3976	0.4435
<b>TimeGate</b>	0.0977	<b>0.0919</b>	22.019	<b>0.4658</b>	<b>0.5497</b>	0.5897

boxed image in Figure 4, or mountain ranges in blue boxed image in Figure 5). The baseline algorithm was not able to fill some areas and even introduced artifacts from clouds (e.g., the cloud artifacts in blue boxed images in Figure 3 and 4). The 2d-classic model tended to paint everything green or grey and thus lowering variations and miss-coloring objects (e.g., buildings in the red boxed images in Figure 3 and 4 as well as in the green boxed images in Figure 5). While the geography in Taipei (a mega-city surrounded by mountains) was not present in the training data (Denmark), the appearance of

our preferred model (TimeGate FEAT) was still fair and less affected by artifacts than the baseline.

## 5 CONCLUSION

We propose a deep learning approach to create cloud- and distortion-free imagery from temporal data. Our three-phase model first calculates temporal features from different time steps, then fuses the resulting feature maps via time gating mechanism, and finally employs the U-Net architecture to post-process these temporal features to obtain one coherent image. The aggregation via the time layer is conditioned on cloud masks from the target time during training, but these masks are not required during testing. Our experimental results suggest a better performance w.r.t. the SSIM and on three tile locations from different areas (Denmark, Germany, and Taiwan) and also shows clear visual improvements over the baseline algorithm. Our results are critical for applied research in the field of remote sensing. Despite the harmonization of existing Landsat archives, the number of cloud-free observations, particularly, for the years with non-overlapping Landsat satellites, can be low in areas prone to clouds, such as Central Europe, Central Africa, South America, South-East Asia and sub-Arctic regions [17]. Therefore, the deep image processing and more advanced thematic information extraction are possible solutions to benefit from sometimes patchy, cloud-contaminated and distorted satellite data archives.

## ACKNOWLEDGMENTS

Stefan Oehmcke and Fabian Gieseke acknowledge support by the Villum Foundation through the project Deep Learning and Remote Sensing for Unlocking Global Ecosystem Resource Dynamics (DeReEco).

## REFERENCES

- [1] Zhe Zhu. 2019. The US government might charge for satellite data again? Here's why that would be a big mistake. The Conversation, <http://theconversation.com/the-us-government-might-charge-for-satellite-data-again-heres-why-that-would-be-a-big-mistake-113021>.
- [2] A Arbelles and T R Raviv. 2019. Microscopy cell segmentation via convolutional LSTM networks. In *Int. Symp. on Biomed. Imaging (ISBI)*. IEEE, 1008–1012.
- [3] Asim Banskota, Nilam Kayastha, Michael J Falkowski, Michael A Wulder, Robert E Froese, and Joanne C White. 2014. Forest monitoring using Landsat time series data: a review. *Canadian Journal of Remote Sens.* 40, 5 (2014), 362–384.
- [4] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. 2018. Recurrent neural networks for multivariate time series with missing values. *Scientific reports* 8, 1 (2018), 1–12.
- [5] Jin Chen, Xiaolin Zhu, James E. Vogelmann, Feng Gao, and Suming Jin. 2011. A simple and effective method for filling gaps in Landsat ETM+ SLC-off images. *Remote Sens. of Env.* 115, 4 (April 2011), 1053–1064.
- [6] Tzu-Hsin Karen Chen, Alexander V Prishchepov, Rasmus Fensholt, and Clive E Sabel. 2019. Detecting and monitoring long-term landslides in urbanized areas with nighttime light data and multi-seasonal Landsat imagery across Taiwan from 1998 to 2017. *Remote Sens. of environment* 225 (2019), 317–327.
- [7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR abs/1412.3555* (2014).
- [8] N Flood. 2013. Seasonal Composite Landsat TM/ETM+ Images Using the Medoid (a Multi-Dimensional Median). *Remote Sens.* 5, 12 (Dec. 2013), 6481–6500.
- [9] P Griffiths, S van der Linden, T Kuemmerle, and P Hostert. 2013. A Pixel-Based Landsat Compositing Algorithm for Large Area Land Cover Mapping. *Journal of Selected Topics in Appl. Earth Obs. and Remote Sens.* 6, 5 (Oct. 2013), 2088–2101.
- [10] Claas Grohnfeldt, Michael Schmitt, and Xiaoxiang Zhu. 2018. A conditional generative adversarial network to fuse sar and multispectral optical data for cloud removal from sentinel-2 images. In *Int. Geoscience and Remote Sens. Symp. (IGARSS)*. IEEE, 1726–1729.
- [11] Brent N Holben. 1986. Characteristics of maximum-value composite images from temporal AVHRR data. *Int. journal of Remote Sens.* 7, 11 (1986), 1417–1434.
- [12] Huabing et al. Huang. 2017. Mapping major land cover dynamics in Beijing using all Landsat images in Google Earth Engine. *Remote Sens. of Env.* 202 (2017), 166–176.
- [13] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. 2017. Image-to-Image Translation with Conditional Adversarial Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 5967–5976.
- [14] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 2012. 3D convolutional neural networks for human action recognition. *Trans. on pattern analysis and machine intelligence* 35, 1 (2012), 221–231.
- [15] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual losses for real-time style transfer and super-resolution. In *Europ. conf. on computer vision (ECCV)*. Springer, 694–711.
- [16] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR abs/1412.6980* (2014).
- [17] V Kovalsky and David P Roy. 2013. The global availability of Landsat 5 TM and Landsat 7 ETM+ land surface observations and implications for global 30 m Landsat data product generation. *Remote Sens. of Env.* 130 (2013), 280–293.
- [18] Salim Malek, Farid Melgani, Yakoub Bazi, and Naif Alajlan. 2017. Reconstructing cloud-contaminated multispectral images with contextualized autoencoder neural networks. *Trans. on Geoscience and Remote Sens.* 56, 4 (2017), 2270–2282.
- [19] Brian L Markham, James C Storey, Darrel L Williams, and James R Irons. 2004. Landsat sensor performance: history and current status. *IEEE Trans. on Geoscience and Remote Sens.* 42, 12 (2004), 2691–2694.
- [20] Diganta Misra. 2019. Mish: A Self Regularized Non-Monotonic Neural Activation Function. *CoRR abs/1908.08681* (2019).
- [21] Charlotte Pelletier, Geoffrey I Webb, and François Petitjean. 2019. Temporal convolutional neural network for the classification of satellite image time series. *Remote Sens.* 11, 5 (2019), 523.
- [22] Francesco Pirotti, Filiz Sunar, and M. Piragnolo. 2016. Benchmark of Machine Learning Methods for Classification of a Sentinel-2 Image. *Int. Journal of Geo-Information (ISPRS) XLI-B7* (06 2016), 335–340.
- [23] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention*. Springer Int. Publishing, 234–241.
- [24] D.P. et al. Roy. 2014. Landsat-8: Science and product vision for terrestrial global change research. *Remote Sens. of Env.* 145 (April 2014), 154–172.
- [25] Marc Rußwurm and Marco Körner. 2018. Multi-temporal land cover classification with sequential recurrent encoders. *Int. Journ. of Geo-Inf.* 7, 4 (2018), 129.
- [26] Stephen Sagar, Dale Roberts, Biswajit Bala, and Leo Lymburner. 2017. Extracting the intertidal extent and topography of the Australian coastline from a 28 year time series of Landsat observations. *Remote Sens. of Env.* 195 (2017), 153–169.
- [27] Vishnu Sarukkai, Anirudh Jain, Burak Uzkent, and Stefano Ermon. 2020. Cloud removal from satellite images using spatiotemporal generator networks. In *The IEEE Winter Conference on Applications of Computer Vision*. 1796–1805.
- [28] Wenzhe Shi, Jose Caballero, Ferenc Husz á r, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. 2016. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Conf. on computer vision and pattern recognition (CVPR)*. 1874–1883.
- [29] Praveer Singh and Nikos Komodakis. 2018. Cloud-gan: Cloud removal for sentinel-2 imagery using a cyclic consistent generative adversarial networks. In *Int. Geoscience and Remote Sens. Symp. (IGARSS)*. IEEE, 1772–1775.
- [30] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems (NeurIPS)*. 3483–3491.
- [31] X Song, J Sexton, C Huang, S Channan, and J Townshend. 2016. Characterizing the magnitude, timing and duration of urban growth from time series of Landsat-based estimates of impervious cover. *Remote Sens. of Env.* 175 (2016), 1–13.
- [32] M Tan, B Chen, R Pang, V Vasudevan, M Sandler, A Howard, and Q V Le. 2019. Mnasnet: Platform-aware neural architecture search for mobile. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2820–2828.
- [33] Ashish et al. Vaswani. 2017. Attention is all you need. In *Advances in neural information processing systems (NeurIPS)*. 5998–6008.
- [34] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. on image processing* 13, 4 (2004), 600–612.
- [35] Michael A. et al. Wulder. 2016. The global Landsat archive: Status, consolidation, and direction. *Remote Sens. of Env.* 185 (Nov. 2016), 271–283.
- [36] Y Xu, Q Kong, Q Huang, W Wang, and M D Plumbley. 2017. Convolutional gated recurrent neural network incorporating spatial features for audio tagging. In *Int. Joint Conf. on Neural Networks (IJCNN)*. IEEE, 3461–3466.
- [37] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. 2019. Self-Attention Generative Adversarial Networks. In *Int. Conf. on Machine Learning (ICML)*, Vol. 97. PMLR, 7354–7363.
- [38] Zhe Zhu, Shixiong Wang, and Curtis E Woodcock. 2015. Improvement and expansion of the Fmask algorithm: Cloud, cloud shadow, and snow detection for Landsats 4–7, 8, and Sentinel 2 images. *Remote Sens. of Env.* 159 (2015), 269–277.
- [39] Zhe Zhu and Curtis E. Woodcock. 2012. Object-based cloud and cloud shadow detection in Landsat imagery. *Remote Sens. of Env.* 118 (March 2012), 83–94.