IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 22 May 2023, accepted 26 June 2023, date of publication 6 July 2023, date of current version 13 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3292966

RESEARCH ARTICLE

WindSR: Improving Spatial Resolution of Satellite Wind Speed Through Super-Resolution

ASHUTOSH KUMAR[1], TANVIR ISLAM[2,3], JUE MA[1], TAKEHIRO KASHIYAMA[4], YOSHIHIDE SEKIMOTO[1], AND CHRIS MATTMANN[2]

[1]Institute of Industrial Science, The University of Tokyo, Tokyo 153-8505, Japan
[2]Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109, USA
[3]Okta Inc., Seattle, WA 98004, USA
[4]Faculty of Economics, Osaka University of Economics, Osaka 533-8533, Japan

Corresponding author: Ashutosh Kumar (ashutosh@iis.u-tokyo.ac.jp)

This work was supported by JST SPRING under Grant JPMJSP2108.

ABSTRACT Prediction of accurate wind speed is necessary for a variety of applications such as energy production, agriculture, climate modeling, and weather forecasting. Various satellites orbiting the earth measure the wind speed, which is particularly useful as they provide measurements of wind speed over large areas and in remote locations that might be difficult to measure using other methods. However, satellite-based wind speed measurements have relatively low spatial resolution compared to other methods, such as ground-based radar. In this research, we develop WindSR and a lightweight tiny-WindSR to improve the resolution of satellite wind speed data by four times from the NASA's GEOS-5 Nature Run dataset. WindSR has SRResNet-based architecture consisting of several Residual-in-Residual Dense Blocks to compute features from low spatial resolution (28 km) wind speed for upscaling. We train WindSR with more than 20,000 pairs of low-resolution (28 km) and corresponding high-resolution (7 km) wind speed data and evaluate its performance on the validation set consisting of 2,102 wind speed images. Experimental results show that WindSR outperforms classical upsampling algorithms, such as Bicubic interpolation and Lanczos interpolation by 17.89% and general-purpose super-resolution GANs such as BSRGAN and SwinIR by up to 11.35% on the RMSE metric. The dataset developed in this research is publicly available at: https://github.com/sekilab/WindSR_Dataset.

INDEX TERMS CNN, GAN, super-resolution, wind speed.

I. INTRODUCTION

As one of the most important renewable energy sources, wind is an inexhaustible, no-cost, and sustainable energy source. It has become one of the focus of energy studies due to the availability of megawatt-size wind machines, accessible management facilities, ease of maintenance, tax benefits, and government subsidies [1]. Wind, however, is not a controllable energy that can be scheduled and planned and the prediction and calculation of wind speed are essential for a variety of applications, such as energy production, weather forecasting, agriculture, transportation, etc.

Large-scale wind speed on the earth can be measured mainly by two techniques, such as ground-based radar and earth-observing satellites. Ground-based radar measures the wind speed by sending out radio waves that bounce off the atmospheric particles. By measuring the Doppler shift [2] of the returned signal, wind speed is calculated. Ground-based radar techniques, however, cannot measure wind speed in remote locations [3]. Satellite-based techniques use a similar principle to measure wind speed and other atmospheric variables [4] from earth-observing satellites. They are particularly useful in covering a large area and remote locations, where ground-based radar cannot be used. Compared to ground-based radar techniques that generally have a high resolution of less than one kilometer, satellite-based wind speed measurement techniques have a low spatial

The associate editor coordinating the review of this manuscript and approving it for publication was Gerardo Di Martino.

resolution, ranging from hundreds of meters to tens of kilometers [3].

Deep learning techniques, particularly, convolutional neural networks have been shown to be successful in a variety of tasks such as object detection, image classification, image generation using GANs, etc [5]. Recently, transformers have also been shown to be successful for image processing tasks [6], [7], [8], [9]. Such deep learning techniques for image super-resolution are useful since they can learn from a large amount of training dataset and generalize well on the unseen/test dataset. A wide range of techniques have been developed utilizing GANs to reconstruct a high-resolution image from a low-resolution image [10], [11], [12], [13], [14], [15], [16], [17].

Present methods for image super-resolution tasks mainly focus on super-resolution problems for standard real-world images. Some studies [18] have explored the use of other types of data, such as wind speed and solar data limited to benchmarking existing super-resolution techniques [12], [19]. Existing super-resolution techniques utilizing GANs consider natural image, which has textures and shapes. Satellite data, on the other hand, have different properties than natural images (e.g., no regular texture or shapes) and the use of such satellite data for super-resolution tasks has not been fully explored.

In this research, we develop WindSR and tiny-WindSR based on the SRResNet architecture [12], [14] to improve the resolution of wind speed by four times from 28 km to 7 km using the satellite wind speed data from the NASA's GEOS-5 Nature Run dataset. Specifically, we

- Develop a large-scale wind speed dataset consisting of more than 20,000 pairs of Low Resolution-High Resolution (LR-HR) images for super-resolution from NASA's GEOS-5 Nature Run dataset
- Propose an SRResNet-based architecture – WindSR as well as a small version of it, tiny-WindSR for real-time applications to upscale wind speed from 28 km spatial resolution to 7 km
- Compare the accuracy of the proposed architecture with classical upsampling techniques as well as state-of-the-art GAN-based algorithms, such as BSRGAN [20] and SwinIR [21] on a validation set consisting of more than 2,100 wind speed images
- Understand the performance of discriminator networks and loss metrics for remote sensing data, such as wind speed, compared to standard three-channel real-world images

II. METHODOLOGY

A. GEOS-5 NATURE RUN DATASET

The GEOS-5 Nature Run dataset [4] contains gridded output files at a resolution of 7 km from a non-hydrostatic, global mesoscale simulation produced by the non-hydrostatic version of the GEOS-5 Atmospheric Global Climate Model (AGCM). The GEOS-5 Nature Run dataset offers two resolutions: reduced-resolution and full-resolution.

TABLE 1. Full-resolution GEOS-5 Nature Run dataset details.

Attribute	Value
Frequency	30-minute from 23:30 UTC
Spatial Grid	2D
Resolution	Longitude=5760, Latitude = 2881 and Time = 1
File type	NetCDF-4
Total variables	75
Collection time	May 2005 to June 2007

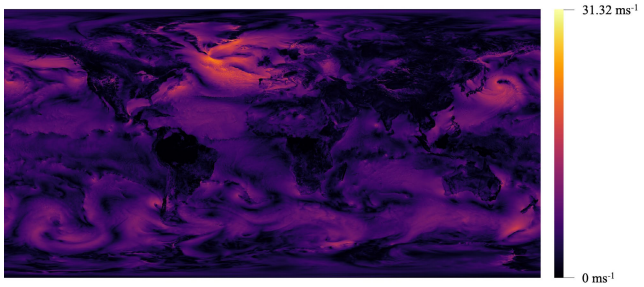


FIGURE 1. We calculate the wind speed for each grid/pixel using the 2-meter eastward and northward component in the GEOS-5 Nature Run dataset. The resolution of the dataset is 7 km.

A reduced-resolution dataset contains hourly data intervals, while a full-resolution dataset contains data at 30-minute intervals. In this study, we make use of the full-resolution dataset for conducting research studies, which has a frequency of 30 minutes and starts from 23:30 UTC. The GEOS-5 Nature Run dataset contains a total of 75 variables such as cloud top pressure, cloud top temperature, total precipitation, snowfall, northward wind, eastward wind, etc. The values for those variables are contained in the grid with a resolution of 7 km. The description of the full-resolution GEOS-5 Nature Run dataset is shown in Table 1. As the GEOS-5 Nature Run dataset is available for several months at intervals of 30 minutes, we randomly download the dataset for different days and times to ensure a wide range of wind speeds for better generalization, and process it as explained in the next subsection.

B. PREPARATION OF DATASET

From the GEOS-5 Nature Run dataset, we consider two variables U2M and V2M, which are 2-meter eastward and 2-meter northward wind, respectively. The unit of both U2M and V2M is ms^{-1}. Using U2M and and V2M components, we calculate the wind speed for each grid as, $\mathcal{W} = \sqrt{U2M^2 + V2M^2}$. The visualization of the wind speed is shown in Figure 1. From Figure 1, we find that the wind speed varies greatly across the globe with some places having wind speed above 30 ms^{-1}. We also notice that the wind speed in oceans is particularly high compared to the land.

From the calculated wind speed shown in Figure 1, we extract patches/crops of resolution 400 × 400 in such a way that no two patches intersect in any way and are exclusive. We then use bilinear interpolation [22], [23] to down-sample the crops to 100 × 100. Since the original

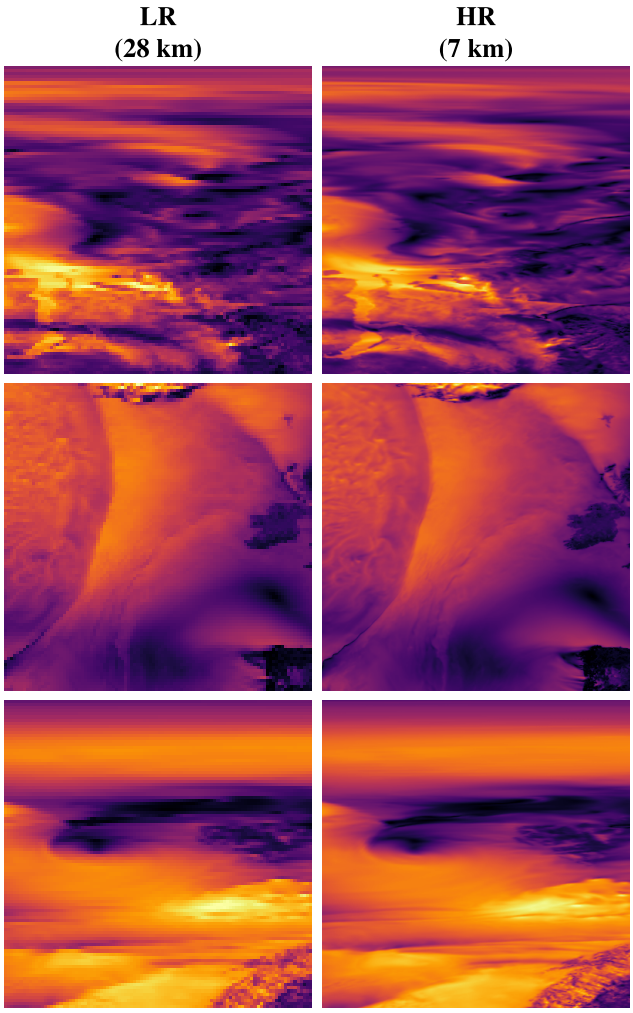


FIGURE 2. Examples of low-resolution (LR) at 28 km and high-resolution (HR) at 7 km wind speed present in the training dataset. LR wind images are scaled four times to match the same dimensions as HR images for visualization.

resolution of the wind speed is 7 km, the resolution after 4× down-sampling is 28 km. The 7 km high-resolution (HR) and the corresponding 28 km low-resolution (LR) pairs are, then, used to train the super-resolution resolution model. In total, we prepare 20,046 pairs (LR-HR) for training and independently prepare another 2,102 pairs (LR-HR) for validation. To consider the robustness of the approach, the 2,102 pairs for validation are prepared from different dates and times used for training. Figure 2 shows the visualization of the LR-HR pair.

While it is common to use the compressed form of data representations such as JPEG, PNG, etc. in general computer vision tasks for super-resolution, the same is not applicable in this problem since the value of each pixel represents wind speed value, and retrieval of exact value is necessary. Thus, we need a lossless format to save all the wind speed values in raw format without any loss of information. For this reason, we save the LR-HR wind speed values in raw numpy [24] format.

C. DEVELOPMENT OF WINDSR

1) WINDSR AND TINY-WINDSR ARCHITECTURE

WindSR and tiny-WindSR have SRResNet-based architecture [14] (LR→ CONV2D→ RESIDUAL BLOCKS → CONV2D→ UPSAMPLE→ CONV2D→ HR), which consists of multiple residual blocks [25] to upsample low-resolution images. We replace the residual blocks in the generator of the SRResNet with Residual-in-Residual-Dense-Blocks (RRDB) [12] for upscaling lower resolution data to higher resolution, as shown in Figure 3. Figure 3 shows that WindSR and tiny-WindSR compute the maximum feature in the LR space through RRDB blocks before upsampling. Both WindSR and tiny-WindSR have 128 filters in each of the convolutional layers in the generator, including those inside the RRDB blocks. tiny-WindSR consists only of one RRDB block and 128 filters in each of the convolutional layers. Being very lightweight, tiny-WindSR can be used to upscale input wind speed for real-time applications.

Both WindSR and tiny-WindSR do not contain batch normalization layers after convolution blocks that have been seen to generate artifacts in the image [12]. The effect of batch normalization is presented in the results section.

2) LOSS FUNCTION

During the development of the WindSR and tiny-WindSR architecture, we consider two common types of loss functions in image generation, which are pixel losses ($\mathcal{L}^1$ and $\mathcal{L}^2$) and content loss ($\mathcal{L}^C$). In pixel loss, image differences are measured at the pixel level, and high-level features such as texture and structure are not taken into account, which may result in extremely smooth and blurry images. $\mathcal{L}^1$ pixel loss considers the absolute difference between pixel values of the generated and ground truth image, as shown in Eq. 1, while $\mathcal{L}^2$ pixel loss considers the squared difference between the pixel values, as shown in Eq. 2.

$$\mathcal{L}^1 = \mathbb{E}\left[\left|\mathcal{G}(LR^i) - HR^i\right|\right] \tag{1}$$

$$\mathcal{L}^2 = \mathbb{E}\left[\left(\mathcal{G}(LR^i) - HR^i\right)^2\right] \tag{2}$$

In Eq. 1 and Eq. 2, $\mathcal{G}(LR^i)$ represents the high-resolution wind speed image (7 km) generated by the generator ($\mathcal{G}$) for the input low-resolution LR^i at 28 km. HR^i corresponds to the ground-truth high-resolution wind speed data at 7 km for the low-resolution wind speed LR^i.

To consider high-level features, content loss ($\mathcal{L}^C$) is used, which measures the difference between the feature representations of the generated image and the ground truth image, typically obtained from a pre-trained convolutional neural network, such as VGG-19 [26], as shown in Eq. 3. Content loss is more effective in preserving the global structure and texture of the image, and can produce more visually appealing results than pixel loss.

$$\mathcal{L}^C = \mathbb{E}\left[\left|\mathcal{F}^{VGG^4}\left(\left(\mathcal{G}(LR^i)\right) - \mathcal{F}^{VGG^4}\left(HR^i\right)\right|\right] \tag{3}$$

TABLE 2. Hyperparameters during training.

Hyperparameter	Values
Learning rate	0.001, 0.0001
Residual scaling	0.1, 0.2, 0.5
RRDB	1, 4, 16, 48, 128, 256
Filters	1, 4, 16, 64, 128
Loss	Pixel, Content
Discriminator	True/False
Batch Normalization (BN)	True/False

In Eq. 3, $\mathcal{F}^{VGG^4}((\mathcal{G}(LR^i))$ refers to the feature map obtained for the generated high-resolution image ($\mathcal{G}(LR^i)$) image by the 4^{th} convolution layer in the VGG-19 network. Similarly, $\mathcal{F}^{VGG^4}(HR^i)$ refers to the feature map obtained for the ground-truth high-resolution (HR^i) wind speed at 7 km resolution.

3) HYPERPARAMETERS AND TRAINING DETAILS

We train WindSR with different hyperparameters for 300 epochs on Tesla A100 GPUs [27] and consider various hyperparameters, as shown in Table 2. The discriminator during the experiment has the same architecture as presented in [12]. Residual scaling [19], [28] in Table 2 refers to scaling down the residual values in the RRDB blocks before adding to the main path to prevent instability of training very deep neural networks. It is important to note here that for the calculation of content loss ($\mathcal{L}^C$), we use the pre-trained VGG network on the ImageNet dataset. Since the input has three channels in the VGG dataset, as compared to the one channel present in the wind speed dataset, we simply repeat the wind speed values along three axes for both the generated image and ground truth image and compare the feature difference in the intermediate VGG layer.

D. BASELINE METHODS

For the comparison of the proposed models, we use four classical upsampling techniques and two GAN-based techniques utilizing the state-of-the-art convolutional neural network and transformer backbones.

1) CLASSICAL UPSAMPLING TECHNIQUES

a: NEAREST NEIGHBOR

Nearest neighbor upsampling is one of the simplest upsampling techniques that copies the value of the nearest pixel to increase the resolution of the images. It is also known as the pixel replication method and can preserve sharp image details, but it can substantially introduce non-uniform texture to images that were previously smooth.

b: BILINEAR

The bilinear sampling method estimates each new pixel in the upsampled image by averaging the four nearest pixels in the original image. The weights for each pixel are determined by the distance between the new pixel and each of its nearest neighbors.

c: BICUBIC

In bicubic interpolation, cubic splines or other polynomial techniques are used to upsample images. The new pixels in the upsampled image are assigned values based on a weighted average of the nearest 16 pixels in the original image, according to a cubic polynomial function that takes into account the distance between the new pixel and its neighbors. Compared to the standard bilinear algorithm, it preserves fine detail more effectively.

d: LANCZOS

Lanczos method is helpful when the upscaled image is used to identify features or boundaries. Edge and linear elements may be detected more easily with Lanczos resampling. During upsampling, zeros are inserted between the pixels of the original image, and then the Lanczos kernel Eq. 4 is used to interpolate the missing values.

$$L(x) = \begin{cases} \text{sinc}(x) \cdot \text{sinc}(x/a) & \text{if } |x| \leq a \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

In Eq. 4, $\text{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$, and a refers to the width of the kernel. We consider 8×8 pixels neighborhood for the Lanczos interpolation.

2) GAN-BASED DEEP LEARNING TECHNIQUES

Classical upsampling techniques have several limitations, such as the introduction of artifacts, lack of details, and irregular quality due to fixed rules of algorithms. A deep learning-based upsampling technique upsample images several times without losing quality due to its generalization by training on a large-scale dataset. In this research, we make use of two deep learning upsampling techniques for comparison with our developed WindSR. It should be noted that we consider the real wind speed values using the raw image formats instead of using lossy formats, such as JPG used for training standard images.

a: BSRGAN

All super-resolution algorithms first degrade the high-resolution image to low-resolution using techniques, such as bicubic interpolation, bilinear interpolation, etc. and try to reconstruct the high-resolution image from the degraded image. Blind Super Resolution with Generative Adversarial Networks (BSRGAN) [20] introduces novel techniques to degrade high-resolution images to low-resolution using more practical approaches for blurring, downsampling and introducing noise compared to traditional techniques such as bicubic interpolation commonly used in downsampling images. BSRGAN tries to reduce the mismatch between the assumed degradation model, such as bicubic compared to how the images are degraded in the real-world scenario.

The BSRGAN architecture utilizes the widely used ESRGAN [12] architecture to train a PSNR-oriented [29] BSRNet model, followed by a perceptual quality-oriented BSRGAN model. The PSNR-oriented model focuses mainly

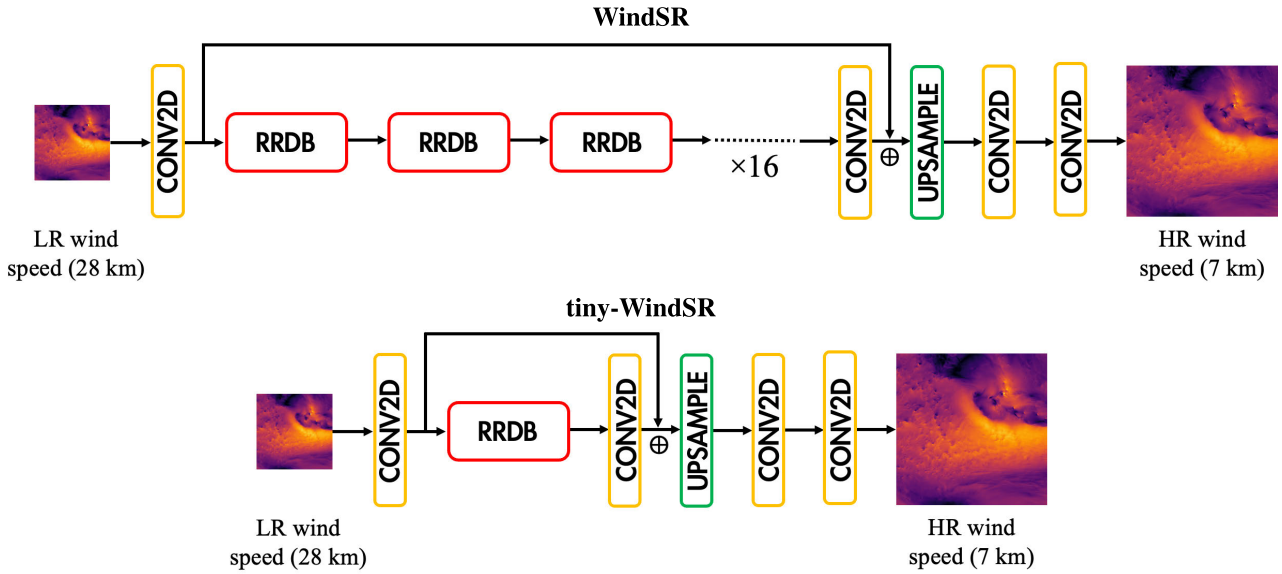


FIGURE 3. The network architecture of WindSR and tiny-WindSR consists of 16 and 1 RRDB modules, respectively to compute features from low-resolution wind speed at 28 km. Based on the experiment results, we remove the batch normalization layer (BN) from the RRDB block. The RRDB layer consists of CONV2D→ReLU→CONV2D blocks and BN is not used after convolution, as also presented in [12]. Both WindSR and tiny-WindSR have 128 filters in each of the CONV2D layers.

on pixel-level reconstruction and for real-world images is less practical, as it leads to the smoothing of the image. From the perspective of wind speed super-resolution, pixel-level values are significant rather than the natural perceptual quality of the image. For this reason, in our experiment, we only report PSNR-oriented BSRNet model results.

b: SWINIR

While the majority of super-resolution techniques use the convolutional neural network, SwinIR [13] utilizes transformer networks to achieve the same goal. SwinIR uses a multi-scale strategy to process images at different levels of detail, such as shallow feature extraction, deep feature extraction, and high-resolution image reconstruction. Each level is processed by a separate instance of the swin transformer [21]. In particular, the deep feature extraction module consists of residual swin transformer blocks. The residual swin transformer has a convolutional layer at the top, which enhances the translational equivariance of SwinIR [13]. The residual connection allows the aggregation of different levels of features using identity-based connections between different blocks and the reconstruction module. SwinIR has been shown to outperform state-of-the-art super-resolution techniques on several real-world data sets such as DIV2K [30], ImageNet [31], and Flickr2K [32].

Similar to the BSRGAN, we train PSNR-oriented SwinIR without any discriminator network with the GEOS-5 Nature Run dataset [4] since our target is not on the visual appearance of the image, but the values of individual pixels.

III. RESULTS

We evaluate the performance of various upsampling techniques on the validation dataset consisting of 2,102 wind speed images using RMSE metric and present the results in Table 3. In Table 3, we report BSRGAN and SwinIR results only for the pixel-wise loss since having a discriminator increases the pixel-wise loss further. From Table 3, we notice that WindSR achieves the lowest RMSE on the validation set. Classical upsampling techniques, such as the nearest neighbor have a large RMSE compared to deep learning-based algorithms, such as BSRGAN, SwinIR, etc. From Table 3, we notice that tiny-WindSR also outperforms BSRGAN and SwinIR models.

TABLE 3. Comparison of RMSE values of various upsampling algorithms on the validation dataset.

Models	RMSE
Nearest neighbor	0.352
Bilinear	0.259
Bicubic	0.257
Lanczos	0.264
BSRGAN (PSNR only)	0.238
SwinIR (PSNR only)	0.254
tiny-WindSR	**0.222**
WindSR	**0.211**

In Table 4, we show the results of various hyperparameters on the performance of WindSR. We find that changing pixel loss to L2 in the case of WindSR_V1 increases the RMSE loss slightly. Decreasing the number of filters to 16 in WindSR_V2 and having a batch normalization in the convolutional layers in WindSR_V3 increases the RMSE loss by 9% in both cases compared to WindSR.

From Table 4, we notice that having a discriminator network introduces a significant increase in the RMSE value. Further, the introduction of content loss on the extracted features from a pre-trained VGG-19 network increases RMSE

TABLE 4. RMSE on the validation dataset and GFLOPs of the generator with various hyperparameters. Training a discriminator, if present, is done using content loss, while training a generator is done using pixel loss.

Model name	RRDB	Filters	Loss	BN	Discriminator	GFLOPs	RMSE
WindSR	16	128	L1 (Pixel)	No	No	1,117	**0.2107**
WindSR_V1	16	128	**L2 (Pixel)**	No	No	1,117	0.2168
WindSR_V2	16	**16**	L1 (Pixel)	No	No	17.53	0.2296
WindSR_V3	16	128	L1 (Pixel)	**Yes**	No	1,117	0.2297
WindSR_V4	16	128	**L1 (Pixel/Content)**	No	**Yes**	1,117	**0.3954**
WindSR_V5	16	128	L1 (Pixel/Content)	No	No	1,117	**0.3623**
WindSR_V6	16	128	**L1 (Pixel)**	No	**Yes**	1,117	0.5113
tiny-WindSR	**1**	128	L1 (Pixel)	No	No	128.19	**0.2217**

loss significantly, even in the absence of the discriminator network. This could mean that content loss increases the RMSE, rather than the discriminator itself in WindSR_V4. However, this is not true, because when we remove the content loss and keep the discriminator in the case of WindSR_V6, the RMSE increases by 29.21% compared to WindSR_V4 with both pixel and content loss.

In the case of tiny-WindSR, even after reducing the number of RRDB to one, the RMSE does not increase considerably, but merely 5.22%. This is significant because tiny-WindSR has an 88.52% reduction in the number of parameters to 128.19 GFLOPs compared to the original WindSR with 1,117 GFLOPs.

Figure 4 shows the comparison results of the high-resolution images generated using WindSR and tiny-WindSR (7 km) from low-resolution images at 28 km resolution compared to the ground truth (7 km). We find that generated wind speed looks close to the ground truth for a range of wind speed values, but with slightly smooth edges and overestimated values.

IV. DISCUSSIONS

From the results presented in Table 3, we notice that both tiny-WindSR and WindSR significantly outperform classical upsampling techniques and GAN-based super-resolution algorithms. GAN-based algorithms, such as BSRGAN, and SwinIR have lower RMSE compared to classical upsampling algorithms as they can learn more complex features of low-resolution images for upsampling through training on a large amount of data sets.

From Table 4, we notice that WindSR has the lowest RMSE on the validation dataset compared to other models. Changing the pixel-wise loss function during training to L2 (WindSR_V1) increases the RMSE on the validation set, which suggests that the absolute difference in the value of pixels (L1) helps minimize the pixel-wise error. WindSR_V2 with a reduced number of filters has a higher value of RMSE. tiny-WindSR, however, with a reduced number of RRDB has a smaller increment in the RMSE value compared to WindSR_V2, which suggests that the number of features in a shallow network is more important compared to the deeper network with several RRDB blocks. The introduction of batch normalization (BN) layers in WindSR_V3 increases the RMSE by approximately 9% compared to the original network. This phenomenon is also observed in super-resolution real-world images as they lead to the occurrence of artifacts [12] in the generated images.

WindSR_V4 is based on the real-world super-resolution GANs [12], [14], which has both a generator and a discriminator. Apart from the pixel loss, it also has content loss which is obtained by calculating features from an intermediate VGG-19 layer. The introduction of the discriminator, however, significantly increases the RMSE loss on the validation set by 73.33%. In the super-resolution of wind speed, the primary goal is to only learn the values inside each pixel. This is, however, not the case for real-world images because the generated image should look more natural to human eyes compared to the values of individual pixels. A discriminator may increase the pixel loss value, however, it may help with other metrics (such as Perceptual Index [33]) to make the generated image look more natural. WindSR_V5 with the introduction of content loss and no discriminator performs better than WindSR_V4 but has significantly higher RMSE compared to the original WindSR. This is because content loss is calculated by computing features extracted from the VGG-19 layer, which is trained using the ImageNet dataset [31] and the computed high-level features are not a good representative of the wind speed data and are not necessary.

Interestingly, we notice that if we remove the content loss and keep only the L1 pixel loss with the discriminator in the case of WindSR_V6, the RMSE increases significantly by 29.31% compared to WindSR_V4. Thus, the loss function must incorporate high-level features for training the discriminator.

The tiny-WindSR model contains only one residual block and 128 filters in each of the convolutional layers, making it a lightweight model. However, the increase in the RMSE loss is 5.22% compared to a decrease in the GFLOPs over 88.52%. This clearly shows that we do not need to have a very deep network for super-resolution tasks that require working on data (e.g., satellite wind speed, precipitation) where the value of the individual pixel is important compared to the overall perception of the visual appearance. tiny-WindSR could be helpful for increasing the resolution of wind speed models in real-time.

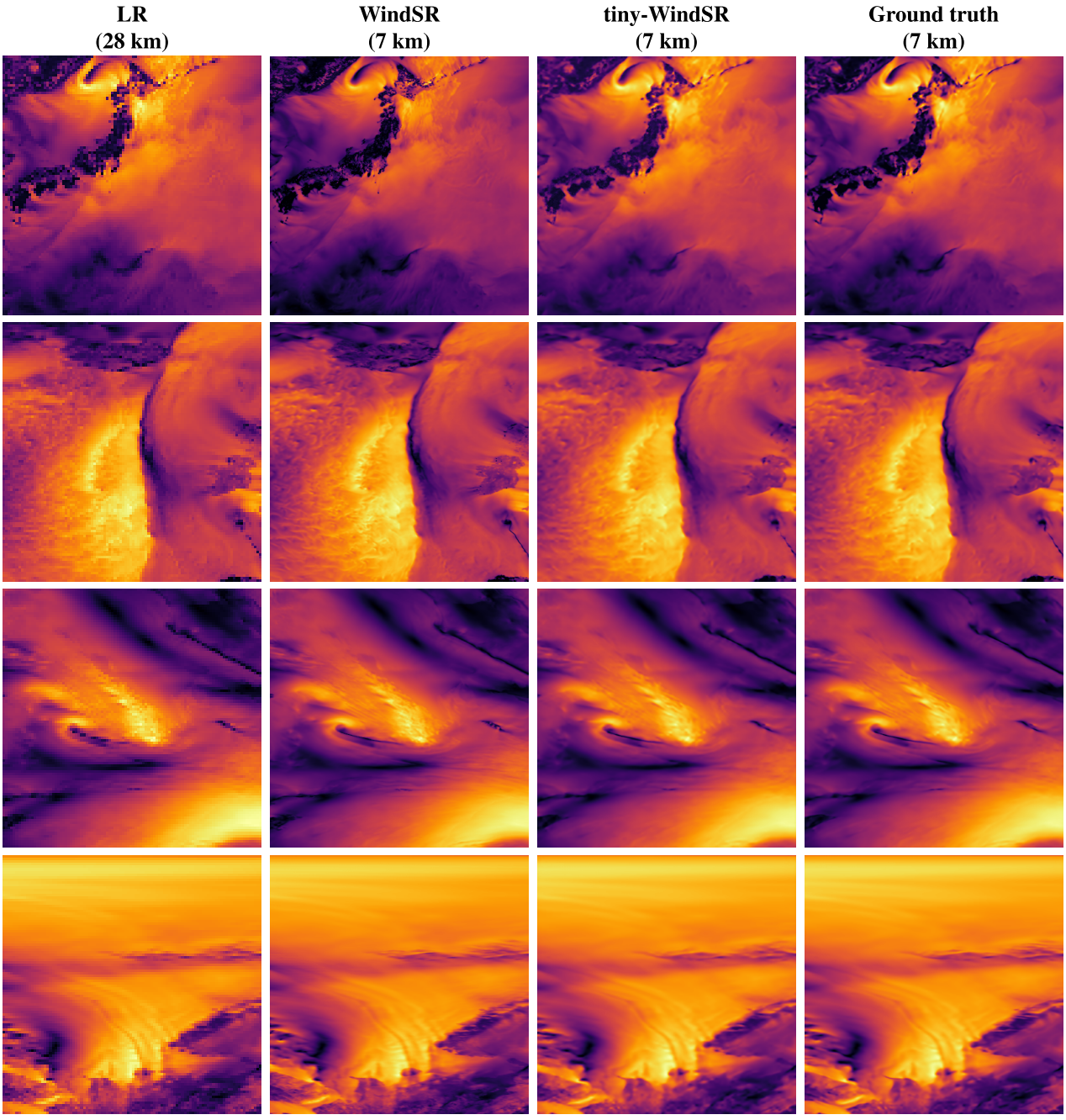


FIGURE 4. Comparison of the visualization of the WindSR and tiny-WindSR generated high-resolution (7 km) wind speed images with ground truth data.

V. CONCLUSION

In this research, we propose a novel framework WindSR and tiny-WindSR to increase the resolution of the satellite wind speed data using super-resolution. WindSR consists of several RRDB modules to compute features from low-resolution wind speeds for upsampling. We train the proposed networks and baseline GAN networks – BSRGAN and SwinIR using NASA's GEOS-5 Nature Run dataset with low-resolution wind speed at 28 km resolution as input and high-resolution wind speed at 7km resolution as the output of the network. Both WindSR and tiny-WindSR outperform traditional upsampling interpolation techniques, such as the nearest neighbor, bilinear, etc., and also deep learning-based techniques such as BSRGAN, and SwinIR by up to 17.89% on the validation dataset.

Through the experimental results, we found that the use of the discriminator for super-resolution of data such as wind speed hurts the performance of the accuracy, which is due to an increase in pixel-wise loss. Pixel-wise loss is less significant for normal images because texture and quality are

important for the generated image to look realistic, which is supported by the presence of a discriminator. However, for the remotely sensed wind speed data, we find that pixel-wise loss alone is sufficient for upsampling. In the continuation of our research work, we would evaluate the performance of WindSR in other domains for super-resolution tasks. Wind speed from the satellite is often collected at low resolutions due to the limitations of the sensors. Findings from this study would be useful for improving the resolution of satellite wind speed models as well as using super-resolution in other domains.

ACKNOWLEDGMENT

This study used high-performance NVIDIA Tesla A100 GPU cluster nodes provided by the Information Technology Center, The University of Tokyo, Japan.

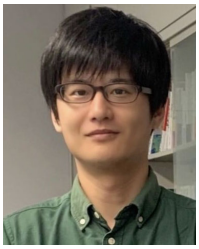
REFERENCES

[1] M. A. Mohandes, T. O. Halawani, S. Rehman, and A. A. Hussain, "Support vector machines for wind speed prediction," *Renew. Energy*, vol. 29, no. 6, pp. 939–947, May 2004.
[2] A. E. Blaugrund, "Notes on Doppler-shift lifetime measurements," *Nucl. Phys.*, vol. 88, no. 3, pp. 501–512, Nov. 1966.
[3] A. Kumar, T. Islam, Y. Sekimoto, C. Mattmann, and B. Wilson, "Convcast: An embedded convolutional LSTM based architecture for precipitation nowcasting using satellite data," *PLoS ONE*, vol. 15, no. 3, Mar. 2020, Art. no. e0230114.
[4] A. M. D. Silva, W. Putman, and J. Nattala, "File specification for the 7-km GEOS-5 nature run, ganymed release non-hydrostatic 7-km global mesoscale simulation," Global Model. Assimilation Office, Earth Sci. Division, NASA Goddard Space Flight Center, Greenbelt, MD, USA, Tech. Rep. GSFC-E-DAA-TN19, 2014.
[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, K. Aiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 5998–6008.
[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.
[8] H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, and H. Jégou, "Going deeper with image transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 32–42.
[9] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM Comput. Surv.*, vol. 54, no. 10s, pp. 1–41, Jan. 2022.
[10] T. Yang, P. Ren, X. Xie, and L. Zhang, "GAN prior embedded network for blind face restoration in the wild," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 672–681.
[11] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.
[12] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and A. C. Loy, "ESRGAN: Enhanced super-resolution generative adversarial networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV) Workshops*, 2018, pp. 63–79.
[13] J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte, "SwinIR: Image restoration using Swin transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2021, pp. 1833–1844.
[14] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 105–114.
[15] X. Wang, L. Xie, C. Dong, and Y. Shan, "Real-ESRGAN: Training real-world blind super-resolution with pure synthetic data," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2021, pp. 1905–1914.
[16] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, "Plug-and-play image restoration with deep denoiser prior," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 6360–6376, Oct. 2022.
[17] K. Zhang, L. Van Gool, and R. Timofte, "Deep unfolding network for image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 3214–3223.
[18] R. Kurinchi-Vendhan, B. Lütjens, R. Gupta, L. Werner, and D. Newman, "WiSoSuper: Benchmarking super-resolution methods on wind and solar data," 2021, *arXiv:2109.08770*.
[19] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 1132–1140.
[20] K. Zhang, J. Liang, L. Van Gool, and R. Timofte, "Designing a practical degradation model for deep blind image super-resolution," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 4771–4780.
[21] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9992–10002.
[22] E. J. Kirkland and E. J. Kirkland, "Bilinear interpolation," in *Advanced Computing in Electron Microscopy*. New York, NY, USA: Springer, 2010, pp. 261–263.
[23] P. R. Smith, "Bilinear interpolation of digital images," *Ultramicroscopy*, vol. 6, no. 2, pp. 201–204, Jan. 1981.
[24] S. van der Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: A structure for efficient numerical computation," *Comput. Sci. Eng.*, vol. 13, no. 2, pp. 22–30, Mar. 2011.
[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[27] T. Suzumura et al., "mdx: A cloud platform for supporting data science and cross-disciplinary research collaborations," 2022, *arXiv:2203.14188*.
[28] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-ResNet and the impact of residual connections on learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 31, 2017, pp. 4278–4284.
[29] A. Hore and D. Ziou, "Image quality metrics: PSNR vs. SSIM," in *Proc. 20th Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 2366–2369.
[30] E. Agustsson and R. Timofte, "NTIRE 2017 challenge on single image super-resolution: Dataset and study," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 1122–1131.
[31] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[32] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier, "From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions," *Trans. Assoc. Comput. Linguistics*, vol. 2, pp. 67–78, Dec. 2014.
[33] Y. Blau, R. Mechrez, R. Timofte, T. Michaeli, and L. Zelnik-Manor, "The 2018 PIRM challenge on perceptual image super-resolution," in *Computer Vision—ECCV 2018 Workshops*, L. Leal-Taixé and S. Roth, Eds. Cham, Switzerland: Springer, 2019, pp. 334–355.

ASHUTOSH KUMAR received the B.Tech. degree from the Indian Institute of Technology (IIT) Kanpur, and the master's degree from The University of Tokyo, Japan, where he is currently pursuing the Ph.D. degree. He is also a Visiting Student with the Senseable City Laboratory, Massachusetts Institute of Technology (MIT), USA. During the master's study, he was a Visiting Researcher with the NASA Jet Propulsion Laboratory and developed Convcast for precipitation nowcasting using IMERG satellite images. His main research interests include urban monitoring and CO_2 emission estimation on urban road networks using moving camera videos utilizing visual AI. He received the Best Paper Award at the ISPRS Geospatial Week 2019 for his internship work on point cloud data classification from Heidelberg University, Germany. He also received the Academic Excellence Award for Outstanding Academics during the bachelor's study, the Proficiency Medal for the Best Undergraduate Project, and the Furuichi Kimitake Prize for Outstanding Master's Thesis.

TANVIR ISLAM is currently a Staff Data Scientist with Okta Inc., Seattle, USA. Prior to joining Okta Inc., he was with the NASA Jet Propulsion Laboratory (JPL), with specialization in remote sensing observations. At JPL, he has engaged with the development of advanced microwave calibration and retrieval algorithms for NASA's Earth Observing Missions. Prior to joining NASA/JPL, in 2015, he was with the NOAA/NESDIS/STAR, and worked on the development of satellite remote sensing algorithms, with an emphasis on microwave vibrational inversion techniques, from 2013 to 2015. He was a Visiting Scientist with The University of Tokyo, as a part of the NASA/JAXA Precipitation Measurement Missions (PMM) Algorithm Development Team, in 2012; and the University of Calgary, in 2015.

JUE MA is a Visiting Ph.D. Student with the MIT Media Laboratory, School of Engineering, The University of Tokyo. She is also with the City Science Group, MIT Media Laboratory. She has a background in landscape architecture and civil engineering. Her research interests include urban planning support tools and urban simulation and decentralized autonomous organization (DAO) for supporting public participation in communities.



TAKEHIRO KASHIYAMA received the Ph.D. degree in information science from Kansai University, Japan. He is an Associate Professor with the Osaka University of Economics, Osaka, Japan. Prior to joining Osaka University of Economics, he was an Assistant Professor with the Institute of Industrial Science, The University of Tokyo, Japan. He is interested in research about urban dynamics reproduction. His current research interests include creating synthetic human mobility data in urban areas and predicting human behavior during a disaster.

YOSHIHIDE SEKIMOTO received the Ph.D. degree in civil engineering from The University of Tokyo, in 2002. He directs the Human-Centered Urban Informatics Laboratory, established in April 2013, which is a part of the Institute of Industrial Science (IIS), The University of Tokyo, where he is currently the Vice Director of the Center for Spatial Information Science (CSIS). Previously, he was with the National Institute for Land, Infrastructure and Management, from 2002 to 2007; and the Center for Spatial Information Science, The University of Tokyo, from 2007 to 2013.

CHRIS MATTMANN is the Manager of the Open Source Programs and Applications Office and a Principal Data Scientist of the Engineering Administrative Office, NASA's Jet Propulsion Laboratory (JPL). He is responsible for influencing engineering data system designs and facilitating the infusion of new technologies to meet the laboratory's future challenges. He is also a JPL's first Principal Scientist of data science. He is also the Director of the Information Retrieval and Data Science (IRDS) Group, USC, and an Adjunct Associate Professor. He teaches graduate courses in content detection and analysis and in search engines and information retrieval. He has materially contributed to understanding of the deep web and dark web through the DARPA MEMEX Project. His work helped uncover the Panama Papers scandal.