Generating Synthetic Multispectral Satellite Imagery from Sentinel-2

Tharun Mohandoss
Radiant Earth Foundation
San Francisco, CA 94105
tharun96@utexas.edu

Aditya Kulkarni
Radiant Earth Foundation
San Francisco, CA 94105
adkulkar@eng.ucsd.edu

Daniel Northrup
Benson Hill
Saint Louis, MI
dan@northrup.ag

Ernest Mwebaze
Sunbird AI
Kampala, Uganda
emwebaze@gmail.com

Hamed Alemohammad
Radiant Earth Foundation
San Francisco, CA 94105
hamed@radiant.earth

Abstract

Multi-spectral satellite imagery provides valuable data at global scale for many environmental and socio-economic applications. Building supervised machine learning models based on these imagery, however, may require ground reference labels which are not available at global scale. Here, we propose a generative model to produce multi-resolution multi-spectral imagery based on Sentinel-2 data. The resulting synthetic images are indistinguishable from real ones by humans. This technique paves the road for future work to generate labeled synthetic imagery that can be used for data augmentation in data scarce regions and applications.

1 Introduction

In many supervised learning problems with satellite imagery, such as agricultural monitoring, training data are generated from data collected on the ground (a.k.a. ground reference). Ground reference data collection is an extensive effort, and extremely scarce in remote and dangerous areas that would most benefit from remote sensing applications. As a result, there are limited number of training datasets available for these problems which narrows application of machine learning (ML) based techniques with satellite imagery to specific parts of the world while these images are available at global scale.

Techniques such as transfer learning [1] and data augmentation [2] have been used to mitigate this issue. Transfer learning has shown some promising results for crop type classification in regions with homogeneous agricultural practices such as the US [3]. However, this is not necessarily scalable to other parts of the world, such as regions that are dominated by smallholder farming. Similar study for a road detection problem shows that a model trained in Las Vegas, US is not easily transferable to Khartoum, Sudan [4].

Several studies have used GANs with satellite imagery for applications such as super-resolution and data augmentation [5, 6, 7]. These show the success of applying GAN to high-resolution satellite imagery with 3-4 multispectral bands.

GANs are generally used to reproduce an RGB image distribution but remote sensing data augmentation applications would require producing images with more than 3 bands along with their corresponding labels. Generalizing from RGB imagery to labelled multi-band imagery can be achieved in two steps by first producing more than 3 bands and then incorporating label generation. Here, we tackle the first step and present a new framework for generating synthetic satellite imagery. We introduce a new GAN architecture that incorporates ten bands with varying resolutions from

AI for Earth Sciences Workshop at NeurIPS 2020.

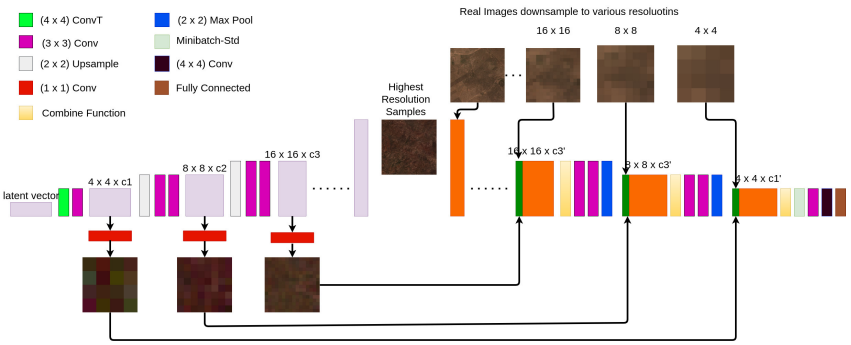

Figure 1: Block Diagram of the MSG-ProGAN model

Sentinel-2 satellite imagery, and generates realistic multispectral imagery for data augmentation. Multispectral bands are essential for many land surface monitoring applications, therefore this model can be applied to augment data across multiple applications.

2 Dataset

We use multispectral imagery from Sentinel-2 satellites (details of the individual bands are listed in the Table A.1). Our study region is western Kenya, where we have ground reference data for the next phase of this project. Therefore, for the experiments of this study we selected 105k Sentinel-2 images from this region at 256×256 pixels to be used as training images. The selected images consist of several types of land cover including barren land, urban areas, forests, croplands, and water. We filter out samples that contain cloud and samples where water bodies cover a significant portion of the image. Figure 2 shows sample real RGB images used in this study, and Figure A.1 shows all ten bands from sample images.

3 Methodology

GANs suffer from a variety of issues such as training instability and mode collapse that require careful setting of hyperparameters for successful training. [8] suggests WGAN-GP training loss which can mitigate the issues caused by insufficient overlap between generated and target distributions leading to an increased stability of training. Karnewar et al. [9] builds on the previous work [10] and proposes the Multi-Scale Gradients (MSG) method where images are produced in exponentially growing resolutions starting from the smallest size of 4×4 till the final output size allowing the discriminator to provide feedback in all resolutions instead of just one which further mitigates these issues.

3.1 Model

We adopt the MSG-GAN architecture and WGAN-GP loss. Specifically, we use the MSG-ProGAN architecture used in [9] (and variants of it) and modify it to produce multispectral satellite images of size 256×256 instead of RGB images (which differs in the number of output channels). The exact architectures are shown in Tables A.2 and A.3.

4 Experiments

Satellite imagery has coarser pixel size (a.k.a. resolution) compared to common imagery in computer vision, and they usually have more than 3 bands, possibly with varying resolutions. Due to these differences, a straightforward port of GAN models from existing literature might not achieve the desired results. Therefore, we broke down our approach to smaller experiments and incrementally increased complexity of the model towards the final goal of producing multispectral satellite images.

We start with experiment 1 where we generate 4×4 resized versions of the RGB bands of the target images using the smaller parts of the Generator an Discriminator in the original MSG-ProGAN

Table 1: List of experiments

No	Experiment	Result
1	4×4 RGB	Synthetic and real images are indistinguishable.
2	256×256 RGB	Synthetic images can fool humans.
3A	256×256 ten bands 1	Less than satisfactory results.
3B	256×256 ten bands 2	Less than satisfactory results.
3C	256×256 ten bands 3	Best results for the ten bands case.
4A	256×256 RGB fewer convolutions	Satisfactory Results.
4B	256×256 RGB fewer filters	Image quality almost as good as Experiment 2.
5	256×256 ten bands fewer filters	Image quality almost as good as Experiment 3C.

network. Results are satisfactory (not shown) and the distribution of the 4×4 images produced by the generator is visually indistinguishable from the 4×4 versions of the original images. Using this as a toy experiment, we chose RMSProp as the optimizer, 10^{-3} as the learning rate and a weight factor (λ) of 10 for the gradient penalty for the rest of experiments. More details on the training is provided in sectionB.

Table 1 shows the list of all experiments conducted. We start with 256×256 RGB imagery and then generate 256×256 ten band imagery. Finally, we experiment with smaller versions of the MSG-ProGAN model to optimize the model resource footprint (E.g. GPU memory requirement).

4.1 Experiment 2: Generating 256×256 RGB images

Here, we use the full MSG-ProGAN architecture for 256×256 images described in Tables A.2 and A.3 by setting m = 3 to produce a 3 channel RGB output. We train the model for 12 epochs on the RGB channels with a batch size of 4. The resulting synthetic images are extremely convincing. In an online experiment with 106 unique individuals attempting to distinguish real images from synthetic ones, a majority of the them (>70) scored an average in the range of 50%-70%. Despite the GAN having learned well, there are some classes that are missing in the outputs such as urban areas and regular croplands. This could be due to the class imbalance in our dataset (regular croplands and urban areas are underrepresented). Figure 2 shows example synthetic images from this experiment.

4.2 Experiment 3: Generating 256×256 ten band images

Here, we expand the model to ten bands consisting of four bands of 10m resolution and six bands of 20m resolution. The 10m bands form a 256×256 image and the 20m bands form a 128×128 image due to the difference in resolutions. In experiment 3A, we simply modify the model from experiment 2 and use nearest neighbor to interpolate the 128×128 images into 256×256 images and treat all the ten bands the same. Next, we set $m = 10$ in the architecture of MSG-ProGAN to produce ten bands per image instead of three. The results are less than satisfactory. Some examples shown in Figure A.2.

For experiment 3B, we grouped together bands based on their resolutions (forming 2 groups, one for 10m bands, one for 20m bands). In the generator, we produce 256×256 output for the 10m bands as usual and 128×128 output for the 20m bands by using an extra pooling layer after the convolution layer that produces the image for this group. In the discriminator, we process these groups in parallel branches and merge the output features together forming a neural network described in Figure A.3. The images produced by this architecture are of poor quality and suffer from various issues such as presence of artefacts and loss of correlation between the bands of different resolutions in the generated images. Some results (only the RGB bands) are shown in Figure A.2.

To improve the results and ensure bands with varying resolution are correlated, we revised the model from 3A to generate only the 10m bands in the highest resolution (256×256). The proposed discriminator model for this is presented in Figure A.4. This architecture produces the best quality images out of the three experiments (3A, 3B, and 3C) designed to generate ten band images. The images from different bands are well synced and the images are almost realistic although there is some observable loss of variety in the images. Figure 2 shows samples generated from this model.

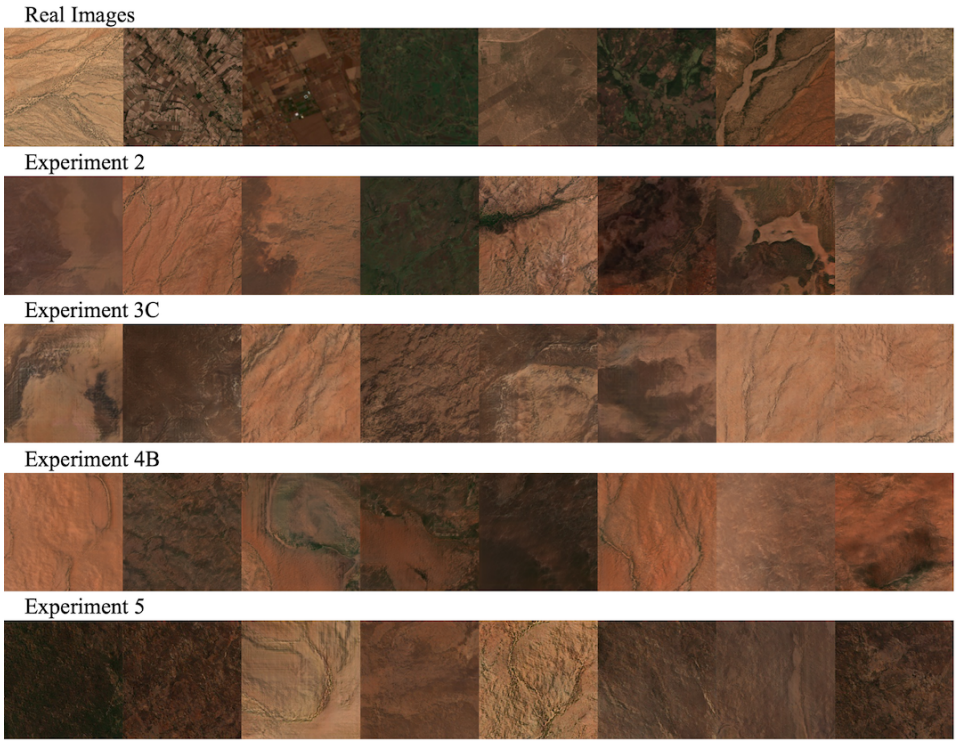

Figure 2: Sample real and synthetic images from experiments 2, 3C, 4B, and 5 (only RGB bands shown)

4.3 Experiments 4A, 4B, and 5: Resource efficient models

To reduce the resource footprint of our model, we designed two experiments for RGB images. First, we investigate the effect of reducing the number of convolutions in the model by removing every other convolution and also reducing the number of images produced by the generator and passed to the discriminator (only 4×4, 16×16, 64×64, 256×256 ones as shown in Figure A.5). Next, we investigate the effect of reducing the number of filters in every layer of the model by half.

We find that both of these approaches are viable strategies to reduce the model's resource footprint i.e they both produce satisfactory results but reducing the number of filters is the preferred approach and the performance in this case is very close to the original model. We also observe by modifying the model from Experiment 3C in the same way that this result generalizes to the ten bands case.

5 Discussion

Satellite imagery have unique properties that distinguishes them from common images used in the computer vision field. As a result, existing GAN architectures cannot be applied to these imagery off the shelf. With the goal of generating synthetic satellite imagery for data augmentation, we designed multiple GAN architectures of varying complexity to generate synthetic multispectral multi-resolution satellite imagery. Results show:

1. A straightforward port of models from existing GAN literature is sufficient to generate RGB bands of satellite images but these models have high resource requirements.
2. Our modified GAN architecture can generate realistic images with ten bands at varying resolutions.
3. Significant GAN model size reduction can be achieved by using fewer number of filters with minimal reduction in quality.

This study demonstrates a state-of-the-art GAN architecture that can be used to generate diverse multispectral satellite imagery with varying band resolutions. Such a model is an essential step to the next phase of our experiment which is generating these images with their corresponding labels.

Acknowledgements

This research is funded by a grant awarded to Radiant Earth Foundation through the 2019 Grand Challenges Annual Meeting Call-to-Action from the Bill & Melinda Gates Foundation. The findings and conclusions contained within are those of the authors and do not necessarily reflect positions or policies of the Bill & Melinda Gates Foundation.

References

[1] Anna X. Wang, Caelin Tran, Nikhil Desai, David Lobell, and Stefano Ermon. Deep transfer learning for crop yield prediction with remote sensing data. In *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, COMPASS '18, New York, NY, USA, 2018. Association for Computing Machinery.

[2] Xingrui Yu, Xiaomin Wu, Chunbo Luo, and Peng Ren. Deep learning in remote sensing scene classification: a data augmentation enhanced convolutional neural network framework. *GIScience & Remote Sensing*, 54(5):741–758, 2017.

[3] Sherrie Wang, George Azzari, and David B. Lobell. Crop type mapping without field-level labels: Random forest transfer and unsupervised clustering techniques. *Remote Sensing of Environment*, 222(November 2018):303–317, mar 2019.

[4] Yoni Nachmany and Hamed Alemohammad. Detecting roads from satellite imagery in the developing world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[5] D. Lin, K. Fu, Y. Wang, G. Xu, and X. Sun. Marta gans: Unsupervised representation learning for remote sensing image classification. *IEEE Geoscience and Remote Sensing Letters*, 14(11):2092–2096, 2017.

[6] Dongao Ma, Ping Tang, and Li-Jun Zhao. Siftinggan: Generating and sifting labeled samples to improve the remote sensing image scene classification baseline in vitro. *CoRR*, abs/1809.04985, 2018.

[7] Kui Jiang, Zhongyuan Wang, Peng Yi, Guangcheng Wang, Tao Lu, and Junjun Jiang. Edge-Enhanced GAN for Remote Sensing Image Superresolution. *IEEE Transactions on Geoscience and Remote Sensing*, 57(8):5799–5812, aug 2019.

[8] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017.

[9] Animesh Karnewar, Oliver Wang, and Raghu Sesha Iyengar. MSG-GAN: multi-scale gradient GAN for stable image synthesis. *CoRR*, abs/1903.06048, 2019.

[10] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.

[12] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc., 2016.

A Supplementary Figures and Tables

Table A.1: Description of the bands in Sentinel-2 satellite images. 10m and 20m bands are used in this study.

Band	Spatial Resolution
Band 1 – Coastal aerosol	60
Band 2 – Blue	10
Band 3 – Green	10
Band 4 – Red	10
Band 5 – Vegetation red edge 1	20
Band 6 – Vegetation red edge 2	20
Band 7 – Vegetation red edge 3	20
Band 8 – NIR	10
Band 8A – Narrow NIR	20
Band 9 – Water vapour	60
Band 10 – SWIR - Cirrus	60
Band 11 – SWIR1	20
Band 12 – SWIR2	20

Table A.2: Generator Architecture. After every block in the generator, a 1×1 Conv layer is used to convert the output activation volume into an image which is passed onto the discriminator.

Block	Operation	Activation	Output Shape
	Latent Vector	Norm	512 x 1 x 1
1.	Conv 4x4	LReLU	512 x 4 x 4
	Conv 3x3	LReLU	512 x 4 x 4
	Upsample	-	512 x 8 x 8
2.	Conv 3x3	LReLU	512 x 8 x 8
	Conv 3x3	LReLU	512 x 8 x 8
	Upsample	-	512 x 16 x 16
3.	Conv 3x3	LReLU	512 x 16 x 16
	Conv 3x3	LReLU	512 x 16 x 16
	Upsample	-	512 x 32 x 32
4.	Conv 3x3	LReLU	512 x 32 x 32
	Conv 3x3	LReLU	512 x 32 x 32
	Upsample	-	512 x 64 x 64
5.	Conv 3x3	LReLU	256 x 64 x 64
	Conv 3x3	LReLU	256 x 64 x 64
	Upsample	-	256 x 128 x 128
6.	Conv 3x3	LReLU	128 x 128 x 128
	Conv 3x3	LReLU	128 x 128 x 128
	Upsample	-	128 x 256 x 256
7.	Conv 3x3	LReLU	64 x 256 x 256
	Conv 3x3	LReLU	64 x 256 x 256

B Model and Training Details

B.1 Weight initialization: Equalized Learning Rate

We follow the weight initialization technique mentioned in [10] to use a trivial N (0, 1) initialization and then explicitly scale the weights at run-time [11]. We set $w'_i = w_i/c$, where w_i are the weights and c is the per-layer normalization constant from He's initializer. This ensures that the dynamic range, and thus the learning speed, is the same for all weights.

Table A.3: Discriminator Architecture. m is the number of output channels which varies across different experiments.

Block	Operation	Activation	Output Shape
	Image 1	-	m x 256 x 256
	From RGB	-	64 x 256 x 256
1.	MiniBatchStd	-	65 x 256 x 256
	Conv 3x3	LReLU	64 x 256 x 256
	Conv 3x3	LReLU	128 x 256 x 256
	Max Pool	-	128 x 128 x 128
	Image 2	-	m x 128 x 128
	Concat	-	(128+m) x 128 x 128
2.	MiniBatchStd	-	(128+m+1) x 128 x 128
	Conv 3x3	LReLU	128 x 128 x 128
	Conv 3x3	LReLU	256 x 128 x 128
	Max Pool	-	256 x 64 x 64
	Image 3	-	m x 64 x 64
	Concat	-	(256+m) x 64 x 64
3.	MiniBatchStd	-	(256+m+1) x 64 x 64
	Conv 3x3	LReLU	256 x 64 x 64
	Conv 3x3	LReLU	512 x 64 x 64
	Max Pool	-	512 x 32 x 32
	Image 4	-	m x 32 x 32
	Concat	-	(512+m) x 32 x 32
4.	MiniBatchStd	-	(512+m+1) x 32 x 32
	Conv 3x3	LReLU	512 x 32 x 32
	Conv 3x3	LReLU	512 x 32 x 32
	Max Pool	-	512 x 16 x 16
	Image 4	-	m x 16 x 16
	Concat	-	(512+m) x 16 x 16
5.	MiniBatchStd	-	(512+m+1) x 16 x 16
	Conv 3x3	LReLU	512 x 16 x 16
	Conv 3x3	LReLU	512 x 16 x 16
	Max Pool	-	512 x 8 x 8
	Image 5	-	m x 8 x 8
	Concat	-	(512+m) x 8 x 8
6.	MiniBatchStd	-	(512+m+1) x 8 x 8
	Conv 3x3	LReLU	512 x 8 x 8
	Conv 3x3	LReLU	512 x 8 x 8
	Max Pool	-	512 x 4 x 4
	Image 6	-	m x 4 x 4
	Concat	-	(512+m) x 4 x 4
7.	MiniBatchStd	-	(512+m+1) x 4 x 4
	Conv 3x3	LReLU	512 x 4 x 4
	Conv 3x3	LReLU	512 x 4 x 4
	Fully Connected	Linear	1 x 1 x 1

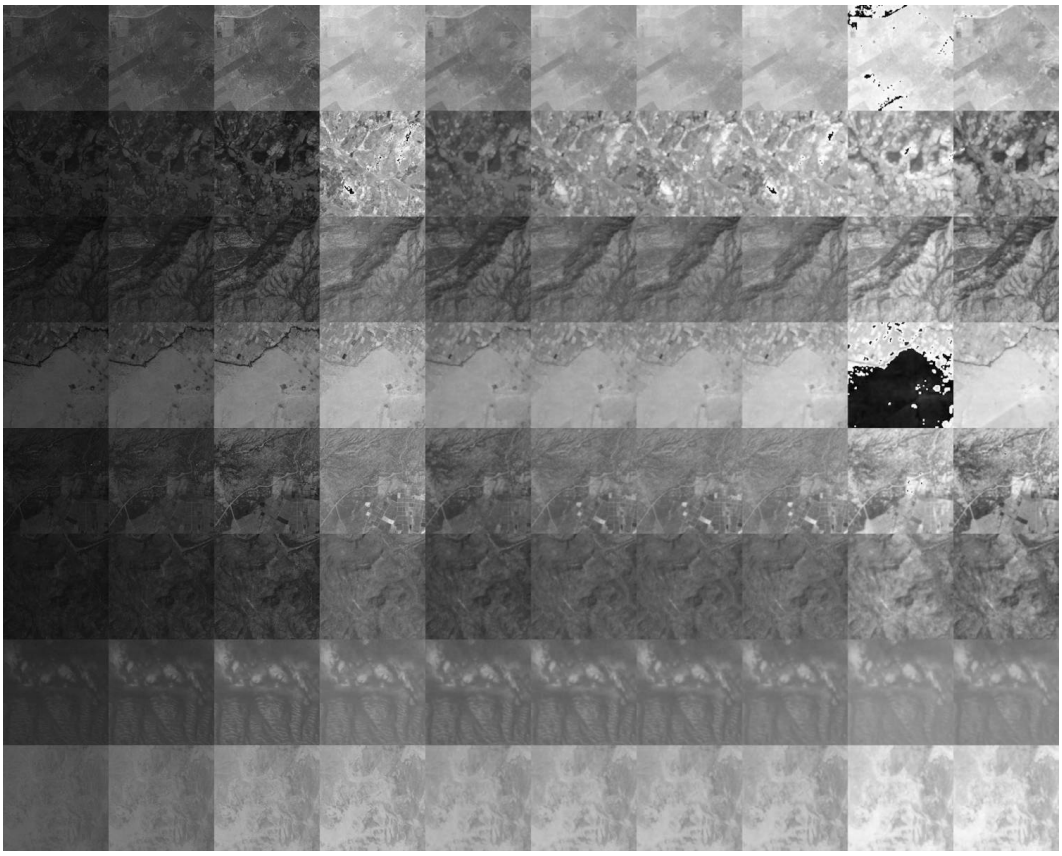

Figure A.1: Samples of real ten band images in our dataset: Each row shows one sample and each column is a band from left to right: blue, green, red, NIR , red-edge, red-edge-2, 'red-edge-3', 'red-edge-4', 'SWIR1', 'SWIR2'. The first 4 bands are 10m bands (256×256) and the remaining six are 20m bands (128×128). The 20m bands have been resized to 256×256 for ease of visualization.

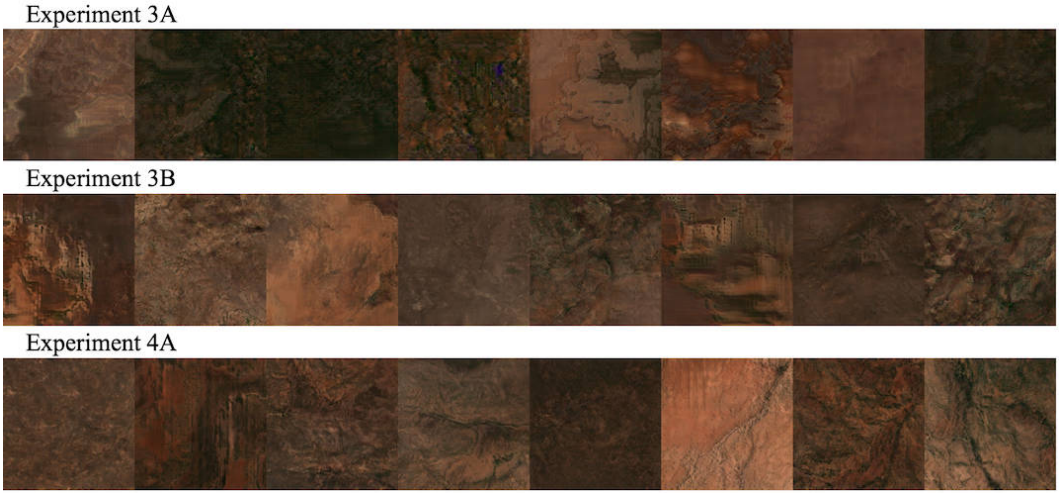

Figure A.2: Sample synthetic images from experiments 3A, 3B and 4A (only RGB bands shown)

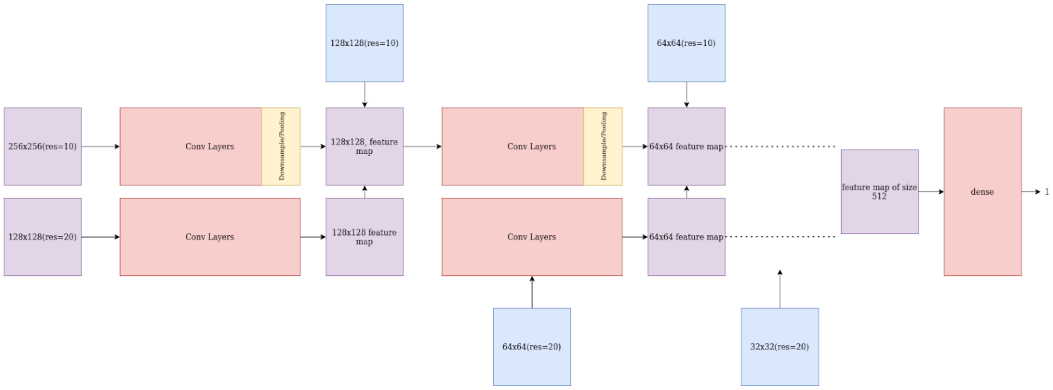

Figure A.3: Discriminator model architecture for experiment 3B

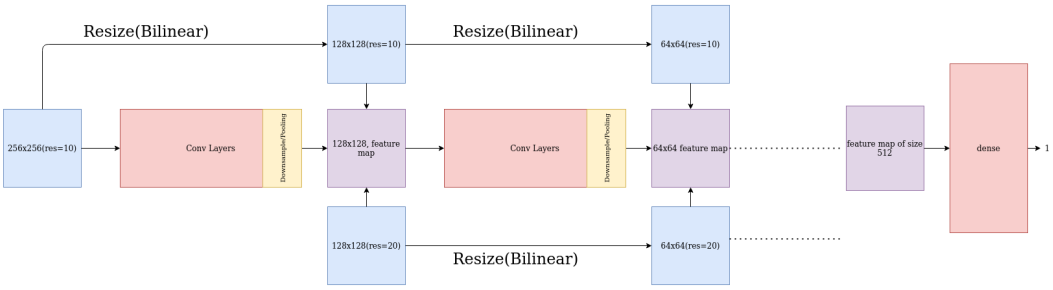

Figure A.4: Discriminator model architecture for experiment 3C

B.2 Pixel-wise Feature Vector Normalization in Generator

We also apply the layer-wise vector normalization in [10] after every 3x3 convolution in the generator to disallow the scenario where the magnitudes in the generator and discriminator spiral out of control as a result of unhealthy competition between the discriminator and generator.

B.3 Minibatch standard deviation

Introduced first in the [12] minibatch discrimination can help GANs prevent mode collapse. We follow [9] in their use of minibatch standard deviation as a simpler variant of the above technique to achieve the same goal. We first compute the standard deviation for each feature in each spatial location over the minibatch. We then average these estimates over all features and spatial locations to arrive at a single value. We replicate the value and concatenate it to all spatial locations and over the

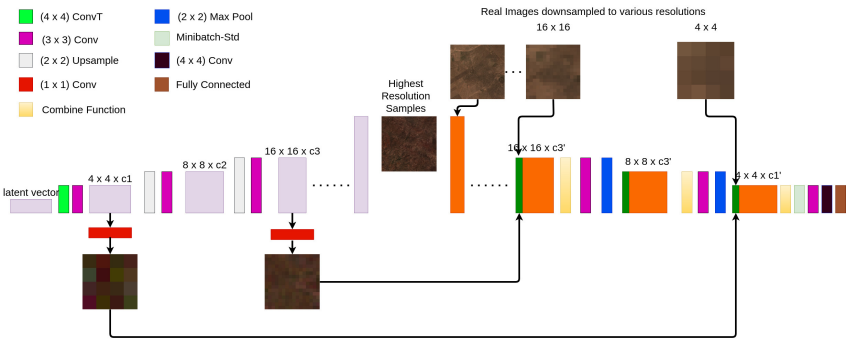

Figure A.5: Block Diagram of the modified MSG-ProGAN model with fewer convolutions.

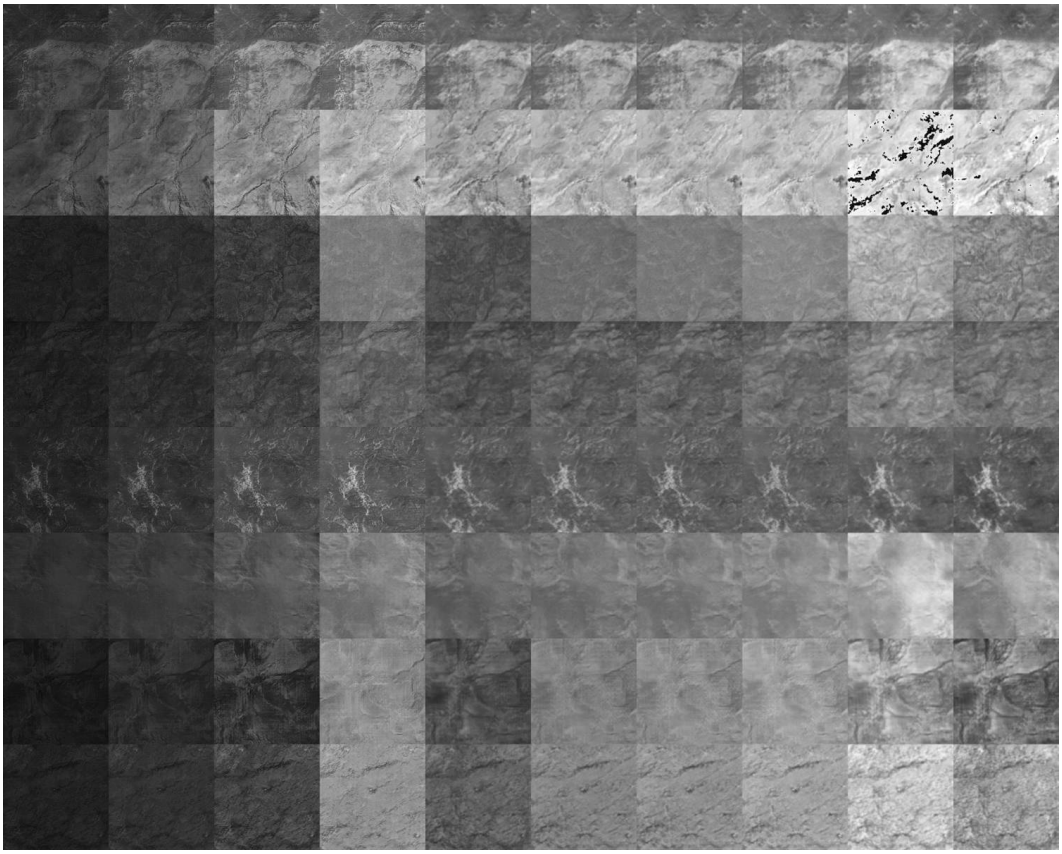

Figure A.6: Results of experiment 3C: Each row shows one sample and each column is a band from left to right: blue, green, red, NIR , red-edge, red-edge-2, 'red-edge-3', 'red-edge-4', 'SWIR1', 'SWIR2'. The first 4 bands are 10m bands (256×256) and the remaining six are 20m bands (128×128). The 20m bands have been resized to 256×256 for ease of visualization.

minibatch, yielding one additional (constant) feature map. This additional layer is added after every concatenation step.

B.4 Training Details

We use Pytorch's RMSProp optimizer with a learning rate of 10^{-3} for both the generator and the discriminator. The total loss for the generator and discriminator are as follows where D(x) represents the discriminator output for input image x :

$$\begin{aligned} G_{loss} &= -D(generated_{image}) \\ D_{loss} &= D(generated_{image}) - D(real_{images}) + \lambda \times GP \end{aligned} \tag{1}$$

where GP denotes the Gradient Penalty corresponding to the WGAN-GP loss. We set the relative importance parameter λ to 10.