BTP
Spectral Reconstruction from RGB images

Guide:
Dr. Viswanath P

Members:
Praveen P
Vishruth T
Dinesh kumar K
Harsha vardhan S

TABLE OF CONTENTS

INTRODUCTION

- **RGB images** are taken from the **380 nm - 750 nm wavelength range** of visible spectrum of light.
- Here we want to **reconstruct 31 spectral images** in range of 380 nm - 750 nm wavelength.
- **Bands we are focusing on are :**
  - `[400 410 420 430 440 450 460 470 480 490 500 510 520 530 540 550 560 570 580 590 600 610 620 630 640 650 660 670 680 690 700]`

Previous Work

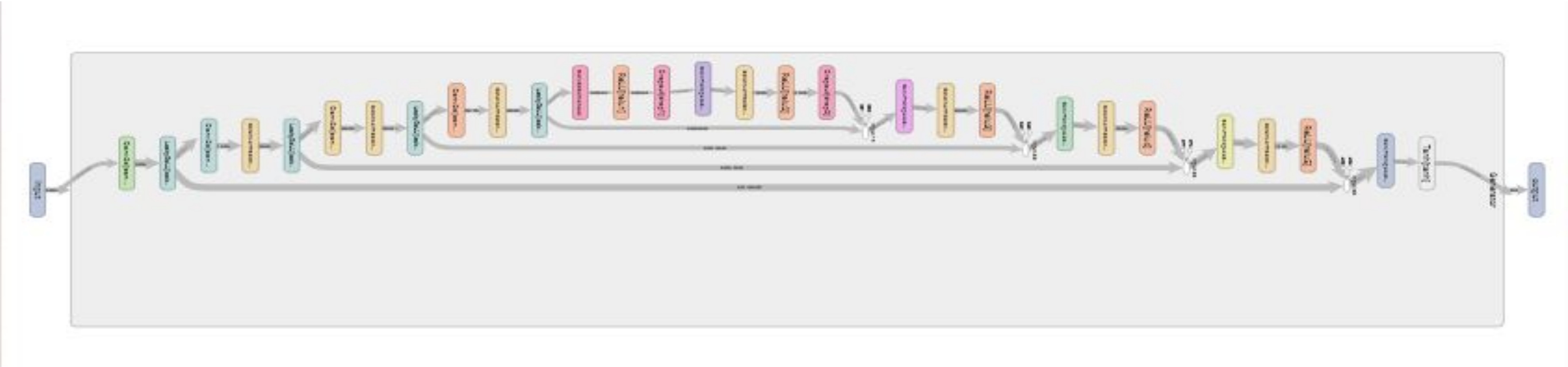
- **Dataset collection**, **Data Preprocessing** like changing its resolution to make it easy for further processing.
- We built a **GAN based model** for the spectral images generation.
- **Generator of the GAN** was built using the **U-Net architecture**.
- **Discriminator** was built using a the **Patch GAN**.
- Used **NTIRE 2020 dataset** for training of the models.
- **Resolution** was set from **512x512** to **256x256** due to **memory constraints**.
- **Among 31 spectral channels** we generated **first 3 channels** as the output of the **Generator network**.

Work Done

- We implemented 3 different GAN models with variation in their Generator models.
- **Deep ConvNet** is the first model in which Generator is U-Net with normal GAN loss functions are used for training. All the training parameters are manually selected.
- **HyperUNet,** it is very similar to first model but here we did **hyper parameter optimization** for the **lambda** (penalty constant).It is **optimized on rmse values** of the model.
- **Resnet**, Generator of this GAN model contains ResBlocks instead of U-Net. **1 Initial Layer** and **3 ResBlocks Layers** and **1 Final Layer**.

Generator (U-Net architecture)

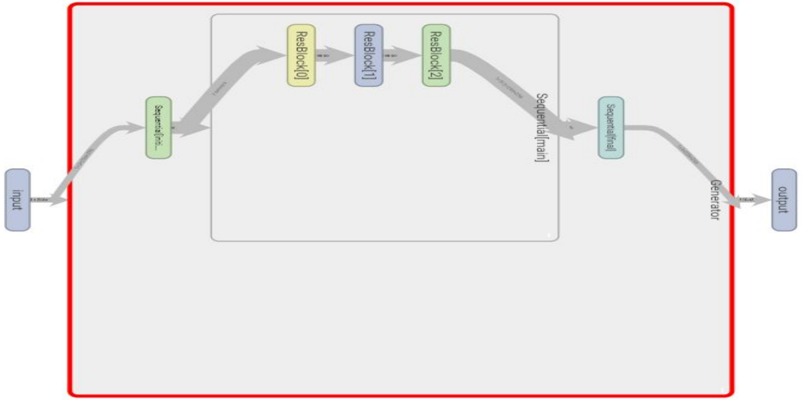
- For the **Generator** we used the **U-Net** architecture , it involves **downsampling** of **3x256x256** images to bottleneck layer **(512x8x8)** from there again **upsampling** will be done to retain **3x256x256**.

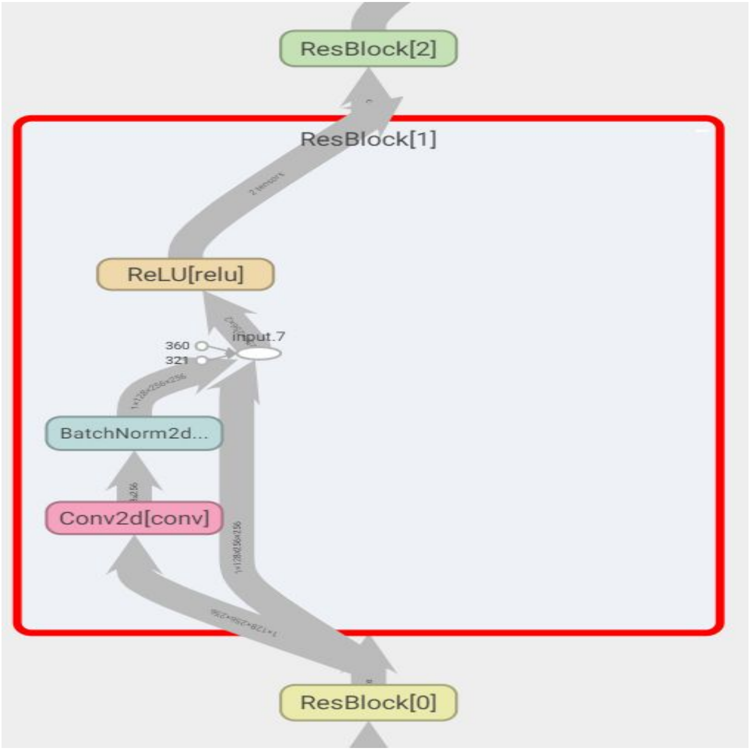


Generator (Resnet architecture)

- **Initial convolution layer** and **3 resnet blocks** and **final convolution layer**.
- Each **resnet block** contains **1 conv layer, batch normalization and then relu activation** function.
- **256x256x3 --→ 256x256x64 --→ 256x256x128 --→ 256x256x256 --→ 256x256x512 --→ 256x256x3.**
- Keeping the **resolution same** we **changed the depth of the image** in each layer.

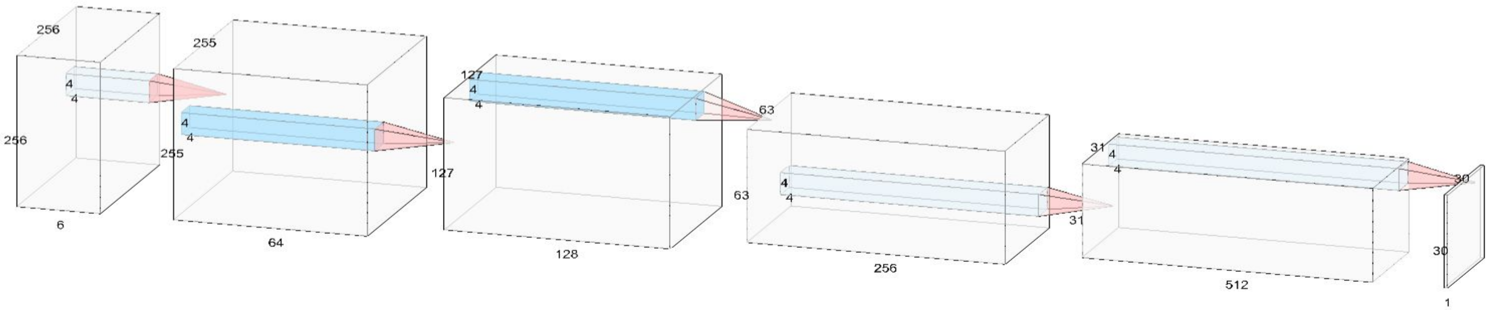
Generator (Resnet architecture)





Discriminator (Patch GAN architecture)

- For the **Discriminator network** we used the **Patch GAN.**
- From the given input after **applying some downsampling** , it will outputs **30x30x1 patch with probability of the patch being real.**
- For the **real image** our **target is to get patch of 1's** and for **fake images** our **target is to get patch of 0's.**
- **Each cell in the patch** corresponds to **70x70 patch in input.**



Generator Loss Functions

- **Generator Loss function**.
  1. **Adversarial loss** ----→
  2. **L1 Loss**

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))]$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1]$$

3. **Total Loss** ---→

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda\mathcal{L}_{L1}(G)$$

- **Optimizer Used** : **Adam**, **learning rate 2e-4** , **betas = (0.5,0.999)**.

Discriminator Loss Function

- **Discriminator Loss Function**.

$$Loss = (Y)(-log(Y_{pred})) + (1 - Y)(-log(1 - Y_{pred}))$$

Remains when Y = 1 | Remains when Y = 0

Removed when Y = 0 | Removed when Y = 1

- **Optimizer used** : **Adam** , **learning rate 2e-4** , **betas = (0.5,0.999).**

Dataset Information

- **NTIRE 2020 Spectral Reconstruction.**
- **31 spectral channels** from **400nm** to **700nm**.
- train : validation : test **450 : 10 : 10**.

- **Cave multispectral images dataset.**
- **31 spectral channels** from **400nm** to **700nm**.
- train : validation : test **22 : 5 : 5**.

- **NUS multispectral images dataset.**
- **31 spectral channels** from **400nm** to **700nm**.
- train : validation : test **50 : 8 : 8**.

Evaluation Metrics

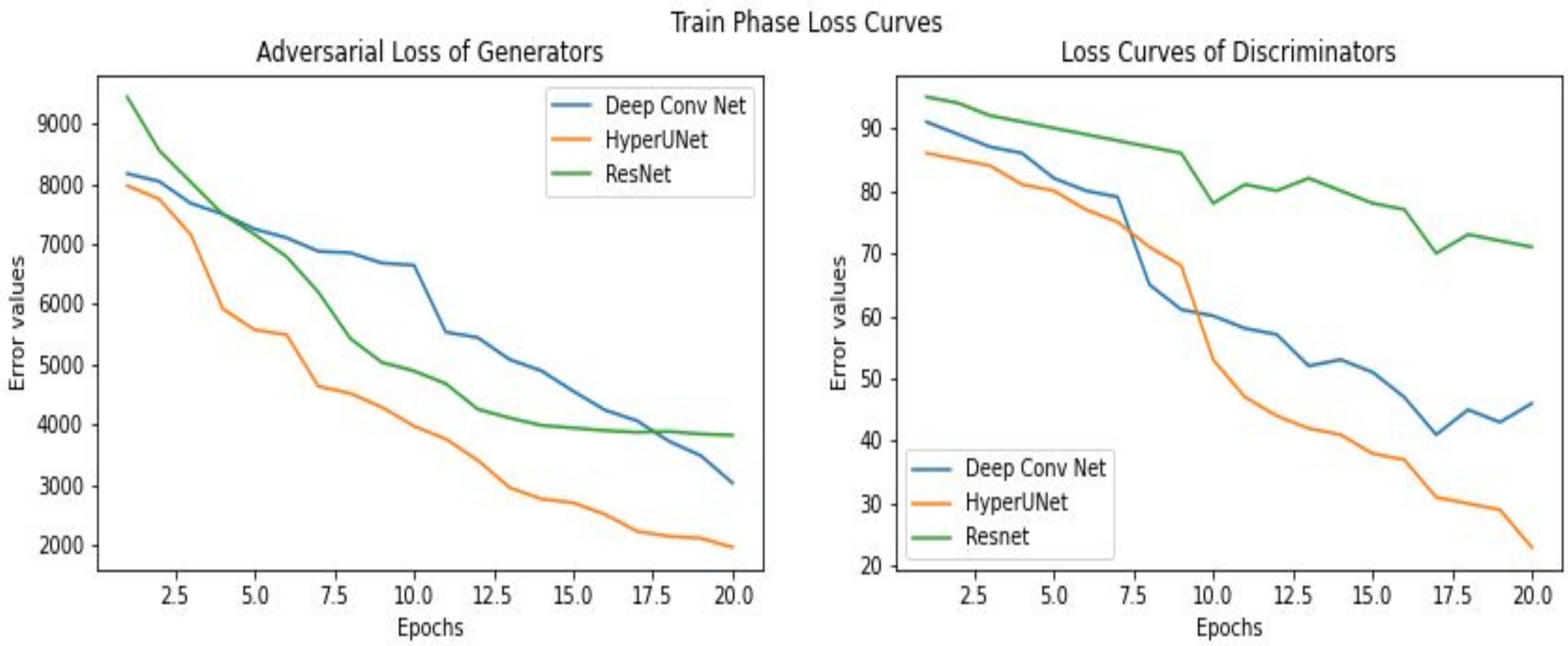
RMSE

1. Root mean square error.
2. Pgtic and Precic denote the value of the c spectral channel of the i-th pixel in the ground truth and the reconstructed image
3. $$RMSE = \sqrt{\frac{\sum_{i,c}\left(P_{gt_{i_c}} - P_{rec_{i_c}}\right)^2}{|P_{gt}|}},$$
4. |Pgt| is the size of the ground truth image (pixel count × number of spectral channels).

RRMSE

1. Relative root mean square error.
2. This indicator is calculated by dividing RMSE with average value of measured data.
3. $$rRMSE = \frac{1}{n}\sum_{i=1}^{n}\left(\sqrt{(I_E^{(i)} - I_G^{(i)})^2 / I_G^{(i)}}\right.$$
4. Relative RMSE by dividing the error by the ground truth pixels value, thus preventing a bias towards low errors in lower pixel values.

Loss Curves of Generators and Discriminators



Results and Evaluation

		NTIRE 2020 dataset (Primary)		CAVE dataset		NUS dataset	
		train	valid	train	valid	train	valid
Deep ConvNet	**RMSE**	16.451	17.998	17.451	16.998	16.221	17.023
	rRMSE	0.767	0.643	0.737	0.713	0.761	0.839
Hyper UNet	**RMSE**	5.148	5.023	7.993	7.148	5.392	5.109
	rRMSE	0.473	0.464	0.703	0.695	0.656	0.612
Res Net	**RMSE**	10.352	11.821	14.894	15.821	14.237	15.452
	rRMSE	0.631	0.642	0.731	0.732	0.791	0.603

Results and Evaluation

- **Comparing the HyperNet with existing models**

	NTIRE 2020				CAVE dataset				NUS dataset			
	Arad	A+	CNN	HyperNet	Arad	A+	CNN	HyperNet	Arad	A+	CNN	HyperNet
RMSE	–	–	–	5.085	5.61	2.74	2.55	7.0705	4.44	2.92	2.86	5.7505
rRMSE	–	–	–	0.418	0.499	0.426	0.469	0.769	0.190	0.142	0.152	0.684

Outputs Spectral Images Of HyperNet

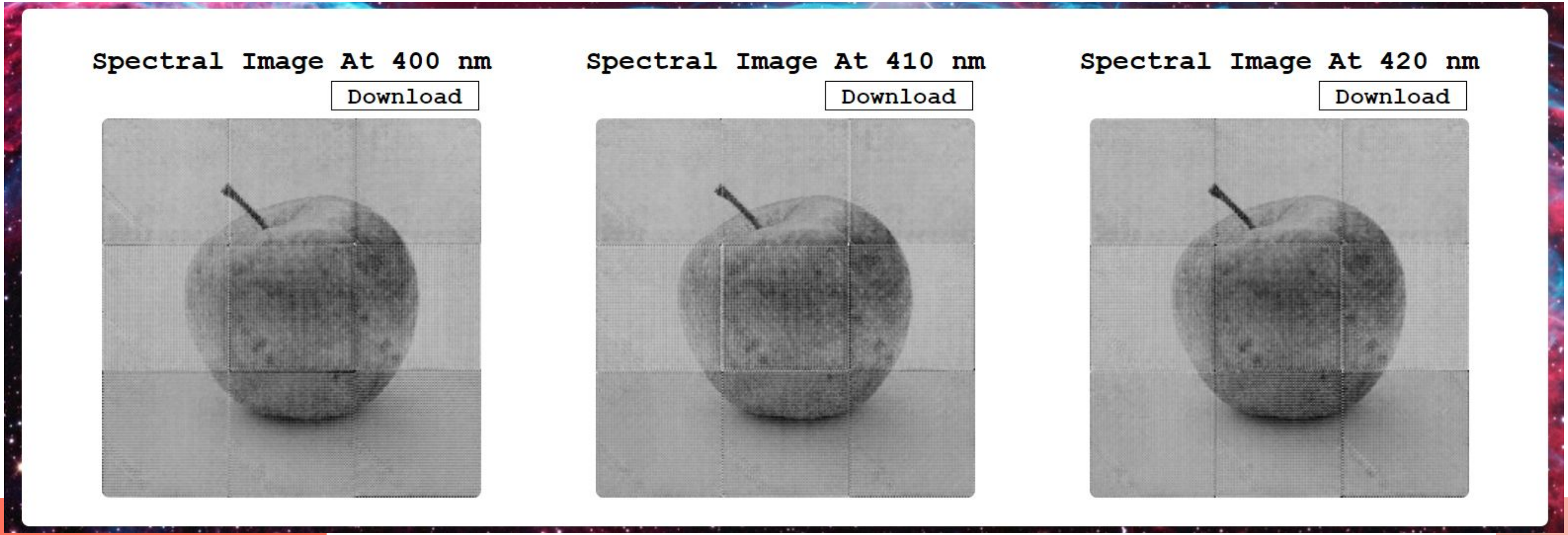
First image contains images from 430 nm, 440 nm, 450 nm.
Second image contains images from 560 nm, 570 nm, 580 nm.

Web Application Demo

- Web application was built using django framework.
- HyperUNet model weights are saved and used in the backend.

Web Application Demo

- This is the final Output Generated Images. We can download them directly.



References

- Can, Yigit Baran, and Radu Timofte. "An efficient CNN for spectral reconstruction from RGB images." arXiv preprint arXiv:1804.04647 (2018).
- Zhao, Yuzhi, et al. "Hierarchical regression network for spectral reconstruction from RGB images." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2020.
- Koundinya, Sriharsha, et al. "2d-3d cnn based architectures for spectral reconstruction from rgb images." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2018.
- Stiebel, Tarek, et al. "Reconstructing spectral images from rgb-images using a convolutional neural network." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2018.
- Arad, Boaz, et al. "Ntire 2020 challenge on spectral reconstruction from an rgb image." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2020.

References

- Goodfellow, Ian J., Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial networks." arXiv preprint arXiv:1406.2661 (2014).
- Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. "Image-to-image translation with conditional adversarial networks." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1125-1134. 2017.
- Liu, Xu, et al. "Multispectral image reconstruction from color images using enhanced variational autoencoder and generative adversarial network." IEEE Access 9 (2020): 1666-1679.
- Aeschbacher, Jonas, Jiqing Wu, and Radu Timofte. "In defense of shallow learned spectral reconstruction from rgb images." Proceedings of the IEEE International Conference on Computer Vision Workshops. 2017.

THANK YOU!!!!!