AeroRIT: A New Scene for Hyperspectral Image Analysis

Aneesh Rangnekar, Student Member, IEEE, Nilay Mokashi, Emmett Ientilucci, Senior Member, IEEE, Christopher Kanan, Senior Member, IEEE, and Matthew J. Hoffman

Abstract—Hyperspectral imagery oriented research like image super-resolution and image fusion is often conducted on open source datasets captured via point and shoot camera setups (ICVL [1], CAVE [2]) that have high signal to noise ratio. In contrast, spectral images captured from aircrafts have low spatial resolution and suffer from higher noise interference due to factors pertaining to atmospheric conditions. This leads to challenges in extracting contextual information from the captured data as convolutional neural networks are very noise-sensitive and slight atmospheric changes can often lead to a large distribution spread in spectral values overlooking the same object.

To understand the challenges faced with aerial spectral data, we collect and label a flight line over the university campus, AeroRIT, and explore the task of semantic segmentation. To the best of our knowledge, this is the first comprehensive large-scale hyperspectral scene with nearly seven million semantic annotations for identifying cars, roads and buildings. We compare the performance of three popular architectures - SegNet, U-Net and Res-U-Net, for scene understanding and object identification. To date, aerial hyperspectral image analysis has been restricted to small datasets with limited train/test splits capabilities. We believe that AeroRIT will help advance the research in the field with a more complex object distribution. We release the data and starter codes at https://github.com/aneesh3108/AeroRIT.

I. INTRODUCTION

For aerial scene analysis, standard RGB color cameras often do not suffice. Despite RGB cameras having a greater spatial resolution, the greater spectral resolution of hyperspectral imaging (HSI) systems can enable material properties to be better analyzed, especially at high altitudes. The fine spectral resolution records a contiguous spectrum, usually in steps of 1 or 5 nanometers (nm), that details the contents present in the scene and increases discrimination capability. However, HSI sensors are significantly more expensive than their RGB counterparts, leading to HSI data being restricted to domains such as precision agriculture and environmental monitoring. HSI enables material discrimination on a pixel level where panchromatic or RGB imagery fails due to limited spectral resolution.

Convolutional neural networks (CNNs) are now widely used for analyzing remote sensing imagery [3]. Hughes et al. used a siamese CNN architecture to match high resolution optical images with their corresponding synthetic aperture radar images [4]. Kleynhans et al. compared the performance of predicting top-of-atmosphere thermal radiance by using forward modeling with radiosonde data and radiative transfer modeling (MODTRAN [8]) against a multi-layer perception (MLP) and CNN and observed better performance from MLP and CNN in all experimental cases [5]. Kemker et al. used multi-scale independent component analysis and stacked convolutional autoencoders as self-supervised learning tasks before performing semantic segmentation on multispectral and hyperspectral imagery [6], [7].

Uzkent et al. adapted correlation filters trained on RGB images with HSI bands to successfully track cars in moving platform scenarios [3]. Hughes et al. used a siamese CNN architecture to match high resolution optical images with their corresponding synthetic aperture radar images [4]. Kleynhans et al. compared the performance of predicting top-of-atmosphere thermal radiance by using forward modeling with radiosonde data and radiative transfer modeling (MODTRAN [8]) against a multi-layer perception (MLP) and CNN and observed better performance from MLP and CNN in all experimental cases [5]. Kemker et al. used multi-scale independent component analysis and stacked convolutional autoencoders as self-supervised learning tasks before performing semantic segmentation on multispectral and hyperspectral imagery [6], [7].

The lack of diverse benchmarks in HSI makes understanding CNN functionality very difficult. The top three datasets, Indian Pines, Salinas and University of Pavia, have nearly distinctive end-members and hence, learning a discriminative boundary...
Mixed spectra, where multiple materials can present in single-pixel subject to ground sampling distance (GSD), are particularly challenging in remote sensing imagery due to their varying nature of the occurrence. Various spectral unmixing methods (survey: Bioucas-Dias et al. [9]) have been applied to separate mixed pixels but most assume the composition of all elements in the scene, referred to as end-members, are previously known. However, it is impossible to have all information about end-members when the scene is constantly changing. For example, in a moving camera setup with a push-broom sensor, each scene typically contains multiple colored cars and buildings, and applying spectral unmixing becomes difficult if the end-member signatures cannot be predetermined. We do not consider pseudo end-members for the scope of this paper and we do not tackle spectral unmixing as a problem, but address mixed pixels as noise uncertainty within the input that the CNNs ultimately learn to handle.

To test the discriminative potential for spectral data in CNNs and understand some of the above factors, we flew an aircraft equipped with a hyperspectral sensor and obtained multiple flight lines at different time stamps. We chose the flight line with the best combination of spatial and spectral quality and annotated every pixel in the - we named the collect 'AeroRIT' (Fig. 1a). We focus on being able to distinguish between 5 classes: 1) roads, 2) buildings, 3) vegetation, 4) cars, and 5) water. This is the first dataset having challenging mixed pixels as noise uncertainty within the input that the CNNs ultimately learn to handle.

A. Datasets for Hyperspectral Remote Sensing Imagery

Table I briefly reviews the current extent of aerial hyperspectral datasets available for analysis. Other hyperspectral datasets include ICVL (Arad and Ben-Shahar [1]) and CAVE (Yasuma et al. [2]) - however, we do not include them in the table as they are non-nadir and do not have pixel-wise labels for the data. The most commonly used aerial datasets are (1) Indian Pines, (2) Salinas Valley, and (3) Univ. of Pavia. The first two primarily contain vegetation and the third contains classes typically found around a university - for example, trees, soil, and asphalt. In all three cases, the small spatial extent often leads researchers to use Monte-Carlo (MC) cross-validation splits for benchmarking the performance of various CNN-based architectures. Recently, Nalepa et al. showed that MC splits can often lead to near-perfect results as there tends to be pixel overlap (leakage) between the training and test sets [10]. The paper also introduces a new routine that ensures minimum to no leakage between generated data splits. However, there is still a possibility of the network overfitting on the training set as the number of samples is significantly small (Table 1). In our scene, we label every pixel of the flight line and create an overall hyperspectral dataset package that contains the radiance image, reflectance image, and the semantic label for every pixel. We also provide a training, validation and test set that can be used to benchmark performance without the need for cross-validation splits. We describe the data collection for our scene in Section III.

B. Semantic Segmentation

Semantic segmentation in HSI is often treated as a pixel classification problem due to a lack of sufficient samples. Most approaches fall under three categories: (1) spectral classifiers, (2) spatial classifiers and (3) spectral-spatial classifiers. Hu et al. used 1D-CNNs to extract the spectral features of HSIs and establish a baseline [11]. The 1D-CNN takes a pixel spectral vector as an input, followed by a convolution layer and a max pooling layer to compute a final class label. Li et al. proposed to extract pixel-pair features and treats classification as a Siamese network problem [12]. Hao et al. designed a two-stream architecture, where stream1 used a stacked denoising autoencoder to encode the spectral values of each input pixel of a patch and stream2 used a CNN to process the patch’s spatial features [13]. Zhu et al. used a generative adversarial networks (GANs) to create robust classifiers of hyperspectral signatures [14]. Recently, Roy et al. proposed using a 3D-CNN followed by a 2D-CNN to learn better abstract level representations for HSI scenes [15]. We refer readers to Li et al. for an in-depth overview of recent methods for HSI classification [16]. As the above methods do not perform semantic segmentation in the truest sense (classification: encoder → class label, segmentation: encoder → decoder), we do not include them in our network comparisons.

III. AERO-RIT

The AeroRIT scene was captured by flying two types of camera systems over the Rochester Institute of Technology's
### Table I: Popular benchmark HSI datasets used for semantic segmentation (or pixel classification), with information on the spatial and spectral resolution. Our dataset is highlighted and as observed, is significantly bigger than its counterparts. (Acronyms: AVIRIS - Airborne visible/infrared imaging spectrometer, ROSIS - Reflective Optics System Imaging Spectrometer, HYDICE - Hyperspectral digital imagery collection experiment)

| Dataset          | Sensor       | Spatial Dimensions [px] | Spectral Dimensions [nm] | Spectral Bands | No. of classes |
|------------------|--------------|-------------------------|--------------------------|----------------|----------------|
| Indian Pines     | AVIRIS       | $145 \times 145$        | 400 - 2500               | 224            | 16             |
| Salinas Valley   | AVIRIS       | $512 \times 217$        | 400 - 2500               | 224            | 16             |
| Univ. of Pavia   | ROSIS        | $610 \times 340$        | 430 - 838                | 103            | 9              |
| KSC              | AVIRIS       | $512 \times 614$        | 400 - 2500               | 224            | 13             |
| Samson           | -            | $952 \times 952$        | 401 - 889                | 156            | 3              |
| Jasper Ridge     | AVIRIS       | $512 \times 614$        | 380 - 2500               | 224            | 4              |
| Urban            | HYDICE       | $307 \times 307$        | 400 - 2500               | 210            | 6              |
| AeroRIT          | Headwall Micro E | $1973 \times 3975$    | 397 - 1003               | 372            | 5              |

Fig. 2: Challenges present in the AeroRIT scene: (a,b) low resolution, (c,d) glint, and (e,f) shadow. Each figure shows the RGB-visualized hyperspectral chip and its corresponding semantic map.

university campus in a Cessna aircraft. The first camera system consisted of an 80 megapixel (MP), RGB, framing-type silicon sensor while the second system consisted of a visible near-infrared (VNIR) hyperspectral Headwall Photonics Micro Hyperspec E-Series CMOS sensor. The entire data collection took place over a couple of hours where the sky was completely free of cloud cover, except the last few flight lines at the end of the day where there was some sparse cloud cover. The aircraft was flown over the campus at an altitude of approximately 5,000 feet, yielding an effective GSD of about 0.4m for the hyperspectral imagery. The RGB data was ortho-rectified onto the Shuttle Radar Topography Mission (SRTM) v4.1 Digital Elevation Model (DEM) while the HSI was rectified onto a flat plane at the average terrain height of the flight line (that is, a low resolution DEM). Both data sets were calibrated to spectral radiancy in units of $W m^{-2} sr^{-1} \mu m^{-1}$. The pixels were labeled with ENVI using individual hyperspectral signatures and the geo-registered RGB images as references. As the RGB images do not form a continuous flight line (framing camera pattern) and are more in short burst captures format, we only labeled the hyperspectral scene and use it in our analysis.

Some important challenges associated with the scene are:

- **Low-resolution**: CNNs have been known to learn edge and color related features in the early to mid layers [17]. In our case, the low pixel resolution coupled with mixed pixels, makes discriminative feature learning relatively difficult. (Fig. 2a, 2b)

- **Glint**: Sun glint occurs due to bidirectional reflectance and the surface paint directly reflecting sunlight into the camera sensor. We observe this only occurs in certain parts of the imagery and is almost always associated with vehicles. As identifying pixels of the vehicles class is one of the end objectives, handling glint is an important topic. (Fig. 2c, 2d)

- **Shadows**: High rise structures (trees, buildings) often cast shadows that act as natural occlusions in scene understanding. Fig. 2f shows an image where a car is stationed right beside a building, but is nearly invisible to the human eye.

#### Conversion into reflectance data. We calculate the surface reflectance from the calibrated radiance image using the software, ENVI. Calibration panels were deployed in the scene during the various overpasses (Fig. 3). The reflectance of these black and white uniform calibration panels was measured using a field deploy-able point spectrometer. The panels were large enough to produce full pixels in the image data (i.e., minimal pixel mixing). These full pixels enabled us to produce a linear spectral (i.e., per-band) lookup table (LUT) for the mapping of radiance to reflectance. That is, an LUT is generated for every band. This in-scene technique is used.
Fig. 3: Targets (cyan) placed in the scene as calibration panels. We use the ground versus aerial signatures to draw a linear mapping between radiance and corresponding reflectance units.

Fig. 4: A comparison of signals obtained from the radiance and reflectance domains. As seen, radiance-\( a \) has a varying range of amplitudes while reflectance-\( b \) is restricted to the 0 – 100 percentage range. The x-axis on the graph denote the bands, and the y-axis on the graph denote the value.

often called the Empirical Line Method (ELM). One of the key assumptions with this technique is that the atmospheric mapping of radiance to reflectance over the in-scene panels used to define the mapping, also applies, spatially, to the rest of the image. This assumption holds fairly true for our case as the atmospheric conditions were so clear.

**IV. NETWORK ARCHITECTURES**

With respect to hyperspectral imagery, the model architectures are constrained by the following requirements: (1) They should be able to process low resolution features very well due to the nature of the data, (2) They should be able to propagate information to all layers of the network so that valuable information is not lost during sampling operations and, (3) They should be able to make the most out of limited data samples. The natural choice of selection would be a U-Net [19] (Fig. 5b), as the skip connections help propagate additional information from the encoder to the decoder. In technical terms, each skip connection concatenates all channels at layer \( i \) with those at layer \( n - i \), where \( n \) is the total number of layers.

A. Squeeze-and-Excitation block

This layer block (Fig. 5c) was proposed by Hu et al. to scale network responses by modeling channel-wise attention weights [21]. This is similar to a residual layer (Fig. 5c) used in ResNets, except that the latter focuses on spatial information as compared to channel information. The workings of this layer are as follows: For any given feature block \( x \), it is passed through global average pooling to obtain a channel feature vector, which embeds the distribution of channel-wise feature amplitudes while reflectance-\( a \) has a varying range of amplitudes while reflectance-\( b \) is restricted to the 0 – 100 percentage range. The x-axis on the graph denote the bands, and the y-axis on the graph denote the value.

\[ z_c = F_{\text{squeeze}}(x_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} x_c(i, j). \] (1)

This is followed by two fully connected layers \( (W_1, W_2) \) and a sigmoid layer \( (\sigma) \), in which the channel-specific weights can be learned through a self-gating mechanism (Eqn. 2):

\[ s = F_{\text{excite}}(z, W_1, W_2) = \sigma(W_2 \delta(W_1 z)), \] (2)

where \( \delta \) refers to the ReLU non-linearity [23], \( W_1 \in \mathbb{R}^{C \times C}, W_2 \in \mathbb{R}^{C \times r} \) and \( r \) is the reduction ratio to vary the capacity of the block. This is referred to as the excitation block. The output of the squeeze-and-excitation block is obtained by reshaping the learned channel weights (Eqn. 2) to the original spatial resolution and multiplying with the feature block:
\[ \tilde{x}_c = s_c x_c. \quad (3) \]

The final representation \( \tilde{x} \) is the combination of all \( \tilde{x}_c \) (Eqn. 3) and provides a more effective channel-weighted feature map that can be passed to the next set of layers.

### B. Conditional Generative Adversarial Networks

Conditional GANs (cGANs) were first proposed by Mirza and Osindero [24], and have been used widely for generating realistic looking synthetic images [25], [26], [27], [28]. We first discuss the base generative adversarial network (GAN) and then proceed to cGANs framework. A typical GAN (Fig. 3) consists of a generator (G) and a discriminator (D), both modeled by CNNs, tasked with learning meaningful representations to surpass each other. The generator learns to generate new fake data instances (e.g. images, audio signals) that cannot be distinguished from the real instances, while the discriminator learns to evaluate whether each instance belongs to the actual training dataset or is fake/synthetic (created by the generator). Formally, we can write the objective loss function as:

\[
\mathcal{L}_{GAN}(G, D) = \mathbb{E}_y[\log D(y)] + \mathbb{E}_z[\log(1 - D(G(z))],
\]

where the input to \( G \) is sampled from a noise distribution \( z \) (e.g. normal, uniform, spherical) and \( \mathbb{E}_y \) is the expectation over the sample distribution (in this case, \( y \)). The generator learns a mapping \( G : z \rightarrow y \) and tries to minimize the loss, while the discriminator tries to maximize it.

In a cGAN setting, the input to generator is no longer just from a noise distribution, but instead appended with a source label \( x \). It now learns a mapping \( G : \{x, z\} \rightarrow y \) and the corresponding loss function becomes:

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

Eqn. 5 shows us that the source label \( x \) is also passed on the discriminator, which uses this additional information to perform the same task as in GAN. We use the cGAN-based image to image translation framework of Isola et al. [28], with the final objective of the generator as follows:

\[
G^* = \arg \min_{G} \max_{D} \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{other}(G).
\]

that is, generate samples of a quality that lowers the discriminator’s ability to identify if the sample is from the real or fake distribution. The other loss in Eqn. 6 is an additional term imposed on the generator, which forces the generated image to be as close to the ground truth as possible. We use the standard L1-loss as \( \mathcal{L}_{other}(G) \).

### V. Experiments and Results

#### A. Experiment Configurations

We use the PyTorch library [29] for all our experiments. We split the scene into training, validation, and test as follows: the original flight line was 1973 × 3975. We drop the first 53 rows and 7 columns and get a flight line of 1920 × 3968. We use the first 1728 columns (and all rows) for training, the next 512 columns as validation and the last 1728 as the test split. We sample 64 × 64 patches (with 50% overlap) to create a training set and non-overlapping patches for validation and test set. Fig. 6 shows the number of samples present in each class – the scene is heavily imbalanced with reference to class cars. We adopt basic data augmentation techniques, random flip, and rotation, and extend the dataset by a factor of four. We use a batch size of 100, and train for 60 epochs with a learning rate of 1e-4. We also use a multi-step decay of factor 0.1 at epoch 40 and 50.

We sample every 10th band from 400 nm to 900 nm (i.e., 400 nm, 410 nm, ..., 900 nm) to obtain 51 bands from the entire band range. As the 372 band centers are not aligned in perfect order, we use ENVI for extracting near accurate bands centers. In preliminary experiments, we found that the last set of bands (from 900 nm to 1000 nm) did not provide useful discriminative information (intuitively due to the low signal to noise ratio in the channels), so they were removed for all experiments. We normalize all data between 0 to 1 by clipping to a max value of 214 (16384).

#### B. Loss and Metrics

We use weighted categorical cross-entropy to minimize the segmentation map and ignore the unspecified class label. The weights are calculated using median frequency balancing (Eigen and Fergus [30]), where the number of pixels in the scene belonging into a particular class are also taken into consideration. This helps overcome the class imbalance shown in Fig. 6.

![Ivy 2](image)

We use the following sets of metrics: overall accuracy (OA), mean per-class accuracy (MPCA), mean Jaccard Index (mIOU) and mean Srensen Dice coefficient (mDICE). They are defined as follows:

\[ mIOU = \frac{2TP}{2TP + FP + FN} \]

\[ mDICE = \frac{2TP}{2TP + FP + FN} \]

where TP, FP, and FN represent true positives, false positives, and false negatives, respectively.
where $n_{ij}$ is the number of pixels of class $i$ predicted as class $j$, where there are $n_{cl}$ different classes, and let $t_i = \sum_j n_{ij}$ be the total number of pixels of class $i$. OA and MPCA report the percentage of pixels correctly classified. However, they are still slightly prone to a dataset bias when class representation is small and hence, we also report mIOU and mDICE. mIOU is the class-wise mean of the area of intersection between the predicted segmentation and the ground truth divided by the area of union between the predicted segmentation and the ground truth. Correlated to mIOU, mDICE also focuses on intersection over union and is often used as a secondary metric for measuring a network’s performance on the task of semantic segmentation. We adopt mIOU as the primary metric for measure of performance.

### C. Model Hyperparameters

SegNet and U-Net both have encoders with 4 max pooling layers and gradually increasing channels by power of 2 ($C64 – MP – C128 – MP – C256 – MP – C512 – MP – BottleNeck$). Res-U-Net blocks are built upon U-Net and, conventionally, Res-U-Net ($N$) contains $N$ identity mapping residual blocks for better information passing. In our experiments, we use $N = 6$ and $N = 9$ following [28]. We also use smaller versions of SegNet and U-Net, called SegNet-m, U-Net-m, that drop the number of max pooling layers from 4 to 2 to compensate for the scene’s low spatial resolution and increase the channel by a factor of 2.

### D. Results

We compare all the models trained for the task of semantic segmentation in Table II. We observe that 6-block Res-U-Net achieves the best performance, but as U-Net-m has nearly four times fewer parameters and roughly the same performance, we adopt U-Net-m as the baseline in this study. We develop on this baseline and achieve a better performing U-Net-m version that outperforms all previous baselines.

We discuss the approaches used to further improve the performance of the U-Net-m architecture (Table III, Fig. 7). We adopt the Inception-variant of Squeeze-and-Excitation (SE) block [21] with a reduction ratio $r = 2$ - we do not sum the output of the SE block with the original channel space as a skip connection, but use it as the importance-weighted channel output. We add a SE block after every conv – batchnorm – relu combination on the encoder side of the network. This increases the U-Net-m performance by almost 4 points. We further replace every ReLU activation with parameterized ReLU (PReLU) [31] and observe a slight performance boost.

### VI. Challenges

Along with the promising performance that CNNs have achieved in the hyperspectral domain [33], [13], [14], [15], there are several important challenges that have not been previously addressed:

1) **Radiance or Reflectance:** We compared the performance of U-Net-m-SE on the radiance and reflectance sets of images and obtain an mIOU of nearly 5 points less when using reflectance (reflectance-mIOU: 69.90 vs radiance-mIOU: 75.35). We hypothesize that the difference in discriminative signatures (Fig. 5) might be one of the reasons behind the performance loss. As reflectance is the atmosphere rectified version and is theoretically less noisy, the performance drop is intriguing. However, as our primary domain of interest is the radiance
Fig. 7: Results with various additions to normal U-Net-m. The y-axis is the IOU measure. SE-block and its additions improves the performance of water pixel identification by nearly 20 points over the baseline, and the overall modifications improve the performance of car pixel identification by 8 points. Self-supervised learning is the factor that contributes to the large improvement in car pixel identification.

Fig. 8: Procedure for image reconstruction from a corrupted image. The generator is the network under consideration (U-Net-m-SE-PReLU), and the discriminator has 5 convolution layers followed by batch normalization and leaky ReLU.

Fig. 9: Successful cases: Outputs for a set of images among all networks. The racetrack image (row 2) shows that the cGAN trained network is the only one that is able to understand that the red unseen track patch is not a car or building.

Fig. 10: Failure cases: Outputs for a set of images through all trained networks. All networks predict building in between the road in Row 2, and misclassify zebra crossing as a car.

2) Data augmentation: We identify some of the tricky cases within the flight line in Fig. 10. Rows 1 – 3 show images partly under the shadow that have misclassified pixels, Row 4 shows a pedestrian crossing misclassified as a vehicle and Row 5 shows an object shining inside the fountain due to glint. The signatures vary heavily in amplitude due to shadows and glint and hence cause the networks to confuse between different classes. Conventional RGB augmentations (brightness, contrast) rely on the assumption of uniform scene illumination irrespective of image size. However, as hyperspectral signatures can vary drastically under varying atmospheric conditions and are subject to the adjacency effect, it is not possible to directly apply RGB-based augmentations for improving performance.

Hence, there is a need to understand the why and how of CNNs with respect to HSI.

3) Understanding the workings of CNNs: Learning task-relevant representations has been the forte of CNNs and a lot of techniques have been proposed to understand their internal workings [34], [35], [36]. However, none of these techniques have been applied towards hyperspectral imagery and advanced CNN architectures are treated as black-box approximators giving constant performance improvements. This approach is not favorable - knowing why a particular pixel has been classified as belonging to ‘cars’ or why limiting max pooling layers to 2 (and indirectly, the receptive field) boosted the performance, can in turn help design better architectures. Hence, there is a need to understand the why and how of CNNs with respect to HSI.
VII. Conclusion

This paper introduces AeroRIT, the first large-scale aerial hyperspectral scene with pixel-wise annotations. Our scene is nearly eight times bigger than the previously largest scene and is composed of challenging factors like shadows, glint and mixed pixels that make inference difficult. We trained networks for semantic segmentation and established a baseline using squeeze and excitation block, self-supervision and PReLU activation. We believe AeroRIT can be used for future work in multiple areas of remote sensing, including but not limited to data augmentation and network architecture designing.

REFERENCES

[1] B. Arad and O. Ben-Shahar, “Sparse recovery of hyperspectral signal from natural rgb images,” in European Conference on Computer Vision. Springer, 2016, pp. 19–34.

[2] F. Yasuma, T. Mitsunaga, D. Iso, and S. K. Nayar, “Generalized assorted pixel camera: postcapture control of resolution, dynamic range, and spectrum,” IEEE transactions on image processing, vol. 19, no. 9, pp. 2241–2253, 2010.

[3] B. Uzkenk, A. Rangnekar, and M. J. Hoffman, “Tracking in aerial hyperspectral videos using deep kernelized correlation filters.” IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 1, pp. 449–461, 2018.

[4] L. H. Hughes, M. Schmitt, L. Mou, Y. Wang, and X. X. Zhu, “Identifying corresponding patches in sar and optical images with a pseudo-siamese cnn.” IEEE Geoscience and Remote Sensing Letters, vol. 15, no. 5, pp. 784–788, 2018.

[5] T. Kleinmans, M. Montanaro, A. Gerace, and C. Kanan, “Predicting top-of-atmosphere thermal radiance using merra-2 atmospheric data with deep learning.” Remote Sensing, vol. 9, no. 11, p. 1133, 2017.

[6] R. Kemker and C. Kanan, “Self-taught feature learning for hyperspectral image classification.” IEEE Transactions on Geoscience and Remote Sensing, vol. 55, no. 5, pp. 2693–2705, 2017.

[7] R. Kemker, C. Salvaggio, and C. Kanan, “Algorithms for semantic segmentation of multispectral remote sensing imagery using deep learning.” ISPRS journal of photogrammetry and remote sensing, vol. 145, pp. 60–77, 2018.

[8] A. Berk, P. Conforti, R. Kennett, T. Perkins, F. Hawes, and J. Van Den Bosch, “Modtran6: A major upgrade of the modtran© radiative transfer code,” in 2014 6th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS). IEEE, 2014, pp. 1–4.

[9] J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, “Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches.” IEEE journal of selected topics in applied earth observations and remote sensing, vol. 5, no. 2, pp. 354–379, 2012.

[10] J. Nalpoto, M. Myller, and M. Kavulok, “Validating hyperspectral image segmentation.” IEEE Geoscience and Remote Sensing Letters, 2019.

[11] W. Hu, Y. Huang, L. Shi, F. Zhang, and H. Li, “Deep convolutional neural networks for hyperspectral image classification.” Journal of Sensors, vol. 2015, 2015.

[12] W. Li, G. Wu, F. Zhang, and Q. Du, “Hyperspectral image classification using deep pixel-pair features.” IEEE Transactions on Geoscience and Remote Sensing, vol. 55, no. 2, pp. 844–853, 2016.

[13] S. Hao, W. Wang, Y. Ye, T. Nie, and L. Bruzzone, “Two-stream deep architecture for hyperspectral image classification.” IEEE Transactions on Geoscience and Remote Sensing, vol. 56, no. 4, pp. 2349–2361, 2017.

[14] J. Zhou, Y. Chen, P. Ghamisi, and J. A. Benediktsson, “Generative adversarial networks for hyperspectral image classification.” IEEE Transactions on Geoscience and Remote Sensing, vol. 56, no. 9, pp. 5046–5063, 2018.

[15] S. K. Roy, G. Krishna, S. R. Dubey, and B. B. Chaudhuri, “Hybridrns: Exploring 3-d-2-d cm feature hierarchy for hyperspectral image classification.” IEEE Geoscience and Remote Sensing Letters, 2019.

[16] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, “Deep learning for hyperspectral image classification: An overview.” IEEE Transactions on Geoscience and Remote Sensing, 2019.

[17] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks.” in European conference on computer vision. Springer, 2014, pp. 818–833.

[18] V. Badrinarayanan, A. Kendall, and R. Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for image segmentation.” arXiv preprint arXiv:1511.00561, 2015.

[19] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, 2015, pp. 234–241.

[20] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2016, pp. 770–778.

[21] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7132–7141.

[22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in Advances in neural information processing systems, 2014, pp. 2672–2680.

[23] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in Proceedings of the 27th international conference on machine learning (ICML-10), 2010, pp. 807–814.

[24] M. Mirza and S. Osindero, “Conditional generative adversarial nets.” arXiv preprint arXiv:1411.1784, 2014.

[25] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution.” in European Conference on Computer Vision. Springer, 2016, pp. 694–711.

[26] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas, “Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks.” arXiv preprint arXiv:1612.03242, 2016.

[27] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang et al., “Photo-realistic single image super-resolution using a generative adversarial network.” arXiv preprint arXiv:1609.04802, 2016.

[28] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks.” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[29] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in PyTorch.” in NIPS Autodiff Workshop, 2017.

[30] D. Eigen and R. Fergus, “Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture.” in Proceedings of the IEEE international conference on computer vision, 2015, pp. 2650–2658.

[31] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.” in Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1026–1034.

[32] L. Jing and Y. Tian, “Self-supervised visual feature learning with deep neural networks: A survey.” arXiv preprint arXiv:1902.06162, 2019.

[33] R. Kemker, R. Luu, and C. Kanan, “Low-shot learning for the semantic segmentation of remote sensing imagery.” IEEE Transactions on Geoscience and Remote Sensing, vol. 56, no. 10, pp. 6214–6223, 2018.

[34] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization.” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2921–2929.

[35] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization.” in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 618–626.

[36] S. Srivivas and F. Fleuret, “Full-gradient representation for neural network visualization,” in Advances in Neural Information Processing Systems, 2019.