Complex-valued Butterfly Transform for Efficient Hyperspectral Image Processing

Utkarsh Singhal and Stella X. Yu

***Abstract*—Hyper-spectral imaging (HSI) is a critical remote sensing modality that captures high-resolution spectral information in addition to high spatial resolution. Due to its ability to capture rich information about the material properties of the target, HSI has found applications such as agriculture, ecological monitoring, urban planning, and medicine. However, HSI images tend to have high spatial resolution and many channels, thus requiring models that can integrate information over a large context and identify spectral signatures. HSI classification datasets are also highly imbalanced, sparsely labeled, and significantly smaller than standard vision datasets like ImageNet or CIFAR, thus motivating smaller models with high sample efficiency.**

This work combines the strengths of multi-scale representations and data-driven feature learning. We generalize the butterfly transform to a learned complex-valued butterfly layer, allowing for parameter-efficient extraction of hierarchical complex-valued features from 1D signals. This allows us to create a lean yet highly accurate hyperspectral image classification model. Benchmarked on the Indian Pines, ROSIS-03 Pavia University, and Salinas datasets, our method demonstrates accuracy on par with SSDGL while using 7x fewer parameters. On the most imbalanced and sparsely labeled dataset, our method outperforms SSDGL.

***Index Terms*—complex-valued deep learning, FFT, butterfly transform, signal processing, hyper-spectral imaging**

I Introduction

HYPER-SPECTRAL imaging (HSI) is an important remote sensing modality that extends regular photography by capturing high-resolution spectral information in addition to high spatial resolution. Recent advances in hyper-spectral imaging technology produce images with hundreds of channels, each capturing a narrow band of the electromagnetic spectrum. Since different materials have different reflectivity profiles (Figure 1), the spectral fingerprint contains rich information about the material properties of the target surface. As a result, hyperspectral images prove invaluable in remote sensing applications such as agriculture [1], ecological monitoring [2], urban planning [3], and medicine [4].

Hyper-spectral image classification is the task of classifying each pixel of an input image into a finite pre-defined set of classes. Specifically, we aim to learn a function $\mathbf{f} : \mathbb{R}^{H\times W\times C} \rightarrow \mathbb{R}^{H\times W\times L}$, which takes an image with height H and width W, and classifies each C-dimensional pixel of this image into L classes (see Figure 1). Unlike RGB images, HSI images have hundreds of channels.

HSI presents several challenges for machine learning approaches [5]. HSI images tend to have high spatial resolution and many channels, thus requiring models that can integrate information over a large context and identify spectral signatures. However, HSI classification datasets are also highly imbalanced, sparsely labeled (with as little as 5 pixels for some categories), and significantly smaller than standard vision datasets, thus motivating smaller models with high sample efficiency.

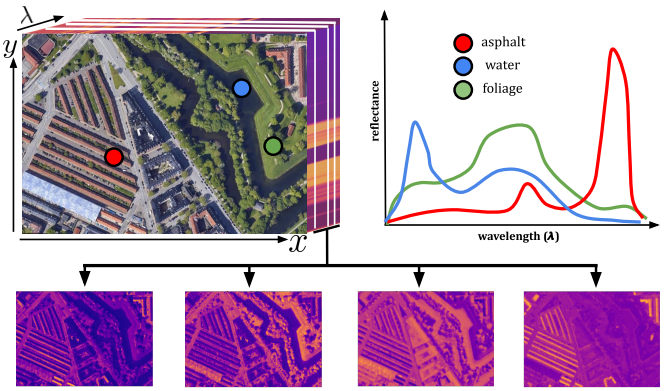

Fig. 1: A hyper-spectral image contains high-resolution measurements of both spectral and spatial information. **Top left**: Illustration of a hyper-spectral image as a 3D tensor (source image credit: Google Earth). **Top right**: Any single pixel contains spectral information which encodes a unique signature of the target material. This spectrum can be seen as a 1D signal. **Bottom**: Any single channel corresponds to a narrow band of the electromagnetic spectrum, and contains important spatial cues which may be more apparent in some bands than others. Different materials produce significantly different spectral signatures, and discriminative features extracted from these signals combined with spatial cues can enable accurate large-scale classification in remote sensing applications.

Recent advances in deep learning have made significant progress in HSI classification [6], [7]. *Spectral-Spatial Dependent Global Learning* (SSDGL) [7] introduces several changes, including hierarchically-balanced sampling for better training, a global joint attention module to extract salient information, and a convolutional LSTM module to process the spectral information in the neighborhood of each pixel. In particular, the ConvLSTM formulation treats the spectral information as a generic 1D sequence, building the representation piecemeal. This approach has a few drawbacks: LSTMs are notoriously difficult to train, and the LSTM only processes channel information at one scale. Additionally, the LSTM processes the spectral signature as a generic 1D sequence without taking into account its redundant structure. This redundant structure could be exploited to increase parameter efficiency further. This raises an important question:

Can we create a more parameter-efficient learned

representation for spectral signatures?

We propose a method that is parameter-efficient and considers the data at multiple scales. We take inspiration from the multi-scale representations in classic signal processing theory [8]. Representations like Fourier, Wavelet, and Scattering Transforms [9], [10] provide principled and efficient methods for processing 1D signals. In particular, the Fast Fourier Transform algorithm [8] computes the Discrete Fourier Transform of a sequence within $O(n \log n)$ time using a divide-and-conquer approach. The data-flow diagram is called a *butterfly diagram* (see Figure 3). These classic representations have a critical flaw: The basis vectors are fixed and uniformly represent the entire signal and frequency space, even though the data may vary in a small area or a few frequencies. For applications where the input data lies in a low-dimensional subset of inputs, a data-driven representation is thus of great utility.

Previous deep learning literature has successfully adapted this network topology for efficient deep learning [11]. However, these methods view the butterfly topology simply as an efficient linear layer. This view ignores the ability of Fourier butterflies to extract multi-scale features from 1D signals. The butterfly topology, which naturally implements hierarchical feature extraction, is especially suited to signal analysis applications such as HSI. Additionally, [11] considers only real-valued inputs and twiddle factors. This formulation cannot exploit the benefits of complex-valued algebra and has a limited class of expressible functions.

This work combines the strengths of complex-valued multi-scale representations and data-driven feature learning with a complex-valued butterfly transform.

Our contributions: 1) We generalize the idea of the Butterfly Transform [11] to create our *learned complex-valued butterfly* layer, which allows for parameter-efficient extraction of complex-valued multi-scale features from 1D signals, **2)** We extend the butterfly architecture to include local spatial information in the form of *butterfly convolutional layers* **3)** We replace the convolutional LSTM encoder used in SSDGL with our proposed layers, achieving a simpler and leaner model. We benchmark our new model on Indian Pines, Pavia University, and Salinas datasets, demonstrating accuracy on-par with [7] on each dataset while using **7x fewer** parameters. Additionally, on the most imbalanced and sparsely labeled dataset (Indian Pines), our method achieves higher accuracy than [7].

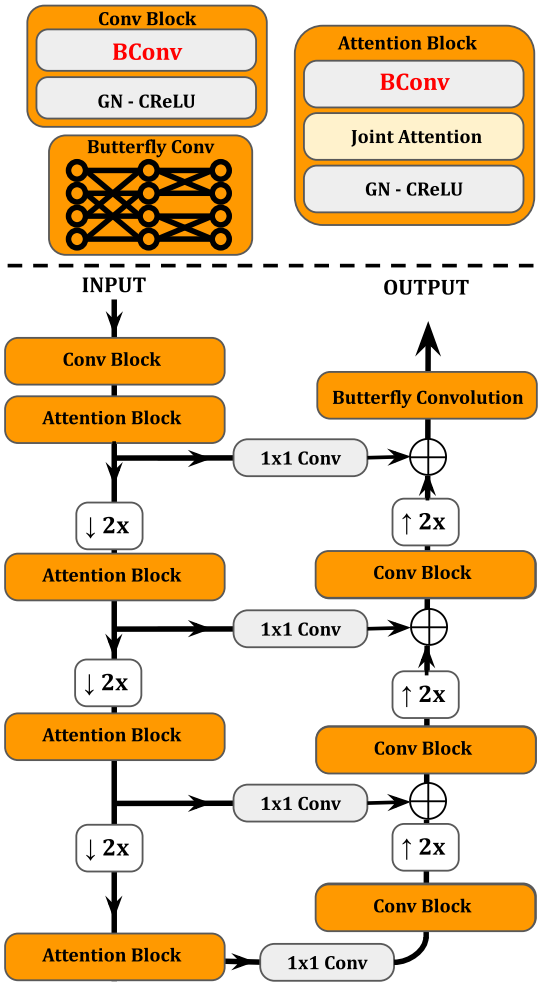

Fig. 2: **Our proposed architecture** is based SSDGL [7], but with large convolutions and ConvLSTM modules replaced with butterfly convolutions, resulting in a leaner yet highly-accurate model. This model works in four stages, each produced through a 2x downsampling of the previous layer. The features at each layer are processed using complex-butterfly layers and joint attention module before being aggregated and upsampled. The final result is a pixel-wise classification map. Our modifications on top of SSDGL are highlighted in orange/red (Butterfly Convolutions), and yellow (joint attention)

II Related Work

Hyper-spectral imaging: HSI as a remote sensing modality used in applications like agriculture [1], urban planning [3], and medicine [4]. In agriculture, hyper-spectral imaging can be used for crop health assessment, diseases monitoring, and soil quality estimation [1]. In urban planning, HSI allows for crucial tasks such as estimating land cover and agricultural development [12]. Hyper-spectral imaging has also proven useful for environmental monitoring and protection [13]. In geology, HSI approaches are used for large-scale mineral density mapping [2]. Common medical applications of HSI include organ segmentation and tissue diagnostics [4].

Deep learning for HSI classification: Recent deep learning approaches have led to significant improvements in classification accuracy for common HSI datasets [6], [7], [14], [15]. Traditional CNN-based methods like [14], [16], [17] struggle with the large number of channels in HSI. Recurrent variants like [15], [18] use RNNs, and SSGDL [7] uses global convolutional LSTM modules to process spectral information. However, this approach results in models of higher architectural complexity, and recurrent neural networks can lead to training instability. Another important challenge in HSI classification is the lack of labeled data, which has been tackled through semi-supervised and unsupervised learning methods [19]–[21].

Finally, in contrast to popular semantic segmentation tasks, HSI images are sparsely labeled. The small number of labels results in lower gradient diversity due to redundant gradients, and the resulting models experience worse convergence. [6], [7] tackle this problem through a *stratified pixel sampling strategy*. In this strategy, the set of labeled pixels is sampled in a stratified manner, ensuring a balanced distribution of pixel samples from each class and a higher gradient diversity. We refer the reader to [7] for a more detailed review of recent deep learning advances for HSI classification.

Complex-valued processing: Complex-valued representations are a cornerstone of engineering and physics [8], [22], and complex analysis has a rich mathematical history [23]. In the field of deep learning, complex-valued representations have shown similar utility. [24] shows that a single complex-valued neuron can solve the XOR problem, indicating higher representational capacity than real-valued counterparts. [25] analyzes the form of critical points in a hierarchical complex-valued network, showing that all critical points are saddle points for non-regular loss functions. [26] evaluates single layer complex-valued neural networks on problems such as symmetry detection. [27], [28] encode geometric properties of data such as depth ordering and confidence in a complex vector and learn a metric for this embedding. [29] discusses convergence and stability guarantees for complex-valued neural networks and shows better generalization for simulated and real-world problems. [30] uses complex values to encode spike timing information in a neural network. [31] tackles the problem of ultrasound image reconstruction using complex-valued CNNs. [32] addresses the problem of complex-valued scaling ambiguity by building complex-scale equivariant and invariant layers. We refer the reader to [33] for a more detailed account of the benefits of complex-valued deep learning. Our work largely builds upon [32], [33].

Butterfly Transform and FFT: Fast Fourier Transform [8], also commonly known as the Cooley-Tukey algorithm, is an efficient method for computing the Discrete Fourier Transform of a finite-length vector. FFT follows a divide-and-conquer scheme, decomposing the DFT of a single vector into smaller sub-vector DFT computations. These sub-vector DFTs are scaled using fixed *twiddle factors* and linearly combined to form the full DFT. Due to the divide-and-conquer nature of this computation, FFT achieves $O(n \log n)$ complexity as opposed to $O(n^2)$ achieved by a naive matrix-multiplication algorithm. However, Fourier Transform is limited to extracting a set of fixed, complete, and orthogonal features, each with the same amount of spectral energy. In practical signal-analysis applications *dimensionality reduction* may be desired, and in contrast to the Fourier Transform, learned layers are capable of adapting to the underlying data and using only necessary features. Butterfly Transform [11] uses a similar network topology as FFT butterflies, additionally using learned twiddle factors to create a linear layer with $O(n \log n)$ complexity. However, the twiddle factors in [11] are real-valued, preventing this layer from exploiting the benefits of complex-valued algebra. The real-valued twiddle factors also limit the class of functions expressed by this layer, introducing additional challenges to learning spectral features. Finally, we note that the Butterfly Transform has been studied primarily as a substitute for fully connected layers, and its applications to signal processing have been overlooked.

III Complex-Valued Butterfly Transform

In this section, we describe the generalization of the Butterfly transform [11] to complex-valued matrices. For a complex-valued, n-dimensional input vector $\mathbf{x} \in \mathbb{C}^n$, complex-valued butterfly layer can be written as a block matrix $\mathbf{B}^{(n,k)} \in \mathbb{C}^{n \times n}$ with each block of size $\frac{n}{k} \times \frac{n}{k}$. We define $\mathbf{B}^{(n,k)}$ as:

$$\mathbf{B}^{(n,k)} = \begin{bmatrix} \mathbf{B}_1^{(\frac{n}{k},k)}\mathbf{D}_{11} & \dots & \mathbf{B}_1^{(\frac{n}{k},k)}\mathbf{D}_{1k} \\ \vdots & \ddots & \vdots \\ \mathbf{B}_k^{(\frac{n}{k},k)}\mathbf{D}_{k1} & \dots & \mathbf{B}_k^{(\frac{n}{k},k)}\mathbf{D}_{kk} \end{bmatrix} \tag{1}$$

where each $\mathbf{D}_{ij} \in \mathbb{C}^{(\frac{n}{k} \times \frac{n}{k})}$ is a learned complex-valued diagonal matrix, and each $\mathbf{B}_i^{(\frac{n}{k},k)}$ is another complex-valued butterfly matrix of size $\frac{n}{k} \times \frac{n}{k}$. This recursive structure is responsible for the parameter efficiency of this layer.

Thus, given an input vector $\mathbf{x} \in \mathbb{C}^n$, our method computes $\mathbf{B}^{(n,k)}\mathbf{x}$. As shown in [11], this computation can be simplified as:

$$\mathbf{B}^{(n,k)}\mathbf{x} = \begin{bmatrix} \mathbf{B}_1^{(\frac{n}{k},k)} \sum_{j=1}^{k} \mathbf{D}_{1j}\mathbf{x}_j \\ \vdots \\ \mathbf{B}_k^{(\frac{n}{k},k)} \sum_{j=1}^{k} \mathbf{D}_{kj}\mathbf{x}_j \end{bmatrix} \tag{2}$$

where each $\mathbf{x}_i$ is a $\frac{n}{k}$-dimensional sub-vector of $\mathbf{x}$. This parametrization uses $O(n)$ parameters at each level of recursion, resulting in a total of $O(n \log n)$ parameters used to define the entire matrix. This parametrization uses significantly fewer parameters for large n such as in HSI while extracting rich features. We note that the critical difference between our proposed layer and [11] is that our version is complex-valued and is thus able to learn a larger class of functions.

Butterfly Convolutions: The formulation described in Equation 2 works for vectors, and is thus equivalent to a 1×1 convolution. For HSI, this limits the butterfly layer to using only spectral information for each pixel. We extend this formulation by adding a spatial component to Equation 2. Given an input patch of size $m \times m$ indexed by pixel p, we define the new *spatial-spectral butterfly transform* as:

$$\mathbf{B}^{(n,k)}\mathbf{x} = \begin{bmatrix} \mathbf{B}_1^{(\frac{n}{k},k)} \sum_{j=1}^{k} \sum_{p=1}^{m \times m} \mathbf{D}_{1jp}\mathbf{x}_j^{(p)} \\ \vdots \\ \mathbf{B}_k^{(\frac{n}{k},k)} \sum_{j=1}^{k} \sum_{p=1}^{m \times m} \mathbf{D}_{kjp}\mathbf{x}_j^{(p)} \end{bmatrix} \tag{3}$$

Here, $x_j^{(p)}$ refers to the j-th sub-vector of p-th pixel in the patch, and $\mathbf{D}_{ijp}$ is a 3-dimensional tensor containing complex-valued twiddle factors. Here, the extra dimension indexes over pixels. Unlike Equation 2 which only considers a single pixel, this formulation sums over pixels in a $m \times m$ neighborhood, thus utilizing both spectral and spatial information.

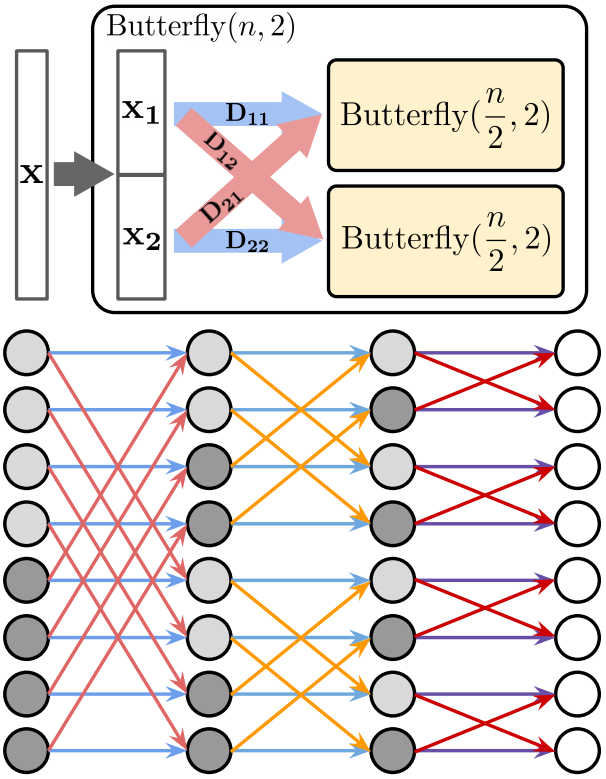

Fig. 3: **Complex Butterfly Transform**. **top:** Block diagram of the layer. The butterfly transform is defined by a sparse block-wise operation on the sub-vectors of the input vector $\mathbf{x}$, followed by recursive calls to butterfly transform on each sub-vector. Given a vector $\mathbf{x}$ with sub-vectors $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}$, we define complex-valued diagonal matrices $D_{11}, D_{12}, \ldots, D_{kk}$ containing the learned twiddle factors. These matrices act on the corresponding sub-vectors, and the resulting combined sub-vectors are processed through recursive calls to Butterfly$(\frac{n}{2}, 2)$. [11] uses a similar formulation, but with real-valued twiddle factors. This limits the class of functions learned by the butterfly transform. Our complex-valued twiddle factors allow for a larger class of functions. In order to include spatial information, we also introduce aggregation over patches instead of single pixels (see Equation 3). **bottom:** A network topology diagram of the butterfly layer for a vector with length 8 and base 2. Each edge represents a complex-valued twiddle factor, and the valued represented by all edges coming to each node are summed to determine the value contained in the node. Different node colors indicate different groups and use different diagonal matrices. In the general case, for a butterfly transform with base k, each intermediate node has k input and output edges. After multiplication by twiddle factors, butterfly transforms are applied to the sub-vectors recursively.

In order to further reduce parameter consumption, we decompose the term $\mathbf{D}_{ijp} = \mathbf{d}_p \mathbf{D}_{ij}$. This decomposed version is equivalent to performing a complex-valued butterfly transform on each pixel followed by depth-wise convolution. This reduces the total parameter count from $O(m^2 n \log_k(n))$ to $O(m^2 + n \log_k(n))$. As our approach is complementary to the interleaving approach used in [11], we combine these approaches into a butterfly convolution block.

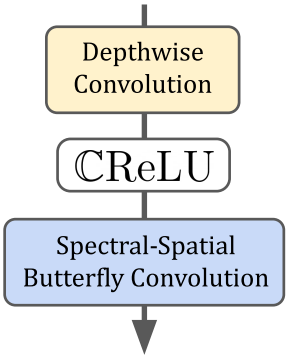

Fig. 4: **Complex Butterfly Conv Block**: Since the naive butterfly transform can only perform 1×1 convolutions, [11] interleaves the butterfly transform with depth-wise separable convolution layers. Our butterfly convolution formulation accumulates over a local patch in addition to using complex-valued butterfly transform on the channels. This formulation is complementary to this approach, so we combine the two methods to form the Complex Butterfly Conv Block (CBConv).

IV Architecture

Our model is a fully convolutional neural network based on SSDGL [7]. It consists of four stages, each operating at a different resolution. We use a series of 2×2-strided complex-butterfly convolutions and joint-attention blocks to downsample the input image by 2x at each stage. The features at each stage are processed using a 1×1 convolution and then up-sampled using bilinear sampling. We use $\mathbb{C}$ReLU [33] as the non-linear activation. We refer the reader to Figure 2 for an architecture diagram. Our proposed model uses only $0.31M$ parameters, 7x fewer than SSDGL. This section describes how we modify each SSDGL module to allow for complex-valued processing.

Joint attention Module: We follow [7], combining the spectral and spatial attention layers into a single block. However, unlike [7], our proposed Joint Attention Module does not rely on a Convolutional LSTM. We replace this LSTM with a Complex Butterfly Convolution, which can extract complex-valued hierarchical features in a feature-efficient manner. We also modify the spectral and spatial attention submodules for complex-valued processing.

Spectral Attention: We modify the spectral attention module described in SSDGL [7] to make it compatible with complex-valued features. The role of this module is to re-weight different parts of the spectrum, modulating each channel by a scaling constant which lies between 0 and 1. The spectral attention module first aggregates features over all the pixels to estimate these channel-wise scaling constants. This module uses two types of aggregation: global average pooling and global max pooling. While averages are well-defined for complex number sets, the same is not true for maximum since the complex

plane cannot be well-ordered. As a result, there are multiple useful definitions of the "maximum" operation for complex numbers and thus multiple possible max-pooling strategies. We note that in CNNs, the salience of a feature is represented by its magnitude. This motivates the "Magnitude MaxPool" strategy of [32], which aggregates the elements with the largest magnitude (and thus the highest salience). Following [7], the averaged-pooled features are processed using complex-valued MLP, added together, and passed through a thresholding function. We apply the sigmoid thresholding function to the real-valued part of the feature vector as this operation incorporates both magnitude and phase information.

For a complex-valued input feature $\mathbf{f} \in \mathbb{C}^{H \times W \times C}$ of height H and width W and containing C channels, we define the complex spectral attention layer as:

$$M_c(\mathbf{f}) = \sigma(\mathfrak{Re}\big[(N_1(Avg(\mathbf{f})) + N_2(MagPool(\mathbf{f}))\big]) \quad (4)$$

where N_1 and N_2 are complex-valued MLPs, and $\mathfrak{Re}\big[\cdot\big]$ extracts the real part of a complex-valued input.

Spatial Attention: We also create a complex-valued version of SSDGL's Spatial Attention layer. This module selectively re-weights different input image pixels, highlighting relevant areas by multiplying them with 1 and eliminating features from irrelevant areas by multiplying them with 0. For each pixel, we aggregate information over all channels to produce a single-channel attention map $\mathbf{M_s} \in \mathbb{C}^{H \times W}$. Like Channel Attention, we do this through two methods: averaging the channels and computing the maximum absolute value over channels. These two feature maps are concatenated and processed using a convolutional layer and the complex thresholding function to produce a spatial attention map.

For a complex-valued input feature $\mathbf{f} \in \mathbb{C}^{H \times W \times C}$ of height H and width W and containing C channels, we define the complex-valued spatial attention layer as:

$$M_s(\mathbf{f}) = \sigma(\mathfrak{Re}\big[(L(AvgPool(\mathbf{f}); MagPool(\mathbf{f})\big]) \quad (5)$$

where L is complex-valued convolution layer, and $\mathfrak{Re}$ extracts the real part of a complex-valued input.

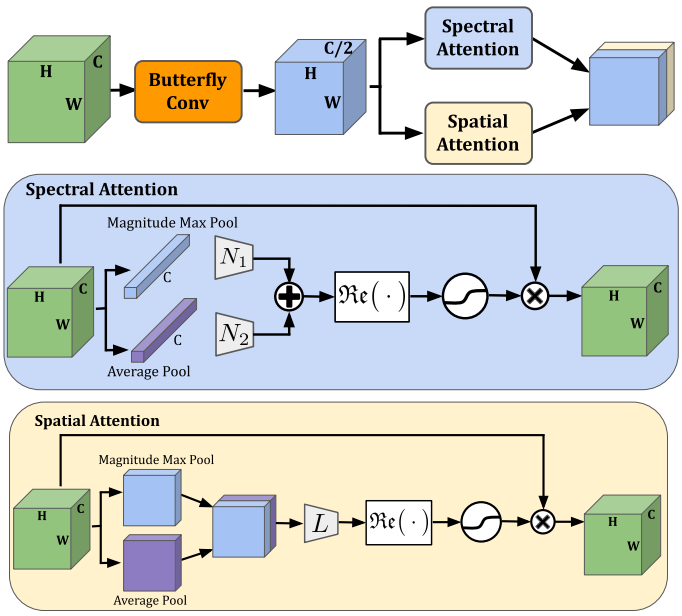

Fig. 5: Visualization of our modified version of JAM [7]. **Joint Attention Module**: The joint attention module combines spatial and spectral attention into a single block. Our proposed version uses a complex butterfly convolution instead of ConvLSTM [7]. **Spectral attention**: The spectral attention layer generates a weighting between 0 and 1 for each individual channel. Features from the entire image are pooled using average and magnitude max pooling, and the resulting features are summed and thresholded in order to predict the channel-wise modulation. **Spatial attention**: The spatial attention layer generates a weighting between 0 and 1 for each individual pixel. The feature vector corresponding to each pixel is aggregated using average and magnitude max pooling, and the resulting features are concatenated and processed through a convolutional layer. The final feature map is then thresholded in order to predict the pixel-wise modulation.

V Experiments

We benchmark our model on three datasets: **Pavia University** (Figure 6), **Indian Pines** (Figure 7), and **Salinas** (Figure 8). Indian Pines is the most imbalanced, with as little as 5 pixels of training data in some categories, whereas Salinas and Pavia are more balanced. In this section, we describe each dataset and the corresponding experimental results. Our method is on-par with SSDGL on Salinas and Pavia, but outperforms SSDGL on Indian Pines.

V-A Indian Pines Dataset

The Indian Pines [34] dataset consists of an aerial image captured in Indiana, the USA in 1992 using the AVIRIS spectrometer. This dataset contains a single 145×145 pixel image with 220 channels. This image has a spatial resolution of 20 m, and the channels correspond to the EM spectrum in the visible and IR range, spanning wavelengths from 400 nm to 2500 nm. Each labeled pixel is classified into 16 different types of land cover for pixel-wise classification, including various crops like alfalfa, corn, and wheat. This process is followed by background pixel removal, retaining $10,249$ pixels. Figure 7 contains a visualization of the original dataset, masked ground truth, and test predictions made by our model and SSDGL [7].

Following the training strategy of SSDGL [7], we use 5% of the labeled pixels as training data, sampled using the hierarchically balanced pixel sampling strategy. We evaluate our method using various accuracy statistics and tabulate them in Table I. The tabulated metrics include per-class test accuracy, average accuracy, overall accuracy, and kappa coefficient. Despite being significantly leaner, our method achieves the highest accuracy in every class, including the average metrics.

V-B Pavia University Dataset

The Pavia University dataset [35] contains an image of the University of Pavia, Italy, captured using the Reflective Optics System Imaging Spectrometer (ROSIS). This 610×340

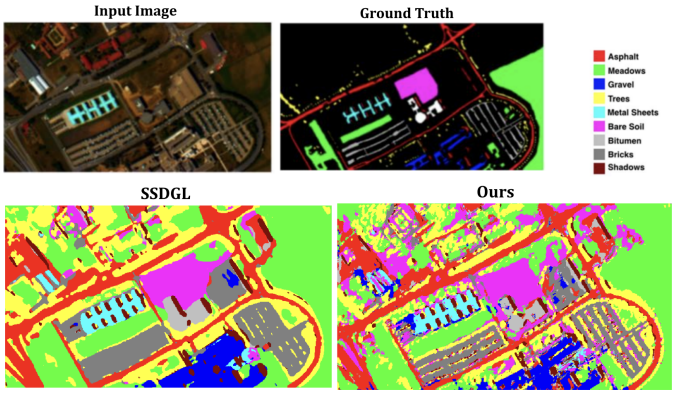

Fig. 6: **Pavia University dataset**: Test predictions made by our model and SSDGL. The ground truth image shows the segmentation, with the black color designating unlabeled areas. While both methods make accurate predictions in the labeled areas, SSDGL produces significantly coarser maps. For example, SSDGL completely misses the large groups of trees in the parking lot in the lower-left half of the image. Likewise, SSDGL's coarse segmentation map groups several entities into larger blobs in several other areas. This indicates our method is comparatively more sensitive to fine changes in spectral information than SSDGL, which relies heavily on spatial priors. Multi-band composite image and legend reproduced from [7].

image has 103 bands covering visible and near-IR parts of the EM spectrum, with wavelengths in the $430\,\text{nm}$-$860\,\text{nm}$ range. This dataset contains $42,776$ labeled pixels corresponding to 9 classes, and we use 1% of this data for training. Figure 6 visualizes test predictions for our model and SSDGL. We evaluate the models using various accuracy metrics and tabulate them in Table III. On this dataset, our performance closely matches that of SSDGL, and in all classes and metrics except *Asphalt*, our method stays within 0.01% of SSDGL.

V-C Salinas Dataset

The Salinas dataset [35] consists of an aerial image from Salinas Valley, California, acquired using the AVIRIS sensor. This dataset contains a single 512×145 image with 224 channels. The channels cover the infrared and visible spectrum, with wavelengths in the $400\,\text{nm}$-$2500\,\text{nm}$ range. This dataset contains 16 classes corresponding to crops like corn, lettuce, and grapes. Figure 8 includes test prediction visualizations for our method along with SSDGL.

We follow the same training configuration as for the Indian Pines dataset and show the results in Table II. We note that the Salinas dataset has been saturated due to recent methods achieving nearly 100% accuracy. Our method follows the same trend as SSDGL [7] and FPGA [6], staying within 0.03%.

VI Conclusion

We propose a learned complex-valued butterfly convolution. In contrast to the Fourier Transform, our method can learn useful features from the data. Our approach is compatible with complex-valued deep learning and capable of learning a larger

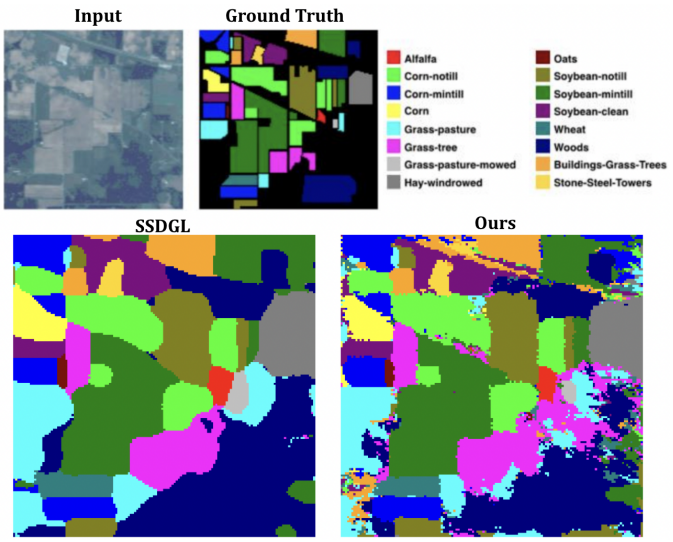

Fig. 7: **Indian Pines dataset**: Test predictions from our model and SSDGL. This dataset contains a satellite image with various crops and land types labels. The ground truth image indicates the segmentation, with the black color designating unlabeled areas. Both methods achieve nearly perfect accuracy in the labeled areas, but our method produces sharper segmentation maps compared with SSDGL. Multi-band composite image and legend reproduced from [7].

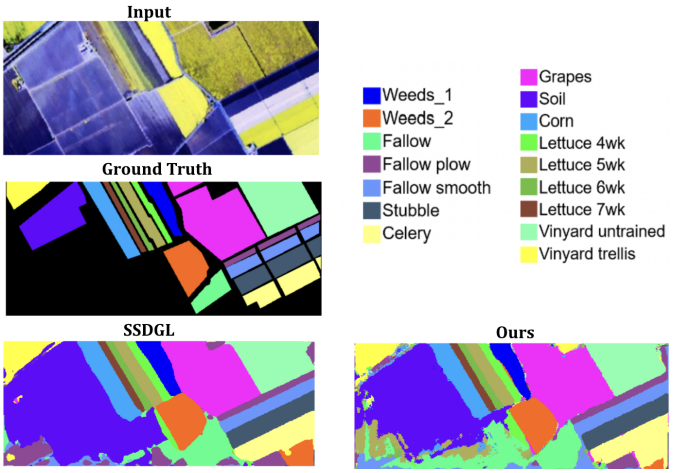

Fig. 8: **Salinas dataset**: This dataset contains a high resolution image with various crop types labeled in the ground truth segmentation map. Black color designates unlabeled areas, whereas each other color designates a crop or land type. Multi-band composite image and legend reproduced from [6].

set of features compared to Butterfly Transform. This combination of complex-valued multi-scale feature representation and data-driven feature learning allows for leaner yet accurate models on HSI classification. On the most imbalanced and sparsely labeled dataset (Indian Pines), our method outperforms SSDGL [7]. These results demonstrate the benefits of complex-valued butterfly transforms in signal processing and indicate exciting directions in complex-valued deep learning.

TABLE I: Results for Indian Pines Dataset. Our model achieves the highest accuracy for every class and accuracy metric (OA, AA, Kappa). despite using significantly fewer parameters compared to SSDGL. Baseline accuracy numbers reproduced from [7].

Class	CNN-based					FCN-based			
	SVM	SS-CNN	SSRN	DBMA	MCNN	U-Net	FPGA	SSDGL	Ours
1	70.32	72.14	75.57	90.37	94.36	97.67	97.22	**100.0**	**100.0**
2	69.63	90.42	90.65	92.72	92.84	92.48	93.07	99.63	**99.92**
3	58.26	81.48	97.01	95.63	93.02	84.77	89.46	99.24	**99.61**
4	45.22	71.23	93.36	89.35	95.32	89.33	**100.0**	**100.0**	**100.0**
5	75.48	83.62	98.56	96.92	92.13	81.00	95.63	99.56	**99.78**
6	96.14	97.19	98.94	99.18	98.86	94.08	97.56	**100.0**	**100.0**
7	95.79	91.01	84.21	79.57	84.83	**100.0**	**100.0**	**100.0**	**100.0**
8	87.72	92.34	98.36	99.11	98.63	98.90	**100.0**	**100.0**	**100.0**
9	75.03	96.39	97.61	97.91	92.47	78.95	**100.0**	**100.0**	**100.0**
10	66.25	81.75	81.03	92.08	94.76	89.49	96.64	99.68	**100.0**
11	77.62	87.39	93.02	95.15	96.28	97.81	96.74	99.36	**99.66**
12	67.28	83.03	95.72	90.71	94.12	86.50	91.65	99.11	**99.64**
13	96.93	97.42	99.81	99.81	96.95	98.97	**100.00**	**100.00**	**100.0**
14	95.07	95.31	95.79	97.11	98.79	98.58	99.91	**100.00**	**100.0**
15	35.48	74.04	92.25	88.13	92.83	92.08	99.72	**100.00**	**100.0**
16	97.61	94.61	96.57	97.05	87.32	93.18	**100.00**	**100.00**	**100.0**
OA	75.31	89.82	92.21	94.43	94.78	93.20	96.18	99.63	**99.85**
AA	71.12	83.73	93.03	93.81	93.37	92.11	97.33	99.79	**99.91**
Kappa	0.7173	0.8783	0.9115	0.9365	0.9437	0.9222	0.9564	0.9958	**0.9982**

TABLE II: Results for the Salinas Dataset. Our model achieves close to perfect accuracy (within 0.03%), whereas SSDGL achieves 100% accuracy on every metric. Baseline accuracy numbers reproduced from [6].

Class	Patch-based					Patch-free		
	SVM	S-CNN	Gabor-CNN	DFFN	3D-GAN	FPGA	SSDGL	Ours
1	99.61	**100**	**100**	99.99	75.35	**100**	**100**	**100**
2	99.69	97.75	88.06	99.94	98.49	**100**	**100**	**100**
3	99.56	98.88	99.25	**100**	**100**	**100**	**100**	**100**
4	99.41	**100**	**100**	**100**	99.76	99.92	**100**	**100**
5	98.72	99.93	98.88	99.11	**100**	**99.11**	**100**	**100**
6	99.77	89.48	**100**	99.95	99.97	**100**	**100**	99.97
7	99.52	99.30	98.94	99.43	99.45	**100**	**100**	**100**
8	76.74	98.69	99.48	99.56	98.30	**99.94**	**100**	99.99
9	99.37	99.34	97.27	**100**	99.95	**100**	**100**	**100**
10	95.13	100	99.21	99.79	99.79	99.77	**100**	**100**
11	99.32	99.93	**100**	99.48	**100**	**100**	**100**	**100**
12	99.70	99.89	**100**	99.84	**100**	**100**	**100**	**100**
13	99.15	88.62	**100**	99.96	**100**	**100**	**100**	**100**
14	98.38	88.72	99.79	99.95	**100**	**100**	**100**	**100**
15	75.56	90.62	94.31	99.45	99.52	99.99	**100**	99.96
16	99.23	99.96	93.38	99.96	90.25	99.88	**100**	**100**
OA	90.92	97.62	97.63	99.71	98.22	99.92	**100**	99.99
AA	96.18	96.94	98.04	99.78	97.75	99.91	**100**	**100**
Kappa	0.8985	0.9510	0.9734	0.9967	0.9793	0.9991	**1.0**	0.9999

TABLE III: Results for Pavia University Dataset. Our model shows on-par accuracy compared to SSDGL, with OA, AA, and Kappa within 0.01% of the accuracy demonstrated by SSDGL [7]. Baseline accuracy numbers reproduced from [7].

Class	CNN-based					FCN-based			
	SVM	SS-CNN	SSRN	DBMA	MCNN	U-Net	FPGA	SSDGL	Ours
1	85.82	92.21	97.41	96.26	96.53	94.06	97.83	**100.0**	99.74
2	96.02	90.27	99.10	98.31	99.26	98.67	99.95	**100.00**	99.99
3	65.46	79.82	88.61	89.16	87.15	78.30	91.28	**100.00**	99.90
4	81.24	92.67	99.81	97.53	93.68	96.01	95.85	**99.67**	**99.67**
5	99.23	99.41	100.00	99.72	97.26	**100.00**	**100.00**	**100.00**	**100.0**
6	67.69	86.86	95.51	97.98	96.41	99.80	99.76	**100.00**	**100.0**
7	53.86	79.85	92.16	85.51	88.74	79.56	99.73	**100.00**	**100.0**
8	86.28	92.83	89.03	80.70	93.81	99.58	98.05	**99.92**	**99.92**
9	99.92	93.93	99.96	91.12	93.79	99.25	97.86	**100.00**	**100.0**
OA	86.54	90.14	96.55	95.13	96.28	96.09	98.68	**99.97**	99.92
AA	73.52	83.52	95.73	93.48	94.69	93.47	97.82	**99.95**	99.91
Kappa	0.8192	0.8768	0.9543	0.9353	0.9481	0.9480	0.9825	**0.9996**	0.9990

Acknowledgements: This work was supported, in part, by US Government fund through Etegent Technologies on Automated Learning from Unsupervised Repositories of Data.

REFERENCES

[1] T. Adão, J. Hruška, L. Pádua, J. Bessa, E. Peres, R. Morais, and J. J. Sousa, "Hyperspectral imaging: A review on uav-based sensors, data processing and applications for agriculture and forestry," *Remote Sensing*, vol. 9, no. 11, p. 1110, 2017.

[2] R. Resmini, M. Kappus, W. Aldrich, J. Harsanyi, and M. Anderson, "Mineral mapping with hyperspectral digital imagery collection experiment (hydice) sensor data at cuprite, nevada, usa," *International Journal of Remote Sensing*, vol. 18, no. 7, pp. 1553–1570, 1997.

[3] C. Weber, R. Aguejdad, X. Briottet, J. Avala, S. Fabre, J. Demuynck, E. Zenou, Y. Deville, M. S. Karoui, F. Z. Benhalouche *et al.*, "Hyperspectral imagery for environmental urban planning," in *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2018, pp. 1628–1631.

[4] G. Lu and B. Fei, "Medical hyperspectral imaging: a review," *Journal of biomedical optics*, vol. 19, no. 1, p. 010901, 2014.

[5] 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*, vol. 57, no. 9, pp. 6690–6709, 2019.

[6] Z. Zheng, Y. Zhong, A. Ma, and L. Zhang, "Fpga: Fast patch-free global learning framework for fully end-to-end hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 8, pp. 5612–5626, 2020.

[7] Q. Zhu, W. Deng, Z. Zheng, Y. Zhong, Q. Guan, W. Lin, L. Zhang, and D. Li, "A spectral-spatial-dependent global learning framework for insufficient and imbalanced hyperspectral image classification," *IEEE Transactions on Cybernetics*, 2021.

[8] A. V. Oppenheim, *Discrete-time signal processing*. Pearson Education India, 1999.

[9] S. Mallat, "Understanding Deep Convolutional Networks," *Philosophical Transactions A*, vol. 374, p. 20150203, 2016. [Online]. Available: http://arxiv.org/abs/1601.04920

[10] R. Kondor and S. Trivedi, "On the generalization of equivariance and convolution in neural networks to the action of compact groups," *arXiv preprint arXiv:1802.03690*, 2018.

[11] A. Prabhu, A. Farhadi, M. Rastegari *et al.*, "Butterfly transform: An efficient fft based neural architecture design," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 024–12 033.

[12] B. Xu and P. Gong, "Land-use/land-cover classification with multispectral and hyperspectral eo-1 data," *Photogrammetric Engineering & Remote Sensing*, vol. 73, no. 8, pp. 955–965, 2007.

[13] M. B. Stuart, A. J. McGonigle, and J. R. Willmott, "Hyperspectral imaging in environmental monitoring: a review of recent developments and technological advances in compact field deployable systems," *Sensors*, vol. 19, no. 14, p. 3071, 2019.

[14] W. Song, S. Li, L. Fang, and T. Lu, "Hyperspectral image classification with deep feature fusion network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 6, pp. 3173–3184, 2018.

[15] H. Wu and S. Prasad, "Convolutional recurrent neural networks forhyperspectral data classification," *Remote Sensing*, vol. 9, no. 3, p. 298, 2017.

[16] A. B. Hamida, A. Benoit, P. Lambert, and C. B. Amar, "3-d deep learning approach for remote sensing image classification," *IEEE Transactions on geoscience and remote sensing*, vol. 56, no. 8, pp. 4420–4434, 2018.

[17] Q. Zhu, Y. Zhong, L. Zhang, and D. Li, "Adaptive deep sparse semantic modeling framework for high spatial resolution image scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 10, pp. 6180–6195, 2018.

[18] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3639–3655, 2017.

[19] Q. Sun, X. Liu, and S. Bourennane, "Unsupervised multi-level feature extraction for improvement of hyperspectral classification," *Remote Sensing*, vol. 13, no. 8, p. 1602, 2021.

[20] H. Wu and S. Prasad, "Semi-supervised deep learning using pseudo labels for hyperspectral image classification," *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1259–1270, 2017.

[21] Q. Zhu, Z. Li, Y. Zhang, and Q. Guan, "Building extraction from high spatial resolution remote sensing images via multiscale-aware and segmentation-prior conditional random fields," *Remote Sensing*, vol. 12, no. 23, p. 3983, 2020.

[22] J. Mathews and R. L. Walker, *Mathematical methods of physics*. WA Benjamin New York, 1970, vol. 501.
[23] T. Needham, *Visual complex analysis*. Oxford University Press, 1998.
[24] T. Nitta, "The computational power of complex-valued neuron," in *Joint International Conference ICANN/ICONIP*, 2003. [Online]. Available: https://link.springer.com/content/pdf/10.1007/3-540-44989-2_118.pdf
[25] ——, "On the critical points of the complex-valued neural network," in *Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP'02.*, vol. 3. IEEE, 2002, pp. 1099–1103.
[26] M. F. Amin and K. Murase, "Single-layered complex-valued neural network for real-valued classification problems," *Neurocomputing*, vol. 72, no. 4-6, pp. 945–955, 2009.
[27] S. X. Yu, "Angular embedding: from jarring intensity differences to perceived luminance," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2302–9.
[28] ——, "Angular embedding: A robust quadratic criterion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 158–73, 2012.
[29] A. Hirose and S. Yoshida, "Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence," *IEEE Transactions on Neural Networks and learning systems*, vol. 23, no. 4, pp. 541–551, 2012.
[30] D. P. Reichert and T. Serre, "Neuronal synchrony in complex-valued deep networks," *arXiv preprint arXiv:1312.6115*, 2013.
[31] J. Lu, F. Millioz, D. Garcia, S. Salles, D. Ye, and D. Friboulet, "Complex convolutional neural networks for ultrasound image reconstruction from in-phase/quadrature signal," *arXiv preprint arXiv:2009.11536*, 2020.
[32] U. Singhal, Y. Xing, and S. X. Yu, "Co-domain symmetry for complex-valued deep learning," 2021.
[33] C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal, "Deep complex networks," *arXiv preprint arXiv:1705.09792*, 2017.
[34] M. F. Baumgardner, L. L. Biehl, and D. A. Landgrebe, "220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3," Sep 2015. [Online]. Available: https://purr.purdue.edu/publications/1947/1
[35] M. Graña, M. Veganzons, and B. Ayerdi, "Hyperspectral remote sensing scenes." [Online]. Available: http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes

=2