# Hyperspectral and LiDAR fusion using Deep Neural Networks

Prafful Sanjay Pawar (Matr. Nr. 4899570), Supervisor - Dr. Behnood Rasti, Examiner - JProf. Dr. Matthias Forkel

***Abstract*—This report presents a Convolutional Neural Network based fusion technique that fuses the Hyperspectral and LiDAR imagery for land classification. The architecture consists of two CNNs, one CNN is designed to learn spectral–spatial features from hyperspectral data, and the other one is used to capture the elevation information from LiDAR data. Both of them consist of Atrous Spatial Pyramid Pooling block followed by three convolutional layers, where the last two convolutional layers are coupled together via a parameter-sharing strategy. In the fusion phase, feature-level and decision-level fusion methods are simultaneously used to integrate these heterogeneous features. A weighted summation strategy is used for the decision-level fusion where weights are determined on the bases of classification accuracy of each output. In this study, the model is evaluated on three datasets i.e. Houston 2013, Houston 2018 and Trento dataset. The results are compared with the state-of-the-art in terms of classification accuracies i.e. overall accuracy, average accuracy and kappa for different classes for input images. An improvement in the accuracy of the land classification can be observed for Houston 2018 dataset, with an overall accuracy of 93.97% compared to 88.56%.**

***Index Terms*—CNNs, Atrous Spatial Pyramid Pooling, decision fusion, feature fusion, hyperspectral data, LiDAR, parameter sharing**

## I. Introduction

DUE to the recent advances in sensing, the availability of multimodal data is no more a question for various applications in remote sensing (RS), where many data types like multispectral imagery (MSI), hyperspectral imagery (HSI), LiDAR etc. are available.

The improvement in advanced sensing technologies has enabled simultaneous acquisition of multimodal data for the same objective. This is important in the field of remote sensing (RS), owing to presence of satellite image data from several sources like multispectral (MSI), hyperspectral (HSI), and LiDAR sensors etc. Each source provides different kind of information about the same region, which can aid in tasks relating to total scene understanding. For example, HSI provide detailed spectral information through hundreds of spectral channels, which can be used to accurately classify diverse materials of interest. The detailed spectral information from HSI is commonly used to discriminate various materials based on their reflectance values, finding applications in agricultural monitoring, environment-pollution monitoring, land-use pattern. In a similar manner LiDAR data is used to obtain the elevation information, which is useful to distinguish objects within the same material. Since the attributes of these modalities complement each other, they are extensively used in a cumulative manner for multimodal learning in remote sensing area.

## II. Related work

### A. Conventional Methods

*1) Subspace Sensor Fusion (SubFus):* This work proposes a novel idea of modeling the multimodal datasets in different subspaces using common fused features. It includes three main steps for acheiving the final classification map. The first step is to extract spatial information. The second step is to estimate the low-dimensional fused features in unknown subspaces and the final step is to apply a spectral classifier on the fused features.

In the first step, for extraction of spatial information, morphological profiles (MPs) are used as they have a simple but effective approach to model the spatial dependencies of adjacent pixel. MPs are produced by considering the sequential use of opening/closing operations with a structuring element (SE) of increasing size, leading to the creation of a "morphological spectrum" for each pixel. The MP's for data from sensors are obtained by applying morphological operators to the dataset from the sensors. The resultant are matrices that contain morphological features from the sensors in a vectorized form.

Let H and L be the multisensor data from sensors 1 and 2, respectively. Then

$$\mathbf{H} = MP(H), \quad \mathbf{L} = MP(L), \tag{1}$$

are matrices that contain the vectorized morphological features from sensor 1(HSI) and sensor 2(LiDAR), respectively.

In the second step, they have assumed that multisensor data can be modeled using fused features, but in different subspaces. The extracted features contains a lot of redundant information, so an assumption is made that the multisensor features have low-rank representations. Therefore, the features from the sensors are modeled in lower- dimensional space as

$$\mathbf{H} = \mathbf{F}\mathbf{V}_1^T + \mathbf{N}_1, \mathbf{L} = \mathbf{F}\mathbf{V}_2^T + \mathbf{N}_2 \tag{2}$$

where matrices $\mathbf{H}$ ($n$ x $p_1$) and $\mathbf{L}$ ($n$ x $p_2$) contain the vectorized features for sensors 1 and 2, respectively, in their $i^{th}$ columns. $\mathbf{N}_1$ and $\mathbf{N}_1$ are ($n$ x $p$) matrices representing the noise and model error, $\mathbf{F}$ is an ($n$ x $r$) matrix containing the fused features, $\mathbf{V}_1$ and $\mathbf{V}_2$ are ($p_1$ x $r$ and $p_2$ x $r$, respectively) subspace projection matrices. In this paper, they have suggested the maximum possible rank i.e., $r = \min(n, p_1, p_2)$, to achieve the maximum reduction in the feature space.

In the final step of subspace feature fusion, they estimate unknown matrices $\mathbf{F}$, $\mathbf{V}_1$, and $\mathbf{V}_2$ using a novel constraint penalized cost function (CPCF)

$$(\hat{\mathbf{F}}, \hat{\mathbf{V}_1}, \hat{\mathbf{V}_1}) = arg \min_{F,V_1,V_2} \boldsymbol{J}(\mathbf{F}, \mathbf{V}_1, \mathbf{V}_2)$$

such that $\mathbf{V}_1^T\mathbf{V}_1 = \mathbf{I}$ and $\mathbf{V}_2^T\mathbf{V}_2 = \mathbf{I}$, where

$$\boldsymbol{J}(\mathbf{F}, \mathbf{V}_1, \mathbf{V}_2) = \frac{1}{2}\|\mathbf{H}-\mathbf{F}\mathbf{V}_1^T\|_F^2 + \frac{\lambda_1}{2}\|\mathbf{L}-\mathbf{F}\mathbf{V}_2^T\|_F{}^2 + \lambda_2\|\mathbf{F}\|_{TV}$$

Here the TV-norm ($\|.\|_{TV}$) ensures piecewise smoothness on F wherease $\lambda_1$ and $\lambda_2$ control the tradeoff between the fidelity of the sensor 2 and the smoothness of the subspace features, respectively with respect to the fidelity of the data from sensor 1. Following this they use the alternating direction method of multipliers to solve for matrices $\mathbf{F}$, $\mathbf{V}_1$, and $\mathbf{V}_2$. The variable $\mathbf{F}$ is split and penalty method is applied to enforce the constraint for $\mathbf{F} = \mathbf{S}$ to the minimization problem. After splitting the variables they solve the problem w.r.t. one unknown matrix at a time while the other ones are assumed to be fixed (a cyclic descent (CD) algorithm) leading to five steps, each for $\mathbf{F}$, $\mathbf{S}$, $\mathbf{V}_1$, $\mathbf{V}_2$ and the fifth step comprising of updating the multipliers.

### *B. Deep learning methods*

*1) FusAtNet:* The usage of attention learning mechanism has shown remarkable performance gain for different visual inference tasks [1], [2], [3]. Ideally, the attention modules highlight the prominent features while suppressing the irrelevant features through a self-supervised learning paradigm. FusAtNet[4] is a network that takes attention mask from one modality and uses it to enhance the representations of other modality (Fig. 1). It is an attention based multimodal fusion network for land-cover classification given an HSI-LiDAR pair as input. The method extracts the spectral features using "self-attention" in HSI and incorporates multimodal attention using the proposed "cross-attention" mechanism that uses LiDAR modality to derive an attention mask that highlights the spatial features of the HSI. It performs pixel based classification by using the spectral and spatial information from HSIs and the depth and intensity information encoded in LiDAR.

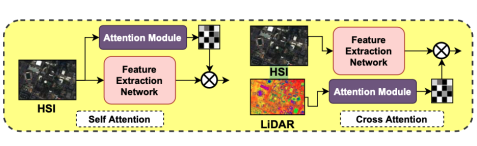



Fig. 1. Self-attention vs cross-attention for multimodal fusion. The self-attention module (left) works only on single modality. In this the hidden representations as well as the attention mask are derived from the same modality (HSIs). In the cross-attention module (right), the attention mask is derived from a different modality (LiDAR) and is used to enhance the latent features from the first modality.[4]

The attention modules selectively highlight the hotspots in the extracted hyperspectral features in order to increase the interclass variance and thus improve the classification accuracy. This is achieved in two steps: In the first step, the HSI feaures are passed through a feature extractor and spectral attention module, and their combination is used to highlight the spectral information in the HSI features. Simultaneously, the LiDAR features are passed through a spatial attention framework and the resultant mask accentuates the spatial characteristics of HSI. In the second step, the highlighted features are combined with the original features and passed

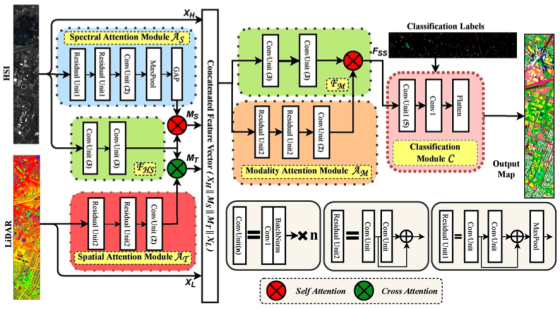



Fig. 2. Schematic of FusAtNet. The hyperspectral training samples $X_H$ are sent to the feature extractor $F_{HS}$ to get latent representations and to spectral attention module $A_S$ to generate spectral attention mask. Simultaneously, the corresponding LiDAR training samples $X_L$ are sent to spatial attention module $A_T$ to get the spatial attention mask. The attention masks are then individually multiplied to the latent HSI representations to get $M_S$ and $M_T$ . $M_S$ and $M_T$ are then concatenated with $X_H$ and $X_L$ and sent to modality feature extractor $F_M$ and modality attention module $A_M$. The outputs from the two are then multiplied to get $F_{SS}$, which is then sent to the classification module $C$ for pixel classification.[4]

through modality extraction and modality attention modules, the outputs of which are combined to highlight the important sections of the two modalities. The resultant features are then sent to the classification module to obtain the classification map.

The network architecture contains six modules that are used in three phases. In the first phase, hyperspectral feature extractor $F_{HS}$, spectral attention module $A_S$ and spatial attention module $A_T$ are used to jointly extract and highlight the spatial-spectral features from the HSI. In the second phase, modality feature extractor $F_M$ and modality attention module $A_M$ are used to selectively highlight the modality specific features. In the third phase, the modality specific spectral-spatial features are sent to the classification module $C$. All the modules are inherently CNN modules where the size of all the kernels is fixed to 3×3 and non-linearity is fixed to ReLU. A brief overview of the modules is given below:

**Hyperspectral feature extractor** $F_{HS}$ consists of a 6 layer CNN and is used to extract the spectral-spatial features from the HSIs. The first five layers contain 256 filters while the sixth layer has 1024 number of filters. All the convolution operations are applied with zero padding. Output of each convolution operation is operated on by batch normalisation.

**Spectral attention module** $A_S$ is a CNN with 3 convolution blocks, with 2 convolution layers each. In addition, first and second convolution block are followed by a residual block each. There is a maxpooling layer after each residual block and the sixth convolution layer. The last layer of this module is a global average pooling (GAP) layer.

**Spatial attention module** $A_T$ is a 6-layer CNN that generates attention mask from the LiDAR modality. The first 3 layers consist of 128 filters each while each of the last three layers has 256 number of filters. There are two residual layers, each after second and fourth convolution layer. All the convolution layers are followed by a batch normalisation operation.

**Modality feature extractor** $F_M$ follows the same structure

as that of $F_{HS}$.

**Modality attention module** $A_M$ has same architecture as that of $A_T$. The work of this module is to create an attention mask that focuses on specific traits of each modality and therefore the input is kept the same as that of $F_M$.

**Classification module** $C$: The input to $C$ module are the final spectral-spatial features $F_{SS}(\mathbf{x}_H^i,\mathbf{x}_L^i)$ which is obtained from multiplying the outputs from $F_M$ and $A_M$. The Classification module $C$ is a 6-layer fully convolutional neural network where first four layers consist of 256 filters each while the fifth and sixth layers respectively contain 1024 and K filters, where K is the number of classes. The filter size for the last layer is set to 1×1 and no padding is used in any layer. All the layers except last one are operated on by ReLU activation function and batch normalisation while the last layer is the softmax layer. The output of $C$ is a vector of size $1 \times \mathbf{K}$. The output from $C$ is subjected to a categorical cross-entropy loss which is backpropagated to train the FusAtNet model in an end-to-end fashion.

$$L_C = -\mathbb{E}_{(\mathbf{x}_H^i,\mathbf{x}_L^i,y^i)}[y^i log C(\theta_C, F_{SS}(\mathbf{x}_H^i,\mathbf{x}_L^i))] \tag{3}$$

where, $L_C$ is the classification loss. During the testing phase, the given test sample is passed through the fusion module and follows the same path as that of the training samples. The resultant output is sent to the classification module $C$ where it is assigned the predicted class label.

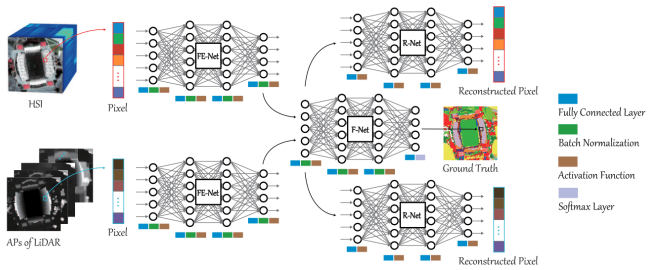



Fig. 3. Illustration of EndNet for classificaiton of HSI and LiDAR data.[6]

*2) EndNet:* EndNet[6] fuses the multimodal information by enforcing the fused features to reconstruct the multimodal input in turn. Such a reconstruction strategy is capable of better activating the neurons across modalities compared with some conventional and widely used fusion strategies, e.g., early fusion, middle fusion, and late fusion. EndNet framework consists of three parts: feature extraction (or encoder) network (FE-Net), fusion module connecting with labels (F-Net), and reconstruction (or decoder) network (R-Net).

*a) FE-Net:* extracts features for each modality in hierarchical manner. Consider two input modalities $\mathbf{X}_1 \in \mathbb{R}^{d_1 \times N}$ and $\mathbf{X}_2 \in \mathbb{R}^{d_2 \times N}$ with $d_1$ and $d_2$ being dimensions by $N$ pixels. The extracted features for the $i^{th}$ pixel in the $l^{th}$ layer, denoted as $\mathbf{z}_{s,i}^{(l)}$ can be written as

$$\mathbf{z}_{s,i}^{(l)} = \begin{cases} f(\mathbf{W}_s^{(l)}\mathbf{x}_{s,i} + \mathbf{b}_s^{(l)}), & \text{if } l = 1 \\ f(\mathbf{W}_s^{(l)}\mathbf{z}_{s,i}^{(l-1)} + \mathbf{b}_s^{(l)}), & \text{if } l = 2, ..., p \end{cases} \tag{4}$$

where $s = 1, 2$ stands for the different modalities; $f(.)$ is the non linear activation (sigmoid, ReLU, etc) w.r.t the to-be-estimated network parameters:$\{\mathbf{W}_s^{(l)}\}_{l=1}^p$ and $\{\mathbf{b}_s^{(l)}\}_{l=1}^p$. Batch normalization is used before activation function.

*b) F-Net:* The features extracted from FE-Net $\mathbf{a}_i$ are formulated as:

$$\mathbf{a}_i^{(l+1)} = h([\mathbf{z}_{1,i}^{(l)}, \mathbf{z}_{2,i}^{(l)}]) \tag{5}$$

Similar to $f(.)$, $h(.)$ can be seen as a set of block that consists of the encoder layer, the Batch Normalization layer and the ReLU activation layer. The output in F-Net is connected with one-hot encoded labels through a softmax layer.

*c) R-Net:* Inspired from the encoder-decoder architecture of [5], they enforce to reconstruct the input modalities from the fused features to improve the fusion level of the features extracted from FE-Net, which results into:

$$\mathbf{x}_{s,i} = g(\mathbf{a}_i^{(l+1)}), s = 1, 2 \tag{6}$$

where $g(.)$ denotes the reconstruction block with sigmoid activation without the batch normalization layer seen in R-Net.

*d) Objective Function:* The objective function in EndNet is given as:

$$\min_{\phi,\varphi} \underbrace{\sum_{s=1}^{2} \|\mathbf{X}_s - g_\varphi(f_\phi(\mathbf{X}_s))\|_F^2}_{\text{encoder-decoder}} + \underbrace{\frac{-1}{N}\sum_{i=1}^{N}[\mathbf{y}_i \log \mathbf{a}_i^{(l+1)} + (1 - \mathbf{y}_i)\log(1 - \mathbf{a}_i^{(l+1)})]}_{\text{cross-entropy}}$$

where $\mathbf{y}_i$ is defined as the one-hot encoded label vector with $C$ categories in the $i^{th}$ pixel, and the encoder–decoder term formulates how the reconstruction idea is embedded in the proposed end-to-end network

*3) Coupled CNNs:* This [7] architecture uses two coupled convolutional neural networks (CNNs), where one is designed to learn spectral-spatial features from HSI data, the other one is used to capture the elevation information from LiDAR data. Both consists of three convolutional layers with small kernels (3 x 3) where the last two convolutional layers are coupled together using a parameter-sharing strategy. The coupled convolution layers reduce the number of parameters and more importantly guide the two CNNs learn from each other and hence facilitate feature fusion.

In the fusion phase, they use feature-level and decision-level fusion strategies simultaneously. For the feature-level fusion, they have proposed summation and maximization fusion methods along with concatenation method. To enhance the discriminative ability of learned features, they have added two output layers to the CNNs, respectively. These three output results are finally combined together via a weighted summation method, whose weights are determined by the classification accuracy of each output on the training data.

Each of the two networks include an input module, a feature learning module, and a fusion module. For the HS network,

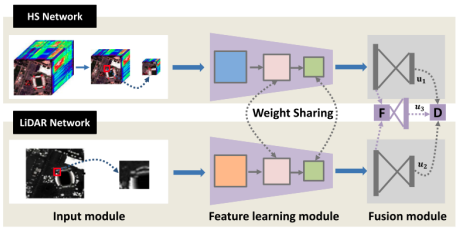



Fig. 4. Flowchart of Coupled CNNs model[7]

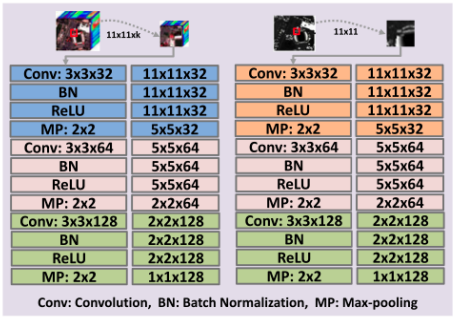



Fig. 5. Coupled CNNs model architecture[7]

PCA is firstly used to reduce the redundant information of the original hyperspectral data, and then a small cube is extracted surrounding the given pixel. For the LiDAR network, an image patch is directly extracted at the same spatial position as the hyperspectral data. In the feature learning module, three convolutional layers are used where the last two of them share parameters. In the fusion module, three classifiers are constructed. Each CNN has an output layer, and their fused features are also fed into an output layer.

*a) Feature Learning via Coupled CNNs:* Consider hyperspectral image $\mathbf{X}_h \in \mathbb{R}^{m\times n\times b}$ and corresponding LiDAR image $\mathbf{X}_l \in \mathbb{R}^{m\times n}$, where $m$ and $n$ represent the height and width, respectively, of the two images and $b$ represents the number of spectral bands of hyperspectral image covering the same region. To sufficiently fuse the information from $\mathbf{X}_h$ and $\mathbf{X}_l$, firstly PCA is used over $\mathbf{X}_h$. This reduces redundant spectral information and then for each pixel a small cube $\mathbf{x}_h \in \mathbb{R}^{p\times p\times k}$ and a small patch $\mathbf{x}_l \in \mathbb{R}^{p\times p}$ centered at it are chosen from $\mathbf{X}_h$ and $\mathbf{X}_l$ respectively. These small patches are fed into three convolutional layers to learn features. For the first convolutional layer, two different convolution operators (the blue box and the orange box) are used to obtain an initial representation of $\mathbf{x}_h$ and $\mathbf{x}_l$, respectively. This convolutional layer is sequentially followed by a batch normalization (BN) layer to regularize and accelerate the training process, a rectified linear unit (ReLU) to learn a nonlinear representation, and a max-pooling layer to reduce the data variance and the computation complexity.

In the second layer both the networks share parameters which makes the model efficient by reducing number of parameters and also enables the two networks to learn from each other. For the third convolutional layer, similar strategy is used which further improves the discriminative ability of the learned representation from the second convolutional layer. All the convolutional layers have padding operators to make the output size the same as the input size.

*b) Data Fusion:* Post extracting the feature representations of $\mathbf{x}_h$ and $\mathbf{x}_l$, a novel combination strategy is used based on feature-level and decision-level fusions. Consider the learned features for $\mathbf{x}_h$ and $\mathbf{x}_l$ are represented by $\mathbf{R}_h \in \mathbb{R}^{128\times 1}$ and $\mathbf{R}_l \in \mathbb{R}^{128\times 1}$ respectively. $\mathbf{R}_h$ and $\mathbf{R}_h$ are combined to generate a new feature representation and then these three feature representations are input into output layers separately. In the end all these output layers are integrated together to produce the final result which is given as:

$$\mathbf{O} = D[f_1(\mathbf{R}_h;\mathbf{W}_1), f_2(\mathbf{R}_l;\mathbf{W}_2), f_3(F(\mathbf{R}_h,\mathbf{R}_l);\mathbf{W}_3);U] \quad (7)$$

where $\mathbf{O} \in \mathbb{R}^{C\times 1}$, where $C$ is the number of classes represents output of final fusion module. $D$ and $F$ are are decision-level and feature-level fusions, respectively; $f_1, f_2$, and $f_3$ are three output layers connected to $\mathbf{R}_h$, $\mathbf{R}_l$ and $F(\mathbf{R}_h, \mathbf{R}_l)$, respectively; $\mathbf{W}_1 \in \mathbb{R}^{C\times 128}, \mathbf{W}_2 \in \mathbb{R}^{C\times 128}$ and $\mathbf{W}_3 \in \mathbb{R}^{C\times 128}$ denote the connection weights for $f_1, f_2$ and $f_3$ respectively whereas $\mathbf{U} \in \mathbb{R}^{C\times 3}$ corresponds to the fusion weight for $D$.

*c) Objective function:* The whole network is trained in an end-to-end manner using the training set $\{(\mathbf{x}_h^{(i)}, \mathbf{x}_l^{(i)}, \mathbf{y}^{(i)})|i = 1, 2, ..., N\}$ where $N$ denotes number of training samples and $\mathbf{y}^{(i)}$ denotes the groundtruth for the $i_{th}$ sample. Cross entropy loss function is used to compute loss after obtaining the three outputs for each sample. The loss value for one output can be computed as:

$$L_j = -\frac{1}{N}\sum_{i=1}^{N}[\mathbf{y}^i \log(\hat{\mathbf{y}}_j^{(i)}) + (1-\mathbf{y}^{(i)})\log(1-\hat{\mathbf{y}}_j^{(i)})] \quad (8)$$

where $j = 1, 2, 3$ represents the respective output number. Here $L_1$ and $L_2$ are responsible for the Hyperspectral and LiDAR features whereas $L_3$ is designed to supervise the learning process of the fused feature between the two. The final loss value $L$ is combination of $L_1$,$L_2$ and $L_3$, where $\lambda_1$ and $\lambda_2$ represent the weight for $L_1$ and $L_2$ respectively.

$$L = \lambda_1 L_1 + \lambda_2 L_2 + L_3 \quad (9)$$

## III. Proposed Methodology

### A. Atrous Spatial Pyramid Pooling (ASPP)[10]

*1) Atrous Convolution:* In the task of image segmentation, the steps of using CNN network to segment the image are convolution first and then pooling, which reduces the size of image and enlarges the receptive field. Since image segmentation prediction is a pixel-wise output, it is necessary to upsample the pooled image to the size of the original image for preprocessing. It can be seen that there are two key points in this traditional processing method: one is to reduce the size of the image by pooling, and the other is to restore the image to its original size by upsampling. In the above two processes, a lot of useful information will be lost and the ideal segmentation effect will not be achieved.

Long et al. [9] show that atrous convolution can systematically aggregate multi-scale context information without losing resolution.

Atrous convolution is applied to one-dimensional or two-dimensional information input data $x[i]$. After filtering $w[k]$, the output $y[i]$ is obtained as follows.

$$y[i] = \sum_{k} x[i + r \cdot k]w[k] \tag{10}$$

Here, $i$ is the location of the pixels, $r$ is the dilated rate of the atrous convolution, and $k$ is the size of the convolution kernel. Standard convolution is a special atrous convolution with a dilated rate of 1. Different dilated rates can be set to adjust the range of the receptive field. The smaller the rate, the more detailed the segmentation of the rough feature map, but more time will be spent in training.

Atrous convolution does not pad the blank pixels between the pixels, but skips some pixels on the existing pixels, or keeps the input unchanged, adding some weights of 0 to the parameters of the convolution kernel, so as to expand the receptive field.

If the void rate of a void convolution is $r$ and the size of the convolution nucleus is $k$, then the size of the receptive field $F$ obtained is:

$$F = (r - 1)(k - 1) + k \tag{11}$$

We use the parallel atrous convolution layers with different dilated rates in the pyramid model to capture multi-scale information. The smaller rate correlates the nearest pixels, while the larger rate correlates the long-range pixels.

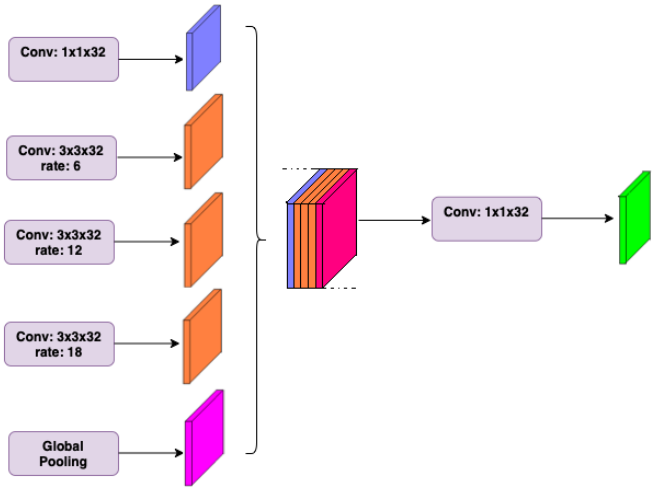



Fig. 6. Visualization of Atrous Spatial Pyramid Pooling (ASPP)

## B. Framework of the Proposed Model

The proposed model is essentially a modification of the CoupledCNN [7] architecture. ASPP is applied at the beginning of the Feature Learning module to enhance the feature representation obtained from the two modalities. As shown in Fig. 10, the proposed model mainly consists of two networks: an HS network for spectral–spatial feature learning and a LiDAR network for elevation feature learning. Each of them includes an input module, a feature learning module, and a fusion module. For the HS network, PCA is firstly used to reduce the redundant information of the original hyperspectral data, and then a small cube is extracted surrounding the given pixel. For the LiDAR network, we can directly extract an image patch at the same spatial position as the hyperspectral data. In the feature learning module, we use ASPP block followed by three convolutional layers, and the last two of them share parameters. In the fusion module, we construct three classifiers. Each CNN has an output layer, and their fused features are also fed into an output layer.

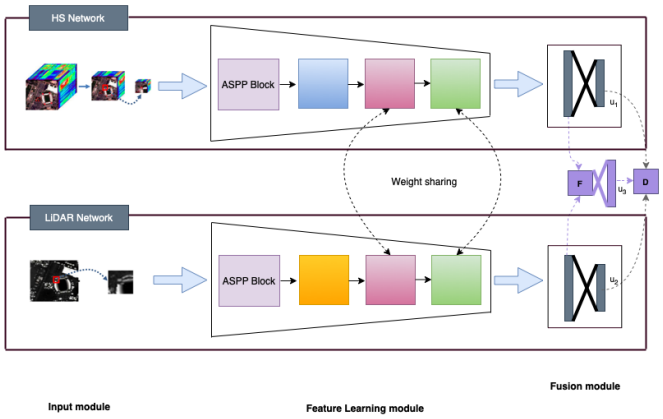



Fig. 7. Flowchart of Proposed architecture

*1) Feature Learning via Coupled CNNs:* Consider hyperspectral image $\mathbf{X}_h \in \mathbb{R}^{m \times n \times b}$ and corresponding LiDAR image $\mathbf{X}_l \in \mathbb{R}^{m \times n}$, where $m$ and $n$ represent the height and width, respectively, of the two images and $b$ represents the number of spectral bands of hyperspectral image covering the same region. To sufficiently fuse the information from $\mathbf{X}_h$ and $\mathbf{X}_l$, firstly PCA is used over $\mathbf{X}_h$. This reduces redundant spectral information and then for each pixel a small cube $\mathbf{x}_h \in \mathbb{R}^{p \times p \times k}$ and a small patch $\mathbf{x}_l \in \mathbb{R}^{p \times p}$ centered at it are chosen from $\mathbf{X}_h$ and $\mathbf{X}_l$ respectively. These small patches are then fed into Feature Learning module to learn features. The convolutional layer is sequentially followed by a batch normalization (BN) layer to regularize and accelerate the training process, a rectified linear unit (ReLU) to learn a nonlinear representation, and a max-pooling layer to reduce the data variance and the computation complexity.

In the second convolutional layer both the networks share parameters which makes the model efficient by reducing number of parameters and also enables the two networks to learn from each other. For the third convolutional layer, similar strategy is used which further improves the discriminative ability of the learned representation from the second convolutional layer. These two convolutional layers are then followed by BN, ReLU and max-pooling operators. All the convolutional layers have padding operators to make the output size the same as the input size.

*2) Data Fusion:* Post extracting the feature representations of $\mathbf{x}_h$ and $\mathbf{x}_l$, a novel combination strategy is used based on


Fig. 8. Proposed architecture: CoupledCNN with ASPP

feature-level and decision-level fusions. Consider the learned features for $\mathbf{x}_h$ and $\mathbf{x}_l$ are represented by $\mathbf{R}_h \in \mathbb{R}^{128\times 1}$ and $\mathbf{R}_l \in \mathbb{R}^{128\times 1}$ respectively. $\mathbf{R}_h$ and $\mathbf{R}_l$ are combined to generate a new feature representation and then these three feature representations are input into output layers separately. In the end all these output layers are integrated together to produce the final result which is given as:

$$\mathbf{O} = D[f_1(\mathbf{R}_h;\mathbf{W}_1), f_2(\mathbf{R}_l;\mathbf{W}_2), f_3(F(\mathbf{R}_h,\mathbf{R}_l);\mathbf{W}_3);U] \tag{12}$$

where $\mathbf{O} \in \mathbb{R}^{C\times 1}$, where $C$ is the number of classes represents output of final fusion module. $D$ and $F$ are decision-level and feature-level fusions, respectively; $f_1$, $f_2$, and $f_3$ are three output layers connected to $\mathbf{R}_h$, $\mathbf{R}_l$ and $F(\mathbf{R}_h,\mathbf{R}_l)$, respectively; $\mathbf{W}_1 \in \mathbb{R}^{C\times 128}, \mathbf{W}_2 \in \mathbb{R}^{C\times 128}$ and $\mathbf{W}_3 \in \mathbb{R}^{C\times 128}$ denote the connection weights for $f_1, f_2$ and $f_3$ respectively whereas $\mathbf{U} \in \mathbb{R}^{C\times 3}$ corresponds to the fusion weight for $D$. Figure 12 shows the structure of the Data fusion module.

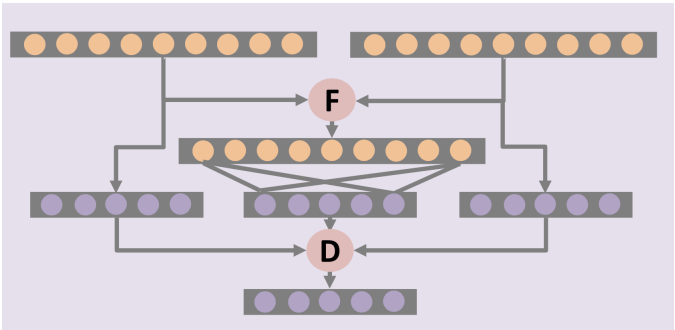



Fig. 9. Structure of fusion module

*3) Network Training and Testing:* The whole network is trained in an end-to-end manner using the training set $\{(\mathbf{x}_h^{(i)}, \mathbf{x}_l^{(i)}, \mathbf{y}^{(i)})|i = 1, 2, ..., N\}$ where $N$ denotes number of training samples and $\mathbf{y}^{(i)}$ denotes the groundtruth for the $i_{th}$ sample. Cross entropy loss function is used to compute loss after obtaining the three outputs for each sample. The loss value for one output can be computed as:

$$L_j = -\frac{1}{N}\sum_{i=1}^{N}[\mathbf{y}^i \log(\hat{\mathbf{y}}_j^{(i)}) + (1-\mathbf{y}^{(i)})\log(1-\hat{\mathbf{y}}_j^{(i)})] \tag{13}$$

where $j = 1, 2, 3$ represents the respective output number. Here $L_1$ and $L_2$ are responsible for the Hyperspectral and LiDAR features whereas $L_3$ is designed to supervise the learning process of the fused feature between the two. The final loss value $L$ is combination of $L_1$,$L_2$ and $L_3$, where $\lambda_1$ and $\lambda_2$ represent the weight for $L_1$ and $L_2$ respectively.

$$L = \lambda_1 L_1 + \lambda_2 L_2 + L_3 \tag{14}$$

where $\lambda_1$ and $\lambda_2$ represent the weight parameters for $L_1$ and $L_2$ , respectively. In the experiments, we empirically set them to $0.01$ because it can achieve satisfactory performance. Similar to most CNN models, $L$ can be optimized using a backpropagation algorithm. Note that $L_1$ and $L_2$ can also be considered as regularization terms for $L_3$, thus reducing the overfitting risk during the network training process.

Once the network is trained, it can be used to predict the label of each test sample. First, $u_j, j \in \{1, 2, 3\}$ is computed on the training set. Its $i^{th}$ element $u_{ji}$ can be derived as

$$\mathbf{a}_{ji} = \frac{\sum_{l=1}^{N}\sum_{y(l)=i}\mathbf{I}(\hat{\mathbf{y}}_j^{(l)} = \mathbf{y}^{(l)})}{\sum_{l=1}^{N}\mathbf{I}(\mathbf{y}^{(l)} = i)} \tag{15}$$

$$\mathbf{u}_{ji} = \frac{\mathbf{a}_{ji} + 10^{-5}}{\mathbf{a}_{1i} + \mathbf{a}_{2i} + \mathbf{a}_{3i} + 10^{-5}} \tag{16}$$

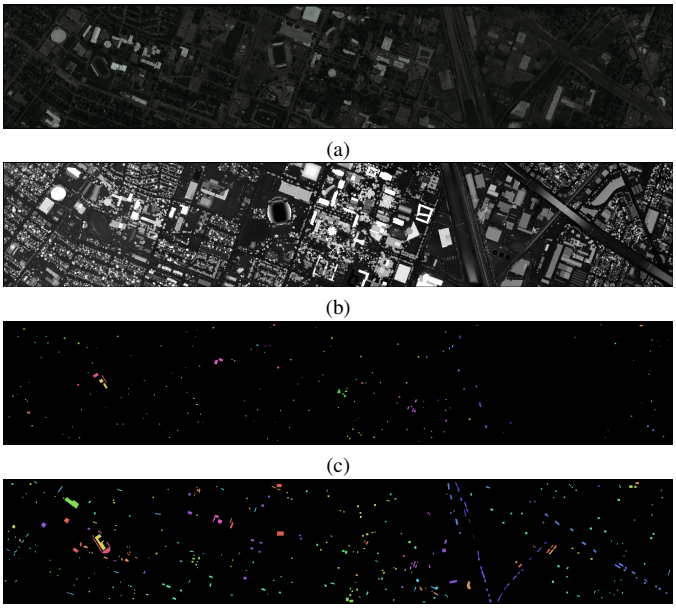
where $a_{ji}$ is the $i^{th}$ class accuracy of the $j^{th}$ output, and $I$ is an indicator function, the value of which equals 1 when the condition exists and 0 otherwise. Second, for the $t^{th}$ test sample, we are able to obtain three output values $\hat{\mathbf{y}}_1^{(t)}, \hat{\mathbf{y}}_2^{(t)}$ and $\hat{\mathbf{y}}_3^{(t)}$ via a feed-forward propagation. Finally, the output value can be derived by using (12).

## IV. Experiments

### A. Data Description

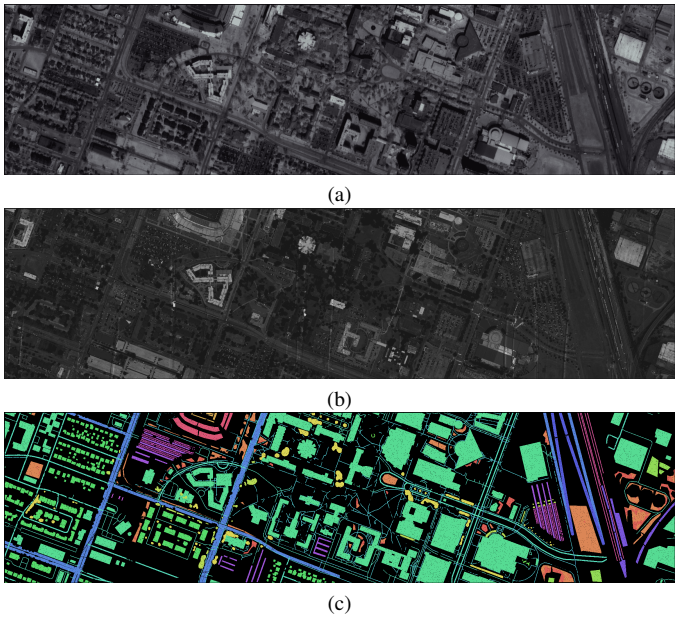
*1) Houston 2013 Data:* This dataset was acquired over the University of Houston campus and the neighboring urban area in June 2012 [8]. It consists of a hyperspectral image and LiDAR data, both of which contain 349 × 1905 pixels with a spatial resolution of 2.5 m. The number of spectral bands for the hyperspectral data is 144. Figure [6] demonstrates a pseudocolor image of the hyperspectral data, a grayscale image of the LiDAR data, and groundtruth maps of the training and test samples.

*2) Houston 2018 Data:* This dataset, widely identified as the GRSS_DFC_2018 dataset was distributed originally for the 2018 GRSS Data Fusion Contest and was acquired over the University of Houson campus. The dataset consists of hyperspectral, multispectral LiDAR, and very high-resolution RGB images. For the hyperspectral imaging, an ITRES CASI 1500 was used with the spectral range of 380–1050 nm with 48



(a) (b) (c) (d)

Fig. 10. Visualization of the Houston 2013 data. (a) Pseudo-color image for the hyperspectral data (band number 64, 43 and 22 used for R,G and B respectively) (b) Grayscale image for the LiDAR data (c) Training data map (d) Test data map.



(a) (b) (c)

Fig. 11. Visualization of DFC_2018 data. (a) Pseudo-color image for the hyperspectral data (band number 40, 20 and 10 used for R,G and B respectively) (b) Grayscale image for the LiDAR data (c) Ground reference.

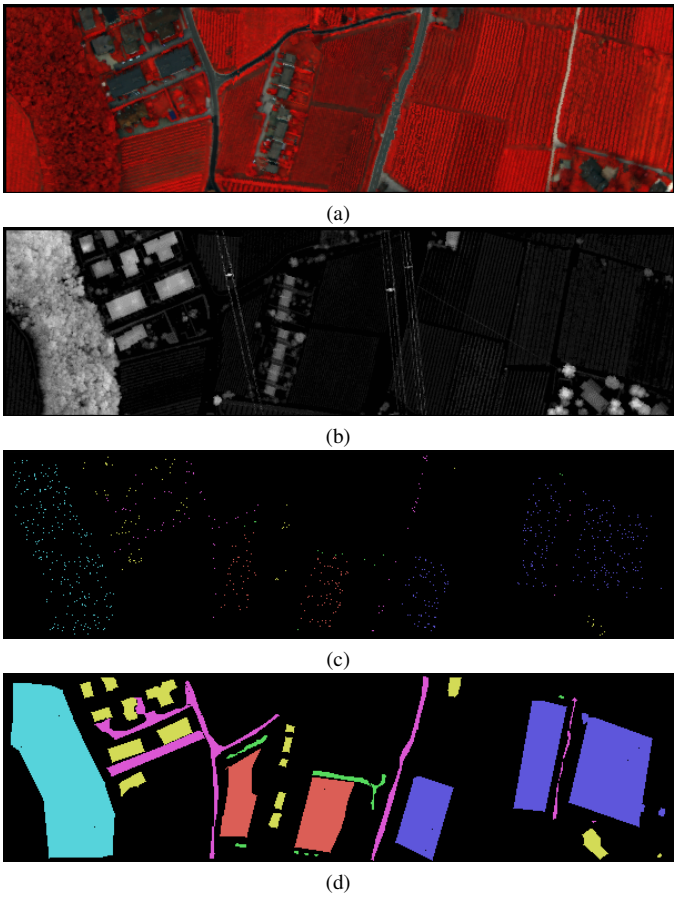
bands at a 1 m ground sampling distance (GSD). For the multispectral LiDAR, an Optech Titam MW (14SEN/CON340) with an integrated camera which operates at three different laser wavelengths was used. The multispectral LiDAR data include point cloud data at 1550, 1064, and 532 nm, as well as an intensity image from the first return per channel, and DSMs at a 50 cm GSD. The RGB was acquired with a very high-resolution RGB imager (DiMAC ULTRALIGHT) with a 70 mm focal length. The very high resolution color image includes Red, Green, and Blue bands at a 5 cm GSD. Houston dataset contains 601 × 2384 pixels. The dataset consist of 20 land-cover classes.

*3) Trento:* The Trento dataset was acquired over an area in the city of Trento, Italy. The dataset consists of hyperspectral and LiDAR data. The LiDAR DSM data was acquired using the Optech ALTM 3100EA sensor and the hyperspectral data was captured using the AISA Eagle sensor, all with the spatial resolution of 1 m. The hyperspectral data consist of 63 bands (600 × 166 pixels) ranging from 402.89 to 989.09 nm, where the spectral resolution is 9.2 nm. The dataset consists of six classes including Building, Woods, Apple trees, Roads, Vineyard, and Ground. Fig. 2 shows a false-color composite representation of the hyperspectral data and the corresponding training and test samples.

### *B. Experimental Results and Analysis*

In addition to two single-source models (i.e., RF on HSI and RF on LiDAR), the effectiveness of feature-level fusion models is also tested. Table I-III reports the detailed comparison results of different models in terms of OA, AA, and Kappa coefficients of eight models on the Houston 2013, Houston 2018 and Trento dataset respectively. Several conclusions can be observed from it. First, for the single-source models, Random Forest applied to HSI achieves significantly better results than when applied to LiDAR except in the case of Houston 2018 dataset. It indicates that the spectral–spatial information in the hyperspectral data is more discriminative than the elevation information in the LiDAR data. Secondly, all of the six feature-level fusion models (i.e., SubFus, FusAtNet, EndNet, CoupledCNN, ASPP on LiDAR, ASPP on both networks) obtain higher accuracies than the Random Forest. Interesting observations can be made while comparing traditional method like SubFus with modern DL-based methods. It can be observed in Table I-III, SubFus (OA = $84.2\%$) comes close to EndNet(OA=$87.65\%$) in terms of overall accuracy for Houston 2013 dataset. SubFus(OA=$99.51\%$) performs better than FusAtNet(OA=$98.95\%$), EndNet(OA=$93.16\%$) when tested on Trento dataset. FusAtNet(OA=$88.47\%$), EndNet(OA=$84.45\%$), CoupledCNN(OA=$88.56\%$) perform better than SubFus(OA=$73.87\%$). Qualitative results can be observed in Fig. 17-19. When comparing for Houston 2013 dataset in Fig. 17, it can be observed that, SubFus17c smoothens certain surfaces and struggles in differentiating between Residential and Commercial spaces. For Houston 2018 dataset, Fig. 18 demonstrates the classification maps for all models, SubFus smoothens certain regions but performs on par with the DL-based methods.

*1) Comparison with State-of-the-art Models:* It can be observed in Table I, the proposed model(OA = $94.62\%$) performs slightly better than CoupledCNN(OA=$93.83\%$) in terms of Overall Accuracy. In case of Houston 2018 dataset(Table II), the proposed technique(OA=$93.97\%$) outperforms CoupledCNN (OA = $88.56\%$) by a significant margin. This improvement can be attributed towards increased receptive field captured by ASPP block. It expands the receptive field with the help of parallel atrous convolution layers. In case of Trento



(a) (b) (c) (d)

Fig. 12. Visualization of Trento data. (a) Pseudo-color image for the hyperspectral data (band number 40, 20 and 10 used for R,G and B respectively) (b) Grayscale image for the LiDAR data (c) Training data map (d) Test data map.

Dataset(Table III), proposed method(OA=99.78%) performs on par with rest of the method .Qualitative results can be observed from Fig. 14, 15 and 16. In case of Trento dataset, proposed method(OA=99.63%) performs slightly worse than CoupledCNN(OA=99.75%). It can be visually verified that the classification maps obtained from proposed method tend to be less noisy and have smooth interclass transitions.

*2) Sensitivity Analysis:* Fig. 13 depicts the sensitivity of the overall accuracy (Y-axis) w.r.t the selection of the Learning rate. Analysis for learning rate is carried out for values $\{0.001, 0.01, 0.1\}$. The maximum overall accuracy $(94.62\%, 93.97\%, 99.78\%)$ for all the datasets is obtained at a learning rate of 0.01. The overall accuracy drops significantly at learning rate of 0.1.

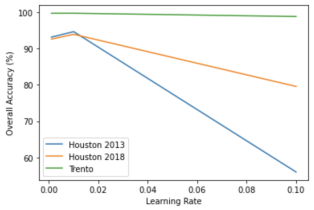



Fig. 13. The sensitivity of the overall accuracy w.r.t learning rate applied to Houston 2013, Houston 2018 and Trento datasets

Fig.14-16 demonstrates the effects of the selection of $\lambda_1$ and $\lambda_2$ on the OA when proposed model is applied on the training samples from different datasets. The maximum OA $(94.62\%, 93.97\%, 99.78\%)$ takes place at $\lambda_1 = \lambda_2 = 0.01$. As can be seen, the OA is more sensitive to the selection of $\lambda_1$, OA drops significantly for change in value for $\lambda_1$ at constant value of $\lambda_2 = 0.01$.

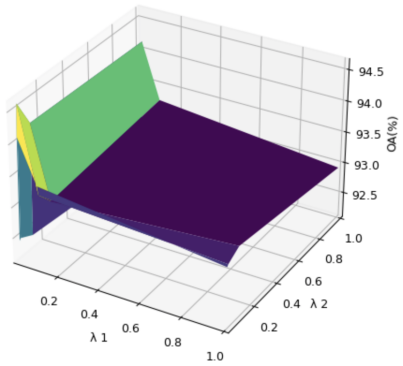



Fig. 14. The sensitivity of the overall accuracy w.r.t. the tuning parameters $\lambda_1$ and $\lambda_2$ of the proposed model applied on the Houston 2013 dataset.

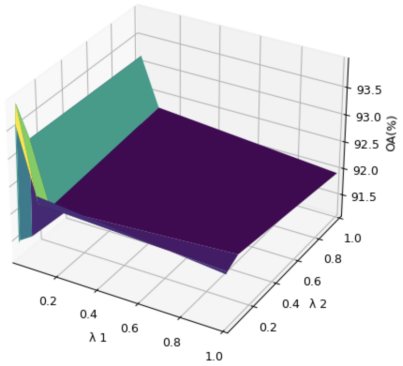



Fig. 15. The sensitivity of the overall accuracy w.r.t. the tuning parameters $\lambda_1$ and $\lambda_2$ of the proposed model applied on the Houston 2018 dataset.

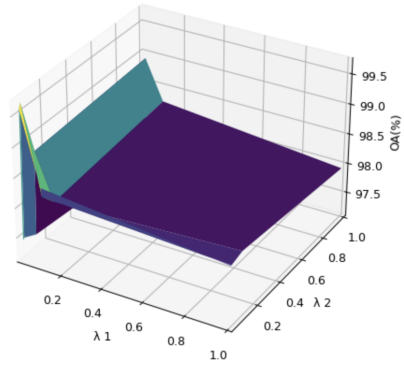



Fig. 16. The sensitivity of the overall accuracy w.r.t. the tuning parameters $\lambda_1$ and $\lambda_2$ of the proposed model applied on the Trento dataset.

TABLE I
HOUSTON 2013 RESULTS

| Classes | RF on HSI | RF on LiDAR | SubFus | FusAtNet | EndNet | CoupledCNN | ASPP on LiDAR network | ASPP on both networks |
|---|---|---|---|---|---|---|---|---|
| Healthy Grass | 83.57 | 13.48 | 80.53 | 83.00 | 84.81 | 82.62 | 82.67 | 83.09 |
| Stressed Grass | 98.21 | 16.25 | 79.41 | 100.00 | 94.45 | 100.00 | 99.71 | 100.00 |
| Synthetic Grass | 98.01 | 56.63 | 99.80 | 100.00 | 100.00 | 95.84 | 96.43 | 97.42 |
| Trees | 97.34 | 45.26 | 76.32 | 99.53 | 97.44 | 99.05 | 99.14 | 99.81 |
| Soil | 96.59 | 58.05 | 100.00 | 98.86 | 98.77 | 100.00 | 100.00 | 99.81 |
| Water | 97.20 | 58.04 | 100.00 | 100.00 | 95.10 | 100.00 | 95.80 | 97.20 |
| Residential | 81.90 | 39.55 | 68.28 | 97.48 | 81.72 | 90.95 | 91.60 | 93.56 |
| Commercial | 41.02 | 28.96 | 78.82 | 90.31 | 78.06 | 91.93 | 95.25 | 91.73 |
| Road | 69.97 | 15.76 | 76.86 | 89.33 | 78.19 | 86.12 | 92.72 | 95.09 |
| Highway | 59.61 | 11.87 | 76.73 | 100.00 | 68.92 | 94.50 | 92.66 | 95.17 |
| Railway | 77.32 | 39.27 | 98.76 | 93.55 | 96.58 | 97.82 | 97.62 | 96.20 |
| Parking Lot 1 | 50.24 | 09.90 | 89.43 | 95.00 | 85.30 | 91.74 | 91.93 | 89.91 |
| Parking Lot 2 | 60.70 | 12.98 | 64.21 | 92.98 | 77.19 | 89.12 | 83.50 | 88.42 |
| Tennis Court | 99.59 | 80.16 | 100.00 | 99.60 | 100.00 | 97.97 | 92.30 | 93.93 |
| Running Track | 97.25 | 75.89 | 100.00 | 99.15 | 98.73 | 98.30 | 100.00 | 99.79 |
| OA | 77.83 | 32.02 | 84.02 | 95.22 | 87.65 | 93.83 | 94.36 | 94.62 |
| AA | 80.57 | 37.48 | 85.95 | 95.92 | 89.02 | 94.40 | 94.09 | 94.71 |
| $\kappa$ | 0.7601 | 0.2687 | 0.8265 | 0.9481 | 0.8660 | 0.9329 | 0.9388 | 0.9416 |

TABLE II
HOUSTON UNIVERSITY 2018 RESULTS

| Classes | RF on HSI | RF on LiDAR | SubFus | FusAtNet | EndNet | CoupledCNN | ASPP on LiDAR network | ASPP on both networks |
|---|---|---|---|---|---|---|---|---|
| Healthy Grass | 92.16 | 82.21 | 89.41 | 92.55 | 84.60 | 93.88 | 91.71 | 93.41 |
| Stressed Grass | 85.72 | 77.56 | 82.20 | 92.44 | 96.04 | 90.60 | 93.14 | 97.06 |
| Synthetic Grass | 99.25 | 100.00 | 100.00 | 73.54 | 100.00 | 100.00 | 100.00 | 100.00 |
| Evergreen Trees | 90.18 | 92.69 | 93.36 | 98.47 | 93.29 | 99.73 | 99.15 | 99.63 |
| Deciduos Trees | 73.24 | 87.04 | 88.15 | 98.97 | 82.61 | 98.57 | 99.56 | 99.09 |
| Soil | 84.14 | 85.88 | 99.48 | 97.21 | 96.18 | 99.64 | 100.00 | 100.00 |
| Water | 96.80 | 95.41 | 99.86 | 100.00 | 94.25 | 100.00 | 100.00 | 100.00 |
| Residential | 64.20 | 80.28 | 95.36 | 92.97 | 82.39 | 90.67 | 95.13 | 96.09 |
| Commercial | 31.12 | 77.00 | 70.47 | 89.54 | 93.78 | 90.05 | 96.62 | 96.79 |
| Sidewalk | 23.04 | 20.91 | 45.36 | 74.93 | 63.84 | 75.67 | 78.05 | 80.34 |
| Crosswalk | 22.36 | 30.51 | 53.29 | 77.45 | 59.81 | 75.73 | 78.64 | 83.75 |
| Major Thoroughfares | 38.79 | 52.62 | 78.60 | 95.98 | 19.49 | 88.32 | 79.98 | 93.80 |
| Road | 25.71 | 20.48 | 71.10 | 82.27 | 72.83 | 83.20 | 93.27 | 90.20 |
| Highway | 70.86 | 49.02 | 98.62 | 98.63 | 72.68 | 99.09 | 89.09 | 99.75 |
| Railway | 90.73 | 81.56 | 99.77 | 99.05 | 97.34 | 99.91 | 99.09 | 99.97 |
| Paved Parking Lot | 68.47 | 67.29 | 97.24 | 97.67 | 89.64 | 96.01 | 99.97 | 98.67 |
| Gravel Parking Lot | 97.07 | 95.29 | 100.00 | 100.00 | 94.41 | 100.00 | 98.09 | 100.00 |
| Cars | 55.18 | 80.51 | 94.84 | 98.77 | 51.85 | 98.43 | 100.00 | 98.72 |
| Trains | 60.09 | 96.82 | 99.00 | 98.71 | 76.57 | 99.72 | 97.70 | 100.00 |
| Seats | 82.57 | 93.61 | 100.00 | 97.07 | 96.65 | 100.00 | 100.00 | 99.99 |
| OA | 42.97 | 64.32 | 73.87 | 88.47 | 84.45 | 88.56 | 93.03 | 93.97 |
| AA | 67.59 | 73.33 | 87.80 | 92.81 | 80.91 | 93.96 | 95.52 | 96.06 |
| $\kappa$ | 0.3673 | 0.5683 | 0.6825 | 0.8536 | 0.7971 | 0.8543 | 0.9101 | 0.9221 |

TABLE III
TRENTO DATASET RESULTS

| Classes | RF on HSI | RF on LiDAR | SubFus | FusAtNet | EndNet | CoupledCNN | ASPP on LiDAR network | ASPP on both networks |
|---|---|---|---|---|---|---|---|---|
| Apple Trees | 77.65 | 27.46 | 99.64 | 100.00 | 80.03 | 99.97 | 100.00 | 100.00 |
| Buildings | 83.60 | 28.47 | 98.37 | 95.47 | 95.75 | 99.07 | 98.93 | 97.94 |
| Ground | 89.91 | 04.50 | 99.35 | 99.40 | 87.34 | 98.92 | 92.06 | 99.78 |
| Wood | 92.26 | 77.23 | 100.00 | 98.75 | 99.73 | 100.00 | 100.00 | 100.00 |
| Vineyard | 90.27 | 65.78 | 99.92 | 99.80 | 92.06 | 100.00 | 100.00 | 99.98 |
| Roads | 73.02 | 44.36 | 97.70 | 98.56 | 93.20 | 98.70 | 98.76 | 99.12 |
| OA | 86.72 | 57.31 | 99.51 | 98.95 | 93.16 | 99.75 | 99.51 | 99.63 |
| AA | 84.45 | 41.30 | 99.16 | 98.66 | 91.35 | 99.44 | 99.39 | 99.43 |
| kappa | 0.8224 | 0.4313 | 0.9935 | 0.9861 | 0.9088 | 0.9967 | 0.9963 | 0.9965 |

## V. CONCLUSION

This work proposed a deep learning-based fusion technique for the classification of hyperspectral and LiDAR data fusion. ASPP provides multi-resolution features which helps in enhancing the feature extraction. Small convolution kernels and parameter sharing layers were used for making the model more efficient and effective. In the fusion phase, feature-level and decision-level fusion strategies are used simultaneously. For the decision-level fusion, a weighted summation method is used whose weights depend on the performance of each output layer. Several experiments were performed on three data sets. The experimental results show that the proposed model can

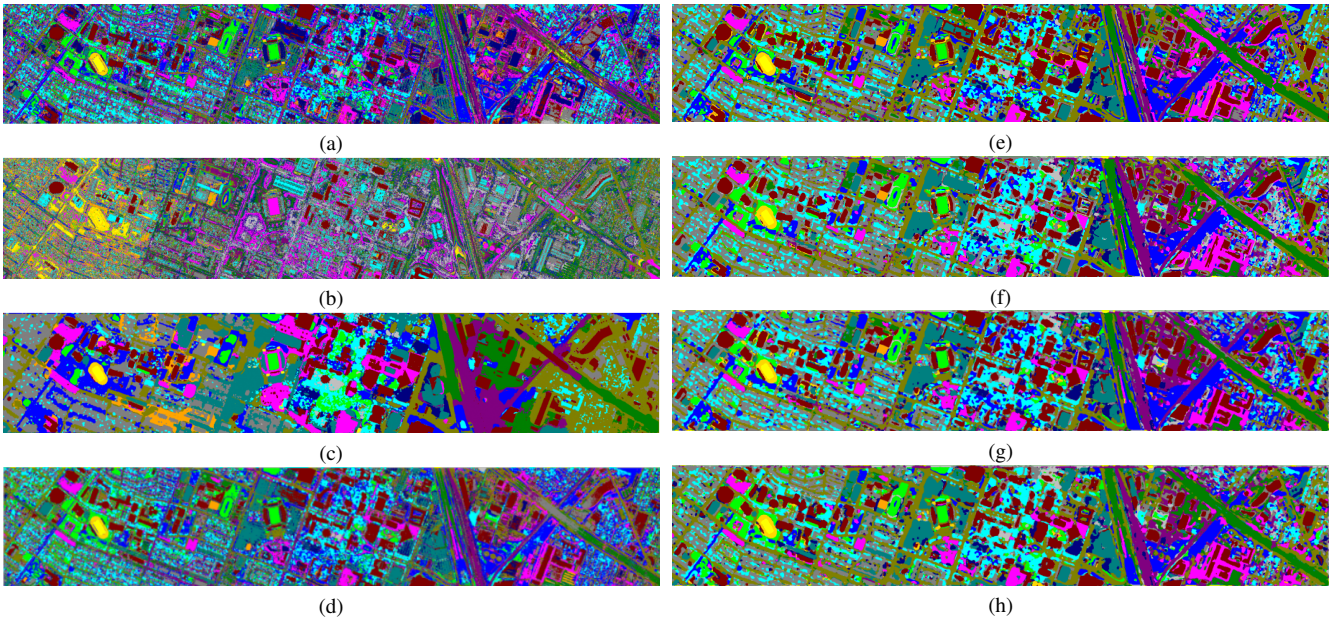



Fig. 17. Classification maps obtained for Houston 2013 dataset. (a) RF Classifier on HSI (b) RF Classifier on LiDAR (c) SubFus (d) EndNet (e) FusAtNet (f) CoupledCNN (g) CoupledCNN with aspp on LiDAR network (h) CoupledCNN with aspp on HS and LiDAR networks

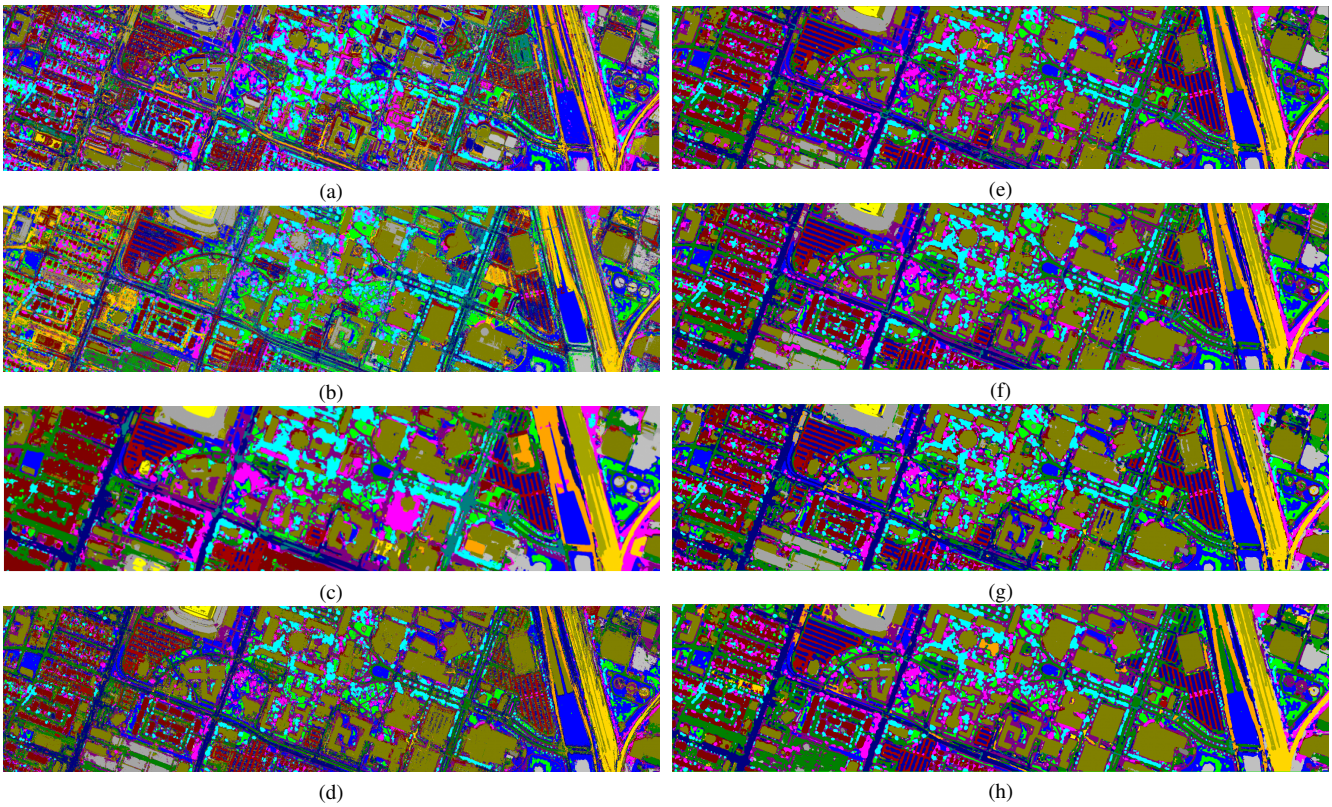



Fig. 18. Classification maps obtained for Houston 2018 dataset. (a)RF Classifier on HSI (b)RF Classifier on LiDAR (c) SubFus (d) EndNet (e) FusAtNet (f) CoupledCNN (g) CoupledCNN with ASPP on LiDAR network (h) CoupledCNN with ASPP on both HS and LiDAR networks

achieve significantly better performance on the Houston 2018 data. Additionally, the effects of different hyperparameters are evaluated on the classification performance. In the future, features extracted at different scales by ASPP can be fused which possibly improves the classification accuracy due to the multiscale deep representation for the fused features.

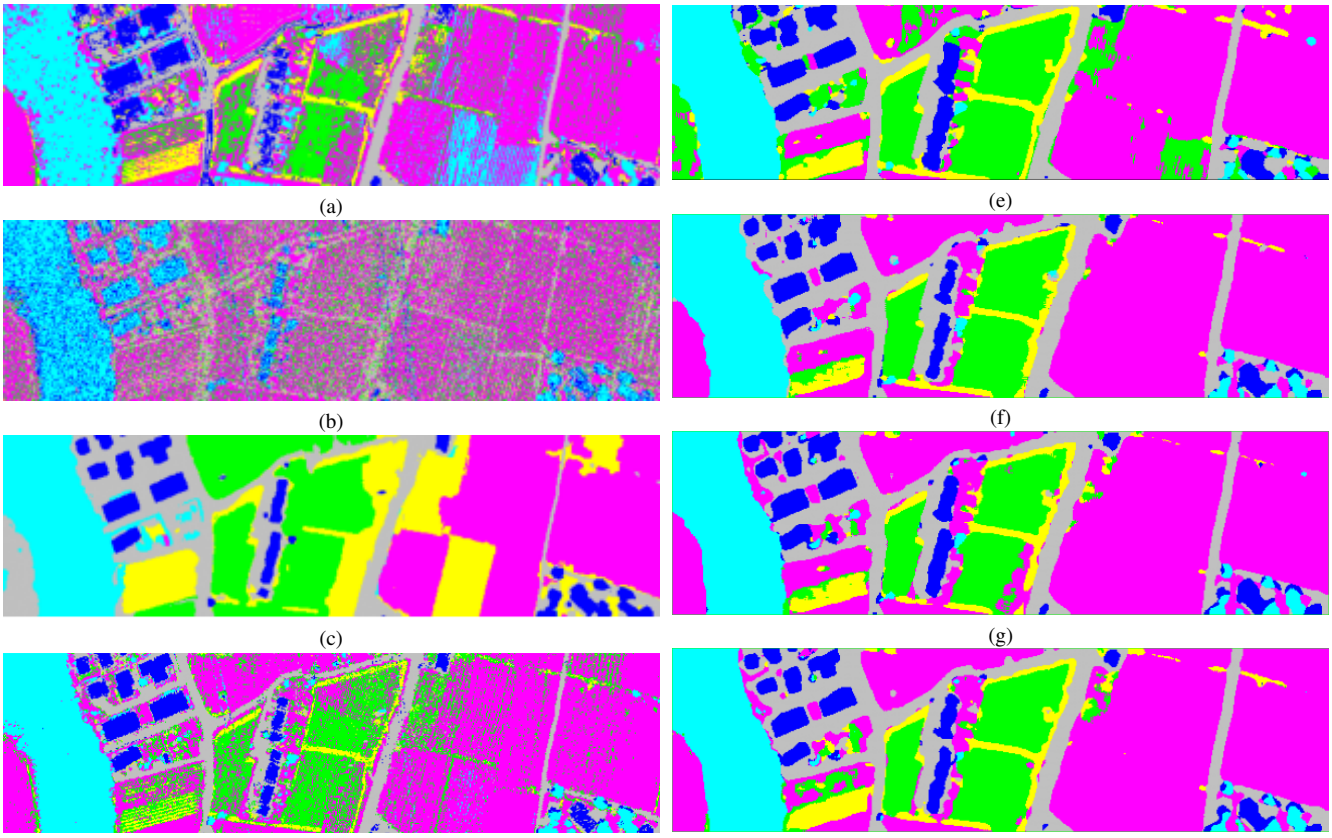


Fig. 19. Classification maps obtained for Trento dataset. (a) RF Classifier on HSI (b) RF Classifier on LiDAR (c) SubFus (d) EndNet (e) FusAtNet (f) CoupledCNN (g) CoupledCNN with ASPP on LiDAR network (h) CoupledCNN with ASPP on both HS and LiDAR networks

## REFERENCES


[1] Saumya Jetley, Nicholas A Lord, Namhoon Lee, and Philip HS Torr. Learn to pay attention. *arXiv preprint arXiv:1804.02391, 2018*

[2] Lichao Mou and Xiao Xiang Zhu. Learning to pay attention on spectral domain: A spectral attention module-based convolutional network for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing, 2019.*

[3] Juan Mario Haut, Mercedes E Paoletti, Javier Plaza, Antonio Plaza, and Jun Li. Visual attention-driven hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing, 57(10):8065–8080, 2019*

[4] S. Mohla, S. Pande, B. Banerjee and S. Chaudhuri. FusAtNet: Dual Attention based SpectroSpatial Multimodal Fusion Network for Hyperspectral and LiDAR Classification. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2020, pp. 416-425, doi: 10.1109/CVPRW50498.2020.00054.*

[5] R. Hang, Z. Li, P. Ghamisi, D. Hong, G. Xia, and Q. Liu, Classification of hyperspectral and LiDAR data using coupled CNNs, *IEEE Trans. Geosci. Remote Sens., vol. 58, no. 7, pp. 4939–4950, Jul. 2020.*

[6] D. Hong, L. Gao, R. Hang, B. Zhang and J. Chanussot. Deep Encoder-Decoder Networks for Classification of Hyperspectral and LiDAR Data. *IEEE Geoscience and Remote Sensing Letters, doi: 10.1109/LGRS.2020.3017414*

[7] R. Hang, Z. Li, P. Ghamisi, D. Hong, G. Xia and Q. Liu. Classification of Hyperspectral and LiDAR Data Using Coupled CNNs. *IEEE Transactions on Geoscience and Remote Sensing, vol. 58, no. 7, pp. 4939-4950, July 2020, doi: 10.1109/TGRS.2020.2969024.*

[8] C. Debes et al., Hyperspectral and LiDAR data fusion: Outcome of the 2013 GRSS data fusion contest. *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 7, no. 6, pp. 2405–2418, Jun. 2014.*

[9] J. Long, E. Shelhamer, T. Darrell, in Fully convolutional networks for semantic segmentation (IEEEBoston, 2015). *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015), pp. 3431–3340*

[10] Chen, Liang-Chieh & Papandreou, George & Kokkinos, Iasonas & Murphy, Kevin & Yuille, Alan. (2016). DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence. PP. 10.1109/TPAMI.2017.2699184.*