Shared Manifold Learning Using a Triplet Network for Multiple Sensor Translation and Fusion With Missing Data

Aditya Dutt, *Graduate Student Member, IEEE*, Alina Zare, *Senior Member, IEEE*, and Paul Gader, *Fellow, IEEE*

***Abstract*—Heterogeneous data fusion can enhance the robustness and accuracy of an algorithm on a given task. However, due to the difference in various modalities, aligning the sensors and embedding their information into discriminative and compact representations is challenging. In this article, we propose a contrastive learning-based multimodal alignment network to align data from different sensors into a shared and discriminative manifold where class information is preserved. The proposed architecture uses a multimodal triplet autoencoder to cluster the latent space in such a way that samples of the same classes from each heterogeneous modality are mapped close to each other. Since all the modalities exist in a shared manifold, a unified classification framework is proposed. The resulting latent space representations are fused to perform more robust and accurate classification. In a missing sensor scenario, the latent space of one sensor is easily and efficiently predicted using another sensor's latent space, thereby allowing sensor translation. We conducted extensive experiments on a manually labeled multimodal dataset containing hyperspectral data from AVIRIS-NG and NEON and light detection and ranging (LiDAR) data from NEON. Finally, the model is validated on two benchmark datasets: Berlin Dataset (hyperspectral and synthetic aperture radar) and MUUFL Gulfport Dataset (hyperspectral and LiDAR). A comparison made with other methods demonstrates the superiority of this method. We achieved a mean overall accuracy of 94.3% on the MUUFL dataset and the best overall accuracy of 71.26% on the Berlin dataset, which is better than other state-of-the-art approaches.**

***Index Terms*—Classification, contrastive learning, hyperspectral image (HSI), light detection and ranging (LiDAR), missing sensor, multimodal, remote sensing, robust data fusion, sensor translation, shared manifolds, synthetic aperture radar (SAR), triplet networks.**

I. Introduction

MULTIMODAL information fusion architectures have significantly outperformed unimodal models and achieved outstanding results on tasks such as land-use and land-cover classification (LULC) [1], [2], mineral exploration [3], [4], [5], urban planning [6], biodiversity conservation [7], sentiment detection [8], [9], [10], speech recognition, word sense disambiguation, fact extraction, and media description. In certain situations, one sensor is not sufficient to obtain robust performance. The conventional approach in multimodal fusion is to concatenate the representations of different modalities. This can further be divided into three categories, as shown in Fig. 1.

1) *Early Fusion:* In early fusion, the low-level features are extracted from each modality which are fused before being classified. However, the fusion of heterogeneous data sources into a fixed-size representation is challenging. Additionally, the model can lose important information to generate a common representation.
2) *Late Fusion:* In late fusion, the representation from each modality is classified, and a decision is made using methods such as majority voting [11], [12]. This method is also called *decision fusion*.
3) *Intermediate Fusion:* Intermediate fusion or Hybrid fusion is the most reliable and flexible fusion method. In this case, the intermediate representations of modalities are merged. In the context of a neural network, these representations are generated by the convolutional layers and fused gradually to form a shared representation layer.

Fusion methods can be classified into two groups: concatenation and alignment-based methods. Usually, concatenation-based methods extract the features using deep learning or other machine learning models and fuse the information for classification. Several new techniques have recently been developed for joint data classification using concatenated representations. It is essential to extract rich structural and texture information from the sensors to make a robust classification. To address this issue, Liao et al. [13] used morphological features to learn spatial information by using a structured morphological element of predefined size and shape. They proposed a graph-based model to couple the dimension reduction and fusion of information. However, using this method, the cloud-covered regions are not accurately classified because the morphological features of LiDAR are not computed properly. Morphological profiles have other limitations due to a fixed size structuring element. To overcome this issue, Rasti et al. [14] used extinction profiles to extract spatial and spectral information from the LiDAR and

Manuscript received 30 April 2022; revised 20 August 2022 and 25 September 2022; accepted 11 October 2022. Date of publication 27 October 2022; date of current version 9 November 2022. This work was supported by the National Science Foundation under Grant CNS-1747783. *(Corresponding author: Aditya Dutt.)*

Aditya Dutt is with the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL 32611 USA (e-mail: aditya.dutt@ufl.edu).

Alina Zare is with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611 USA (e-mail: azare@ufl.edu).

Paul Gader is with the Department of Computer and Information Science and Engineering, and the Department of Environmental Engineering Sciences, University of Florida, Gainesville, FL 32611 USA (e-mail: pgader@ufl.edu).

Digital Object Identifier 10.1109/JSTARS.2022.3217485

hyperspectral data, which were fused using orthogonal total variation component analysis. Extinction profiles are extrema-oriented filters, and experiments have shown that they perform better than morphological filters. However, after extracting the extinction profiles, a simple stacking of features is not efficient for classification. To address this, Zhao et al. [15] proposed a hierarchical random walk network to capture the spatial and spectral features. The random walk also reduces the problem of weak localization around boundaries. To extract rich spectral features, Zhao et al. [16] used Gabor convolutional layers to extract the multidirectional, multiscale, and semantic change features along with the Octave convolutional layers. Their model is able to capture directional texture features and frequency variation features efficiently.

After extracting the spatial and spectral features, an efficient fusion strategy is required. Hong et al. [17] proposed an end-to-end unified deep learning model for remote sensing imagery classification. They developed two deep learning models: Ex-Net to extract information and Fu-Net for data fusion. They used cross-modality learning and multimodality learning to enhance classification accuracy. However, their model still depends on a large number of samples to yield good results. To reduce the dependence on the number of samples, Hang et al. [18] proposed an unsupervised coupled CNN framework for hyperspectral and LiDAR data. They provided each modality as the input to these CNNs. This coupled convolution guides the CNNs to learn useful representations from each other, which helps with the fusion process. Furthermore, they used both feature level and decision level fusion to enhance the fusion process. Several unsupervised CNN architectures were also proposed [19], [20], [21], [22], [23] to perform sensor fusion.

During the fusion process, it is essential to preserve complementary information from all the heterogeneous modalities. The presence of redundant features during fusion decreases the classification performance. To tackle this problem, Zhang et al. [24] used an information fusion network (IP-CNN) to learn complementary information from heterogeneous sensors. They utilized the Gram matrices to achieve this task. This concept is similar to neural style transfer [25]. They used the gram matrices from LiDAR as a texture reference to the fused embeddings. Similarly, the HSI gram matrix serves as a spectral reference to the fused embeddings. Therefore, the fused embeddings carry spatial and spectral information from the original modalities. Guo et al. [26] also utilized the Gram matrices to control the image texture difference between the clean and degraded images. However, interpretability is a big issue in deep learning models. To increase the interpretability of fusion models, Hong et al. [27] proposed a shared and specific feature learning that is capable of decomposing data into modality-shared and modality-specific components, which enables a better information blending of multiple heterogeneous modalities.

Instead of the direct fusion process, many researchers have tried to develop alignment models where data from all modalities can be mapped onto a shared manifold while preserving the characteristics of all classes. This method provides an advantage by reducing the ambiguity in classification when only one modality is used. Therefore, the modalities with low discriminative ability can also perform a better classification. However, the decoupled parts of embeddings should not contribute when learning the common features. Hong et al. [28] developed a common subspace learning (CoSpace) model that learns shared feature representations from hyperspectral and multispectral correspondences. CoSpace achieved state-of-the-art results on the classification task. Their shared space also allowed sensor translation. Pournemt et al. [29] proposed a semisupervised alignment approach that is based on utilizing only the common knowledge present in the shared representations. They claimed that the decoupled information from different modalities obstructs the alignment process. To achieve this, they used the joint spectral analysis of the graph Laplacians of the different modalities. Their model showed promising results for real-world multimodal problems.

The alignment models are better than the concatenation models because they increase the classification performance in both unimodal and multimodal scenarios. In recent years, contrastive learning has gained much attention. Contrastive learning is an efficient way to address the data scarcity problem and align data from multiple modalities simultaneously. In 2015, Hoffer et al. [30] proposed Triplet Networks, which can learn representations by distance comparisons. After that, Triplet Networks have been successfully applied to several applications. In this article, we explore the usage of contrastive learning to tackle the multimodal manifold alignment challenge. Most of the methods discussed so far focus on joint representation learning and do not perform sensor translation. According to Baltrusaitis et al. [31], there are five technical challenges in a multimodal setting: Representation, Translation, Alignment, Fusion, and Colearning. We simultaneously address the first four challenges in our framework.

This article focuses on generating a discriminative shared manifold for multimodal data. The shared manifold is generated in such a way that the samples of a class from all the modalities are mapped close to each other while dissimilar classes are pushed apart. The proposed architecture contains one Triplet autoencoder and one standard autoencoder. Each autoencoder takes a separate modality as its input. An objective function is proposed based on the triplet loss, which encourages the latent space of both the autoencoders to be similar for the same class and dissimilar for different classes. To further bring the latent space of both modalities close to each other, we proposed a similarity enhancement (SE) term. After training, the resulting latent space embeddings are highly discriminative. For classification, we combine the latent space of all modalities and apply a KNN and shallow neural network. In order to perform sensor translation, we developed a regression network to predict the latent space of a missing sensor from the available sensor's latent space. Since the latent space is already clustered, this process becomes significantly easier. Once the latent space of the missing sensor is predicted, it can be reconstructed to generate the full-scale data. Experimental results show that the overall classification accuracy outperforms all the other models on MUUFL Gulfport [32] and Berlin Datasets [27].

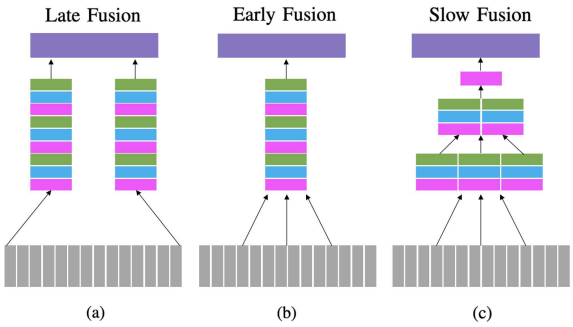


Fig. 1. Data fusion methods. (a) Late fusion. (b) Early fusion. (c) Intermediate/slow fusion.

The main contributions are highlighted as follows.

1) *Shared Manifold Representation/Alignment:* The proposed architecture aligns the embeddings from all the sensors into a common shared manifold. Since an autoencoder is used in this architecture, it is ensured that the latent space embeddings capture all the information necessary to reconstruct the sensor data. Additionally, the latent space is discriminative. The proposed model is not limited to remote sensing applications.
2) *Sensor Translation:* A shallow regression network is developed to predict a missing sensor's embeddings from other available sensors. The predicted embeddings can then be reconstructed using a decoder, allowing sensor translation.
3) *Classification:* Classification is performed using the fused embeddings of all the sensors. Furthermore, classifiers are created for single modalities as well. The representations show high classification accuracy using even a simple model such as KNN.

II. Proposed Method

A. CoMMANet: Shared Manifold Generation Architecture

The proposed framework, CoMMANet, consists of two autoencoders as shown in Fig. 2. The first autoencoder is a triplet autoencoder, which is implemented on *sensor* A. The autoencoder for *sensor* B is a standard autoencoder. The encoder of the *sensor* A is denoted by the embedding function $\mathbf{e}_A(.)$. The decoder of the *sensor* A is denoted by the embedding function, $\mathbf{d}_A(.)$. Similarly, the encoder and decoder of *sensor* B are denoted by the embedding functions $\mathbf{e}_B(.)$ and $\mathbf{d}_B(.)$, respectively. A standard CNN architecture is used for encoders $[\mathbf{e}_A(.)$ and $\mathbf{e}_B(.)]$ and decoders $[\mathbf{d}_A(.)$ and $\mathbf{d}_B(.)]$ in this article.

The inputs for the triplet autoencoder will be three samples from *sensor* A denoted by $\mathbf{S}_A^a$, $\mathbf{S}_A^p$, and $\mathbf{S}_A^n$, where $\mathbf{S}_A^a$ is the anchor, $\mathbf{S}_A^p$ is the positive, and $\mathbf{S}_A^n$ is the negative sample. The anchor and the positive samples share the same label. The negative sample belongs to a class other than the anchor's class. The latent space or embeddings of *sensor* A are denoted by

$$\mathbf{z}_A = \mathbf{e}_A\left(\mathbf{S}_A\right) \tag{1}$$

where $\mathbf{S}_A \in \mathbb{R}^{N_1}$ and N_1 is the dimensionality of data extracted from sensor A, and $\mathbf{z}_A \in \mathbb{R}^D$.

This implies

$$\mathbf{z}_A^t = \mathbf{e}_A\left(\mathbf{S}_A^t\right) \text{ for } t \in \{a, p, n\} \tag{2}$$

where $\mathbf{z}_A^t \in \mathbb{R}^D$ for $t \in \{a, p, n\}$, and D is the dimensionality of the latent space.

The reconstructed outputs of *sensor* A from the decoder are denoted by

$$\tilde{\mathbf{S}}_A^t = \mathbf{d}_A\left(\mathbf{z}_A^t\right) \text{ for } t \in \{a, p, n\} \tag{3}$$

where $\tilde{\mathbf{S}}_A^t \in \mathbb{R}^{N_1}$ for $t \in \{a, p, n\}$.

There is only one input from the *sensor* B, $\mathbf{S}_B^a$, which represents the anchor from *sensor* B. Similarly, the latent space or embeddings of *sensor* B anchor are denoted by

$$\mathbf{z}_B^a = \mathbf{e}_B\left(\mathbf{S}_B^a\right) \tag{4}$$

where $\mathbf{S}_B^a \in \mathbb{R}^{N_2}$, $\mathbf{z}_B^a \in \mathbb{R}^D$, N_2 is the dimensionality of data extracted from sensor B, and D is the dimensionality of the latent space.

Similarly, the reconstructed output of *sensor* B anchor is denoted by

$$\tilde{\mathbf{S}}_B^a = \mathbf{d}_B\left(\mathbf{z}_B^a\right) \tag{5}$$

where $\tilde{\mathbf{S}}_B^a \in \mathbb{R}^{N_2}$.

The goal of training a standard triplet network is to minimize an objective function. The objective function has the following properties.

1) Decrease when the distances between the anchor and positive sample embeddings decrease, i.e., the distance between samples of the same classes decreases.
2) Decrease when the distances between the anchor and negative sample embeddings increase, i.e., the distance between dissimilar classes increases.

However, this triplet loss is limited to one modality only. For a multimodal setting, we propose a multimodal triplet loss objective function. The multimodal triplet loss has the following properties.

1) Decrease when the distances between embeddings of samples from the same class decrease irrespective of the sensor.
2) Decrease when the distances between embeddings of samples from different classes and the same/ different sensors increase.

The objective function to train the CoMMANet is shown as

$$\mathbf{L}_{\text{CommaNet}} = \mathbf{L}_T + \mathbf{L}_E + \mathbf{L}_{\text{SE}}. \tag{6}$$

The working and influence of all these loss function terms are explained as follows.

1) Interpretation of Loss Term $\mathbf{L}_T$*:* The loss function term $\mathbf{L}_T$ is the multimodal triplet loss term, which is described by (7). The multimodal triplet loss function contains two terms: intrasensor triplet loss and intersensor triplet loss.

The intrasensor triplet loss term is the standard triplet loss term to train a triplet network. This term results in discriminative embeddings of *sensor* A. The effect of this term is demonstrated in Fig. 3.

The intersensor triplet loss term is a novel term that is introduced in this article. This term maps the anchor of the *sensor* B

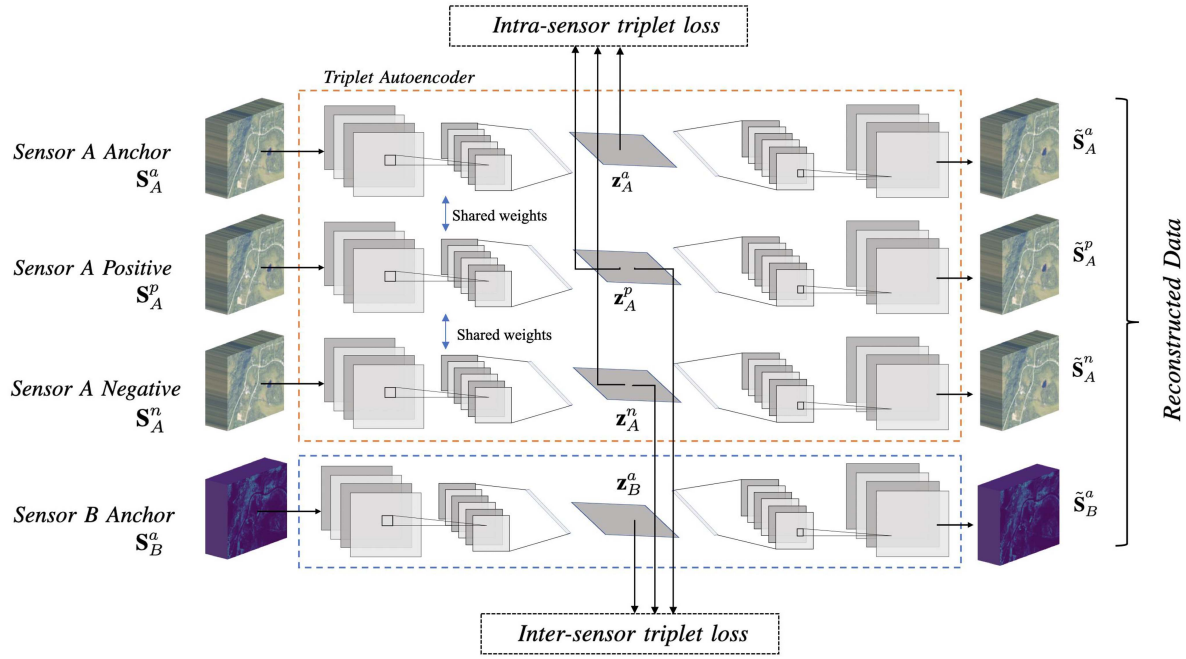


Fig. 2. Proposed CoMMANet architecture for the shared manifold generation.

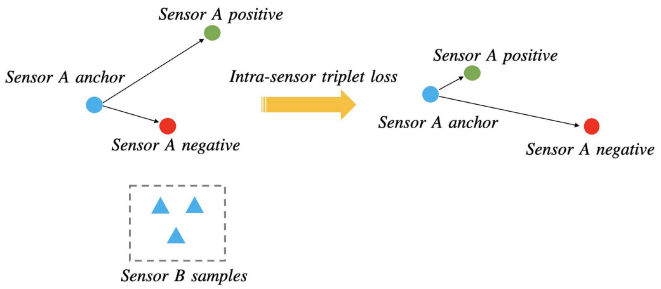


Fig. 3. Effect of the intrasensor triplet loss term is shown here. After applying this term, the network brings the anchor and positive embeddings from *sensor A* close to each other and pushes away the *sensor A* negative sample's embeddings.

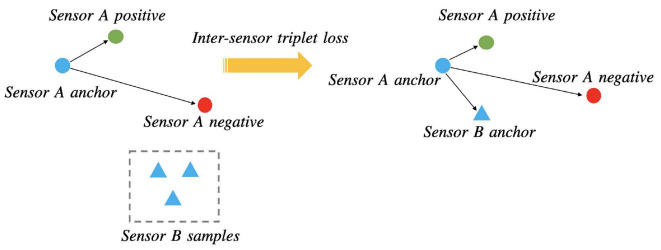


Fig. 4. Effect of the intersensor triplet loss term is shown here. Before applying this term, only the *sensor A* embeddings are clustered. After applying this term, the *sensor B* anchor also moves close to the anchor of *sensor A*.

close to the positive of *sensor A* and pushes away the negative of *sensor A*. In other words, we are treating the anchor of *sensor B* similar to the anchor of *sensor A*. This term is responsible for grouping the cross-modal embeddings in the shared latent space. The effect of the intersensor triplet loss term is demonstrated in Fig. 4.

$$\mathbf{L}_T = \sum_{k=1}^{K} \underbrace{\left\|\mathbf{z}_{A,k}^a - \mathbf{z}_{A,k}^p\right\|^2 - \left\|\mathbf{z}_{A,k}^a - \mathbf{z}_{A,k}^n\right\|^2 + \alpha}_{\text{Intra-sensor triplet loss term}} + \underbrace{\left\|\mathbf{z}_{B,k}^a - \mathbf{z}_{A,k}^p\right\|^2 - \left\|\mathbf{z}_{B,k}^a - \mathbf{z}_{A,k}^n\right\|^2 + \alpha}_{\text{Inter-sensor triplet loss term}} \tag{7}$$

where α is the margin hyperparameter, and there are K random triplets selected from the dataset for training.

2) Interpretation of Loss Term $\mathbf{L}_E$*:* This term consists of reconstruction loss terms of all the autoencoders. The three autoencoders are from the Triplet network for *sensor A*. The fourth autoencoder is for *sensor B*. The loss term $\mathbf{L}_E$ is described as

$$\mathbf{L}_E = \sum_{k=1}^{K} \left\|\mathbf{S}_{A,k}^a - \tilde{\mathbf{S}}_{A,k}^a\right\|^2 + \left\|\mathbf{S}_{A,k}^p - \tilde{\mathbf{S}}_{A,k}^p\right\|^2 + \left\|\mathbf{S}_{A,k}^n - \tilde{\mathbf{S}}_{A,k}^n\right\|^2 + \left\|\mathbf{S}_{B,k}^a - \tilde{\mathbf{S}}_{B,k}^a\right\|^2 \tag{8}$$

where K is the number of triplets selected for training.

3) Interpretation of Loss Term $\mathbf{L}_{SE}$*:* After using the loss term $\mathbf{L}_T$, the embeddings of both the sensors are clustered in the shared manifold. However, the term $\mathbf{L}_T$ is not sufficient to tightly cluster the embeddings of different sensors. The embeddings of the heterogeneous modalities are difficult to bring closer in the shared manifold because they have a different data structure and capture different kinds of information from the region of interest (ROI). Therefore, to enhance the clustering process, an SE term is introduced in the loss function, which is shown as

$$\mathbf{L}_{\text{SE}} = \gamma \left\|\mathbf{z}_A^a - \mathbf{z}_B^a\right\|^2 \tag{9}$$

where γ is the weight parameter, which is assigned a small value between 0 and 1. In our experiments, we usually set the value of γ less than 0.4.

This term is weighted by a parameter γ. However, the first two terms $\mathbf{L}_T$ and $\mathbf{L}_E$ are not weighted using a balancing parameter. The reason is that the term $\mathbf{L}_T$ is primarily responsible for clustering the embeddings of all the classes irrespective of the sensors. The term $\mathbf{L}_E$ is the reconstruction loss of all the autoencoders. Now, both $\mathbf{L}_T$ and $\mathbf{L}_E$ terms are given a weight of 1 as both are equally important. However, the term $\mathbf{L}_{\text{SE}}$ is simply enhancing the work done by the term $\mathbf{L}_T$. It does so by reducing the distance between anchors of both the sensors, which results in more compact clusters, as shown in Figs. 14 and 15. We observed that using small values of γ resulted in a significant improvement in clustering. Using larger values of γ could result in a collapsed model. Therefore, assigning this term an equal weight as the other terms ($\mathbf{L}_T$ and $\mathbf{L}_E$) is not useful. That is why this is the only term to be assigned a weighting/balancing parameter.

The experiments have been conducted with and without the $\mathbf{L}_{\text{SE}}$ term in the loss function and the comparison in performance is shown in Section IV-A.

Algorithm 1: Multimodal offline hard triplet mining.

Input: $\mathbf{S}_A$, $\mathbf{S}_B$
Output: $\mathbf{S}_A^a$, $\mathbf{S}_A^p$, $\mathbf{S}_A^n$, $\mathbf{S}_B^a$
Initialization : Checkpoint counter initialized
1: $i = 1$
2: Select K random triplets. Train the model for a specific number of epochs and save the model as Checkpoint 1.
3: **for** $i = 2$ to N **do**
4: Predict the embeddings of each sensor: $\mathbf{z}_A$, $\mathbf{z}_B$, using the Checkpoint $(i-1)$ model.
5: Select K anchor, positive, and negative from *sensor* A in such a way that: $d(\mathbf{z}_A^a, \mathbf{z}_A^n) < d(\mathbf{z}_A^a, \mathbf{z}_A^p)$, where d is a distance metric. The corresponding *sensor* B anchor samples are selected randomly.
6: Select K anchor $(sensor\ B)$, positive $(sensor\ A)$, and negative $(sensor\ A)$ in such a way that: $d(\mathbf{z}_B^a, \mathbf{z}_A^n) < d(\mathbf{z}_B^a, \mathbf{z}_A^p)$, where d is a distance metric. The corresponding *sensor* A anchor samples are selected randomly.
7: **end for**
8: **return** $\mathbf{S}_A^a$, $\mathbf{S}_A^p$, $\mathbf{S}_A^n$, $\mathbf{S}_B^a$

B. Training Strategy

A naïve approach to train the CoMMANet is randomly selecting multiple triplets. Since this approach is computationally intensive, an offline semihard/hard triplet selection strategy is used to train the network. According to Schroff et al. [33], selecting hard triplets early in training can lead to a collapsed model (i.e., $f(x) = 0$). The hard triplets mining strategy is also not robust to outliers. Therefore, the semihard triplet strategy is primarily used for the experiments (see Section IV-B for ablation study). Providing the network with a mixture of easy and semihard triplets makes the training process more stable, which results in a better performance. We developed a strategy for selecting the semihard/hard multimodal triplets (which is a variation of the standard hard triplet mining strategy).

The multimodal triplet mining is divided into two stages: intrasensor and intersensor triplets selection.

1) In the first stage, the semihard/hard triplets from the *sensor* A are sampled.
2) In the second stage, the anchor of *sensor* B is treated as the anchor of *sensor* A, and again the semihard/hard triplets are sampled. So, essentially, now the anchor is from *sensor* B, and the positive and negative are from *sensor* A.

Note that the triplet selection process requires latent space embeddings and not the original sensor data.

Let $\mathbf{S}_A^a$, $\mathbf{S}_A^p$, and $\mathbf{S}_A^n$ denote the anchor, positive, and negative samples from *sensor* A. $\mathbf{S}_B^a$ is the anchor from the *sensor* B. Let $\mathbf{z}_A$ and $\mathbf{z}_B$ denote the latent space embeddings of *sensor* A and *sensor* B, respectively. The pseudocode of the algorithm is presented in Algorithm 1.

C. Learning Mapping of Additional Sensors

The current architecture allows only two sensors to be mapped onto a shared manifold. However, there can be more than one sensor surveying the ROI, for example, *LiDAR*, *SAR*, and *HSI*. In such cases, first, two sensors are mapped onto the shared manifold. Then, the rest of the sensors are mapped using the following algorithm (architecture shown in Fig. 5).

1) Use the pretrained encoder from any of the first two sensors. Let us say, the *sensor* A pretrained encoder is used. According to (1), for an input $\mathbf{S}_A$, the embeddings $\mathbf{z}_A \in \mathbb{R}^D$ can be expressed as $\mathbf{e}_A(\mathbf{S}_A)$, where D is the dimensionality of the shared latent space.
2) Develop a standard autoencoder for *sensor* C.
3) The latent space of *sensor* C is forced to be similar to the pretrained encoder's latent space. For a given input $\mathbf{S}_C \in \mathbb{R}^{N_3}$, where N_3 is the dimensionality of data extracted from *sensor* C, the encoder of *sensor* C is represented by the function $\mathbf{e}_C(.)$ and the decoder is represented by the function $\mathbf{d}_C(.)$. The latent space $\mathbf{z}_C$ is denoted by $\mathbf{e}_C(\mathbf{S}_C)$, where $\mathbf{z}_C \in \mathbb{R}^D$, and D is the dimensionality of the shared latent space. The reconstructed output of the *sensor* C, $\tilde{\mathbf{S}}_C$, can be expressed as $\mathbf{d}_C(\mathbf{z}_C)$, where $\tilde{\mathbf{S}_C} \in \mathbb{R}^{N_3}$.
4) The model is trained using the following objective function:

$$\mathbf{L}_{sensorC} = \sum_{k=1}^{K} \|\mathbf{z}_{A,k} - \mathbf{z}_{C,k}\|^2 + \underbrace{\left\|\mathbf{S}_{C,k} - \tilde{\mathbf{S}}_{C,k}\right\|^2}_{\text{Reconstruction term}} \tag{10}$$

where K is the number of training samples.
5) After training the network using the above-mentioned objective function, the *sensor* C embeddings are mapped close to the *sensor* A embeddings. All three sensors now have embeddings in the shared manifold in a discriminative manner.

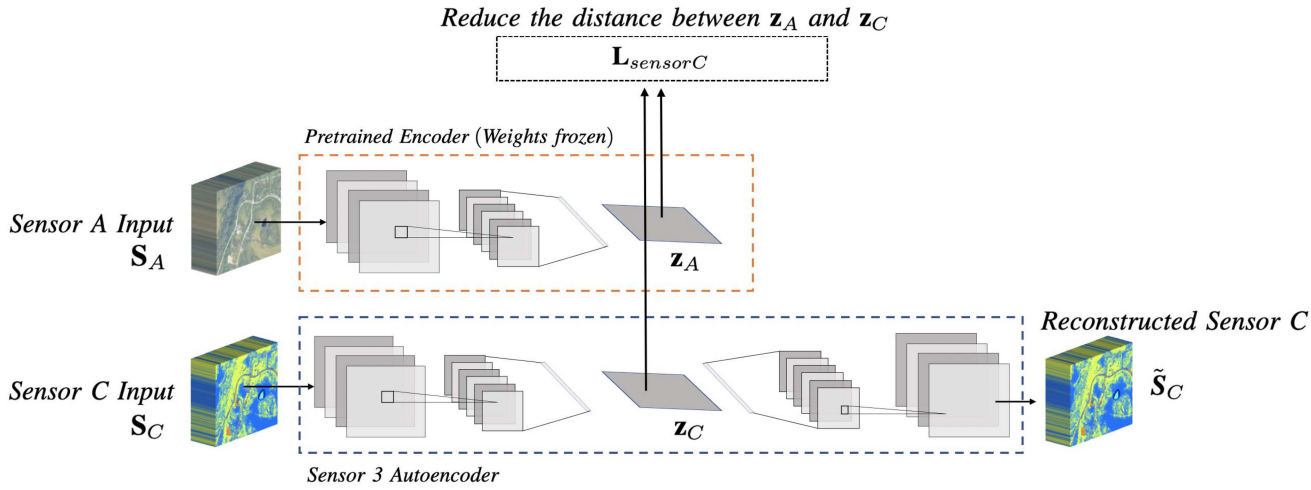


Fig. 5. Architecture to map the additional sensors onto the shared manifold is shown here. The embeddings of a new sensor are brought closer to the pretrained encoder's embeddings by reducing the distance between them.

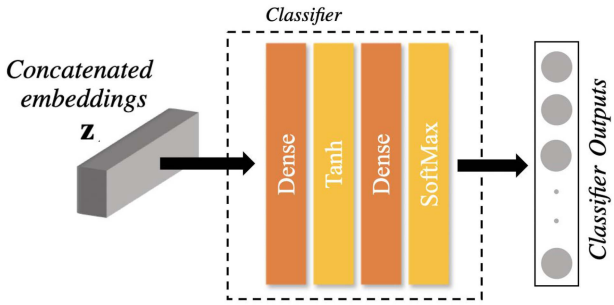


Fig. 6. Shared embeddings classification model. The term "Dense" signifies a fully connected layer whose neurons are connected to every neuron in the previous layer.

D. Classification Strategy

For classification, the embeddings of different modalities are concatenated, and a shallow fully connected neural network is applied. The classifier architecture is shown in Fig. 6. Alternatively, the nearest-neighbor classifier is also used to classify the concatenated embeddings. Since the embeddings are already clustered in the latent space, the nearest-neighbors classifier also gives a good performance which is comparable to the neural network classifier results.

To compute the nearest neighbors, the distance of a test sample is computed with every training sample. All the distances are averaged by the class. Now, the label of the class having the minimum distance is assigned to the test sample. It is not necessary to use all the training samples, and k-nearest neighbors can also be used.

To develop a unified classification model, instead of using the concatenated embeddings, the classification can be performed using the latent space of any of the sensors. Since the embeddings of all the sensors lie close to each other in the shared latent space, a classifier trained on one sensor can be used for other sensors as well. However, the classifier trained on the concatenated embeddings of all the sensors yields more accurate and robust results. Let $\mathbf{z}$ denote the flattened latent space of a single sensor. Let the hidden state $\mathbf{h_1}$ contain $\mathbf{U}$ hidden units (neurons). We use the convention that the addition of a matrix, $A \in \mathbb{R}^{N\times M}$, and a vector, $B \in \mathbb{R}^{1\times M}$, means that the vector B is added to every row of matrix A. The classifier output is given by

$$\mathbf{h}_1 = \tanh\left(\mathbf{z}\mathbf{W}_1 + \mathbf{b}_1\right) \tag{11}$$

where $\mathbf{z} \in \mathbb{R}^{N\times D}$, D is the dimensionality of the latent space, N is the number of samples, $\mathbf{W_1} \in \mathbb{R}^{D\times U}$ is the hidden state weight matrix, and $\mathbf{b_1} \in \mathbb{R}^{1\times U}$ is the bias. Thus, $\mathbf{h_1} \in \mathbb{R}^{N\times U}$.

Let σ be the softmax function applied on the final layer, and $\mathbf{O}$ be the number of units (neurons) in the output layer

$$^{\mathbf{y}} = \sigma\left(\mathbf{h}_1\mathbf{W}_2 + \mathbf{b}_2\right) \tag{12}$$

where $\mathbf{W_2} \in \mathbb{R}^{U\times O}$ is the hidden state weight matrix, $\mathbf{b_2} \in \mathbb{R}^{1\times O}$ is the bias, and $\mathbf{h_1} \in \mathbb{R}^{N\times U}$. Thus, $^{\mathbf{y}} \in \mathbb{R}^{N\times O}$ is the classifier output, and N is the number of samples.

If the latent spaces of both *sensor A* and *sensor B* are fused to perform classification, the input of the classifier will be the concatenated flattened latent space, $\mathbf{z}_{\text{fused}}$, of both the sensors, where $\mathbf{z}_{\text{fused}} \in \mathbb{R}^{N\times(2*D)}$, D is the dimensionality of the latent space of each sensor, and N is the number of samples.

E. Missing Sensor's Embeddings Prediction/Sensor Translation

After following the steps in Section II-A, the embeddings of all the sensors are successfully mapped onto the shared manifold. Let $\mathbf{S}_A$ and $\mathbf{S}_C$ be the two sensors, where $\mathbf{S}_C$ is the missing sensor. The clustered embeddings of both the sensors will lie close to each other in the shared manifold. In an ideal scenario, the sensors would be homogeneous, and their embeddings will completely overlap. However, if the sensors are heterogeneous, the embeddings will not overlap entirely due to the variation in data structure and information captured by different sensors. Therefore, a shallow regression network is developed, which predicts the embeddings of the missing sensor using the available sensor. The architecture of the regression neural network is shown in Fig. 7.

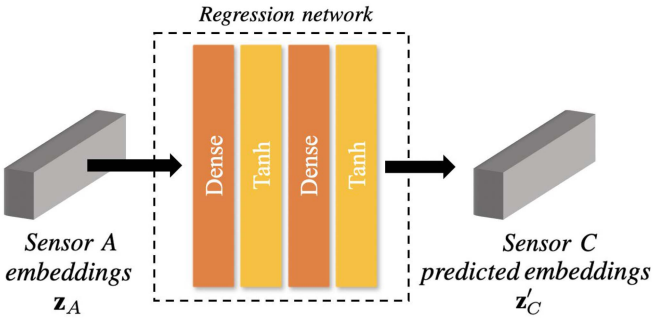


Fig. 7. Missing sensor's embeddings prediction/sensor translation model. The term "Dense" signifies a fully connected layer whose neurons are connected to every neuron in the previous layer.

Let $\mathbf{z}_A$ be the flattened latent space of *sensor* A (the available/predictor sensor). Here also, we use the convention that the addition of a matrix, $A \in \mathbb{R}^{N\times M}$, and a vector, $B \in \mathbb{R}^{1\times M}$, means that the vector B is added to every row of matrix A. The predicted latent space of *sensor* C (the missing sensor) $\mathbf{z}'_C$ is given by

$$\mathbf{h}_3 = \tanh\left(\mathbf{z}_A\mathbf{W}_3 + \mathbf{b}_3\right) \tag{13}$$

where $\mathbf{z}_A \in \mathbb{R}^{N\times D}$, D is the dimensionality of the latent space, and N is the number of samples. $\mathbf{W_3} \in \mathbb{R}^{D\times V}$ is the hidden state weight matrix, V is the number of units (neurons) in the hidden layer $\mathbf{h}_3$, and $\mathbf{b_3} \in \mathbb{R}^{1\times V}$ is the bias. Thus, $\mathbf{h_3} \in \mathbb{R}^{N\times V}$

$$\mathbf{z}'_C = \tanh\left(\mathbf{h}_3\mathbf{W}_4 + \mathbf{b}_4\right) \tag{14}$$

where $\mathbf{W_4} \in \mathbb{R}^{V\times D}$ is the hidden state weight matrix, D is the number of units (neurons) in the output layer $\mathbf{z}'_C$, and N is the number of samples. $\mathbf{h_3} \in \mathbb{R}^{N\times V}$ is the previous layer output, and $\mathbf{b_4} \in \mathbb{R}^{1\times D}$ is the bias. Thus, $\mathbf{z}'_C \in \mathbb{R}^{N\times D}$ is the predicted latent space of *sensor C*.

After predicting the missing sensor's latent space, its decoder can be used to reconstruct the original data. This way sensor translation can be performed. Here, the predicted latent space of *sensor* C can be reconstructed using its decoder $\mathbf{d}_C(.)$. The reconstructed *sensor* C data $\tilde{\mathbf{S}}_C$ can be expressed as

$$\tilde{\mathbf{S}}_C = \mathbf{d}_C\left(\mathbf{z}'_C\right) \tag{15}$$

where $\tilde{\mathbf{S}}_C \in \mathbb{R}^{N\times N_3}$, N is the number of samples, and N_3 is the dimensionality of data extracted from sensor C.

In our experiments, the latent space values are between -1 and 1; therefore, a *tanh* activation is used. However, any other activation function can also be used according to convenience.

III. Experiments

A. Datasets Description

1) *AVIRIS-NG and NEON Data:* For initial experiments, the data are used from AVIRIS-NG and NEON sensors. The HSI data are acquired from NEON and AVIRIS-NG sensors. The LiDAR data are available only from NEON. The data were acquired over Healy, Alaska, in 2017. The NEON HSI data, which are captured by the NEON Airborne Observation Platform Imaging Spectrometer, comprises 426 bands, each with a spectral resolution of 5 nm covering the range from 380 μm to 2510 μm. The LiDAR data were acquired by the Optech Gemini sensor. The data are composed of 1000 $\times$ 1000 pixels with a spatial resolution of 1 m.
The AVIRIS-NG sensor comprises 425 bands covering the range from 380 μm to 2510 μm with a spectral resolution of 5 nm. The spatial resolution of AVIRIS-NG data is 5 m. QGIS software was used to extract the common ROI between NEON and AVIRIS-NG data. The original AVIRIS-NG data are composed of 213 $\times$ 213 pixels, which was upscaled to 1000 $\times$ 1000 pixels. The RGB image of ROI is shown in Fig. 8. The NEON Canopy Height Model was used to generate the ground truth for trees. The rest of the data were manually labeled. For the experiments, four classes are used: mixed ground, road, building, and trees. The distribution of the classes in the dataset is described in Table I.

Fig. 8. RGB image of the ROI used from the AVIRIS-NG and NEON data.

TABLE I
Description of the AVIRIS-NG/NEON Data

No.	Class	Number of Samples
1	Mixed Grass	46324
2	Buildings	22538
3	Trees	19054
4	Road/ Ground	1300
	Total	**89216**

2) *MUUFL Gulfport Data:* The Gulfport dataset was acquired in November 2010 over the University of Southern Mississippi Gulfport Campus, Long Beach, Mississippi, USA. The dataset is composed of coregistered HSI and LiDAR-based digital surface model. The HSI data were acquired by the Compact Airborne Spectrographic Imager (CASI)-1500 sensor and the LiDAR data were acquired by the Gemini airborne laser terrain mapper LiDAR sensor. The data consist of 325 $\times$ 220 pixels with 64 spectral channels covering the range from 375 nm to 1050 nm at a spectral sampling of 10 nm. The spatial resolution of HSI is 0.54 m across the track and 1.0 m along the track. The spatial resolution of LiDAR data is 0.60 m across the track and 0.78 m along the track. In this dataset, 11 classes

TABLE II
DESCRIPTION OF THE MUUFL DATA

No.	Class	Number of Samples
1	Trees	23246
2	Mostly Grass	4270
3	Mixed ground surface	6882
5	Road	6687
6	Water	466
7	Building shadow	2233
8	Building	6240
9	Sidewalk	1385
10	Yellow curb	183
11	Cloth panels	269
	Total	**53687**

TABLE III
DESCRIPTION OF THE HS-SAR BERLIN DATA

No.	Class	Training Sample	Testing Samples
1	Forest	443	54511
2	Residential Area	423	268219
3	Industrial Area	499	19067
4	Low Plants	376	58906
5	Soil	331	17095
6	Allotment	280	13025
7	Commercial Area	298	24526
8	Water	170	6502
	Total	**2820**	**461851**

are investigated for the land cover classification task. The distribution of the classes in the dataset is described in Table II.

3) *HS-SAR Berlin data:* This dataset describes the Berlin urban and its neighboring rural area. It consists of an EnMAP HS image which is simulated from the HyMap HS data. The corresponding Sentinel-1 SAR data of the same region were prepared using the ESA toolbox SNAP after applying orbit profile, radiometric calibration, speckle reduction, and terrain correction. The HSI data consist of 797 $\times$ 220 pixels with 244 spectral bands covering the range from 400 to 2500 nm. The SAR data are a dual-polarSAR containing four bands. The processed SAR image consists of 1723 $\times$ 476 pixels and has a 13.89 m GSD. A nearest-neighbor interpolation is performed on the HSI image to make SAR and HSI images of the same size. The ground truth map is created using OpenStreetMap data. This dataset contains eight categories for the classification task. The distribution of the classes in the dataset is described in Table III.

B. Experimental Setup

To measure the classification performance, Overall Accuracy (OA), Average Accuracy (AA), and Kappa coefficient are used. In the missing sensor scenario, the mean-squared error metric is used to evaluate the performance of predicted latent space embeddings and reconstructed sensor data.

1) *AVIRIS-NG/ NEON Data*
Shared Embeddings: The CoMMANet is applied on AVIRIS-NG and NEON HSI data. A total of 40% of the data from each class is used for training and the rest 60% for testing. From HSI data, a single pixel is used for training. The HSI data from both sensors lie between 0 and 1. The weight for the SE term γ is set to 0.4. The margin α is set to 1. A *tanh* activation is used on the latent space so that the embeddings remain bounded. The embeddings are 32-dimensional vectors. To find the optimal latent space size, experiments are conducted by setting the latent space dimensionality to 16, 32, 64, and 128. The best performance is achieved when the latent space is 32-dimensional. Therefore, 32-dimensional latent space is used to generate shared embeddings.
For training the triplet network, the semihard triplets mixed with a few easy triplets are used instead of hard triplets because hard triplets are sensitive to outliers. The SE term is also added to the loss function. The CoMMANet is trained for 10 checkpoints for 450 epochs. At the beginning of each checkpoint, new triplets are computed to focus on the samples showing a higher loss. In each checkpoint, 100 000 triplets are used. The learning rate is set to 0.0005, the batch size is set to 256, and the model is trained using the Adam optimizer.
Classification: The CoMMANet embeddings of NEON HSI and AVIRIS-NG HSI are concatenated and used for classification. The classification is performed using a single neural network having two hidden layers with 128 and 64 hidden units, respectively. The classification results are given in Table IV. A K-fold Monte Carlo experiment is conducted, and results are reported along with the mean and standard deviation of accuracy.
Additionally, to demonstrate the effectiveness of a unified classification model, a classifier is trained on the embeddings of one sensor and tested on the embeddings of other sensors. Since all the embeddings are clustered in the shared manifold, the models trained on one sensor show comparable accuracy on other sensors as well. The results are given in Table V.
Mapping additional sensor (LiDAR): To map an additional sensor (NEON LiDAR) onto the shared manifold, the pretrained encoder of NEON HSI is used. The size of the input patches from LiDAR is 5 $\times$ 5 pixels. Each channel of LiDAR data is scaled between 0 and 1. The model is trained for 300 epochs with a learning rate of 0.0005 and a batch size of 256. However, the model converges very fast (in approx. 200 epochs). The shared embeddings of NEON

TABLE IV
CLASSIFICATION PERFORMANCE OF THE COMMANET EMBEDDINGS ON THE AVIRIS-NG/ NEON DATA IS SHOWN

No.	Class	NEON HSI + AVIRIS-NG HSI
		Neural Network
1	Mixed Grass	98.3 ± 0.05
2	Buildings	98.1 ± 0.07
3	Trees	99.2 ± 0.02
4	Road/ Ground	98.0 ± 0.05
OA (%)		$\mathbf{98.3 \pm 0.02}$
AA (%)		$\mathbf{98.7 \pm 0.05}$
Kappa (%)		$\mathbf{98.9 \pm 0.02}$

TABLE V
AVIRIS-NG/NEON DATA: THE EFFECTIVENESS OF A UNIFIED CLASSIFICATION MODEL IS SHOWN HERE

Neural Network Classifier *trained* on embeddings of	Evaluation Metric (%)	Neural Network Classifier *tested* on embeddings of		
		NEON HSI	AVIRIS-NG HSI	LiDAR
NEON HSI	OA	98.6 ± 0.06	98.5 ± 0.05	93.16 ± 0.12
	AA	98.4 ± 0.12	95.6 ± 0.15	93.1 ± 0.12
	Kappa	97.7 ± 0.13	97.8 ± 0.02	95.6 ± 0.08
AVIRIS-NG HSI	OA	95.2 ± 0.08	98.5 ± 0.05	96.2 ± 0.10
	AA	94.6 ± 0.10	95.5 ± 0.11	90.5 ± 0.15
	Kappa	92.3 ± 0.12	97.4 ± 0.04	93.8 ± 0.12
LiDAR	OA	97.5 ± 0.10	97.2 ± 0.03	98.4 ± 0.04
	AA	97.1 ± 0.20	94.4 ± 0.16	95.7 ± 0.10
	Kappa	95.9 ± 0.05	95.5 ± 0.10	97.4 ± 0.06

The classifier is trained on embeddings of one sensor and tested on embeddings of another sensor.

TABLE VI
AVIRIS-NG/NEON DATA: THE LATENT SPACE OF ONE SENSOR IS PREDICTED USING ANOTHER SENSOR AND THEN RECONSTRUCTED USING THE SENSOR'S DECODER

Sensor (Predictor)	Sensor (Predicted)	Latent Space MSE (± 0.0005)	Reconstructed Data MSE (± 0.0001)
NEON HSI	LiDAR	0.042	0.008
NEON HSI	AVIRIS-NG HSI	0.048	0.002
AVIRIS-NG HSI	LiDAR	0.056	0.008
AVIRIS-NG HSI	NEON HSI	0.038	0.001
LiDAR	AVIRIS-NG HSI	0.180	0.012
LiDAR	NEON HSI	0.157	0.010

** Latent space values ∈ [-1,1] All sensors data ∈ [0,1]*

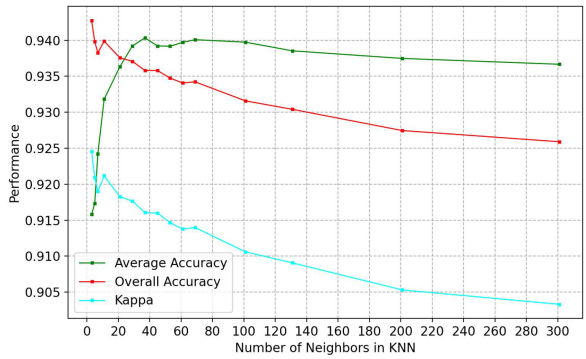


Fig. 9. Sensitivity of the KNN model to the value of k is shown for the MUUFL dataset. The KNN is used for the classification of concatenated HSI and LiDAR embeddings from CoMMANet.

HSI, AVIRIS-NG HSI, and NEON LiDAR are shown in Fig. 14.

Missing sensor prediction/reconstruction: To simulate a missing sensor scenario, one sensor's embeddings are predicted from another sensor's embeddings. The values of the α and γ parameters are the same as used for classification. A shallow neural network with two hidden layers is used to predict the embeddings. The two hidden layers contain 128 and 64 hidden units, respectively. A *tanh* activation is used on the final layer. The batch size is set to 64. The model is trained using fivefold validation for 50 epochs in each fold using the Adam optimizer. The prediction results are given in Table VI. After the embeddings of a sensor are predicted, the decoder is used to reconstruct the original data.

2) *MUUFL Gulfport Data*

Shared embeddings: In this dataset also, 40% of the data from each class is used for training and the rest 60% for testing. For LiDAR, the input patch size of 13×13 pixels is found to be optimal. Similar to the previous dataset, a single pixel is used from HSI. The HSI data already exist between 0 and 1. The LiDAR height and intensity are scaled between 0 and 1. The optimal weight for the SE term γ is 0, and the margin α is set to 0.4. The value of α depends on the number of classes and the activation function used on the latent space. If the number of classes is high, then a higher value of α will lead to a collapsed model (i.e., $f(x) = 0$). The embeddings of each sensor are 32-dimensional vectors, and a *tanh* activation is applied on the latent space so that the embeddings remain bounded between -1 and 1. The semihard triplets mixed with a few easy triplets are used for training the triplet network. The SE term is also added to the loss function. The checkpoints are created in a similar way as the previous dataset. The CoMMANet is trained for 10 checkpoints with 50 epochs in each checkpoint. In each checkpoint, 800 000 triplets are used. The learning rate is set to 0.001, the batch size is set to 1024, and the model is trained using the Adam optimizer.

Classification: The CoMMANet embeddings of both sensors are concatenated and classified using an ensemble of three shallow neural networks, which yielded an overall classification accuracy of $93.1\% \pm 0.15$. The best average classification accuracy is $94.0\% \pm 0.10$, which is achieved using KNN (with $k = 35$). However, the best OA is $94.3\% \pm 0.12$, which is achieved using an ensemble of KNN (with $k = 35$) and three neural networks (initialized with different weights). The sensitivity of the KNN model to the value of k is shown in Fig. 9. Similar to the previous dataset, a K-fold Monte Carlo experiment is conducted here, and the mean and standard deviation of accuracy are reported in Table VII. The classification map and ground truth are shown in Fig. 10.

For this dataset also, unified classification experiments are conducted, and the results are given in Table VIII.

TABLE VII
CLASSIFICATION PERFORMANCE (%) ON THE MUUFL DATA IS SHOWN HERE

No.	Class	HSI + LiDAR		
		Neural Network Accuracy	KNN ($k = 35$) Accuracy	Neural Network + KNN ($k = 35$) Accuracy
1	Trees	96.8 ± 0.03	96.6 ± 0.03	97.0 ± 0.02
2	Mostly Grass	84.5 ± 0.15	87.1 ± 0.15	87.3 ± 0.12
3	Mixed ground surface	86.9 ± 0.25	89.6 ± 0.48	89.2 ± 0.26
4	Dirt and sand	90.6 ± 0.40	94.8 ± 0.32	94.8 ± 0.30
5	Road	95.5 ± 0.18	96.3 ± 0.12	96.3 ± 0.10
6	Water	94.6 ± 0.28	96.2 ± 0.23	96.5 ± 0.21
7	Building shadow	83.5 ± 0.50	80.9 ± 0.21	83.4 ± 0.21
8	Building	72.6 ± 0.08	98.0 ± 0.12	98.0 ± 0.08
9	Sidewalk	82.6 ± 1.10	84.3 ± 0.24	81.5 ± 1.00
10	Yellow curb	85.5 ± 0.52	90.9 ± 0.40	89.9 ± 0.40
11	Cloth panels	95.8 ± 0.35	98.7 ± 0.43	98.1 ± 0.38
OA (%)		$\mathbf{93.1 \pm 0.15}$	$\mathbf{94.0 \pm 0.10}$	$\mathbf{94.3 \pm 0.12}$
AA (%)		$\mathbf{87.6 \pm 0.08}$	$\mathbf{93.1 \pm 0.15}$	$\mathbf{91.9 \pm 0.10}$
Kappa (%)		$\mathbf{90.8 \pm 0.10}$	$\mathbf{92.1 \pm 0.12}$	$\mathbf{92.3 \pm 0.08}$

The concatenated CoMMANet embeddings of HSI and LiDAR are classified using a neural network, KNN, and a KNN and neural network ensemble.

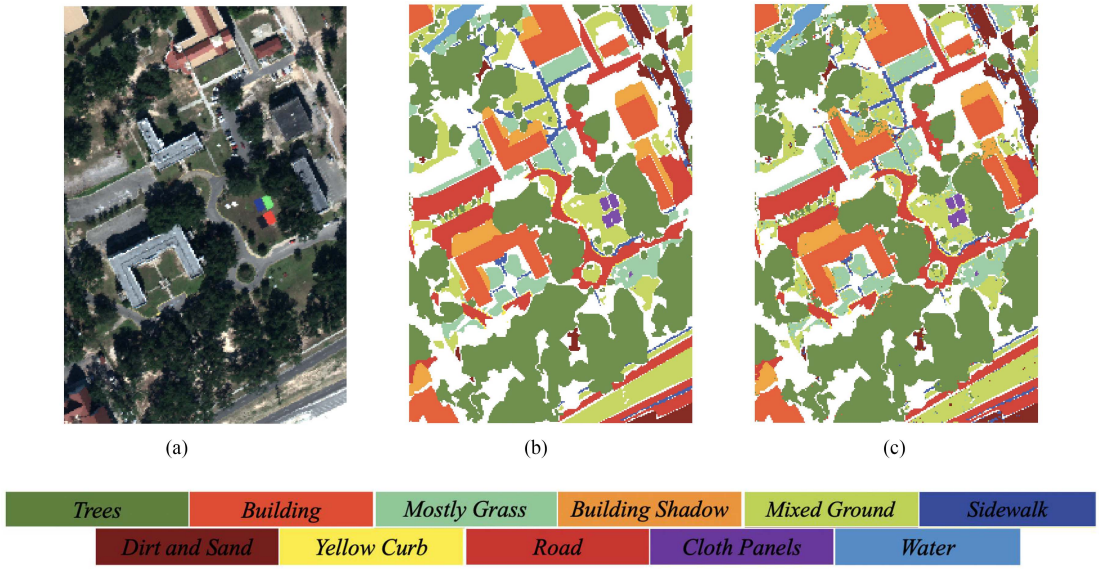


Fig. 10. Visualization of the classification maps from the MUUFL data. (a) RGB image. (b) Ground truth map. (c) Classification map.

TABLE VIII
MUUFL DATA: THE EFFECTIVENESS OF A UNIFIED CLASSIFICATION MODEL IS SHOWN HERE

Neural Network Classifier *trained* on embeddings of	Evaluation Metric (%)	Neural Network Classifier *tested* on embeddings of	
		HSI	LiDAR
HSI	OA	88.22 ± 0.05	87.04 ± 0.10
	AA	84.84 ± 0.45	74.85 ± 0.40
	Kappa	84.58 ± 0.08	82.89 ± 0.21
LiDAR	OA	88.01 ± 0.20	87.47 ± 0.12
	AA	84.82 ± 0.32	76.12 ± 0.42
	Kappa	84.35 ± 0.12	83.50 ± 0.18

The classifier is trained on embeddings of one sensor and tested on embeddings of another sensor.

The classifier trained on HSI embeddings shows a good accuracy on LiDAR embeddings as well and vice versa.

Missing sensor prediction/reconstruction: For the prediction of one sensor's embeddings from another sensor, a shallow neural network is used with two hidden layers. The embeddings are 32-dimensional vectors. The two hidden layers contain 128 and 64 hidden units, respectively. A *tanh* activation is used on the final layer. The batch size is set to 32. The model is trained using fivefold validation for 100 epochs in each fold using the Adam optimizer. The mean-squared error of the latent space predictions are least when α and γ are both set to 0.4 while training the CoMMANet to generate shared embeddings.

After the embeddings of a sensor are predicted, the decoder is used to reconstruct the original data. The reconstructed LiDAR height and intensity are shown in Fig. 11. Similarly, HSI embeddings are predicted using the LiDAR embeddings and then reconstructed using the decoder model. The prediction metrics are given in Table IX.

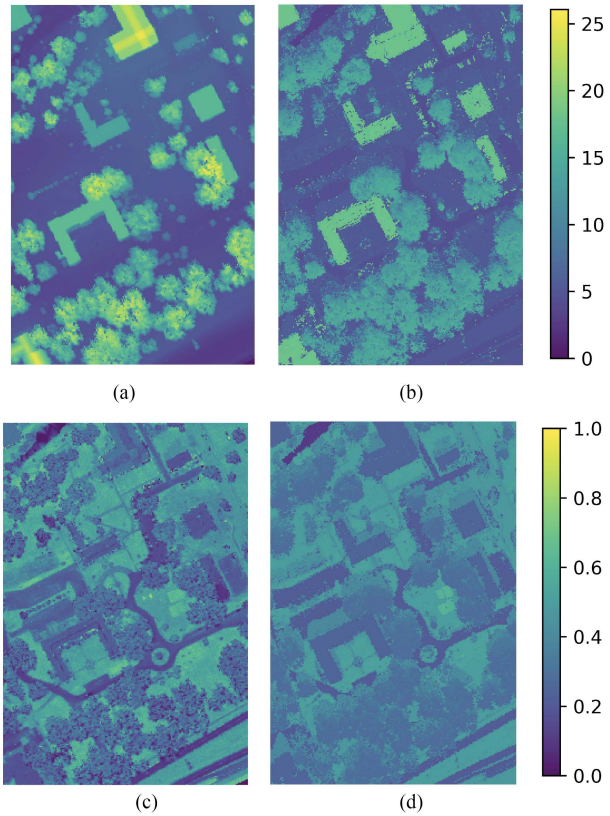


Fig. 11. LiDAR latent space is predicted using the HSI latent space from the MUUFL dataset. Using the LiDAR decoder, the predicted latent space is reconstructed to get the LiDAR height and intensity, which are shown here. (a) LiDAR height ground truth (in meters). (b) LiDAR height predicted from HSI (in meters). (c) LiDAR intensity ground truth (between 0 and 1). (d) LiDAR intensity predicted from HSI (between 0 and 1).

TABLE IX
MUUFL DATA: THE LATENT SPACE OF ONE SENSOR IS PREDICTED USING ANOTHER SENSOR, AND THEN RECONSTRUCTED USING THE SENSOR'S DECODER

Sensor (Predictor)	Sensor (Predicted)	Latent space MSE	Reconstructed Data MSE
HSI	LiDAR	0.10 ± 0.04	0.015 ± 0.003
LiDAR	HSI	0.10 ± 0.06	0.009 ± 0.004

* *Latent space values ∈ [-1,1]* *HSI and LiDAR data ∈ [0,1]*

The images of original and predicted HSI spectra are available here.[1] The data are reconstructed with a low mean-squared error. In LiDAR data reconstruction, the classes having a lot of variation such as trees and mixed ground show slightly higher reconstruction loss. It is slightly more challenging to reconstruct the HSI spectra using LiDAR only. Nevertheless, the reconstruction loss of spectra is very low.

3) *HS-SAR Berlin DataShared embeddings:* In this dataset, the training and testing samples are provided separately. The optimal size of the input patches from SAR is 11 × 11 pixels. A single pixel is used from HSI for training.

[1]https://github.com/GatorSense/AdaptiveManifoldLearning_CBL

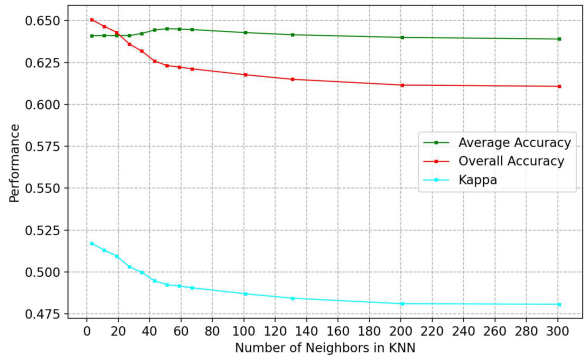


Fig. 12. Sensitivity of the KNN model to the value of k is shown for the HS-SAR Berlin dataset. The KNN is used for the classification of concatenated HSI and SAR embeddings from CoMMANet.

The HSI data already lie between 0 and 1. Each channel of SAR is scaled between 0 and 1. The weight for the SE term γ is set to 0.4 and the margin α is set to 1 for the best performance. For training the triplet network, the semihard triplets mixed with a few easy triplets are used. The SE term is also added to the loss function.

The CoMMANet is trained for 10 checkpoints with 50 epochs in each checkpoint. In each checkpoint, 280 000 triplets are used. The embeddings of each sensor are 32-dimensional vectors, and a *tanh* activation is applied on the latent space. The learning rate is set to 0.001, the batch size is set to 512, and the model is trained using the Adam optimizer. Due to fewer samples in the training dataset, the model is prone to overfitting. Therefore, a tenfold validation is used to avoid overfitting.

Classification: First, the CoMMANet embeddings of HSI and SAR are concatenated and classified using an ensemble of three neural networks and KNN. Using the neural network ensemble, the best overall classification accuracy is 71.26%. The best AA is 63.26%, which is achieved using KNN (with $k = 51$). Since all the previous methods reported their best accuracy, the best accuracy achieved is reported here instead of mean accuracy for the sake of comparison. The sensitivity of the KNN model to the value of k is shown in Fig. 12. The classification performance on the Berlin dataset is given in Table X. The classification map is shown in Fig. 13.

Second, a classifier is trained on one sensor's embeddings and tested on other sensor's embeddings. The results listed in Table XI show that the classification models developed for one sensor's embeddings can accurately predict the embeddings of other sensors also.

Missing sensor prediction/reconstruction: For the missing sensor prediction, the same model as the MUUFL data is used. The mean-squared error of the predictions is least when α is set to 1, and γ is set to 0.4 in the CoMMANet. All the training parameters are the same as the parameters used for the MUUFL data. The prediction metrics are given in Table XII.

After the embeddings of a sensor are predicted, the decoder is used to reconstruct the original data. The images

TABLE X
CLASSIFICATION PERFORMANCE (%) ON THE HS-SAR BERLIN DATA IS SHOWN HERE

No.	Class	HSI + SAR		
		Neural Network Best Accuracy	KNN ($k = 51$) Best Accuracy	Neural Network + KNN ($k = 51$) Best Accuracy
1	Forest	78.64	73.65	78.81
2	Residential Area	81.76	72.82	81.32
3	Industrial Area	35.60	36.96	35.96
4	Low Plants	76.48	74.30	77.05
5	Soil	63.71	69.57	65.96
6	Allotment	32.84	28.83	33.59
7	Commercial Area	18.85	19.58	18.67
8	Water	62.20	61.69	63.06
OA (%)		**71.26**	**63.26**	**71.10**
AA (%)		**60.50**	**64.19**	**61.15**
Kappa (%)		**57.20**	**50.03**	**57.31**

Since all the other methods reported their best accuracy, the best accuracy is shown here for comparison. The concatenated CoMMANet embeddings of HSI and SAR are classified using a neural network, KNN, and a KNN and neural network ensemble.

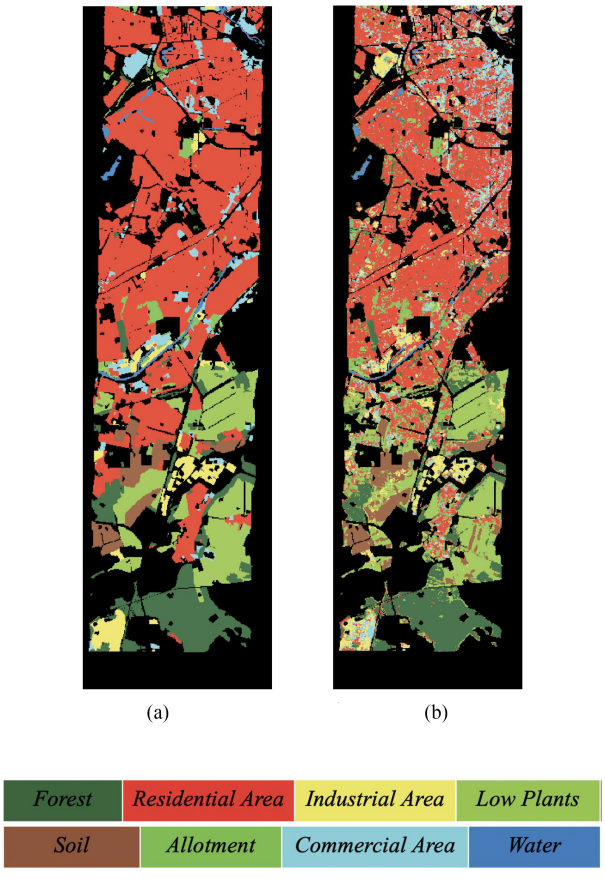


Fig. 13. Visualization of the classification maps from the HS-SAR Berlin data. (a) Ground truth map. (b) Classification map.

of original and predicted SAR and HSI bands are available here.[2] The data are reconstructed with a reasonably low mean-squared error. The SAR data have a smaller reconstruction error compared to the HSI reconstruction. In the testing data, the number of samples is significantly higher than the number of training samples. Therefore, the spectra of testing data have a significant amount of variation, which causes the accuracy of predicted spectra to drop. However, the spectra are successfully predicted with a low mean-squared error.

[2]See footnote 1

TABLE XI
HS-SAR BERLIN DATA: THE EFFECTIVENESS OF A UNIFIED CLASSIFICATION MODEL IS SHOWN HERE

Neural Network Classifier *trained* on embeddings of	Evaluation Metric (%)	Neural Network Classifier *tested* on embeddings of	
		HSI	SAR
HSI	OA	59.01	50.84
	AA	64.24	44.70
	Kappa	46.28	34.29
SAR	OA	59.82	50.78
	AA	64.44	43.96
	Kappa	46.85	33.98

The classifier is trained on embeddings of one sensor and tested on embeddings of another sensor. The best accuracy (%) is shown.

TABLE XII
HS-SAR BERLIN DATA: THE LATENT SPACE OF ONE SENSOR IS PREDICTED USING ANOTHER SENSOR, AND THEN RECONSTRUCTED USING THE SENSOR'S DECODER

Sensor (Predictor)	Sensor (Predicted)	Latent space MSE	Reconstructed Data MSE
HSI	SAR	0.92 ± 0.03	0.026 ± 0.005
SAR	HSI	0.95 ± 0.06	0.023 ± 0.011

* *Latent space values* $\in$ *[-1,1]* *HSI and SAR data* $\in$ *[0,1]*

C. Comparison and Analysis

To validate the effectiveness of the proposed model, a comparison is made with the state-of-the-art models such as CNN-PPF [34], FDSSCN [35], CNN-MRF [36], CRNN[37], IAP [38], Coupled CNN [18], IP-CNN [24], FusAtNet [39], SpectralFormer [40], FIT [41], FrIT [42], and SOTNet [43] using their reported results (see [24, Tab. XI]) and (see [42, Tab. II]). The comparison results are given in Table XIII. Using the MUUFL dataset, it shows 3.33%, 9.47%, 5.36%, 2.92%, 8.25%, 3.37%,

TABLE XIII
COMPARISON OF THE CLASSIFICATION ACCURACY (%) USING THE MUUFL DATA

No.	Class	CNN PPF	FDSSCN	CNN MRF	CRNN	IAP	Coupled CNN	IP CNN	FusAtNet	Spec Former	FIT	FrIT	SOT Net	CoMMANet Neural Network + KNN ($k = 35$)
1	Trees	89.07	87.37	93.04	91.43	85.32	98.90	94.40	98.10	81.47	90.72	89.61	96.50	97.0
2	Mostly Grass	85.71	32.37	60.17	63.16	81.99	78.60	92.26	71.66	87.54	78.88	81.43	88.23	87.3
3	Mixed ground surface	80.15	88.12	90.60	90.20	78.51	90.66	87.96	87.65	57.47	71.08	70.43	87.11	89.2
4	Dirt and sand	93.10	94.51	97.20	93.44	94.63	90.60	97.15	86.42	86.14	92.55	94.03	96.42	94.8
5	Road	88.98	97.84	92.00	87.62	86.81	96.90	94.38	95.09	89.13	83.68	87.99	94.39	96.3
6	Water	98.93	96.20	99.68	95.89	99.79	75.98	99.79	90.73	99.14	98.93	98.93	99.37	96.5
7	Building shadow	89.07	89.92	95.39	90.16	90.91	73.54	96.30	74.27	87.37	90.19	90.42	91.89	83.4
8	Building	92.15	87.44	94.71	89.29	95.46	96.66	96.13	97.55	90.61	87.13	94.57	94.84	98.0
9	Sidewalk	75.45	85.75	30.53	82.91	73.94	64.93	94.01	60.44	66.79	64.04	69.39	90.61	81.5
10	Yellow curb	100.00	72.73	36.36	96.97	98.91	19.47	100.00	9.39	95.08	97.27	73.77	100.00	89.9
11	Cloth panels	100.00	99.16	95.80	96.64	99.63	62.76	99.63	93.02	98.88	99.63	98.88	100.00	98.1
	OA (%)	90.97	84.83	88.94	91.38	86.05	90.93	93.86	91.48	81.21	85.46	86.61	93.87	**94.3**
	AA (%)	90.24	84.70	85.02	88.88	89.63	77.18	**95.64**	78.58	76.20	81.19	82.69	94.49	91.9
	Kappa (%)	84.46	80.24	85.55	84.41	82.12	88.22	91.99	88.65	85.42	86.74	86.31	91.84	**92.3**

The mean accuracy of the proposed model is compared with the other methods' mean accuracy.

0.44%, 2.82%, 13.09%, 8.84%, 7.69%, and 0.43% improvement in OA over CNN-PPF, FDSSCN, CNN-MRF, CRNN, IAP, Coupled CNN, IP-CNN, FusAtNet, SpectralFormer, FIT, FrIT, and SOTNet. The proposed CoMMANet outperforms the other methods in terms of the kappa coefficient and OA on the MUUFL Dataset. Moreover, applying a simple method, such as KNN, also gives a high AA and OA. The CoMMANet even outperforms vision transformer-based models [40], [42].

For many classes, the accuracy is lower compared to the other methods. The reason is that in our model, to extract features from each modality, we use standard convolutional layers instead of Gabor filters [16], [44] and complementary-structure control [24]. Therefore, other methods are able to extract rich spatial information and achieve higher accuracy on some ambiguous classes such as "sidewalk," "yellow curb," and "dirt/sand" from the MUUFL dataset. Gabor filters are known to extract rich texture information from an image. It can increase the robustness of learned representations while reducing the training complexity of the neural networks [45], [46], [47]. In the CoMMANet, increasing the number of CNN layers becomes very computationally expensive because of the triplet networks. Therefore, fewer convolutional layers are used in the model. The information fusion is performed during the classification by simply concatenating the embeddings.

Zhang et al. [24] used Gram matrices to maintain the complementary structure of their fusion block. They used the Gram matrix from LiDAR as a texture reference for the fused features. Similarly, a Gram matrix from HSI was used as a spectral reference for the fused features. The joint Gram matrices preserve the complementary information from both sensors. The concept is similar to Image Style transfer [25]. Several other methods [48], [49], [50] use different techniques to remove redundant information from the fused representation. But, in this article, our goal is to show the effectiveness of the proposed architecture without using any additional components/techniques. The proposed model shows superior results compared to all the other methods in terms of OA and kappa coefficient on the MUUFL Dataset.

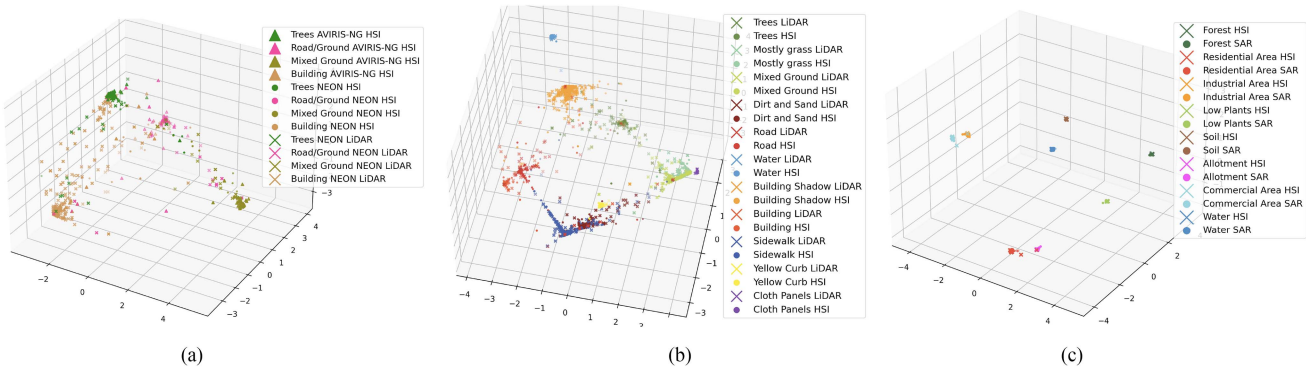
For the Berlin Dataset, the comparison is made with previous methods using (see [51, Tab. V]) and (see [52, Tab. VI]), and the results are given in Table XIV. The proposed model shows 6.73%, 4.55%, 4.71%, 2.75%, 1.41%, 4.95%, 5.02%, and 0.75% improvement in OA over CoSpace [28], LeMA [53], CapsNet [54], Co-CNN [18], CCR-Net [51], ContextCNN [55], DFINet [56], and AsyFFNet [52]. However, classes such as "Commercial Area," "Residential Area," and "Allotment" show a lower accuracy compared to the other state-of-the-art methods. Wu et al. [51] used a cross-channel reconstruction module (CCR) which makes the fusion process more efficient. The CCR module achieves effective information exchange between the two modalities and results in a more compact fusion at the feature level. If the LiDAR or SAR latent space is capable of reconstructing HSI latent space and vice versa, it indicates that the latent space of both the sensors is highly similar and carries enough information about the target class. Due to this factor, they achieve higher accuracy on some classes. Another problem with the Berlin Dataset is that the number of testing samples (461 851) is significantly higher than the number of training samples (2820) (see Table III). Therefore, the difference in training and testing data distributions causes the performance to drop, and the proposed model could not learn the representation of classes having fewer samples effectively. As a result, the AA gets slightly lower than the other methods. However, the proposed model still outperformed all the other methods in terms of OA on the Berlin Dataset.

Additionally, in all three datasets, the classification models trained on one sensor's embeddings are able to classify other sensor's embeddings accurately. But if a classifier is trained on embeddings of a sensor having a low discriminative ability, the classification accuracy on all sensors drops. However, it is successfully demonstrated that using the proposed CoMMANet, the embeddings from different sensors can be aligned, and a unified classification/analysis model is possible, which is sensor agnostic. It eliminates the need to develop separate classification models for every sensor.

TABLE XIV
COMPARISON OF THE CLASSIFICATION ACCURACY (%) USING THE HS-SAR BERLIN DATA

No.	Class	CoSpace	LeMA	CapsNet	CoCNN	CCRNet	Context CNN	DFINet	AsyFFNet	CoMMANet	
										Neural Network	KNN ($k = 51$)
1	Forest	85.09	85.11	84.96	84.09	85.93	77.22	80.29	76.65	78.64	73.65
2	Residential Area	61.60	64.84	65.22	68.48	68.07	63.69	61.93	70.76	81.76	72.82
3	Industrial Area	51.18	48.94	48.42	49.09	53.17	61.44	47.44	60.16	35.60	36.96
4	Low Plants	75.44	80.04	80.80	79.43	82.62	73.77	80.01	74.66	76.48	74.30
5	Soil	82.50	80.66	69.18	81.25	85.10	87.22	77.54	79.18	63.71	69.57
6	Allotment	54.66	54.07	55.08	50.68	63.02	82.88	73.42	79.24	32.84	29.83
7	Commercial Area	28.81	27.40	26.12	26.16	29.23	31.13	49.11	37.94	18.85	19.58
8	Water	60.78	57.75	59.69	59.52	68.78	74.24	77.59	83.90	62.20	61.69
OA (%)		64.53	66.71	66.55	68.51	69.85	66.31	66.24	70.51	**71.26**	63.26
AA (%)		62.51	62.35	61.18	62.34	66.99	68.95	68.42	**70.31**	60.50	64.19
Kappa (%)		50.93	53.12	52.77	54.76	57.16	54.03	53.98	**58.24**	57.20	50.03

For comparison, the best accuracy is shown since all the state-of-the-art methods reported their best accuracy.



(a) (b) (c)

Fig. 14. Visualization of shared embeddings from different datasets with the SE term. (a) AVIRIS-NG/NEON data embeddings. (b) MUUFL data embeddings. (c) Berlin data embeddings.

IV. ABLATION STUDIES

The proposed CoMMANet is affected by several components and hyperparameters. To investigate the effect of each component on the performance of the model, ablation studies are conducted by varying one component at a time.

A. *Effect of the Similarity Enhancement Term*

The similarity enhancement term is added to enhance the clustering process. However, there is a tradeoff between the SE weight parameter γ and the classification accuracy. If the value of γ is high, the ambiguity between the classes is not represented properly because the samples get too close to a particular class. Therefore, the classification accuracy decreases. For example, in MUUFL data, the two classes, "Dirt and Sand" and "Sidewalk," are slightly ambiguous. Therefore, a low value of γ should be chosen for classification tasks.

However, in a missing sensor scenario, it is quite the opposite. In this case, the value of γ should be higher. A higher value of γ will bring the latent spaces of both sensors close to each other and form tightly packed clusters. Now, it is easier to predict one sensor's latent space from another sensor's latent space, which makes the sensor translation process more accurate. In the MUUFL data, for the classification task, γ between 0 and 0.1 shows the best performance. For predicting a missing sensor's latent space, the optimal value of γ was 0.4. For the Berlin dataset, γ of 0.4 worked best for both classification and the missing sensor's latent space prediction. The embeddings with and without the SE term are shown in Figs. 14 and 15, respectively. The embeddings with the SE term appear to be more compact, which shows the improvement in clustering after incorporating the SE term in the loss function. In Fig. 16 also, it is shown that the models trained using the SE term perform significantly better than the models trained without the SE term.

B. *Effect of the Triplet Mining Strategy*

Experiments are conducted using semihard and hard triplets. The silhouette scores are computed for the comparison of performance. If the dataset contains fewer outliers, both semihard and hard triplet strategies show a similar performance. However, if the dataset contains a significant number of outliers, then semihard triplets seem to be better option because the hard triplets strategy is sensitive to the presence of outliers. Additionally, a few easy triplets can be added to the semihard triplets to make the training smoother. The effect of the triplet selection strategy is shown in Fig. 16.

In the case of AVIRIS-NG/NEON data, both the strategies show a similar performance irrespective of the sensor used for the triplet network. It is due to the fewer classes and less ambiguity among them. However, in MUUFL data, both strategies

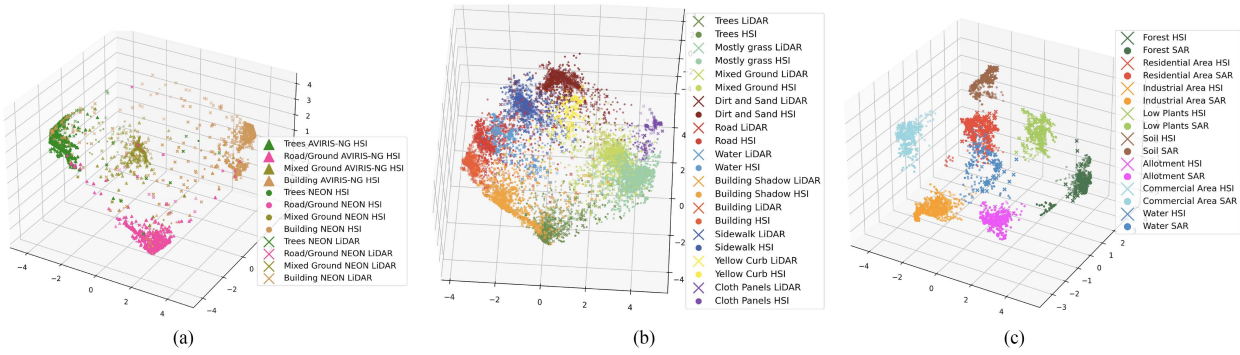


Fig. 15. Visualization of shared embeddings from different datasets without the SE) term. (a) AVIRIS-NG/NEON data embeddings. (b) MUUFL data embeddings. (c) Berlin data embeddings.

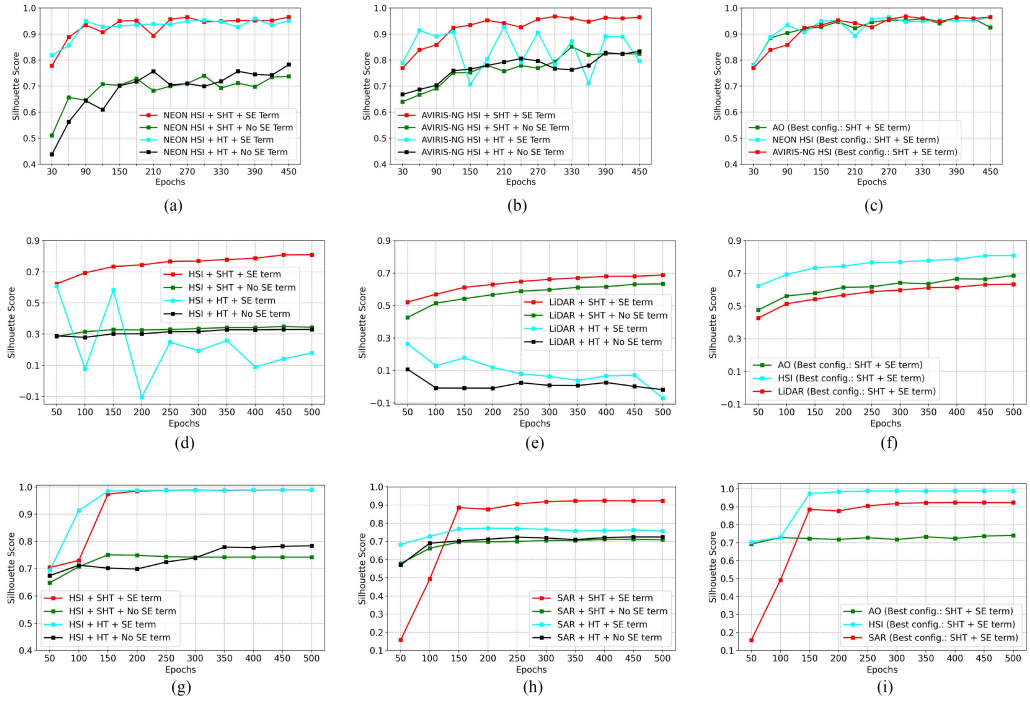


Fig. 16. Results of the Ablation studies conducted on the similarity enhancement term, triplet selection strategy, and sensor used for the triplet network are shown here. The acronyms are: HT for hard triplets, SHT for semihard triplets, SE for similarity enhancement, AO for alternating optimization, HSI for hyperspectral imagery, LiDAR for light detection and ranging, and SAR for synthetic aperture radar. (a) Comparison of models trained on the NEON HSI Data. (b) Comparison of models trained on the AVIRIS-NG HSI Data. (c) AVIRIS-NG/NEON Data: Effect of the Sensor used for the Triplet Network. (d) Models trained on the MUUFL HSI Data. (e) Models trained on the MUUFL LiDAR Data. (f) MUUFL Data: Effect of the Sensor used for the Triplet Network. (g) Models trained on the Berlin HSI Data. (h) Models trained on the Berlin SAR Data. (i) Berlin Data: Effect of the Sensor used for the Triplet Network.

show a similar result when the triplet network is applied on HSI data without using the SE term. But when the SE term is applied and HSI data are used for the triplet network, the results of the semihard triplet mining strategy are significantly better than the hard triplet mining strategy. When the LiDAR data are used for the triplet network, then the semihard triplet strategy also performs significantly better than the hard triplet strategy. For training the MUUFL data, 40% of the samples were used, and there is a significant variation in the distribution of some classes. On such data, using the semihard triplet mining strategy makes the training process more stable. Therefore, the semihard triplet strategy outperforms the hard triplet mining strategy in this case.

The Berlin data also show slightly better performance using semihard triplets when SAR data are used for the triplet network along with the SE term. When the SE term is not applied and the SAR data are used for the triplet network, both the strategies yield a similar result. Both the strategies show a similar performance when the HSI data are used for the triplet network because the Berlin Dataset contains very few samples ($<= 443$) from each class, and there are very few outliers.

C. *Effect of the Sensor Used for the Triplet Network*

The sensor chosen for the triplet network significantly affects the model's performance. If a sensor is more capable

of distinguishing between different classes compared to the other sensors, then using its data as input to the triplet network gives better results. If both the sensors are comparable, then the triplet network can be used on either of the sensors, and a similar performance is observed. A comparison is shown in Fig. 16. Using the triplet network on the better sensor in all the datasets yields better performance. However, the performance drops when a triplet model is applied on a sensor with a low resolution or a low discriminative ability.

D. Effect of Alternating Optimization

One obvious concern is how the performance is affected if a triplet network is used on both sensors. A triplet network can be used on both the sensors, but it is computationally expensive because now we have a large number of triplets from two sensors. However, to reduce the computations significantly, we propose an Alternating Optimization approach. As mentioned in Section II, the CoMMANet is trained in several checkpoints, and triplets are computed at the beginning of every checkpoint. In Alternating Optimization, the triplet network is used on one sensor in one checkpoint and the other sensor in the next checkpoint. The triplet network alternates between both the sensors in this way. This ensures that both the sensors' embeddings are clustered efficiently when the triplet network is applied. A comparison is shown in Fig. 16. The best training parameters configuration from each sensor is used for comparison.

In AVIRIS-NG/NEON data, using a triplet network on NEON HSI, NEON LiDAR, and Alternating Optimization shows almost the same performance. The reason is that both the sensors are hyperspectral sensors. The AVIRIS-NG HSI has a lower resolution, but it still captures more information than LiDAR. Therefore, both NEON HSI and AVIRIS-NG HSI are capable of distinguishing between different classes accurately.

In the case of MUUFL data also, the best results are obtained using the triplet model on HSI. When the LiDAR is chosen for the triplet network, the performance is worst because LiDAR alone cannot identify all land-cover classes. The alternating optimization model lies in the middle. In the case of Berlin data also, using the triplet network on HSI gives the best performance. The second-best model uses SAR on the triplet network. It is because HSI can identify the land-cover classes with more precision compared to SAR. The Alternating Optimization model shows the worst performance. The nature of the Alternating Optimization model is slightly unpredictable. However, the models trained using the better sensor always yield the best results, and Alternating Optimization does not improve the model's clustering performance.

V. Conclusion

In this article, a novel architecture, called CoMMANet, is proposed, which can map data from heterogeneous modalities onto a shared manifold in a discriminative manner. The proposed model can cluster the target classes from all the sensors effectively. Additionally, the proposed architecture allows missing sensor data reconstruction or sensor translation. The fused embeddings from all the sensors allow a robust and accurate classification. The discriminative ability of CoMMANet embeddings allows robust classification using even a simple method, such as KNN, also. However, the proposed CoMMANet is not limited to remote sensing applications. The CoMMANet is a generalized architecture that can be used on any kind of multimodal data, such as audio, video, text, RGB images, and ECG. Additionally, different features, such as Gabor filters, residual connections, dilated convolutions, and the Attention mechanism, can be incorporated into the CoMMANet architecture to improve the performance. Experimental results and comparison with the state-of-the-art multimodal classification methods indicate the effectiveness of the proposed CoMMANet. However, the current architecture still needs some improvements. First, the model requires many samples from each class to generalize on unseen samples. Second, the model needs to be equipped with a better fusion strategy. The classification can further be improved if the embeddings contain complementary information from multiple sources. Third, the sensor translation process needs to be improved to predict the missing sensor data with higher accuracy. Our future work will focus on overcoming these shortcomings and developing a more efficient architecture.

Acknowledgment

The authors would like to thank Geomatics Laboratory, Geography Department, and Humboldt-Universität zu Berlin for providing the simulated EnMAP hyperspectral data for the Berlin Urban area.

References

[1] M. Khodadadzadeh, J. Li, S. Prasad, and A. Plaza, "Fusion of hyperspectral and lidar remote sensing data using multiple feature learning," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 6, pp. 2971–2983, Jun. 2015.

[2] T. Matsuki, N. Yokoya, and A. Iwasaki, "Hyperspectral tree species classification of japanese complex mixed forest with the aid of lidar data," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 5, pp. 2177–2187, May 2015.

[3] A. Guha, "Mineral exploration using hyperspectral data," in *Hyperspectral Remote Sensing*. Amsterdam, The Netherlands: Elsevier, 2020, pp. 293–318.

[4] "Hunting new mining deposits with hyperspectral imaging possibility teledyne imaging," Jun. 2017. [Online]. Available: https://possibility.teledyneimaging.com/hunting-new-mining-deposits-hyperspectral-imaging/

[5] S. Peyghambari and Y. Zhang, "Hyperspectral remote sensing in lithological mapping, mineral exploration, and environmental geology: An updated review," *J. Appl. Remote Sens.*, vol. 15, no. 3, 2021, Art. no. 0 31501.

[6] R. Hänsch and O. Hellwich, "Fusion of multispectral lidar, hyperspectral, and RGB data for urban land cover classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 18, no. 2, pp. 366–370, Feb. 2021.

[7] M. Kolmann et al., "Hyperspectral data as a biodiversity screening tool can differentiate among diverse neotropical fishes," *Sci. Rep.*, vol. 11, no. 1, pp. 1–15, 2021.

[8] F. Chen, Z. Luo, Y. Xu, and D. Ke, "Complementary fusion of multi-features and multi-modalities in sentiment analysis," in *Proc. 3rd Workshop Affect. Content Anal.*, 2020, pp. 82–99.

[9] S. Yoon, S. Byun, and K. Jung, "Multimodal speech emotion recognition using audio and text," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2018, pp. 112–118.

[10] Y.-H. H. Tsai, S. Bai, P. P. Liang, J. Z. Kolter, L.-P. Morency, and R. Salakhutdinov, "Multimodal transformer for unaligned multimodal language sequences," in *Proc. Conf. Assoc. Comput. Linguistics Meeting*, 2019, Paper 6558.

[11] W. Liao, R. Bellens, A. Pižurica, S. Gautama, and W. Philips, "Combining feature fusion and decision fusion for classification of hyperspectral and lidar data," in *Proc. IEEE Geosci. Remote Sens. Symp.*, 2014, pp. 1241–1244.

[12] Y. R. Pandeya and J. Lee, "Deep learning-based late fusion of multimodal information for emotion classification of music video," *Multimedia Tools Appl.*, vol. 80, no. 2, pp. 2887–2905, 2021.
[13] W. Liao, A. Pižurica, R. Bellens, S. Gautama, and W. Philips, "Generalized graph-based fusion of hyperspectral and lidar data using morphological features," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 3, pp. 552–556, Mar. 2014.
[14] B. Rasti, P. Ghamisi, and R. Gloaguen, "Hyperspectral and lidar fusion using extinction profiles and total variation component analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 7, pp. 3997–4007, Jul. 2017.
[15] X. Zhao et al., "Joint classification of hyperspectral and lidar data using hierarchical random walk and deep CNN architecture," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 10, pp. 7355–7370, Oct. 2020.
[16] X. Zhao, R. Tao, W. Li, W. Philips, and W. Liao, "Fractional Gabor convolutional network for multisource remote sensing data classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2021, Art. no. 5503818.
[17] D. Hong et al., "More diverse means better: Multimodal deep learning meets remote-sensing imagery classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 5, pp. 4340–4354, May 2020.
[18] 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.
[19] C. Zhao, X. Gao, Y. Wang, and J. Li, "Efficient multiple-feature learning-based hyperspectral image classification with limited training samples," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 7, pp. 4052–4062, Jul. 2016.
[20] D. Hong, L. Gao, R. Hang, B. Zhang, and J. Chanussot, "Deep encoder-decoder networks for classification of hyperspectral and lidar data," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, 2022, Art. no. 5500205.
[21] M. Zhang, W. Li, Q. Du, L. Gao, and B. Zhang, "Feature extraction for classification of hyperspectral and lidar data using patch-to-patch CNN," *IEEE Trans. Cybern.*, vol. 50, no. 1, pp. 100–111, Jan. 2018.
[22] D. Hong, N. Yokoya, G.-S. Xia, J. Chanussot, and X. X. Zhu, "X-modalnet: A semi-supervised deep cross-modal network for classification of remote sensing data," *ISPRS J. Photogramm. Remote Sens.*, vol. 167, pp. 12–23, 2020.
[23] L. Zhang and L. Zhang, "Artificial intelligence for remote sensing data analysis: A review of challenges and opportunities," *IEEE Geosci. Remote Sens. Mag.*, vol. 10, no. 2, pp. 270–294, Jun. 2022.
[24] M. Zhang, W. Li, R. Tao, H. Li, and Q. Du, "Information fusion for classification of hyperspectral and lidar data using IP-CNN," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2021, Art. no. 5506812.
[25] S. Xu, J. Zhang, and J. Liu, "Image style transfer and content-style disentanglement," version: 1. [Online]. Available: http://arxiv.org/abs/ 2111.15624
[26] D. Guo, Y. Pei, K. Zheng, H. Yu, Y. Lu, and S. Wang, "Degraded image semantic segmentation with dense-gram networks," *IEEE Trans. Image Process.*, vol. 29, pp. 782–795, 2019.
[27] D. Hong, J. Hu, J. Yao, J. Chanussot, and X. X. Zhu, "Multimodal remote sensing benchmark datasets for land cover classification with a shared and specific feature learning model," *ISPRS J. Photogramm Remote Sens.*, vol. 178, pp. 68–80, 2021.
[28] D. Hong, N. Yokoya, J. Chanussot, and X. X. Zhu, "Cospace: Common subspace learning from hyperspectral-multispectral correspondences," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 4349–4359, Jul. 2019.
[29] A. Pournemat, P. Adibi, and J. Chanussot, "Semisupervised charting for spectral multimodal manifold learning and alignment," *Pattern Recognit.*, vol. 111, 2021, Art. no. 107645.
[30] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *Proc. Int. Workshop Similarity-Based Pattern Recognit.*, 2015, pp. 84–92.
[31] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 2, pp. 423–443, Feb. 2019.
[32] P. Gader, A. Zare, R. Close, J. Aitken, and G. Tuell, "MUUFL Gulfport hyperspectral and lidar airborne data set," Univ. Florida, Gainesville, FL, USA, Tech. Rep. REP-2013-570. 2013.
[33] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 815–823.
[34] W. Li, G. Wu, F. Zhang, and Q. Du, "Hyperspectral image classification using deep pixel-pair features," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 844–853, Feb. 2016.
[35] W. Wang, S. Dou, Z. Jiang, and L. Sun, "A fast dense spectral–spatial convolution network framework for hyperspectral images classification," *Remote Sens.*, vol. 10, no. 7, 2018, Art. no. 1068.
[36] X. Cao, F. Zhou, L. Xu, D. Meng, Z. Xu, and J. Paisley, "Hyperspectral image classification with Markov random fields and a convolutional neural network," *IEEE Trans. Image Process.*, vol. 27, no. 5, pp. 2354–2367, May 2018.
[37] H. Wu and S. Prasad, "Convolutional recurrent neural networks for hyperspectral data classification," *Remote Sens.*, vol. 9, no. 3, 2017, Art. no. 298.
[38] D. Hong, X. Wu, P. Ghamisi, J. Chanussot, N. Yokoya, and X. X. Zhu, "Invariant attribute profiles: A spatial-frequency joint feature extractor for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 6, pp. 3791–3808, Jun. 2020.
[39] S. Mohla, S. Pande, B. Banerjee, and S. Chaudhuri, "Fusatnet: Dual attention based spectrospatial multimodal fusion network for hyperspectral and lidar classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2020, pp. 92–93.
[40] D. Hong et al., "Spectralformer: Rethinking hyperspectral image classification with transformers," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2021, Art. no. 5518615.
[41] J. Lee-Thorp, J. Ainslie, I. Eckstein, and S. Ontanon, "FNet: Mixing tokens with Fourier transforms," version: 4. [Online]. Available: http://arxiv.org/abs/2105.03824
[42] X. Zhao et al., "Fractional fourier image transformer for multimodal remote sensing data classification," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, doi: 10.1109/TNNLS.2022.3189994.
[43] M. Zhang, W. Li, Y. Zhang, R. Tao, and Q. Du, "Hyperspectral and lidar data classification based on structural optimization transmission," *IEEE Trans. Cybern.*, early access, doi: 10.1109/TCYB.2022.3169773.
[44] Y. Chen, L. Zhu, P. Ghamisi, X. Jia, G. Li, and L. Tang, "Hyperspectral images classification with Gabor filtering and convolutional neural network," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 12, pp. 2355–2359, Dec. 2017.
[45] Z.-Q. Li, H.-M. Ma, and Z.-Y. Liu, "Road lane detection with Gabor filters," in *Proc. Int. Conf. Inf. Syst. Artif. Intell.*, 2016, pp. 436–440.
[46] M. Rai and P. Rivas, "A review of convolutional neural networks and Gabor filters in object recognition," in *Proc. Int. Conf. Comput. Sci. Comput. Intell.*, 2020, pp. 1560–1567.
[47] A. Alekseev and A. Bobe, "Gabornet: Gabor filters with learnable parameters in deep convolutional neural network," in *Proc. Int. Conf. Eng. Telecommun.*, 2019, pp. 1–4.
[48] F. Jahan, J. Zhou, M. Awrangjeb, and Y. Gao, "Inverse coefficient of variation feature and multilevel fusion technique for hyperspectral and lidar data classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 367–381, 2020.
[49] M. Brell, K. Segl, L. Guanter, and B. Bookhagen, "Hyperspectral and lidar intensity data fusion: A framework for the rigorous correction of illumination, anisotropic effects, and cross calibration," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 5, pp. 2799–2810, May 2017.
[50] Y. Gu and Q. Wang, "Discriminative graph-based fusion of HSI and lidar data for urban area classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 6, pp. 906–910, Jun. 2017.
[51] X. Wu, D. Hong, and J. Chanussot, "Convolutional neural networks for multimodal remote sensing data classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5517010.
[52] W. Li, Y. Gao, M. Zhang, R. Tao, and Q. Du, "Asymmetric feature fusion network for hyperspectral and SAR image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, doi: 10.1109/TNNLS.2022.3149394.
[53] D. Hong, N. Yokoya, N. Ge, J. Chanussot, and X. X. Zhu, "Learnable manifold alignment (LEMA): A semi-supervised cross-modality learning framework for land cover and land use classification," *ISPRS J. Photogramm. Remote Sens.*, vol. 147, pp. 193–205, 2019.
[54] H.-C. Li, W.-Y. Wang, L. Pan, W. Li, Q. Du, and R. Tao, "Robust capsule network based on maximum correntropy criterion for hyperspectral image classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 738–751, 2020.
[55] H. Lee and H. Kwon, "Going deeper with contextual CNN for hyperspectral image classification," *IEEE Trans. Image Process.*, vol. 26, no. 10, pp. 4843–4855, Oct. 2017.
[56] Y. Gao et al., "Hyperspectral and multispectral classification for coastal wetland using depthwise feature interaction network," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2021, Art. no. 5512615.

Aditya Dutt (Graduate Student Member, IEEE) received the M.S. degree in computer science in 2019 from the University of Florida, Gainesville, FL, USA, where he is currently working toward the Ph.D. degree in computer science with the Department of Computer and Information Science and Engineering.

His research interests include machine learning, metric learning, multimodal data fusion, speech analysis, and speech emotion recognition.

Alina Zare (Senior Member, IEEE) received the Ph.D. degree in computer and information science and engineering from the University of Florida, Gainesville, FL, USA, in 2008.

She teaches and conducts research in the area of machine learning and artificial intelligence as a Professor in the Electrical and Computer Engineering Department with the University of Florida. Her research has focused primarily on developing new machine learning algorithms to automatically understand and process data and imagery. Her research work includes automated plant root phenotyping, subpixel hyperspectral image analysis, target detection, and underwater scene understanding using synthetic aperture sonar, LIDAR data analysis, ground penetrating radar analysis, and buried landmine and explosive hazard detection.

Dr. Zare is currently an Associate Editor for the IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE.

Paul Gader (Fellow, IEEE) received the Ph.D. degree in mathematics for image-processing-related research from the University of Florida, Gainesville, FL, USA, in 1986.

He is a Professor with the Department of Computer and Information Science and Engineering and the Engineering School of Sustainable Infrastructure and Environment, University of Florida. He performed his first research in image processing in 1984, working on algorithms for detecting bridges in forward-looking infrared imagery as a Summer Student Fellow at Eglin Air Force Base. He has been a leading figure in handwriting recognition and landmine detection. He led the development of a fifth-ranked handwritten character recognizer and a top-ranked handwritten word recognizer in two National Institute of Standards and Technology competitions in the early 1990s. He has authored or coauthored more than 100 journals and more than 300 total papers and was an Associate Editor for *IEEE Geoscience and Remote Sensing Letters*. He has worked on a wide variety of theoretical and applied research problems, including fast computing with linear algebra, mathematical morphology, fuzzy sets, Bayesian methods, handwriting recognition, automatic target recognition, biomedical image analysis, landmine detection, human geography, hyperspectral and light detection, and ranging image analysis projects.