Performance Evaluation of Support Vector Machine and Stacked Autoencoder for Hyperspectral Image Analysis

Brahim Jabir, Bendaoud Nadif, Isabel De la Torre Díez, Helena Garay, and Irene Delgado Noya

***Abstract*—In the world of remote sensing, hyperspectral imaging has emerged as a powerful tool that captures incredibly detailed information about our environment. These images contain hundreds of spectral bands that reveal what the human eye cannot see, making them invaluable for applications ranging from precision agriculture to environmental monitoring. However, extracting insights from complex data requires sophisticated analytical approaches. Our research dives into the performance comparison of two popular machine learning approaches: the support vector machine (SVM) and the more recent deep learning-based stacked autoencoder (SAE). We wanted to understand which approach works better under different real-world conditions that researchers and practitioners face. Through extensive experiments across five diverse public hyperspectral datasets, we discovered that the choice between these models is not straightforward, it depends significantly on your specific circumstances. When labeled data are scarce, which is a common challenge in remote sensing, SVM proves more reliable and efficient. Conversely, when abundant training data are available, SAE demonstrates impressive capabilities in learning complex patterns. One interesting finding was how active learning as a technique that intelligently selects the most informative samples for labeling, improved SAE's performance on medium-sized datasets, potentially offering a practical solution to the data scarcity problem. The proposed approaches showed vulnerability to noise, highlighting the importance of preprocessing steps in real-world applications. Although SVM generally requires less computational resources, SAE's potential to handle large and complex datasets makes it an attractive option when the appropriate computing infrastructure is available. The model training also achieved high accuracy, compared to other models published in the literature. The results achieved provide a practical path for researchers and practitioners navigating the complex landscape of hyperspectral image analysis to help them choose the most suitable approach for their specific constraints and requirements.**

***Index Terms*—Active learning, deep learning, hyperspectral data classification, remote sensing, stacked autoencoder (SAE), support vector machine (SVM).**

I. Introduction

REMOTE sensing has become such a game-changer across tons of fields, giving people crucial insights for making smarter decisions. Just look at agriculture—farmers are using it to keep an eye on their fields, see how crops are doing, and implement cutting-edge farming techniques like precision irrigation and fertilizing [1]. With all this data now pouring in from drones, ground sensors, and image processing software, we seriously need better ways to analyze everything efficiently [2]. Hyperspectral imagery, with its super-detailed spectral information, is basically the backbone of most remote sensing applications. Classifying these images is not easy though—we are talking about extremely high-dimensional data with plenty of noise, which means we need some pretty sophisticated feature extraction and classification methods, this situation needs powerful machine learning (ML) models that can handle and analyze both the complexity and sheer volume of this data [3].

ML has completely transformed remote sensing, particularly with regard to pixel classification in multispectral and hyperspectral images (HSIs) [4]. Indeed, HSI classification consists of sorting pixels into distinct categories based on their spectral signatures and this task is essential for many applications such as environmental monitoring, agriculture, mining exploration, and military surveillance. Therefore, accurate classification enables not only detailed analysis but also informed decision-making that because of the rich spectral information contained in these images [5]. Among the many ML approaches, two models stand out and dominate this field: support vector machines (SVMs) and stacked autoencoders (SAEs). Although these models are fundamentally different in design, they have both demonstrated remarkable performance in processing hyperspectral data, for that, we have chosen to focus on these two approaches in our comparative study. Furthermore, a thorough comparison of these two models will allow us to better understand their respective strengths and weaknesses as well as the contexts in which one may prove more effective than the other. Although SVMs are no longer considered state-of-the-art for HSI classification on their

Received 8 April 2025; revised 15 May 2025; accepted 15 June 2025. Date of publication 17 June 2025; date of current version 14 July 2025. This work was supported by the European University of the Atlantic, Spain. *(Corresponding authors: Isabel De la Torre Díez; Brahim Jabir).*

Brahim Jabir is with ESIM Team, Polydisciplinary Faculty of Sidi Bennour, Chouaib Doukkali University, El Jadida 24000, Morocco (e-mail: ibra.jabir@gmail.com).

Bendaoud Nadif is with English Department, Sultan Moulay Slimane University, Beni Mellal 23000, Morocco (e-mail: nadif1bendaoud@gmail.com).

Isabel De la Torre Díez is with the Universidad Europea del Atlántico, Isabel Torres 21, Santander 39011, Spain (e-mail: isator@yllera.tel.uva.es, itordie@gmail.com).

Helena Garay is with the Universidad Europea del Atlántico, Isabel Torres 21, Santander 39011, Spain, and also with the Universidade Internacional do Cuanza, Cuito, Bié Province, Angola (e-mail: helena.garay@uneatlantico.es).

Irene Delgado Noya is with the Universidad Europea del Atlántico, Isabel Torres 21, Santander 39011, Spain, and also with the Universidad Internacional Iberoamericana, Campeche 24560, México (e-mail: Irene.delgado@uneatlantico.es).

Digital Object Identifier 10.1109/JSTARS.2025.3580654

own, they continue to serve as strong baselines in recent research. Their reliability, solid generalization capabilities, and relatively low computational demands make them enduring benchmarks, providing a stable foundation upon which more advanced techniques [6], [7]. SVM is still practical when labeled data is hard to come by and computing power is scarce. That said, SVMs do require careful parameter tuning, which can be time-consuming and risks overfiting [8]. Including them in this study gives us a clear, interpretable reference point to compare against more complex deep learning approaches in many real-world remote sensing scenarios and conditions, especially where resources are limited.

The SAE, which is considered a deep learning model, offers a powerful alternative by automatically learning hierarchical representations of data. SAE have shown they are effective at handling very large datasets and denoising data, making them robust to noisy inputs [9]. Despite all these advanced deep learning models that have emerged, like convolutional neural networks (CNNs), transformers, and spectral-spatial attention mechanisms in HSI classification, we chose the SAE as a representative of early deep learning models for this study. SAE hits a nice balance between keeping things simple architecture-wise while still being able to extract hierarchical features from complex hyperspectral data. It is less computationally demanding and has more predictable behavior, which makes it perfect for controlled experiments alongside traditional methods like SVM. Recent research still uses SAE or modified versions for stuff like denoising, feature extraction, and semisupervised learning in remote sensing projects [10], [11], [12].

Even with all the progress in ML for hyperspectral data analysis, we are still missing a thorough comparison between SVM and SAE models especially when working with data collected across diverse electromagnetic wavelengths and over diverse conditions. Our research aims to bridge that gap by conducting a comparative study of SVM and SAE models evaluating their performance across various scenarios like parameter settings, dataset sizes, and also active learning strategies while suggesting improved parameter selection methods to improve model performance. By comparing these traditional methods with contemporary methods like SAE we can illustrate the remarkable enhancement and increase in performance of contemporary deep learning algorithms compared to traditional algorithms. Although SVMs are no longer utilized standalone as classifiers nowadays, but as mentioned in the majority of research, SVMs are still important to observe how HSI classification techniques evolved over the years and by exploring these areas. Our study therefore makes important new contributions to HSI data analysis and to the application of ML in remote sensing, by presenting a comprehensive comparative evaluation of two popularly used spectral classifiers, SVM and SAE with various experimental conditions that will be encountered by practitioners in the HSIR application domain. We comprehensively evaluate their performance on several benchmark datasets based on their parameter sensitivity, time complexity, behavior under active learning, and resistance to noise. This reproducible benchmark is a blueprint for further work by offering a solid empirical foundation for the comparison of newer or more complex models. Our study offers several outstanding benefits like a comprehensive comparison. Unlike previous studies that focus on specific aspects, this research provides a holistic comparison of SVM and SAE models across various conditions, offering a thorough understanding of their performance characteristics. Also, parameter optimization, because the study delves into the impact of parameter settings on the performance of both models, providing insight into optimal configurations for different dataset sizes and conditions. In addition to the dataset size sensitivity by evaluating the models on both small and large datasets, the research highlights the scenarios where each model excels, offering guidance on model selection based on dataset size.

Active learning: The integration of active learning techniques showcases how these methods can enhance model performance, particularly for the SAE model with medium-sized datasets, highlighting the importance of iterative learning processes. The noise sensitivity analysis also has crucial benefits, understanding the models' sensitivity to noise is crucial for practical applications where data quality can vary, and this research provides valuable insights into how each model copes with noisy data. Finally, the computational efficiency aspect: the study compares the computational efficiency of SVM and SAE, providing practical insights into resource allocation and processing time for large-scale data analysis.

The rest of the article is organized as follows. Section II presents a review of existing research on hyperspectral data analysis using ML models, highlighting the gaps and justifying the need for this study. Section III offers a detailed description of the methodologies and tools used for implementing the experiments, including SVM and SAE models, active learning techniques, and datasets. Section IV presents and analyzes the experimental results, comparing the performance of SVM and SAE models under different conditions. It also interprets the results, discussing the implications, strengths, and limitations of the results. Finally, Section V summarizes the research, highlighting the main conclusions and suggesting directions for future work.

II. Related Works

The research literature seems to emphasize the integration of cutting-edge technologies, particularly ML techniques, to address various challenges or explore new opportunities within the field. The advancement of ML has allowed for the solving of various problems across multiple fields through the use of techniques such as deep learning and SVM. The recent advancements in HSI processing have introduced a variety of algorithms with enhanced classification accuracy and robustness [13]. Some of the prominent strategies for our research in this section are outlined later.

Class-aligned and class-balancing generative domain adaptation for HSI classification: It aims to solve the domain adaptation issue in HSI classification by aligning and balancing source and target class distributions. It applies generative models to create a more balanced feature space with improved classification performance on different datasets [14].

CFDRM: Coarse-to-fine dynamic refinement model: CFDRM introduces a dynamic refinement model that is coarse-to-fine in nature, enhancing moving vehicle detection in satellite videos.

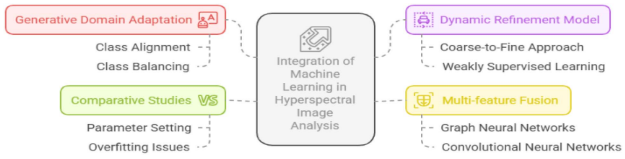

Fig. 1. Integration of ML in HSI analysis.

The weakly supervised approach utilizes minimal labeled data, and therefore it is particularly useful for applications with small annotated datasets [15].

Multifeature fusion: Graph neural network and CNN combining for HSI classification: This approach combines GNNs and CNN to take advantage of both spectral and spatial features in HSI data. The multifeature fusion enhances classification accuracy by capturing complex relationships within the data, and it provides an efficient way for HSI classification tasks [16].

A study by Omer et al. [17] compared deep learning and SVMs but not the two and unsupervised learning. Yang et al. [18] presented the significance of optimal parameter settings in obtaining successful results in machine-learning algorithms. The work conducted by Sankaran et al. [19] was centered on minimizing overfitting when applying unsupervised feature extraction, especially where deep learning models such as autoencoders were involved. The article introduces a novel regularization method known as L2,1-norm-based regularization with the objective of enhancing the learning capacity of autoencoders. The research conducted by Lary et al. [20] introduces the capability of ML in dealing with multivariate, nonlinear, and nonparametric regression or classification, pointing toward its ability to solve intricate geoscience and remote sensing issues. However, in-depth comparative research on parameter mapping-based deep learning and SVMs on regression and classification problems for hyperspectral remote sensing images has not been conducted, Fig. 1 provides the overview of the concept addressed in the literature review. Another comparison between traditional classifiers and CNNs for HSI classification showed that deep architectures generally outperform shallow models in terms of classification accuracy, especially when spatial information is incorporated [21], [22].

Though these studies have compared traditional and deep learning methods for HSI classification, few have provided comprehensive, side-by-side, especially across multiple datasets and performance dimensions such as parameter sensitivity, active learning behavior, noise robustness, and computational efficiency. Our work aims to fill this gap by jointly analyzing classification accuracy, parameter sensitivity, computational efficiency, active learning behavior, and robustness to noise across multiple benchmark datasets. This multidimensional evaluation complements prior comparative studies that typically focus on newer deep models or accuracy alone.

Table I includes a selection of literature that has yielded positive results in relation to this current study.

This highlights the advancement in remote sensing and applications of machine-learning techniques, i.e., deep learning and SVM, to address different problems in different fields. Although other research works have compared the models, they only look at certain features and lack a general comparison. For instance, earlier research works have examined the excellence and shortcomings of deep learning and SVM individually but not compared them in detail for both supervised and unsupervised scenarios.

III. Methods and Tools

This part describes the models and approaches that must be employed in order to perform the experiment. We work with Python as a programming language, TensorFlow, and Keras libraries to implement the proposed algorithms. We worked with a deep learning model based on SAE, and also with an SVM for learning and classification. To improve the performance of the models, we use some techniques such as active learning and Tensorboard for the visualization of statistics and logs [33]. The experiments were conducted in a controlled environment to ensure the consistency and reliability of the results. The hardware and software configurations used for the implementation and evaluation of the SVM and SAE models are as follows:

1) *Processor:* Intel Core i7-11700K CPU @ 3.60 GHz.
2) *RAM:* 32.
3) *GPU:* NVIDIA GeForce GTX 1660 Ti 6GB GDRDR6.

A. HSI Classification

HSI classification is a process that involves assigning each pixel in an HSI to a specific class based on its spectral properties. Unlike traditional RGB images, HSIs capture information across a wide spectrum of wavelengths, providing a detailed spectral signature for each pixel. This richness of data aids in more precise identification and differentiation of material [34]. According to Datta et al. [35], several major problems come up with HSI classification.

High dimensionality: HSIs may contain hundreds of spectral bands, resulting in a high-dimensional feature space. High dimensionality tends to necessitate dimensionality reduction methods to counteract the curse of dimensionality and enhance classification performance.

Noise: Hyperspectral data are likely to be exposed to various forms of noise, such as sensor noise, atmosphere noise, and other environmental noises. The right approach in noise handling is required to enhance accuracy in classification.

Feature extraction: Effective feature extraction methods are crucial to extracting the relevant spectral and spatial data from HSIs. Techniques such as principal component analysis (PCA), independent component analysis (ICA), and deep learning-based techniques are extensively used.

Classification algorithms: The proper selection of classification algorithms is pivotal in terms of high accuracy. SVMs and SAEs are two popular methods that have shown potential in HSI classification. SVM performs well in high-dimensional space, whereas SAE uses deep learning to learn intricate features.

TABLE I
APPLICATIONS OF DEEP LEARNING FOR A GROUP OF PROBLEMS

Ref.	Research problem	Proposed model	Description
[23]	Automatic speech recognition (ASR)	PSO-SVM hybrid training	*Particle swarm optimization (PSO):* Support vector machine (SVM) hybrid training: This approach combines PSO, a social behavior-based optimization algorithm with birds, and SVM, a high-performance classification algorithm, to enhance the training process of ASR systems, with improved accuracy and performance.
[24]	Recognize image	Gaussian binary classification	*Gaussian binary classification:* It uses Gaussian distributions to classify data for binary classification tasks. It performs well in image classification tasks where data can be best approximated using Gaussian distributions.
[25]	System for detecting wheezes in respiratory sounds	SVM classifier and Artix-7 XC7A100T FPGA (Xilinx)	*SVM classifier and Artix-7 XC7A100T FPGA (Xilinx):* It uses an SVM classifier to detect wheezing in respiratory sounds and implements it on an Artix-7 FPGA to provide a hardware-accelerated solution to enhance the processing efficiency and speed.
[26]	Neural seizure detection system	SVM and Xilinx Spartan-6 FPGA	*SVM and Xilinx Spartan-6 FPGA:* An SVM-based seizure detection neural system for classification and deployed on a Xilinx Spartan-6 FPGA, utilizing the real-time data processing and low-latency feature of the FPGA.
[27]	Speaker recognition system	SVM with Modified Sequential Minimal Optimization (MSMO) algorithm	SVM with MSMO algorithm: An SVM optimized using the MSMO algorithm, a novel algorithm that enhances training with the ability to effectively solve SVM's optimization problem.
[28]	Facial expression recognition	SVM multiclass: systolic array architecture	*SVM multiclass: systolic array architecture:* This method applies SVM toward multiclass facial expression recognition, optimized with a systolic array architecture for improved computational speed and efficiency.
[4]	Spectral–Spatial Classification of Hyperspectral Imagery	deep brief network (DBN)	*Deep belief network (DBN):* DBN represents a generative graphical model consisting of multiple layers of hidden, stochastic variables. It is utilized for spectral–spatial classification of hyperspectral images, retrieving complicated data patterns for efficient classification.
[29]	La reconnaissance des fruits.	EfficientNet	*EfficientNet:* A family of convolutional neural networks (CNN) that provides a trade-off between efficiency and accuracy. EfficientNet modifies the depth, width, as well as resolution of the network, making it significantly effective in fruit recognition tasks.
[30]	automatic target recognition (SAR-ATR)	Deep convolutional networks (ConvNets)	*Deep convolutional networks (ConvNets):* ConvNets are specialized neural networks for processing grid-structured data such as images. They are applied for synthetic aperture radar (SAR) image automatic target recognition utilizing their feature learning capability in spatial hierarchies of features.
[31]	Predict soil moisture	CNN and SVM	*Convolutional neural network (CNN) and SVM:* It includes the application of CNN feature extraction capability and SVM classification capability in accurately estimating soil moisture.
[32]	Classification and detection of insects in field crops	ANN, SVM, KNN, and NB	Artificial Neural Network (ANN), SVM, K-Nearest Neighbors (KNN), and Naive Bayes (NB): An approach with a multimodel including various machine-learning algorithms (ANN, SVM, KNN, and NB) for classification and identification of field crop pests using efficient and proper analysis.

B. SVM Algorithm for HSI Classification

SVM is an ML algorithm that can be employed to classify and regression problems. Its primary objective is to find the optimal boundary, or "hyperplane," that optimally separates different classes in the training data. This is the margin that is chosen to be maximized so that the distance from the hyperplane to the nearest data points of every class, known as "support vectors," is maximized. This maximization makes SVMs have improved generalization and stability in classification or regression tasks despite having complex or high-dimensional data. Therefore, SVMs are applied widely across various fields since they can find the best decision boundary under the distribution of data points [36]. The basic idea of SVM involves mapping input data points into a higher-dimensional space where it becomes easier to find a linear boundary that separates different classes or groups of data points. This is achieved through the use of a kernel function that implicitly maps the data into this higher-dimensional space. The objective of an SVM is to find the optimal hyperplane represented by the equation $(w, x) = w.x + b$, where w is a weight vector $x \in R^m$ is the input feature vector, and b is the bias term. This hyperplane serves as a decision boundary that separates the data points of two classes in the given dataset (Function 1).

$$\min \frac{1}{p} w^t w + c \sum_{i=1}^{p} \max\left(0, 1 - y_i' \left(w^T x_i + b\right)\right) \quad (1)$$

where

w^{T} and w called L1 norm or "Manhattan norm."

C: a random value called the penalty parameter; this value is selected using hyperparameter optimization.

y: true label, and $w^T x + b$ i is the predictive function.

The function (2) is represented as L1-SVM, with the standard hinge loss. Its differentiable counterpart, L2-SVM (Function 3) provides more stable results.

$$\min \frac{1}{p} \|w\|_2^2 + c \sum_{i=1}^{p} \max\left(0, 1 - y_i' \left(w^T x_i + b\right)\right)^2 \quad (2)$$

where $\|w\|_2$ is the Euclidean norm (also called L2 norm), with the squared hinge loss.

Given the training data and their corresponding labels (x_n, t_n), $n = 1, \ldots, N, x_n \in \mathbb{R}^D, \mathrm{t}_n \in \{-1, 1\}$, to predict the class label of a test data x SVM uses the formula (3), this formula gives if a new example x belongs to class -1 or class +1.

$$\arg_t \max \left(W^{\mathrm{T}} \mathrm{x}\right) t. \quad (3)$$

As the prediction formula shows, the class prediction is not the same as Softmax, because basic SVM can only predict binary problems. However, there are approaches to make SVM multiclass, to predict multiple objects. The simplest way is to use the so-called one-vs-rest approach. For k-class problems. This approach uses a change of variable of the function $\boldsymbol{W}$ by $\boldsymbol{a}$, then

the output of the SVM is designated as follows (Function 4).

$$a_k\ (x) = W^T\ x. \tag{4}$$

The function a_k presents the different classification functions of the SVM associated with the classes (i.e., uses **k** decision functions) instead of just one function, this function determines the class of each input *x*. After this operation, the predicted class would be (Function 5)

$$\arg_k \max a_k\ (x)\ . \tag{5}$$

It should be noted here that prediction using SVM becomes multiclass, i.e., it offers the possibility of detecting multiple classes, it is no longer a binary problem.

Specific to the problem of HSI classification, SVMs have several important technical advantages. Hyperspectral data present certain special difficulties: high dimensionality (often hundreds of bands), limited number of labeled samples, and high correlation between adjacent bands. The success of these data of SVMs relies on several modified principles [37], [6].

First, unlike parametric classifiers that make assumptions regarding the data distribution, SVMs rely on the support vectors alone and therefore are less affected by multimodal and non-Gaussian distributions of spectral signatures in data. For hyperspectral data, the RBF (radial basis function) kernel

$$K\ (x_i, x_j) =\ \exp(-\gamma \|x_i - x_j\|^2. \tag{6}$$

Equation (6) works best as it encapsulates the high-dimensional nonlinear interdependence between spectral bands without directly calculating the transformation. The "kernel trick" allows the SVM to maintain manageable computational complexity even when faced with high dimensionality [38].

Optimization of the regularization parameter C in the hyperspectral setting is a must: a high one leads to overfitting of small spectral variations (noise or atmospheric variation), while a low one ignores large spectral variations among similar substances. Similarly, the γ parameter for the RBF kernel must be optimized to capture the scale of the useful spectral variations. In practice, in the hyperspectral case, cross-validation with a logarithmic range of values (C, γ) is standard practice, and in most typical datasets such as Pavia and Salinas, optimal values tend to be from C $\in$ [10, 10 000] and $\gamma \in [10^{-3}, 10^{-1}]$. To address the multiclass problem prevalent in remote sensing applications (separating numerous land cover classes), One-Against-One (OAO) and One-Against-All methods are broadly used. For hyperspectral data in particular, the OAO method is favored as it constructs more decision boundaries, allowing more subtle discrimination between spectrally similar classes. The method does entail training k(k-1)/2 binary classifiers for k classes, but is computationally affordable considering the SVM's high-dimensional data efficiency in training [39].

One of the key benefits of SVM in remote sensing is its ability to learn well from a few labeled samples, which is a common case in land cover mapping where the terrain data acquisition process may be expensive. Experience has revealed that SVMs are capable of yielding good classification accuracy from as few as 10–30 samples per class, unlike most other classifiers requiring large sets for training.

C. SAE Fr HSI Classification

As mentioned earlier, one of the main challenges with HSIs is the curse of dimensionality. Each pixel can contain hundreds of spectral bands, making traditional algorithms susceptible to overfitting, especially when the number of annotated samples is not enough. SAEs address this issue by automatically learning compact latent representations capable of capturing the most important factors of spectral variation while ignoring redundancies [12], [40]. In the context of HSI classification, SAEs offer several key advantages.

Nonlinear dimensionality reduction [41]: Unlike linear methods such as PCA, SAEs can capture complex and nonlinear structures present in spectral data, which improves separation between classes.

Hierarchical learning [21]: By stacking multiple autoencoders (SAEs), the network learns increasingly abstract representations, enabling better discrimination between materials with similar spectral signatures.

Robustness to noise [42]: SAE training can include noise (via denoising autoencoders), making them particularly robust to disturbances often present in hyperspectral data acquired under real-world conditions.

Adaptability [43]: Thanks to their architectural flexibility, SAEs can be adapted to different dataset sizes, provided there are sufficient annotated examples to avoid overfitting.

Consequently, in this study, we chose the SAE as a representative of early deep architectures to analyze its performance compared to a classic classifier (SVM) in various scenarios: dataset sizes, active learning, and presence of noise.

An Autoencoder is a unique type of artificial neural network where the output is identical to the input. Through training, the Autoencoder modifies its parameters such that the input samples are transformed into a compressed representation, and then decoded to closely approximate the original features. Equation (7) likely represents the mathematical expression for mapping the input data *x* to the hidden layer h within the Sparse Autoencoder framework. SAE, it constitutes of (encoder, decoder, and hidden layer).

$$h\ = f\ (x) = S_f\ (W + b)\ . \tag{7}$$

$S_f$is the activation function used in the encoder is often a nonlinear function such as the sigmoid function (8)

$$sigmoid\ (z) = \frac{1}{1 + z^{-1}}\ . \tag{8}$$

After the data is encoded into the hidden layer, the decoder function g(h) reconstructs the original data. The output of the decoder is denoted as y (function 9), and it aims to closely approximate the input data (function 8)

$$y\ = g\ (h) = S_g\ (W'h + b_y)\ . \tag{9}$$

S_g is the activation function of the decoder being typically a sigmoid function, allowing the decoder to transform the hidden representation back into the original data space.

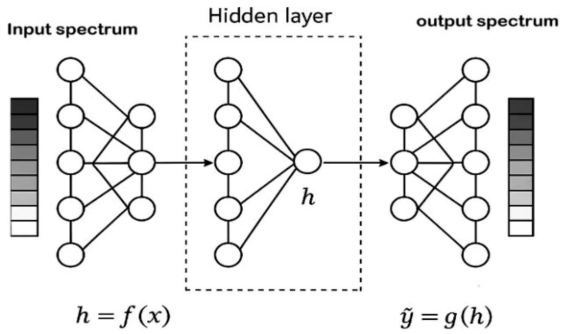

Fig. 2. SAE for HSI classification.

Training objective: The training process of the autoencoder involves optimizing the parameters $\theta = \{W, b_y, b_h\}$ to minimize the reconstruction error on the sample set **D**. The reconstruction error is calculated using a loss function $L(x, g(f(x)))$, which measures the difference between the original input x and the reconstructed output $g(f(x))$.

Loss function: The reconstruction error of the autoencoder is denoted as J_{AE} (function 10)

$$J_{AE} = \sum_{x \epsilon D} L\left(x, g\left(f\left(x\right)\right)\right). \tag{10}$$

The reconstruction error function L in an autoencoder can be expressed using different loss functions depending on the nature of the data and the task at hand. Here are two commonly used loss functions (11) and (12)

$$L\ (x, y) = \|x - y\|^2 \tag{11}$$

$$L\ (x, y) = \sum_{i=1}^{d_x} x_i \log y_i + (1 - x_i) \log\left(1 - y_i\right). \tag{12}$$

The following Fig. 2 illustrates the functioning of a SAE for HSI classification. The encoder transforms the input spectrum into a compressed representation according to the formula h = f(x). The connections between the numerous spectral bands of the input and the intermediate layer can be observed. The hidden layer represents the compressed latent space where the number of neurons is reduced, forcing the network to learn a sparse representation of the data. This layer is denoted by h. The decoder reconstructs the output spectrum from the compressed representation according to the formula $\tilde{y} = g(h)$.

To further place our model selection and underscore their relevance to HSI analysis, we illustrate in Table II how SVM and SAE key features complement the intrinsic properties of hyperspectral data. This complementarity permits us to explain why these models were chosen for examination and how their design activities address key issues such as high dimensionality, noise, few labeled samples, and spectral correlation.

TABLE II
ALIGNMENT OF SVM AND SAE CHARACTERISTICS WITH HSI PROPERTIES

Hyperspectral data property	SVM	SAE
High dimensionality	Handles high-dimensional spaces well using kernel functions (e.g., RBF)	Learns low-dimensional latent representations through unsupervised hierarchical encoding
Interband correlation	Captures complex nonlinear relationships via kernel mapping	Learns hierarchical features that model spectral band dependencies
Limited Labeled Samples	Performs well in small sample sizes due to support vector optimization	Can leverage unlabeled data during pretraining to enhance generalization
Noise sensitivity	Sensitive to noise; mitigated by tuning soft-margin and kernel parameters	Can be trained as denoising autoencoders to improve robustness against spectral noise
Computational complexity	Relatively low for small-to-medium datasets; scalable with efficient implementations	Higher computational cost due to deep architecture and iterative training

D. Active Learning Method

Active learning is a category of ML where the algorithm actively selects data points to label from an unlabeled dataset instead of having a completely random process. The underlying assumption is that the algorithm would select informative or uncertain data points for labeling with the goal of achieving maximum learning efficiency and model performance. After the data is annotated, the model is retrained or trained incrementally on the new batch of labeled data, and so the cycle goes until the level of desired accuracy is achieved. The active learning's primitive assumption is that with a careful choice of points to learn from, the algorithm can be just as good or even better on a smaller set of labeled examples than passive learning techniques. In contrast, passive learning employs an indiscriminate selection of data points from the dataset for labeling, regardless of their informativeness or uncertainty. Active learning methods are especially useful in cases where there is not enough labeled data or where it is too expensive, as they allow the model to make the most out of sparse labeling resources by focusing on the most valuable data points for training [44]. Active learning exploits explicit criteria, i.e., entropy, to assist in selecting data points for labeling. This is not the same as passive learning, which is utilized in training data selection methods (SVM and SAE). Passive learning can result in poor models and requires a large amount of data, resulting in increased memory usage and training time [45]. Active learning addresses these problems as it has the ability to do better with less data and in less time. Moreover, the cost of labeling data is so significant that one must use less labeled data to achieve successful models and hence active learning is a well-worth technique. The novelty of this study in comparing the use of SVM and SAE in active learning, specifically with the use of the entropy heuristic known as the entropy query by bagging, as follows:

$$\hat{x} = \arg\max_{x_i \epsilon U} \left\{ -\frac{1}{\log\left(N\right)} \sum_{\omega=1}^{N} p(y_i^* = \omega | x_i) \log p(y_i^* = \omega | x_i) \right\} \tag{13}$$

where

$\boldsymbol{\omega}$: The label assigned to each class.

SVM/SAE receives the data x_i to predict y_i^*.

$p(\ y_i^* = \omega | x_i)$: The probability for the given input data (x_i) belonging to class ω, as predicted by the classifier.

This equation shows that we can use data annotation to increase the performance of the classifier. To this, we allocate two (14) and (15), the first ($\boldsymbol{X}$) for the labeled data and the second ($\boldsymbol{U}$) for the unlabeled data.

$$X = \{x_i, y_i\}_{i=1}^{n} \tag{14}$$

$$U = \{x_i\}_{i=n+1}^{m}. \tag{15}$$

The entropy query by bagging technique selects the most informative samples for labeling by evaluating prediction uncertainty across an ensemble of models. This section elaborates on the mathematical formulation (13) and provides detailed implementation steps for reproducibility. The method proceeds as follows:

Ensemble construction: We construct an ensemble of N = 10 models (either SVMs or SAEs) each trained on a bootstrap sample drawn with replacement from the current labeled dataset X. For SVMs, all ensemble members share the same kernel parameters (C and γ), but differ in training subsets. For SAEs, we use the same network architecture while introducing slight learning rate variations (±10%) to promote diversity.

Entropy-based uncertainty estimation: For each unlabeled instance xj∈U, the prediction probability distributions from all N models are averaged to obtain a consensus vector.

Sample selection and labeling: At each iteration, the top q = 200 samples with the highest entropy values are selected, labeled, and added to the training set. These samples are removed from U, and all ensemble models are retrained on the updated labeled set X.

For SVMs, we used LIBSVM with Platt scaling (-b 1) to generate probability outputs. To ensure well-calibrated estimates, we applied 5-fold internal cross-validation in each ensemble member. And For SAEs Probabilities are directly extracted from the softmax output layer. To approximate Bayesian uncertainty, we applied Monte Carlo dropout (dropout rate = 0.3, with 20 stochastic forward passes per sample) during inference.

The entropy computations are vectorized using NumPy, enabling efficient processing of batches of 1000 samples at a time to manage memory usage on large datasets. The active learning loop terminates after 20 iterations or earlier if accuracy improvement drops below 0.5% for three consecutive rounds.

The entire framework is implemented in Python 3.8, using TensorFlow 2.4 for SAE models and LIBSVM 3.25 for SVMs. Multiprocessing is used to parallelize the ensemble predictions, which significantly accelerates the process, especially on larger datasets such as Salinas and Pavia University.

Algorithm: Implementation of entropy-based active learning for HSI classification.

```
Input: X = Labeled dataset
U = Unlabeled dataset
N = Number of ensemble models (N = 10)
q = Query batch size (q = 200)
T = Max iterations (T = 20)
t = 0
while t < T and accuracy_improvement > 0.5%:
  B = [train_model(sample_bootstrap(X)) for _ in
   range(N)]
  entropy_scores = []
  for x in U:
    probs = [model.predict_proba(x) for model in B]
    avg_probs = np.mean(probs, axis = 0)
    entropy = -np.sum(avg_probs * np.log(avg_probs +
     1e-10)) / np.log(C)
    entropy_scores.append((x, entropy))
  Q = select_top_q(entropy_scores)
  labels = oracle_label(Q)
  X + = labeled(Q, labels)
  U - = Q
  t + = 1
```

E. Dataset and Models Configuration

This research focuses on training models using multispectral data, with a particular emphasis on the Pavia University dataset, a public hyperspectral dataset containing remote sensing data. This dataset comprises 38 400 labeled pixels divided into 7 classes, with images having a resolution of 1096 × 715 pixels. Out of these pixels, 38 000 are allocated for training, divided into two sets: a training set (denoted as X) and a candidate set (denoted as U). To maintain accuracy and prevent bias, three classes with over 40 000 labeled pixels are excluded from the analysis to avoid potential disproportionate influence on other classes. The dataset is then split into train and test groups, with 2940 images assigned to the train group and 35 460 images to the test group. Within each class, 420 images are randomly chosen as training samples, with the remaining images reserved for testing, ensuring a systematic and unbiased approach to model training and evaluation.

The second dataset used in this experiment is also the Pavia University dataset, which was collected using the Reflective Optics System Imaging Spectrometer, an advanced sensor designed for detecting spectral fine structures in coastal waters. The images in this dataset have a resolution of 610 × 340 pixels and contain 150 bands collected at a close distance between 041 and 0.91 μm relative to the electromagnetic spectrum. The resolution = 14 m per pixel, providing high resolution and avoiding mixed pixels. The dataset has been preprocessed to remove some bands due to noise, leaving around 100 channels suitable for classification. The classes in this dataset are 9, with a total of 25 000 pixels. However, only 3200 pixels were used for training the set X, providing 400 pixels for each class, and the remaining 21 800 pixels were used for the candidate set U. Some classes were also eliminated as they contain more than 10 000 labeled pixels. The third dataset called "KSC Data Sets" contains an image with 512 × 614 pixels spread over 13 classes of different land cover, collected by NASA AVIRIS in 224 bands with a width of 400–2500 nm in the infrared spectrum and 10 nm in the reflected visible at a spatial resolution of 16 m… One-third of the samples are randomly selected for training, and the rest (two-thirds) for testing data.

The fourth dataset used in this study is the Indiana dataset, which was collected using AVIRIS sensors over an Indiana site.

It contains images with a resolution of 145 × 145 pixels and 224 spectral reflectance bands in the wavelength range of 04–25 × 10^{-6} m. The dataset includes sixteen classes, with two-thirds of the samples randomly selected as test samples and the remaining data used for training.

The last dataset used is the Salinas dataset, which includes 43 000 samples divided into 15 classes. Each image has a resolution of 512 × 217 pixels, and the dataset is divided into a training split with 7800 samples and a test split with 35 200 samples, which are randomly selected during the training and testing phase. Four classes were eliminated due to their larger data size than the others, to avoid any issues with the accuracy of other classes.

We trained models (SVM and SAE) on these datasets to compare them with an extension that uses sample selection methods such as maximum uncertainty (MU) and random selection (RS). Since the data in some classes is limited, while others have more data, we used a sample selection method. It is important to note that the models used require parameter settings, making these two steps (sample selection and parameter settings) crucial in the experiment. For the sake of achieving perfect accuracy, it is desirable that there is a great understanding of the model and its parameterizations, especially for the SAE, which is parameter-sensitive at this stage. The parameters required by the SAE are batch size and hidden units.

For each layer, the noise ratio, activation function, learning rate, and iterations. In the case of supervised learning, the learning rate will dictate how quickly the model weights are updated based on the loss gradient during training. If a high learning rate is used, the model will overshoot the optimal weights and diverge or converge. When a low learning rate is used, training will be sluggish. It is generally best to start with a relatively high learning rate and decrease it as training progresses.

In unsupervised learning, the learning rate plays a crucial role in the adaptive adjustment of the model weights in relation to the input data. Compared to supervised learning where the updates to the model are guided by labeled training data, unsupervised learning is done in the absence of direct labels and the aim is to find underlying patterns or structures in the data autonomously. Thus, in the case of unsupervised learning, learning is typically made much smaller compared to supervised learning situations. With the reduced learning rate, the model can update its weights at a slower pace, facilitating slower exploration of the data landscape and preventing sudden wild fluctuations or divergence during learning. When a lower learning rate is employed, unsupervised learning algorithms are able to learn from the data's inherent complexity and converge to useful representations or clusters without being flooded by noise or irrelevant structures.

We configured our model to consist of three layers of 100–200 neurons and altered the momentum from 0 to 1. The learning rate for the unsupervised training ranged between 0,000005 to 01 with the same rate for all the hidden layers, while for the supervised training it was between 0,0005 to 01. For SVM, we used Gaussian kernels with soft-margin parameter (c) and kernel size (σ, denoted by g) ranging from 01 to 1 00 000 and 10^{-7} to 1, respectively. These were all tried and executed using the LIBSVM library, which is well known for simplicity and rapid execution in SVM problems [46]. Fig. 3 gives a rough idea about the datasets and models used.

Fig. 3. Overview of the dataset and the model.

IV. Results and Discussion

In the experiment, we utilized various models and trained them on the selected hyperspectral datasets. The configuration of the models was tailored to each dataset such that the unique characteristics and needs of each dataset were considered. Experimental results of wide coverage for accuracy and parameters of each model and dataset are presented in the following. We varied the parameters systematically to see how they affected the performance of the models, allowing us to infer the best configurations for each data set. The provided tables and figures give an exhaustive overview of the measured performance metrics during the experiments, enabling a good comparison and analysis of how various models and parameter settings affect the outcomes. They assist in better comprehension of the performance of different modeling approaches for HSI analysis.

A. Result

Table III shows the result of the first SVM-RBF model. It shows that accuracy changes with a change of parameters. For dataset Salinas, the highest accuracy was 99.60% when c and g were 1 00 000 and 001, respectively. Similarly, for dataset Pavia, the highest accuracy was 96.20% when c and g were 100 and 1, respectively. On the KSC dataset, it was 93.60% when c and g were 1 00 000 and 1. For the PaviaU dataset, the highest was 92.39% when c and g were set to 1000 and 1. Lastly, on the Indiana dataset, the highest accuracy reached was 91.39% when c and g were set to 1000 and 1.

Table IV shows the results of training using the SAE model. The results reflect the same pattern as the accuracy of training and changes according to the modifications of parameters regarding the number of hidden layers, Nu = (005, 009, 01), respectively, and the learning rates for unsupervised and supervised are 10. The number of iterations was fixed at 1000 epochs.

The results for the Salinas dataset show that the best accuracy was 94.48% when Nu = (200, 200, 200). For the Pavia dataset, the highest accuracy was 91.13% when Nu = (200, 200, 200). The PaviaU dataset had a test accuracy of 87.34% when the parameters of hidden layers were set to Nu = (200, 100, 100). The Indiana dataset had the best accuracy of 86.59% when Nu = (200, 200, 200). Lastly, for the KSC dataset, when Nu = (200, 200, 100), the accuracy was 64.13%.

Table V the impact of various parameters, including the number of iterations, learning rate, and units in the hidden layer,

TABLE III
RESULTS OF THE TRAINED SVM-RBF WITH PARAMETERS APPLIED FOR EACH DATASET

c g	10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1
Salinas dataset								
0,1	15,77%	15,77%	15,77%	15,77%	8,53%	48,59%	72,24%	93,97%
1	15,77%	15,77%	15,77%	8,53%	48,53%	72,18%	93,95%	96,45%
10	15,77%	15,77%	8,53%	48,53%	72,21%	93,90%	96,48%	98,20%
100	15,77%	8,53%	48,52%	72,21%	93,89%	96,47%	98,21%	99,17%
1000	8,53%	48,53%	72,23%	93,90%	96,46%	98,11%	99,15%	99,55%
10 000	48,91%	72,50%	93,89%	96,46%	98,09%	99,19%	99,56%	99,59%
1 00 000	73,45%	93,79%	96,46%	98,06%	99,12%	99,60%	99,60%	99,58%
Pavia dataset								
0,1	6,27%	6,27%	6,27%	6,27%	46,99%	68,10%	84,62%	89,31%
1	6,27%	6,27%	6,27%	46,81%	80,75%	85,16%	89,28%	91,51%
10	5,39%	5,39%	46,79%	80,67%	85,21%	89,27%	91,17%	94,17%
100	6,27%	46,79%	80,65%	85,21%	89,19%	90,81%	93,73%	96,20%
1000	46,84%	80,64%	85,22%	89,18%	90,71%	93,39%	95,92%	95,45%
10 000	80,55%	85,21%	89,18%	90,70%	93,28%	95,47%	95,57%	95,40%
1 00 000	85,59%	89,17%	90,70%	93,31%	95,10%	95,24%	94,88%	95,40%
KSC dataset								
0,1	4,44%6	4,27%	16,80%	28,81%	38,15%	30,54%	60,13%	56,45%
1	5,84%	27,56%	54,48%	49,09%	47,64%	55,70%	58,40%	59,82%
10	15,93%	52,41%	27,64%	28,17%	47,57%	57,81%	45,18%	63,24%
100	34,54%	45,79%	35,81%	44,39%	51,01%	56,57%	59,63%	74,34%
1000	38,08%	34,73%	47,68%	30,72%	51,84%	60,63%	73,06%	80,58%
10 000	33,67%	44,08%	51,29%	29,62%	61,90%	72,72%	79,09%	87,40%
1 00 000	24,52%	49,99%	31,57%	49,57%	73,15%	77,19%	83,41%	93,60%
PaviaU dataset								
0,1	8,17%	8,17%	8,17%	8,17%	41,16%	63,91%	73,38%	79,70%
1	8,17%	8,17%	8,17%	41,21%	64,15%	73,10%	79,81%	84,28%
10	8,17%	8,17%	41,21%	64,13%	73,13%	79,58%	83,97%	89,81%
100	8,17%	41,21%	64,13%	73,13%	79,27%	83,79%	90,76%	92,30%
1000	41,21%	64,10%	73,15%	79,22%	83,70%	90,07%	92,36%	92,39%
10 000	64,07%	72,99%	79,23%	83,69%	89,58%	92,19%	92,20%	91,15%
1 00 000	73,05%	79,38%	83,61%	89,50%	91,13%	91,99%	91,27%	90,78%
Indiana dataset								
0,1	43,96%	42,41%	42,56%	44,24%	51,75%	57,26%	58,33%	58,06%
1	47,74%	46,15%	48,23%	52,07%	53,89%	59,07%	60,51%	71,13%
10	24,66%	44,53%	51,33%	54,39%	58,27%	61,07%	71,45%	85,51%
100	47,98%	50,32%	46,38%	58,54%	61,66%	69,46%	83,52%	90,57%
1000	49,13%	55,99%	58,71%	61,24%	69,29%	82,38%	88,45%	91,39%
10 000	51,84%	58,71%	61,45%	69,19%	81,56%	87,20%	89,65%	91,29%
1 00 000	56,52%	61,48%	69,22%	81,20%	86,84%	88,07%	89,53%	91,30%

TABLE IV
A SUMMARY OF THE RESULTS OBTAINED BY THE SAE MODEL WITH THE PARAMETERS APPLIED FOR EACH DATASET (NUMBER OF UNITS IN THE THREE HIDDEN LAYERS)

Dataset	Salinas dataset							
N_u	100, 100, 100	100, 100, 200	100, 200, 100	100, 200, 200	200, 100, 100	200, 100, 200	200, 200, 100	200, 200, 200
Accuracy	97,29%	97,28%	97,45%	97,43%	97,26%	97,37%	97,17%	97,48%
Dataset	**Pavia dataset**							
N_u	100, 100, 100	100, 100, 200	100, 200, 100	100, 200, 200	200, 100, 100	200, 100, 200	200, 200, 100	200, 200, 200
Accuracy	24,10%	80,29%	84,23%	84,51%	87,88%	89,68%	90,21%	91,13%
Dataset	**Pavia data sets**							
N_u	100, 100, 100	100, 100, 200	100, 200, 100	100, 200, 200	200, 100, 100	200, 100, 200	200, 200, 100	200, 200, 200
Accuracy	85,29%	84,53%	86,35%	86,76%	*87,34%*	86,01%	86,55%	86,41%
Dataset	**Indiana dataset**							
Nu	100, 100, 100	100, 100, 200	100, 200, 100	100, 200, 200	200, 100, 100	200, 100, 200	200, 200, 100	200, 200, 200
Accuracy	85,22%	86,12%	87,31%	85,94%	83,99%	86,05%	84,39%	86,59%
Dataset	**KSC dataset**							
Nu	100, 100, 100	100, 100, 200	100, 200, 100	100, 200, 200	200, 100, 100	200, 100, 200	200, 200, 100	200, 200, 200
Accuracy	70,82%	54,17%	60,28%	15,22%	69,95%	66,12%	64,12%	60,84%

TABLE V
ACCURACY OF SAE TRAINING WITH DIFFERENT PARAMETERS (SUPERVISED AND UNSUPERVISED LEARNING RATE R1 AND R2) APPLIED TO EACH DATASET

Dataset	Pavia dataset												
R1	0,0005	0,005	0,05	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
Accuracy using r1	36,76%	36,86%	82,72%	84,88%	85,43%	85,56%	85,64%	85,71%	85,88%	86,02%	86,13%	88,42%	74,85%
R2	0,000005	0,00005	0,0005	0,005	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,1
Accuracy using r2	87,99%	87,85%	86,87%	85,12%	85,00%	84,88%	84,86%	84,99%	85,08%	85,12%	85,13%	85,12%	85,36%
Dataset	**Salinas dataset**												
R1	0,0005	0,005	0,05	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
Accuracy using r1	19,58%	24,25%	74,16%	71,96%	88,87%	90,87%	83,82%	77,13%	77,39%	76,17%	68,51%	59,57%	54,64%
R2	0,000005	0,00005	0,0005	0,005	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,1
Accuracy using r2	64,12%	64,04%	63,59%	66,72%	68,35%	69,44%	70,84%	71,57%	72,35%	72,88%	73,36%	73,72%	74,15%
Dataset	**PaviaU dataset**												
R1	0,0005	0,005	0,05	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
Accuracy using r1	28,18%	29,77%	66,51%	74,22%	75,61%	76,81%	77,25%	77,52%	77,91%	78,62%	79,13%	79,58%	79,60%
R2	0,000005	0,00005	0,0005	0,005	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,1
Accuracy using r2	62,21%	79,39%	79,32%	79,52%	79,46%	79,59%	79,99%	80,12%	80,16%	80,29%	80,17%	79,94%	80,02%
Dataset	**Indiana dataset**												
R1	0,0005	0,005	0,05	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
Accuracy using r1	39,53%	39,77%	27,80%	29,30%	38,99%	38,39%	36,74%	38,05%	38,77%	39,67%	38,89%	39,13%	40,06%
R2	0,000005	0,00005	0,0005	0,005	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,1
Dataset	**KSC dataset**												
R1	0,0005	0,005	0,05	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
Accuracy using r1	18,36%	18,46%	18,63%	18,58%	18,58%	8,63%	8,65%	8,63%	21,08%	17,05%	16,82%	16,71%	17,08%
R2	0,000005	0,00005	0,0005	0,005	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,1
Accuracy using r2	16,44%	16,43%	20,65%	20,67%	16,73%	16,76%	16,82%	16,89%	17,08%	17,10%	17,05%	17,07%	21,08%

on the accuracy of training using the SAE model. The SAE model is configured with 50 iterations and a specific number of Nu 200 and 100. Additionally, it employs two learning rates for supervised learning and r2 for unsupervised learning. The reported results for each dataset are as follows: for the Pavia dataset, with a fixed, the accuracy was 88.42% at r1 = 09; and with a fixed r1, the accuracy was 87.99% at r2 = 0,000005. For the Salinas dataset, with a fixed r2, the accuracy was 90.87% at r1 = 03; if r1 was fixed, the accuracy was 74.15% at r2 = 01. For the PaviaU dataset, with a fixed r2, the accuracy was 79.60% when r1 = 10; if r1 was fixed, the accuracy reached its maximum 80.29% at r2 = 07. For the Indiana dataset, if we fixed r2, the accuracy was 40,05% at r1 = 10; when r1 was fixed, the accuracy was 52.27% at r2 = 01. For the KSC dataset, if r2 was fixed, the accuracy reached a maximum of 21,08% at r1 = 06; when r1 was fixed, the accuracy was 21,08% at r2 = 10.

Table VI depicts the correlation between the number of iterations and the accuracy of the SAE model across five datasets. The findings reveal that as the number of iterations employed during training increases, there is a corresponding improvement in the accuracy of the SAE model. In this analysis, a fixed supervised learning rate (r1) of 10 and an unsupervised learning rate (r2) of 006 are maintained. Additionally, the number of units in the three hidden layers (Nu) is fixed at 200, 100, and 100, respectively. This setup allows for a controlled examination of how varying the number of iterations impacts the performance of the SAE model across different datasets. The observed trend underscores the importance of iterative refinement in the training process, suggesting that prolonged training iterations contribute to enhanced model accuracy by allowing the network to learn more intricate patterns and representations within the data.

Table VII shows the comparison of the execution time of different learning algorithms, such as SAE and SVM. Execution time is an important consideration for comparing different algorithms. We ran each algorithm eight times, and each time we ran it, we utilized the parameters we had set. The results shown in Table VI are such that overall SAE took longer than SVM, but changing the parameters can alter the execution time.

Table VIII illustrates the impact of noise levels on the learning results. We added different levels of noise, as a percentage, to each image of each dataset for eight executions (eight times of training for each model with different parameters). The noise percentages used were 0,0195, 0,0781, 0,0391. To understand the significance of these percentages, it is important to note that if an image has a pixel value between 0 and 255, a noise percentage of 0,0391 means that the noise variance is: $0{,}0391 \times 256 = 10$.

Figs. 1– 5 demonstrate the training accuracy of SVM with MU sampling as well as SVM with RS sampling. The training curves on the different datasets show that SVM training with the MU method (red curve) generalizes better and gives a higher accuracy compared to SVM with the RS method (blue curve). The batch size used is 200 and the active learning algorithm is executed with a step of 200. The starting value of the accuracies obtained differs from one dataset to another (3200, 2800, 6000, 3000, 1500). The same figures show also that the SAE with MU method generalizes better than SAE with RS, and it gets

TABLE VI
RESULTS OF SAE LEARNING IN RELATION TO THE NUMBER OF ITERATIONS APPLIED TO EACH DATASET

Dataset	Salinas							
Number of iterations	10	50	100	200	300	500	600	1000
Accuracy	18,74%	67,22%	90,87%	96,12%	96,77%	97,29%	97,34%	97,80%
Dataset	**Pavia**							
Number of iterations	10	50	100	200	300	500	600	1000
Accuracy	34,72%	36,04%	88,94%	89,23%	91,13	92,47%	93,23%	94,93%
Dataset	**PaviaU**							
Number of iterations	10	50	100	200	300	500	600	1000
Accuracy	24,10%	80,29%	84,23%	84,51%	87,88%	89,68%	90,21%	91,13%
Dataset	India							
Number of iterations	10	50	100	200	300	500	600	1000
Accuracy	41,59%	52,27%	53,06%	57,53%	62,56%	65,26%	69,38%	71,04%
Dataset	**KSC**							
Number of iterations	10	50	100	200	300	500	600	1000
Accuracy	16,71%	35,17%	20,82%	18,76%	21,25%	17,22%	37,65%	25,00%

As the number of iterations increases, the accuracy of the SAE model also increases

TABLE VII
CONSUMPTION TIME OF EACH ALGORITHM (SAE AND SVM) IN EACH EXECUTION (1,8) ON DATASETS

Execution Order	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
SVM on PaviaU Dataset	14	9	7	6	7	8	13	6
SAE on PaviaU	1232	1472	1195	1385	1627	13411	1745	1805
SVM on Pavia	17	11	7	6	5	6	5	6
SAE on Pavia	1702	1924	1834	2106	2303	2127	2254	2576
SVM on Salinas	109	72	42	32	31	31	31	34
SAE on Salinas	2468	2558	2529	3156	31310	3768	3686	3893
SVM on Indiana dataset	1611	171	191	188	176	173	153	121
SAE on Indiana	1101	1161	1271	1581	1505	1663	1705	1808
SVM on KSC	46	46	47	47	48	48	48	48
SAE on KSC	422	717	771	7874	724	7311	913	998

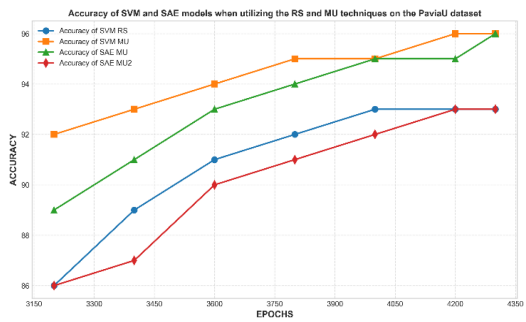

Fig. 4. Performance comparison on the PaviaU dataset.

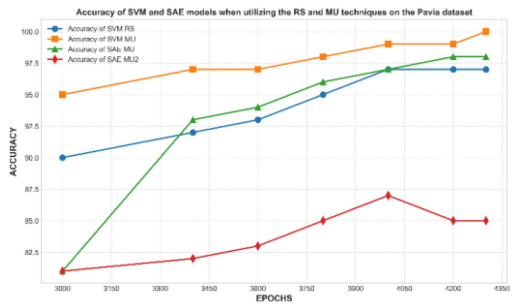

Fig. 5. Performance comparison on the Pavia dataset.

the best results at each iteration. The SVM-MU and SAE-MU algorithms are able to generalize better because they are able to learn features and representations from the data that are more.

This Fig. 4 shows the performance comparison between SVM and SAE on the PaviaU dataset. The orange curve (SVM-MU) achieves the highest accuracy (around 96%), demonstrating that the MU approach allows the SVM to generalize better than the RS method. The green curve (SAE-MU) also shows good progress, reaching an accuracy comparable to SVM-MU toward the end of training. Both MU curves (SVM and SAE) consistently outperform their RS counterparts, confirming the effectiveness of the active learning method.

Fig. 5 illustrates the performance of the Pavia dataset. SVM-MU (orange line) shows superior performance, reaching nearly 98% accuracy toward the end of training. SAE-MU (green line) starts with lower accuracy (around 81%) but quickly progresses to around 98%. Note that SAE with RS (purple line) shows significantly lower performance with stagnation around 85%, suggesting that active learning is particularly beneficial for the SAE model on this dataset.

TABLE VIII
RESULTS OF ADDING NOISE TO THE IMAGES OF THE DATASETS AND ITS EFFECT ON THE LEARNING PROCESS

Execution	1	2	3	4	5	6	7	8
The variance of the noise	0,0195							
SVM on PaviaU Dataset	69,78%	78,96%	82,00%	82,60%	81,08%	81,46%	81,97%	82,79%
SAE on PaviaU Dataset	86,68%	86,82%	83,80%	84,26%	86,99%	83,26%	84,15%	83,20%
SVM on Pavia Dataset	84,78%	88,80%	90,13%	90,78%	89,69%	88,82%	89,39%	91,19%
SAE on Pavia Dataset	88,28%	88,16%	88,05%	88,03%	88,09%	88,19%	88,13%	88,17%
SVM on Salinas Dataset	74,97%	92,12%	94,71%	94,61%	92,28%	91,23%	92,03%	93,67%
SAE on Salinas Dataset	94,36%	94,82%	94,59%	94,73%	94,73%	94,94%	95,08%	95,13%
SVM on Indiana dataset	53,04%	58,52%	64,77%	74,87%	77,37%	76,66%	78,21%	82,39%
SAE on Indiana dataset	84,95%	84,08%	83,30%	83,98%	83,08%	82,78%	77,78%	84,27%
SVM on KSC Dataset	26,60%	25,02%	24,63%	24,10%	24,10%	24,79%	23,33%	24,65%
SAE on KSC Dataset	11,63%	11,92%	12,38%	11,95%	12,55%	12,38%	11,92%	12,46%
The variance of the noise	0,0391							
SVM on PaviaU Dataset	71,72%	78,13%	79,54%	78,32%	76,96%	77,97%	78,01%	78,89%
SAE on PaviaU Dataset	79,76%	79,55%	79,81%	81,07%	80,96%	81,26%	80,26%	80,04%
SVM on Pavia Dataset	83,40%	87,99%	89,05%	88,20%	86,17%	85,91%	86,02%	88,06%
SAE on Pavia Dataset	89,21%	89,16%	89,56%	89,98%	89,49%	89,43%	89,76%	89,83%
SVM on Salinas Dataset	66,84%	88,59%	89,46%	86,55%	83,82%	83,66%	84,57%	87,22%
SAE on Salinas Dataset	90,15%	90,20%	90,19%	90,15%	90,09%	90,15%	90,19%	90,22%
SVM on Indiana dataset	45,81%	50,21%	50,63%	48,98%	49,68%	53,15%	55,30%	57,81%
SAE on Indiana dataset	48,63%	47,89%	47,31%	47,94%	48,13%	48,77%	48,67%	47,39%
SVM on KSC Dataset	16,32%	21,13%	21,72%	20,54%	20,85%	21,79%	20,63%	20,82%
SAE on KSC Dataset	9,44%	9,30%	10,17%	10,49%	10,22%	9,52%	10,08%	9,49%
The variance of the noise	0,0781							
SVM on PaviaU Dataset	73,99%	75,41%	75,16%	73,15%	71,98%	71,45%	73,01%	74,53%
SAE on PaviaU Dataset	96,36%	95,83%	76,27%	75,70%	75,58%	76,47%	76,25%	76,00%
SVM on Pavia Dataset	80,13	85,80%	85,90%	84,25%	82,91%	82,78%	82,87%	84,52%
SAE on Pavia Dataset	84,85%	84,60%	84,25%	84,78%	84,25%	83,57%	83,49%	83,61%
SVM on Salinas Dataset	64,50%	79,57%	77,46%	74,01%	71,40%	71,64%	73,25%	76,81%
SAE on Salinas	77,55%	77,67%	77,69%	77,36%	77,97%	77,69%	77,90%	78,59%
SVM on Indiana dataset	45,83%	47,39%	49,59%	50,69%	50,98%	49,96%	50,32%	50,35%
SAE on Indiana dataset	38,13%	37,80%	38,00%	38,08%	37,48%	37,64%	37,38%	37,69%
SVM on KSC Dataset	20,38%	20,32%	20,66%	20,57%	20,47%	20,44%	20,44%	20,72%
SAE on KSC Dataset	9,25%	8,55%	10,05%	9,60%	9,25%	9,00%	9,79%	9,17%

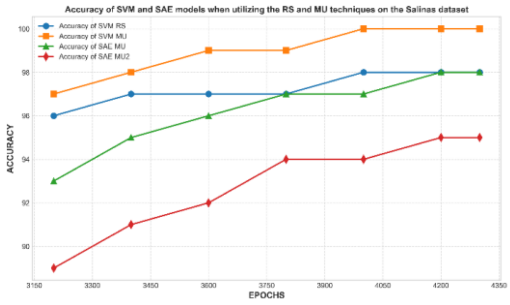

Fig. 6. Performance comparison on the Salinas dataset.

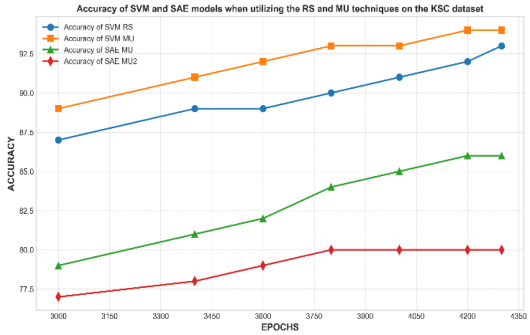

Fig. 7. Performance comparison on the KSC dataset.

On the Salinas dataset, all methods achieve high accuracies. SVM-MU (orange line in Fig. 6) shows the best performance, reaching 99% toward the end of training. SVM-RS (blue line) already starts at a high accuracy of 96%, suggesting that this dataset is relatively easy to classify. SAE models also show good performance, with SAE-MU reaching around 98% and SAE-RS around 95%. This figure demonstrates that on well-structured datasets like Salinas, even RS methods can achieve acceptable performance.

This figure presents the results of the KSC (Kennedy Space Center) dataset. Both SVM models significantly outperform the SAE models. SVM-MU (orange line in Fig. 7) achieves the best accuracy at around 94%, followed closely by SVM-RS at 93%. The SAE models show slower progress, with SAE-MU reaching around 86% and SAE-RS stagnating around 80%. This notable difference in performance between SVM and SAE can be attributed to the complex nature of the KSC hyperspectral data, where SVMs appear to be better suited to capturing the distinctive features of this dataset.

On the Indiana dataset, we observe the greatest disparity between methods. SVM-MU (orange line in Fig. 8) significantly outperforms other approaches, maintaining a high accuracy of around 85%–89% throughout training. SAE-MU (green line) shows a steady improvement, increasing from 64% to 80%.

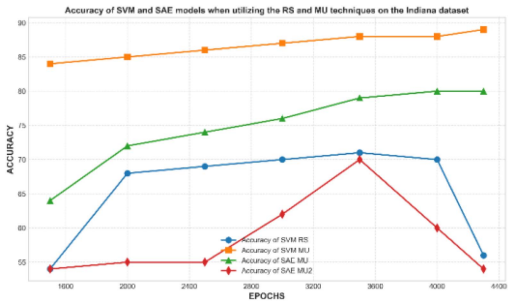

Fig. 8. Performance comparison on the Indiana dataset.

SVM-RS (blue line) and SAE-RS (purple line) exhibit unstable performance with significant fluctuations, notably SVM-RS dropping to 56% at the end after reaching 71%. This instability suggests that the Indiana dataset has complex and heterogeneous features that require an active learning approach for stable and accurate classification.

B. Discussion

The observed performance trends of SAE and SVM are closely tied to the structural and statistical properties of hyperspectral data. SAE's advantage becomes more pronounced with larger datasets due to its ability to learn hierarchical and abstract representations from high-dimensional spectral inputs. This makes it particularly suitable for denoising and compressing redundant spectral bands—an inherent characteristic of hyperspectral imagery. We found that certain configurations of SAE (e.g., three hidden layers with 200 units and learning rates between 0.05 and 0.3) achieved better generalization, suggesting a sweet spot between depth and training stability. These configurations enabled the network to converge effectively without overfitting. The use of dropout during inference also helped capture uncertainty, improving robustness in the presence of noise. On the other hand, SVM's effectiveness in small-sample regimes can be attributed to its reliance on support vectors, making it resilient in high-dimensional, low-sample settings. This behavior aligns with the sparsity and limited annotation issues common in remote sensing. The combined interpretation of these results gives rise to theory-informed explanations beyond levels of general assertions about "deep learning power," linking model performance explicitly to HSI characteristics. The key findings of the work can be summarized as follows:

1) Parameter Comparison Between SVM and SAE: SVM and SAE employ different types of parameters. SAE employs simple parameters such as W and b, while SVM utilizes parameters such as α for the purpose of setting its kernel functions and so forth. SAE also possesses supporting parameters such as the number of hidden layers, hidden layer nodes, and learning rates. Simple parameters are normally tuned at training, while supporting parameters are chosen manually based on experience. Notably, SAE entails more helping parameters with correlated relationships, as seen in observations of Tables IV–VI. Parameters like learning rates and numbers of hidden nodes are tunable in SAE. Contrarily, SVM entails a limited number of parameters whose selection is very simple, as shown in Table III.

Comparison experiments reveal that determining parameters in SAE poses greater challenges compared to SVM, particularly in hyperspectral classification tasks. This difficulty arises from the ambiguity surrounding the importance and optimal range of SAE parameters. While the number of layers in SAE may hold significance to some extent, proving its theoretical importance remains challenging.

2) Data Size and Model Complexity: HSI data files are usually between 10 and 1000 megabytes (MB) in size. In the context of remote sensing, it is common to analyze one scene's worth of image data at a time. Despite this data being relatively large, it is not considered big data. Additionally, the number of samples with known labels, which are used to train machine-learning models, is limited. For traditional SVM classifiers, the requirement for a significant number of training samples is not crucial. This is because of a concept known as the Vapnik–Chervonenkis (VC) dimension, proposed by Vapnik in statistical ML. However, the SAE model, used in this research, has substantially more parameters compared to SVM, especially when the SAE network is deep and has numerous layers. Additionally, the parameters in SAE interact multiplicatively rather than additively, making the model more complex. Hence, autoencoder (SAE) models would have a larger VC dimension than other models, although it is still hard to determine it precisely for deep networks. SAE requires more training samples and computational resources with more parameters. In this study, we found that SAE performs very well when there are enough training samples. However, it should be observed that SAE takes much more training samples than SVM to achieve optimum performance.

3) Computational Efficiency Analysis: When looking at the computational efficiency of SVM and SAE, the iterations to converge are critical. SVM requires fewer iterations than SAE in most instances, as shown in Table VII. Consequently, SVM consumes less computational time, especially in medium or small sizes of datasets. The computational complexity of SAE primarily stems from two factors: its extensive parameter set and its relatively slow convergence rate, even with larger rates. In contrast, SAE shares similarities with online learning, allowing training instances to be processed sequentially. This feature facilitates easy adaptation and implementation on parallel computing platforms such as clusters or GPUs. Looking ahead, advancements in hardware and computational capabilities may enhance the efficiency of SAE, potentially closing the performance gap with SVM in terms of computational time.

4) Dataset Selection and Performance Evaluation: It is essential to employ diverse datasets that adequately cover various aspects of our models. We utilized multiple public hyperspectral datasets to assess the accuracy of both SVM and SAE. However, it is crucial to recognize that classification outcomes might not generalize to Synthetic Aperture Radar (SAR) data, multispectral data, etc. In this study, we evaluated the models using commonly used hyperspectral datasets in the literature to ensure the comprehensiveness of our analysis. It is important to consider the representativeness of these experiments when applying deep network-based classification to other hyperspectral datasets. Our observations, as depicted in Tables IV–VI, reveal that SVM generally outperformed SAE in most cases. However, there were instances where the accuracy of SAE was comparable to SVM.

TABLE IX
COMPARISON WITH PRIOR STUDIES

Study	Compared models	Classification performance (%) (best accuracy achieved)	Spectral–spatial	Active learning	Noise robustness	Parameter sensitivity	Computation analysis
Zhao et al., 2021 [10]	Autoencoder (spectral–spatial)	~95.4	Yes	No	No	No	No
Tian et al., 2024 [11]	Sparse autoencoder (unsupervised)	~90.2	No	No	No	No	No
Zhouhan et al., 2013 [12]	SAE (spectral–spatial)	~96.5	Yes	No	No	No	No
Zhong et al., 2018 [21]	3D CNN vs. SVM	~98.6	Yes	No	No	Partial	No
Chen et al., [22]	CNN vs. SVM/ELM	~97.0	Yes	No	No	Partial	No
Our study	SVM vs. SAE	SVM~ 97 SAE ~99	No	Yes	Yes	Yes	Yes

This discrepancy may be attributed to the fact that SAE tends to excel primarily in scenarios involving very large datasets, whereas its advantages may be less pronounced when dealing with relatively small datasets.

5) Active Learning and Model Performance: The active learning method's ability to improve the model's performance is shown or proven by the data presented in Figs. 4 and 5. SVM and SAE models can utilize an active learning algorithm. This algorithm helps them automatically identify which samples from the dataset are the most informative or useful for improving the classification accuracy of the model. When you use active learning to pick which examples the model learns from, it usually does better than just picking examples randomly. However, how much this helps depends on the type of model. In this case, when SAE and SVM are each provided with an identical set of training data, SVM learns faster and more accurately than SAE, especially when there are only a few examples to learn from. Typically, the number of training samples available for a single scene of hyperspectral data is sufficient for SVM but inadequate for SAE. In Figs. 4 and 5, SVM starts with a high accuracy, but its accuracy does not improve much after that. Consequently, the curves representing SVM in these figures appear relatively flat. Conversely, the curves representing SAE exhibit a steep incline, indicating substantial accuracy improvement with active learning. Hence, we conclude that for datasets of medium size, the active learning approach is more crucial for enhancing the performance of SAE compared to SVM.

6) Noise Sensitivity Analysis: We conducted an assessment of the sensitivity of both methods to noise using various datasets. It was difficult to determine which method is more resistant to noise, but our analysis, illustrated in Table VIII , showed that both SAE and SVM are affected by noise. For instance, on the Indiana dataset, the accuracy exceeded 90% in the absence of noise. However, upon introducing noise, the accuracy of both methods plummeted to below 60%. This trend was consistent across other datasets, indicating that noise has a substantial detrimental impact on the accuracy of both methods. Our experimental results highlight the comparative performance of different classification methods on hyperspectral data, it includes traditional machine-learning methods such as SVM and simpler deep learning models like SAE to provide a comprehensive perspective.

7) Performance Analysis and Comparison With Prior Studies: Table IX presents a comparative summary of recent studies and our proposed work. Even if previous research has investigated either traditional or deep models, most of these studies were limited to classification performance and spectral–spatial representation. On the other side, our study offers a broader experimental design by integrating dimensions such as active learning, noise sensitivity, and parameter tuning, which are essential for the practical deployment of HSI classifiers.

Our study reports peak accuracies of 99.6% for SVM (achieved on the Salinas dataset with optimized parameters c = 1 00 000, g = 0.01) and 97.5% for SAE (achieved with three hidden layers of 200 units each after 1000 iterations). These results are competitive with state-of-the-art approaches like 3D CNN (98.6% reported by Zhong et al.) and CNN (97.0% by Chen et al.).

What is notable is that our models achieve these competitive accuracies without incorporating spatial information, relying solely on spectral features. In contrast, most previous high-performing studies (like Chen, Zhong, and Lin) explicitly leveraged spatial context through specialized architectures like 3D CNNs. This suggests that properly optimized spectral-only models can approach the performance of spectral-spatial models under certain conditions.

The performance differences between datasets are also significant. Both SVM and SAE performed exceptionally well on the Salinas dataset (99.6% and 97.5%, respectively), which has well-separated class structures. However, the performance gap between SVM and SAE widened on more challenging datasets like KSC, where SVM reached 93.6% while SAE managed only 64.1%. This underscores SVM's robustness in scenarios with limited training samples.

Furthermore, our active learning experiments discovered that even though SVM has better accuracy initially, SAE learns much better with entropy-based sample selection, which can be clearly observed from the Indiana dataset results. This demonstrates that diligent training data selection can compensate to some extent

for SAE's need for larger training data sets.

Our comparison of SAE and SVM, notwithstanding the fact that SAE is not the state of the art in deep learning for HSI classification, is nevertheless significant for a few reasons. First, it serves as an exploratory examination of the different architectural impacts on hyperspectral data as a precursor to more sophisticated models. Second, the comparison is of educational value in allowing readers to grasp how the field has progressed from traditional to modern algorithms. Lastly, with resource constraints in consideration, older methods like SVM and simpler deep learning models like SAE are less computationally resource-intensive. By including them in our comparison, we emphasize the tradeoff between classification accuracy and computational resources, demonstrating that even less computationally intensive methods can yield insights of value.

Another key limitation related to these models is that they do not leverage spatial context, which is often critical in HSI classification. Spatial information, such as pixel neighborhood structures or textures, can significantly enhance classification performance, especially in complex environments where spectral signatures alone are ambiguous. Recent studies have shown that combining spectral and spatial features using CNNs, graph convolutional networks (GCNs), and spectral–spatial attention mechanisms leads to superior results achieved by Zhong. However, our objective was to isolate and analyze the behavior of purely spectral models under varying conditions of data size, noise, and active learning, in order to establish a clear performance baseline. We consider this spectral-only focus a necessary first step before extending the analysis to more complex, spatial-aware architectures. Future work should incorporate spatial–spectral models for a more comprehensive evaluation and to align more closely with current state-of-the-art approaches in the field.

V. Conclusion

ML significantly depends on a vast number of labeled samples, but obtaining these samples can be both time-consuming and expensive. Achieving a precise classification map necessitates the availability of a large quantity of high-quality training samples. This requirement is especially critical when training deep networks, where the choice of an appropriate training dataset becomes even more vital compared to the less demanding data needs of SVM. In this study, we compared the performance of SVM and SAE in the classification of hyperspectral remote sensing images. Our findings reveal that while SVM excels in computational efficiency and performs well with smaller datasets, SAE has the potential for better generalization with larger datasets. The study highlights the importance of parameter selection and active learning in enhancing SAE's performance. Noise sensitivity remains a challenge for both models, suggesting the need for robust noise-handling techniques in future research. Future research will benefit from including techniques like attention mechanisms and spectral–spatial fusion architectures (such as CNNs, GCNs, and Transformers) for improving classification accuracy by capturing both local spatial context and global spectral relationships. Additionally, developing hybrid or attention-enhanced SAE variants tailored for hyperspectral applications could address more practical real-world deployment scenarios.

Acknowledgment

The authors would like to extend their heartfelt gratitude to all the participants in their study who generously shared their time, experiences, and insights. Their willingness to engage with the research was essential to the success of the project.

References

[1] E. Omia et al., "Remote sensing in field crop monitoring: A comprehensive review of sensor systems, data analyses and recent advances," *Remote Sens.*, vol. 15, no. 2, Jan. 2023, Art. no. 354, doi: 10.3390/rs15020354.

[2] O. N. Chisom, P. W. Biu, A. A. Umoh, B. O. Obaedo, A. O. Adegbite, and A. Abatan, "Reviewing the role of AI in environmental monitoring and conservation: A data-driven revolution for our planet," *World J. Adv. Res. Rev.*, vol. 21, no. 1, pp. 161–171, Jan. 2024, doi: 10.30574/wjarr.2024.21.1.2720.

[3] Z. Zaman, S. B. Ahmed, and M. I. Malik, "Analysis of hyperspectral data to develop an approach for document images," *Sensors*, vol. 23, no. 15, Aug. 2023, Art. no. 6845, doi: 10.3390/s23156845.

[4] H. Zhang et al., "Hyperspectral classification based on 3D asymmetric inception network with data fusion transfer learning,' 2020, *arXiv:2002.04227*.

[5] G. Vivone, "Multispectral and hyperspectral image fusion in remote sensing: A survey," *Inf. Fusion*, vol. 89, pp. 405–417, Jan. 2023, doi: 10.1016/j.inffus.2022.08.032.

[6] H. N. Mahendra and S. Mallikarjunaswamy, "An efficient classification of hyperspectral remotely sensed data using support vector machine," *Int. J. Electron. Telecommun.*, vol. 68, no. 3, pp. 609–617, Apr. 2022, doi: 10.24425/ijet.2022.141280.

[7] R. Shad, S. T. Seyyed-Al-hosseini, Y. M. Mehrani, and M. Ghaemi, "Ensemble of support vector machines for spectral-spatial classification of hyperspectral and multispectral images," *Multimedia Tools Appl.*, vol. 82, no. 27, pp. 42119–42146, Nov. 2023, doi: 10.1007/s11042-023-14972-3.

[8] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Deep learning for hyperspectral image classification: An overview," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 6690–6709, Sep. 2019, doi: 10.1109/TGRS.2019.2907932.

[9] N. Li et al., "Hierarchical constrained variational autoencoder for interaction-sparse recommendations," *Inf. Process. Manage.*, vol. 61, no. 3, May 2024, Art. no. 103641, doi: 10.1016/j.ipm.2024.103641.

[10] C. Zhao, X. Wan, G. Zhao, B. Cui, W. Liu, and B. Qi, "Spectral-spatial classification of hyperspectral imagery based on stacked sparse autoencoder and random forest," *Eur. J. Remote Sens.*, vol. 50, no. 1, pp. 47–63, Jan. 2017, doi: 10.1080/22797254.2017.1274566.

[11] Q. Tian, C. He, Y. Xu, Z. Wu, and Z. Wei, "Hyperspectral target detection: Learning faithful background representations via orthogonal subspace-guided variational autoencoder," *IEEE Trans. Geosci. Remote Sens.*, vol. 62, 2024, Art. no. 5516714, doi: 10.1109/TGRS.2024.3393931.

[12] Z. Lin, Y. Chen, X. Zhao, and G. Wang, "Spectral-spatial classification of hyperspectral image using autoencoders," in *Proc. 9th Int. Conf. Inf., Commun. Signal Process.*, Dec. 2013, pp. 1–5. doi: 10.1109/ICICS.2013.6782778.

[13] M. J. Khan, H. S. Khan, A. Yousaf, K. Khurshid, and A. Abbas, "Modern trends in hyperspectral image analysis: A review," *IEEE Access*, vol. 6, pp. 14118–14129, 2018, doi: 10.1109/ACCESS.2018.2812999.

[14] J. Feng et al., "Class-aligned and class-balancing generative domain adaptation for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 62, 2024, Art. no. 5509617, doi: 10.1109/TGRS.2024.3367765.

[15] J. Feng, Q. Jiang, J. Zhang, Y. Liang, R. Shang, and L. Jiao, "CFDRM: Coarse-to-fine dynamic refinement model for weakly supervised moving vehicle detection in satellite videos," *IEEE Trans. Geosci. Remote Sens.*, vol. 62, 2024, Art. no. 5626413, doi: 10.1109/TGRS.2024.3403868.

[16] Y. Ding et al., "Multi-feature fusion: Graph neural network and CNN combining for hyperspectral image classification," *Neurocomputing*, vol. 501, pp. 246–257, Aug. 2022, doi: 10.1016/j.neucom.2022.06.031.

[17] G. Omer, O. Mutanga, E. M. Abdel-Rahman, and E. Adam, "Performance of support vector machines and artificial neural network for mapping endangered tree species using WorldView-2 data in Dukuduku Forest, South Africa," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 10, pp. 4825–4840, Oct. 2015, doi: 10.1109/JSTARS.2015.2461136.

[18] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, Nov. 2020, doi: 10.1016/j.neucom.2020.07.061.
[19] A. Sankaran, M. Vatsa, R. Singh, and A. Majumdar, "Group sparse autoencoder," *Image Vis. Comput.*, vol. 60, pp. 64–74, Apr. 2017, doi: 10.1016/j.imavis.2017.01.005.
[20] D. J. Lary, A. H. Alavi, A. H. Gandomi, and A. L. Walker, "Machine learning in geosciences and remote sensing," *Geosci. Front.*, vol. 7, no. 1, pp. 3–10, Jan. 2016, doi: 10.1016/j.gsf.2015.07.003.
[21] Z. Zhong, J. Li, Z. Luo, and M. Chapman, "Spectral–Spatial residual network for hyperspectral image classification: A 3-D deep learning framework," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 2, pp. 847–858, Feb. 2018, doi: 10.1109/TGRS.2017.2755542.
[22] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6232–6251, Oct. 2016, doi: 10.1109/TGRS.2016.2584107.
[23] G. C. Batista, D. L. Oliveira, O. Saotome, and W. L. S. Silva, "A low-power asynchronous hardware implementation of a novel SVM classifier, with an application in a speech recognition system," *Microelectronics J.*, vol. 105, Nov. 2020, Art. no. 104907, doi: 10.1016/j.mejo.2020.104907.
[24] M. Papadonikolakis and C.-S. Bouganis, "A heterogeneous FPGA architecture for support vector machine training," in *Proc. 18th IEEE Annu. Int. Symp. Field-Program. Custom Comput. Machines*, 2010, pp. 211–214. doi: 10.1109/FCCM.2010.39.
[25] O. Boujelben and M. Bahoura, "Efficient FPGA-based architecture of an automatic wheeze detector using a combination of MFCC and SVM algorithms," *J. Syst. Archit.*, vol. 88, pp. 54–64, Aug. 2018, doi: 10.1016/j.sysarc.2018.05.010.
[26] M. A. Elgammal, H. Mostafa, K. N. Salama, and A. Nader Mohieldin, "A comparison of artificial neural network (ANN) and support vector machine (SVM) classifiers for neural seizure detection," in *Proc. IEEE 62nd Int. Midwest Symp. Circuits Syst.*, Aug. 2019, pp. 646–649. doi: 10.1109/MWSCAS.2019.8884989.
[27] L. Feng, Z. Li, Y. Wang, and C. Wang, "A fast on-chip SVM-training system with dual-mode configurable pipelines and MSMO scheduler," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 66, no. 11, pp. 4230–4241, Nov. 2019, doi: 10.1109/TCSI.2019.2929054.
[28] R. A. Patil, G. Gupta, V. Sahula, and A. S. Mandal, "Power Aware hardware prototyping of multiclass SVM classifier through reconfiguration," in *Proc. 25th Int. Conf. VLSI Des.*, Jan. 2012, pp. 62–67. doi: 10.1109/VLSID.2012.47.
[29] D. Thi Phuong Chung and D. V. Tai, "A fruits recognition system based on a modern deep learning technique," *J. Phys.: Conf. Ser.*, vol. 1327, no. 1, Oct. 2019, Art. no. 012050, doi: 10.1088/1742-6596/1327/1/012050.
[30] S. Chen, H. Wang, F. Xu, and Y.-Q. Jin, "Target classification using the deep convolutional networks for SAR images," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 8, pp. 4806–4817, Aug. 2016, doi: 10.1109/TGRS.2016.2551720.
[31] M. K. Gill, T. Asefa, M. W. Kemblowski, and M. McKee, "Soil moisture prediction using support vector machines," *J. Amer. Water Resour. Assoc.*, vol. 42, no. 4, pp. 1033–1046, Aug. 2006, doi: 10.1111/j.1752-1688.2006.tb04512.x.
[32] T. Kasinathan, D. Singaraju, and S. R. Uyyala, "Insect classification and detection in field crops using modern machine learning techniques," *Inf. Process. Agriculture*, vol. 8, no. 3, pp. 446–457, Sep. 2021, doi: 10.1016/j.inpa.2020.09.006.
[33] A. Gullì, A. Kapoor, and S. Pal, *Deep Learning With TensorFlow 2 and Keras: Regression, ConvNets, GANs, RNNs, NLP, and More With TensorFlow 2 and the Keras API*, Second edition. Birmingham Mumbai: Packt, 2019.
[34] X. Yang, Y. Ye, X. Li, R. Y. K. Lau, X. Zhang, and X. Huang, "Hyperspectral image classification with deep learning models," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 9, pp. 5408–5423, Sep. 2018, doi: 10.1109/TGRS.2018.2815613.
[35] D. Datta, P. K. Mallick, A. K. Bhoi, M. F. Ijaz, J. Shafi, and J. Choi, "Hyperspectral image classification: Potentials, challenges, and future directions," *Comput. Intell. Neurosci.*, vol. 2022, pp. 1–36, Apr. 2022, doi: 10.1155/2022/3854635.
[36] W. S. Noble, "What is a support vector machine?," *Nature Biotechnol.*, vol. 24, no. 12, pp. 1565–1567, Dec. 2006, doi: 10.1038/nbt1206-1565.
[37] G. Mercier and M. Lennon, "Support vector machines for hyperspectral image classification with spectral-based kernels," in *Proc. IEEE Int. Geosci. Remote Sens. Symp..*, 2003, pp. 288–290. doi: 10.1109/IGARSS.2003.1293752.
[38] S. Turgay, M. Han, S. Erdoğan, E. S. Kara, and R. Yilmaz, "Evaluating the predictive modeling performance of kernel trick SVM, market basket analysis and Naive Bayes in terms of efficiency," *Wseas Trans. Comput.*, vol. 23, pp. 56–66, Apr. 2024, doi: 10.37394/23205.2024.23.6.
[39] W. Vieira De Oliveira, L. V. Dutra, and S. J. S. Sant'Anna, "Unfolding multilevel agglomerative strategies for SVM classification: A case study in discriminating spectrally similar land covers," *Remote Sens. Lett.*, vol. 15, no. 10, pp. 1035–1046, Oct. 2024, doi: 10.1080/2150704X.2024.2398813.
[40] B. Yan and G. Han, "Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system," *IEEE Access*, vol. 6, pp. 41238–41248, 2018, doi: 10.1109/ACCESS.2018.2858277.
[41] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006, doi: 10.1126/science.1127647.
[42] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103. doi: 10.1145/1390156.1390294.
[43] C. Tao, H. Pan, Y. Li, and Z. Zou, "Unsupervised spectral–spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 12, pp. 2438–2442, Dec. 2015, doi: 10.1109/LGRS.2015.2482520.
[44] H. Ranganathan, H. Venkateswara, S. Chakraborty, and S. Panchanathan, "Deep active learning for image classification," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2017, pp. 3934–3938. doi: 10.1109/ICIP.2017.8297020.
[45] D. U. Jo, S. Yun, and J. Y. Choi, "How much a model be trained by passive learning before active learning?," *IEEE Access*, vol. 10, pp. 34677–34689, 2022, doi: 10.1109/ACCESS.2022.3162253.
[46] A. Bhatt and A. Bhatt, "EEG based emotion recognition using SVM and LibSVM," *Int. J. Comput. Appl.*, vol. 178, no. 45, pp. 1–3, Sep. 2019, doi: 10.5120/ijca2019919352.

Brahim Jabir received the Ph.D. degree in computer engineering from the Faculty of Sciences and Techniques, Sultan Moulay Slimane University, Beni Mellal, Morocco, in 2022.

He is a Professor of computer science with the Polydisciplinary Faculty of Sidi Bennour, Chouaib Doukkali University, El Jadida, Morocco. For five years, he served as a Professor of computer science with the Teacher Training College, Beni Mellal. His research primarily focuses on artificial intelligence (AI) and its diverse applications across various fields.

Mr. Jabir is currently an Associate Editor for the *International Journal of Information Technologies and Systems* at IGI Global. He is a Member of the research team in Computer Science and Modeling (ESIM).

Bendaoud Nadif received the ENS Diploma in ELT, in 2006, several diplomas from the American Language Center of Fez, Fez, Morocco, and the British Council in Rabat, Rabat, Morocco, the B.A. degree in linguistics and the master's degree in staff development and research in higher education from the Faculty of Arts and Humanities, Sidi Mohamed Ben Abdellah University, Fez, Morocco, in 2001 and 2006, respectively, the Ph.D. degree in applied linguistics from the Faculty of Arts and Humanities, Moulay Ismail University, Meknes, Morocco, in 2020.

He is a Lecturer and Teacher Trainer with ESEF, Sultan Moulay Slimane University. He is a Permanent Member with the Research Group on Discourse Analysis, Faculty of Lettres and Human Sciences, SMSU, Beni Mellal. His main research interests are applied linguistics, language development, ICT, CPD, language awareness, cognitive psychology, sociolinguistics, cultural studies, and AI.