Intelligent System for Hyperspectral Image Classification Using Embedded-Label GANs

Victor Sineglazov[1,†], Oleksii Shcherban[1,*,†]

[1] Kyiv Aviation Institute, Liubomyra Huzara Ave. 1, Kyiv, 03058, Ukraine

Abstract

This paper presents an intelligent system for hyperspectral image classification based on an enhanced generative adversarial network with embedded label conditioning. The proposed architecture enables effective augmentation of limited training datasets with class-specific synthetic spectral samples. Key components include label embeddings, spectral regularization, and tailored loss functions designed to improve class separability in the feature space. Experiments on benchmark hyperspectral datasets demonstrate improved classification accuracy, even under scarce supervision. The approach shows strong potential for precision agriculture and vegetation monitoring applications.

Keywords

hyperspectral images, generative adversarial networks, AC-WGAN-GP, classification, synthetic samples, class-aware sampling, label embedding, precision agriculture

1. Introduction

The agricultural sector faces mounting challenges due to rapid population growth, climate variability, and diminishing natural resources. In response, precision agriculture has emerged as a promising paradigm that leverages advanced technologies to optimize crop production and resource utilization [1].

Among these technologies, hyperspectral imaging (HSI) stands out for its ability to capture highresolution spectral profiles across hundreds of contiguous bands. This rich spectral detail enables early detection of crop stress factors such as nutrient deficiencies, pest infestations, and water stress [2, 3]. When combined with thermal imaging, HSI provides complementary information to enhance crop monitoring during critical phenological stages.

Conventional field-based monitoring methods, although accurate, are labor-intensive, time-consuming, and inadequate for large-scale, real-time decision-making [4, 5]. In contrast, artificial intelligence (AI) models have demonstrated strong capabilities in processing high-dimensional remote sensing data efficiently, particularly when applied to data collected from satellites, unmanned aerial vehicles (UAVs), and ground-based platforms [6, 7, 8, 9].

Each platform has inherent trade-offs: satellites provide broad area coverage with lower spatial and temporal resolution [10], UAVs offer high-resolution imagery but are constrained by limited flight duration and operational cost [11], while ground systems provide precise, localized measurements with limited scalability [5].

Despite these limitations, the integration of satellite HSI data with AI-based analysis and sensor fusion has shown promise as a scalable and cost-effective solution for real-time agricultural monitoring over large areas. This study builds upon this foundation, focusing on enhancing hyperspectral image classification through deep generative modeling.

ICST-2025: Information Control Systems & Technologies, September 24-26, 2025, Odesa, Ukraine

[*] Corresponding author.

[†] These authors contributed equally.

svm@kai.edu.ua (V. Sineglazov); oleksiishcherbanrw@gmail.com (O. Shcherban)

0000-0002-3297-9060 (V. Sineglazov); 0009-0004-8702-4917 (O. Shcherban)

2. Hyperspectral and Multispectral Imaging

Hyperspectral imaging (HSI) captures detailed reflectance information by collecting hundreds of narrow, contiguous spectral bands, spanning visible to short-wave infrared (SWIR) wavelengths. This enables detection of subtle physiological variations in crops, such as water stress, chlorophyll deficiency, or early disease presence [2, 13].

By contrast, multispectral imaging (MSI) acquires data in a limited number of wide spectral bands (typically 3–15), often corresponding to red, green, blue (RGB), near-infrared (NIR), and red-edge ranges. While MSI supports widely used vegetation indices such as NDVI and EVI, it lacks the spectral resolution needed to distinguish between spectrally similar vegetation types [5].

This rich spectral-spatial structure makes HSI suitable for classification, target detection, and anomaly identification tasks [18]. However, its high dimensionality also introduces computational and statistical challenges. Specifically, it increases memory consumption and training time, and exacerbates overfitting in machine learning models.

To mitigate these issues, dimensionality reduction methods such as Principal Component Analysis (PCA), Independent Component Analysis (ICA), and deep autoencoders are often applied prior to classification [7, 15].

Modern approaches to HSI classification rely on deep learning architectures—such as 2D/3D CNNs and hybrid transformers—that jointly capture spatial and spectral features [9, 8]. Still, they face difficulties stemming from spectral redundancy, class imbalance, and limited labeled data.

These limitations motivate the use of generative models to augment the training set with synthetic but realistic hyperspectral samples, particularly in low-resource settings.

3. Generative Modeling for HSI Classification

Hyperspectral image (HSI) classification is often hindered by two critical limitations: the scarcity of labeled data and the significant imbalance between common and rare classes. These issues are especially problematic in agricultural monitoring, where data collection is costly and class boundaries are often spectrally ambiguous.

To alleviate these challenges, Generative Adversarial Networks (GANs) have emerged as a viable solution. A classical GAN consists of a generator that synthesizes realistic data samples and a discriminator that distinguishes real from synthetic inputs. When extended with a class-conditioning mechanism via an auxiliary classifier, this framework forms the Auxiliary Classifier GAN (AC-GAN) [11], capable of producing labeled synthetic spectra.

However, training GANs in high-dimensional spectral domains is prone to instability and mode collapse. The Wasserstein GAN with Gradient Penalty (WGAN-GP) introduces improvements by replacing the Jensen–Shannon divergence with the Wasserstein-1 distance, while enforcing the Lipschitz constraint through gradient penalty [12]. This modification enhances convergence and training robustness.

By combining both approaches, the AC-WGAN-GP model provides a promising foundation for hyperspectral classification. It enables stable conditional generation by feeding the generator with Gaussian noise, PCA-reduced spectral vectors, and one-hot encoded class labels. The discriminator learns to distinguish real from fake samples, while the auxiliary classifier encourages semantic consistency in generated outputs.

Nonetheless, several shortcomings remain unresolved. One-hot labels do not capture inter-class similarity, limiting the generator's ability to model transitional spectra. The classifier's feedback to the generator is often underutilized, particularly in class-overlapping or data-sparse regions. Moreover, the architecture does not explicitly mitigate class imbalance, resulting in under-representation of minority categories.

To address these limitations, we propose a series of architectural and training refinements to the AC-WGAN-GP model. These include replacing one-hot labels with learnable class embeddings to reflect semantic proximity, incorporating residual deconvolution and cross-attention in the generator for enhanced spectral fidelity, and upgrading the discriminator with Layer Normalization and minibatch discrimination for better generalization. The classifier is restructured to jointly process both spectral features and label embeddings, enabling more discriminative and robust feature learning.

The following section presents the full specification of the improved model, along with a training strategy designed to avoid data leakage and promote effective representation of rare or spectrally ambiguous classes.

4. Problem Formulation of Hyperspectral Image Classification Using AC-WGAN-GP

In hyperspectral image classification tasks involving generative models, evaluation metrics play a crucial role in objectively comparing model performance and quantifying improvements resulting from architectural modifications. In this study, we employ three widely adopted metrics: Overall Accuracy (OA), Average Accuracy (AA), and the Cohen's Kappa coefficient (κ), which are standard in hyperspectral classification research [21, 20, 18].

While global metrics like OA, AA, and κ assess overall performance, per-class metrics—Precision, Recall, and F1-score—reveal how well individual classes are classified.

Overall Accuracy (OA) is a standard metric in HSI classification that measures the proportion of correctly predicted samples among all test samples:

$$OA = \frac{1}{N}\sum_{i=1}^{\bar{C}} h_{ii}, \tag{1}$$

where N is the total number of test samples, $\bar{C}$ is the number of classes, and h_{ii} represents correctly classified samples of class i (confusion matrix diagonal).

Average Accuracy (AA) evaluates classification performance across all classes equally, regardless of class size. It is calculated as the mean of per-class accuracies:

$$AA = \frac{1}{\bar{C}}\sum_{i=1}^{\bar{C}} \frac{h_{ii}}{N}, \tag{2}$$

where $\bar{C}$ is the number of classes, h_{ii} the correctly classified samples for class i, and N_i the total test samples in class i.

The Kappa coefficient (κ) measures agreement between predicted and true labels while accounting for chance. Unlike OA, it reflects class distribution, making it suitable for imbalanced datasets [21, 20]. It is computed as:

$$\kappa = \frac{N\sum_{i=1}^{\bar{C}} h_{ii} - \sum_{i=1}^{\bar{C}}(h_{i+} \cdot h_{+i})}{N^2 - \sum_{i=1}^{\bar{C}}(h_{i+} \cdot h_{+i})}, \tag{3}$$

where N is the total number of test samples, $\bar{C}$ the number of classes, h_{ii} correct predictions, h_{i+} actual counts, and h_{+i} predicted counts per class.

Precision is the proportion of correctly classified pixels of a given class among all pixels that the model has predicted as belonging to that class. The metric is defined as:

$$Precision_i = \frac{h_{ii}}{h_{+i}}, \tag{4}$$

where h_{ii} – number of samples of class i correctly classified as class i; h_{+i} – total number of samples predicted as class i (regardless of their true class).

Recall is the proportion of correctly classified pixels of a given class among all actual samples of that class. The metric is defined as:

$$Recall_i = \frac{h_{ii}}{h_{i+}}, \tag{5}$$

where h_{ii} — number of samples of class i correctly classified as class i; h_{i+} — total number of samples that truly belong to class i (ground truth labels).

The F1-score is the harmonic mean of precision and recall. It provides a balanced assessment of the classifier's performance, particularly when both false positives and false negatives are of concern. The metric is defined as:

$$F1_i = 2 \cdot \frac{Precision_i \cdot Recall_i}{Precision_i + Recall_i}, \tag{6}$$

where $Precision_i$ — precision for class i; $Recall_i$ — recall for class i.

5. Proposed Method

The proposed model builds upon the AC-WGAN-GP framework by introducing targeted improvements aimed at enhancing class-aware sample generation, increasing spectral diversity, and stabilizing training. The network still comprises three main components — generator (G), discriminator (D), and classifier (C) — but their internal architectures are modified to address key challenges such as class imbalance, spectral overlap, and limited supervision.

The overall structure of the proposed architecture is illustrated in Figure 1, highlighting the internal connections between the generator, discriminator, and classifier modules.

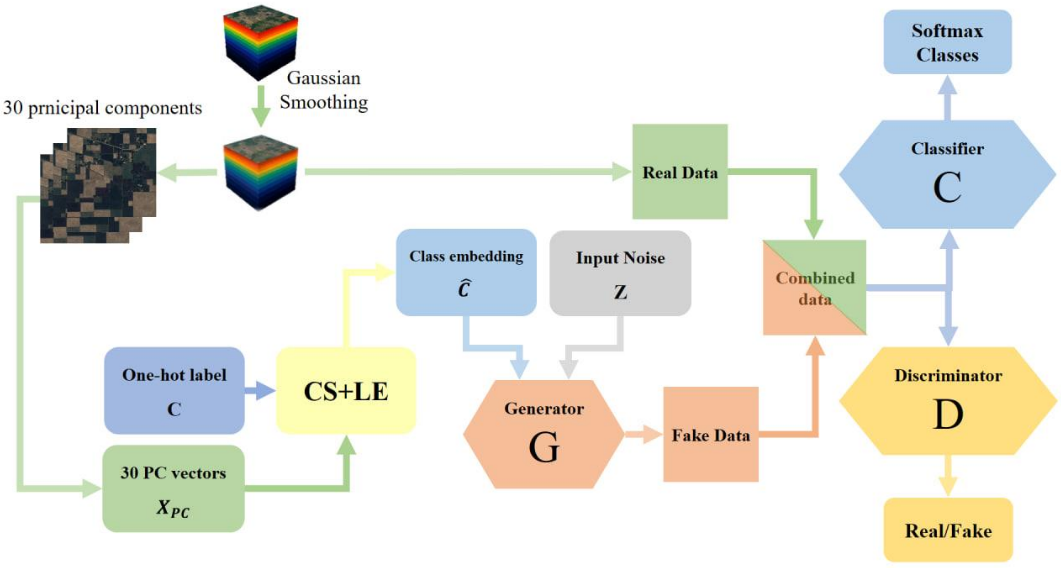

Figure 1: Enhanced architecture of the AC-WGAN-GP.

To capture more informative conditional inputs, we introduce a Class-aware Sampling and Label Embedding (CS+LE) module. Labels are encoded as dense vectors and concatenated with PCA-transformed spectral features and Gaussian noise. This allows the generator to operate in a more structured latent space, facilitating the creation of class-specific and spectrally consistent samples.

The generator architecture is extended with ResNet-style Deconv1D blocks and a cross-attention mechanism that aligns spectral and label embeddings with intermediate features. Spectral Dropout is applied to improve generalization by randomly zeroing entire spectral bands, mimicking sensor noise or occlusion.

In the discriminator, Batch Normalization is replaced with LayerNorm, ensuring stable gradients under gradient penalty regularization. To encourage diversity and mitigate mode collapse, a Minibatch Discrimination layer is included, allowing the model to detect and penalize overly similar samples.

The auxiliary classifier is redesigned with a deeper convolutional stack and uses embedded labels to better reflect inter-class relationships. It outputs both classification scores and internal features used in contrastive and alignment-based losses, promoting more compact and discriminative feature spaces.

To guide training, the loss functions for each module are expanded beyond standard adversarial objectives. The generator incorporates cosine similarity with class PCA centers, categorical loss, and alignment between features and label embeddings. The classifier combines class-weighted cross-entropy, contrastive separation, cosine alignment, and embedding regularization. These terms are weighted by tuned hyperparameters to ensure balanced optimization across classes and objectives.

All synthetic samples are generated in online mode and selected via spectral clustering from real training data. Only the most representative samples are used in training, maintaining separation from test data and ensuring reliable evaluation.

5.1. Architectural Comparison of Baseline and Improved Models

Tables 1 and 2 summarize the architectural differences between the baseline and improved AC-WGANGP models. They detail the input/output dimensions, layers, normalization, and activation functions for each module (G: Generator, D: Discriminator, C: Classifier).

Table 1
Improved architecture of AC-WGAN-GP with embedded labels and residual blocks

Module	№	Input Size	Layer	BN/LN	Stride	Padding	Activation
G	1	Z+PCA+embed	Dense + Reshape	BN	–	–	ReLU
	2	1/8H×1×256	Cross-Attn + Concat	–	–	–	–
	3	1/8H×1×320	ResNet Deconv1d (3×1×320×256)	BN	2×1	SAME	ReLU
	4	1/4H×1×256	ResNet Deconv1d (3×1×256×128)	BN	2×1	SAME	ReLU
	5	1/2H×1×128	ResNet Deconv1d (3×1×128×64)	BN	2×1	SAME	ReLU
	6	H×1×64	Upsampling+Conv1D (3×1×64×1)	–	–	SAME	Tanh
D	1	H×1×1	Conv1d (3×1×1×64)	–	2×1	SAME	LeakyReLU
	2	1/2H×1×64	Conv1d (3×1×64×128)	LN	2×1	SAME	LeakyReLU
	3	1/4H×1×128	Conv1d (3×1×128×256)	LN	2×1	SAME	LeakyReLU
	4	1/8H×1×256	Conv1d (3×1×256×512)	LN	2×1	SAME	LeakyReLU
	5	1/16H×1×512	Minibatch Discrimination	–	–	–	–
	6	1/16H×1×512	Flatten + Dense (1)	–	–	–	Linear
C	1	H×1×1	Conv1d (5×1×1×64)	BN	1×1	SAME	ReLU
	2	H×1×64	Conv1d (3×1×64×128)	BN	2×1	SAME	ReLU
	3	1/2H×1×128	Conv1d (3×1×128×256)	BN	2×1	SAME	ReLU
	4	1/4H×1×256	Flatten	–	–	–	–
	5	1/4H×256	Dense + Label Embed	BN	–	–	ReLU
	6	128	Dense (C)	–	–	–	Softmax

Table 2
Baseline architecture of AC-WGAN-GP

Module	№	Input Size	Layer	BN	Stride	Padding	Activation
G	1	100+30+class	Dense + Reshape	Yes	--	--	ReLU
	2	1/16H×1×512	Deconv1d (3×1×512×256)	Yes	2×1	SAME	ReLU
	3	1/8H×1×256	Deconv1d (3×1×256×128)	Yes	2×1	SAME	ReLU
	4	1/4H×1×128	Deconv1d (3×1×128×64)	Yes	2×1	SAME	ReLU
	5	1/2H×1×64	Deconv1d (3×1×64×1)	No	2×1	SAME	Tanh
D	1	H×1×1	Conv1d (3×1×1×64)	No	2×1	SAME	LeakyReLU
	2	1/2H×1×64	Conv1d (3×1×64×128)	Yes	2×1	SAME	LeakyReLU
	3	1/4H×1×128	Conv1d (3×1×128×256)	Yes	2×1	SAME	LeakyReLU
	4	1/8H×1×256	Conv1d (3×1×256×512)	Yes	2×1	SAME	LeakyReLU
	5	1/16H×1×512	Flatten + Dense	No	--	--	Linear
C	1	H×1×1	Conv1d (15×1×1×64)	No	--	SAME	Tanh
	2	1/15H×1×64	Flatten + Dense	No	--	--	Softmax

5.2. Sample Selection and Label Smoothing

To improve training stability and class balance, we implement a selective sampling approach for synthetic data. For each class, real samples are clustered using KMeans, and synthetic samples are selected based on cosine similarity to cluster centers. Both central and peripheral samples are chosen to ensure diversity. Labels are smoothed using a fixed coefficient ε.

```
Input: Real data $\mathcal{D}$, synthetic data $\mathcal{G}$, classes $C$, smoothing $\varepsilon$, ratio $r$
Output: Filtered samples $\mathcal{X}$, smoothed labels $\mathcal{Y}$
for each class $c$ = 1 to $C$ do
    Cluster $\mathcal{D}_c$ into $K$ clusters;
    foreach cluster do
        Measure similarity between $\mathcal{G}c$ and center;
        Select $r$% most and (1−$r$)% least similar samples;
        Append to $\mathcal{X}$ with label $c$;
    end
end
foreach label $y_i$ in $\mathcal{Y}$ do
    Smooth: $y_i^{smooth} = (1-\varepsilon) \cdot yi + \varepsilon/(C-1)$;
end
      return $\mathcal{X}$,$\mathcal{Y}$
```

Algorithm 1: Simplified sample selection and smoothing.

5.3. Loss Functions of the Improved AC-WGAN-GP

The improved AC-WGAN-GP architecture employs a multi-component loss formulation to enable efficient and stable training across all network modules. Each component of the loss not only incorporates core adversarial objectives common to classical GANs but also introduces domain-specific terms tailored to the challenges of hyperspectral classification.

5.3.1. Generator Loss

Unlike in traditional GANs, the generator in AC-WGAN-GP is optimized not only through adversarial feedback from the discriminator but also by enforcing alignment with class conditions and spectral context.

The basic Wasserstein loss component for the generator is given by:

$$L_{WGAN} = -E_{z,c}[D(G(z,c))], \tag{7}$$

where $E_{z,c}$ is the expectation over all possible combinations of latent noise z and conditional class label c; z denotes the latent noise vector; c is the conditional class label; $G(z, c)$ is the generated spectral sample; $D(G(z, c))$ is the "realism" score assigned by the discriminator.

1. Cosine Similarity with PCA Vectors (Cosine PCA Loss) – ensures that the generated spectrum aligns with the average PCA vector of its target class:

$$L_{PCA} = E[1 - cos(\hat{x}, x_{PCA})], \tag{8}$$

where $\hat{x}$ denotes the generated spectral sample and x_{PCA} is the PCA-transformed class vector.

2. Cosine Alignment Loss – enforces the classifier's internal feature representation f to align with the class embedding vector e:

$$L_{\text{align}} = E[1 - cos(f, e)], \tag{9}$$

where f denotes the feature obtained from the classifier, and e is the class embedding.

3. Categorical Cross-Entropy – penalizes the generator if the classifier fails to recognize the correct class of a generated sample:

$$L_{\text{ce}} = E_{z,c}\big[-\log P_{cls}\big(\, c \mid G(z, c)\,\big)\big], \tag{10}$$

where P_{cls} denotes the probability predicted by the classifier for class c.

The full generator loss is then defined as:

$$L_G = L_{WGAN} + \lambda_{PCA} \cdot L_{PCA} + \lambda_{a\text{lign}} \cdot L_{a\text{lign}} + \lambda_{\text{ce}} \cdot L_{\text{ce}}, \tag{11}$$

where λ_{PCA}, λ_{align}, and λ_{ce} are weighting coefficients that control the contribution of each loss component. These are tuned empirically based on data characteristics, class imbalance, and desired classification performance.

5.3.2. Discriminator Loss

In the AC-WGAN-GP framework, the discriminator functions as a critic that estimates the divergence between real and generated spectral samples. Unlike in classical GANs, where the discriminator performs binary classification, the WGAN formulation approximates the Wasserstein distance between real and synthetic distributions.

The discriminator loss is defined as:

$$L_D = E_{\tilde{x} \sim p(\text{g})}[D(\tilde{x})] - E_{x \sim p(x)}[D(x)] + \lambda \cdot L_{\text{gp}}, \tag{12}$$

$$L_{\text{gp}} = E_{\hat{x} \sim p(\hat{x})}[(\| \nabla_{\hat{x}} D(\hat{x}) \|_2 - 1)^2], \tag{13}$$

where p_g is the distribution of generated samples (from the generator); p_{data} is the distribution of real training samples; $\tilde{x}$ is a generated spectrum $G(z, c)$; x is a real spectral sample; $\hat{x}$ is a linear interpolation between x and $\tilde{x}$; λ is a hyperparameter controlling the weight of the gradient penalty term $\mathcal{L}_{\text{gp}}$ that ensures 1-Lipschitz continuity.

5.3.3. Classifier Loss

The auxiliary classifier in AC-WGAN-GP is responsible for both class prediction and learning discriminative features for regularization. Its loss function comprises several components aimed at maximizing classification accuracy while structuring the feature space.

1. Categorical Cross-Entropy (Class-Weighted) This standard classification loss is weighted to compensate for class imbalance:

$$L_{\mathrm{ce}} = -E_{x,y}[\omega_y \cdot \log P_{cls}(y \mid x)], \tag{14}$$

where x is the spectral sample, y is the true class label, P_{cls} $(y \mid x)$ is the predicted probability, and ω_y is the inverse class frequency weight.

2. Contrastive Loss This term promotes closeness of features from the same class and separation between features from different classes:

$$L_{contrast} = E_{i,j}\begin{cases} \left\|f_i - f_j\right\|^2, & if \quad y_i = y_j \\ \max(0, \left\|f_i - f_j\right\| - \delta)^2, & if \quad y_i \neq y_j \end{cases}, \tag{15}$$

where f_i, f_j are feature vectors and δ is a margin parameter.

3. Cosine Alignment Loss Aligns the feature vector with the corresponding class embedding:

$$L_{\mathrm{align}} = E_{x,y}[1 - cos(f(x), e_y)], \tag{16}$$

where $f(x)$ is the feature vector from the classifier and e_y is the embedding of class y.

4. Embedding Divergence Loss Regularizes class embeddings to prevent their collapse in latent space:

$$L_{div} = \sum_{i \neq j} \frac{1}{\left\|e_i - e_j\right\|^2 + \varepsilon}, \tag{17}$$

where e_i, e_j are embeddings of different classes, and ε is a small positive constant to avoid division by zero.

Total Classifier Loss:

$$L_{\mathrm{C}} = L_{ce} + \lambda_{contrast} \cdot L_{contrast} + \lambda_{align} \cdot L_{align} + \lambda_{\mathrm{div}} \cdot L_{div}, \tag{18}$$

where $\lambda_{contrast}$, λ_{align}, λ_{div} are hyperparameters controlling the contribution of each regularization component. These are selected empirically based on task complexity and class imbalance.

6. Results

6.1. Experimental Setup and Execution Specifics

All experiments were conducted using PyCharm Community Edition 2024.3.4 with Python 3.9 and the TensorFlow 2.19.0 framework. The development environment ran on Windows 10 and local machine specifications were as follows:

- Processor: Intel Core i3-10110U CPU
- RAM: 8 GB

Synthetic samples were generated in online mode without being saved to disk, reducing memory usage and preventing data duplication. Spectral vectors were reduced to H = 30 components using Principal Component Analysis (PCA). The training and test splits were performed with strict class separation, eliminating potential data leakage.

6.2. Analysis of Incremental Improvements in AC-WGAN-GP

Table 3 presents the stepwise impact of architectural enhancements in AC-WGAN-GP on hyperspectral image classification quality. Each modification step (from the baseline model to the inclusion of cosine alignment and embedding divergence loss functions) progressively improves OA, AA, and κ metrics across all datasets.

The most significant improvement is observed on the challenging Indian Pines dataset, where the introduction of the CSLE module (Step 2) raises OA to 67.74%. Adding ResNet-style deconvolutions with Spectral Dropout (Step 3), Minibatch Discrimination and Layer Normalization (Step 4) enhances training stability, particularly on KSC, where accuracy increases to 90.12%.

Incorporating Contrastive Loss (Step 5) and the final loss terms (Cosine Alignment + Embedding Divergence) delivers the highest performance, notably OA = 90.55% on Salinas and κ = 63.63 on Indian Pines. This confirms the effectiveness of the proposed systemic enhancements to both architecture and loss design.

Table 3 summarizes the incremental improvements achieved through architectural modifications.

Table 3
Impact of stepwise improvements on classification performance (Training ratio: 5%)

Step	Improvement	Salinas			Indian Pines			KSC		
		OA	AA	κ	OA	AA	κ	OA	AA	κ
1	Baseline AC-WGAN-GP	89.54	94.46	88.49	66.30	54.48	61.11	89.76	84.61	88.50
2	CSLE	89.73	94.78	88.46	67.74	56.61	63.10	89.81	84.26	88.64
3	ResNet+Spectral Drop.	89.76	94.83	88.54	67.89	56.82	63.28	90.05	85.53	89.18
4	Minibatch Disc. + LN	90.09	95.01	89.05	68.08	56.25	63.36	90.12	85.57	89.27
5	Contrastive Loss	90.44	95.13	89.45	68.10	56.31	63.56	90.16	86.18	89.42
6	Cosine Align. + Diverg.	90.55	95.20	89.50	68.60	58.09	63.63	90.62	86.30	89.55

6.3. Analysis of Incremental Improvements in AC-WGAN-GP

6.3.1. Classification Results Analysis: Salinas

Table 4 presents the classification results on the Salinas dataset using 1%, 5%, and 10% of training samples. A consistent improvement in metrics is observed with increasing training set size: OA increases from 88.70% to 91.27%, AA — from 92.90% to 95.36%, and κ — from 87.41 to 90.07.

Table 5 reports per-class classification metrics for the Salinas dataset using 5% of training samples. The highest F1-scores (above 99%) were achieved for classes with well-defined spectral structures — Stubble, Broccoli, Vineyard soil. In contrast, classes with high spectral variability, such as Untrained vineyard and Untreated vineyard, show lower results (F1 = 78.63% and 70.49%, respectively).

As shown in Figure 2, the classification results for the Salinas dataset visually confirm the effectiveness of the model, especially on classes with clear spectral signatures.

Table 4
Overall classification results (Salinas)

Train/Test	OA (%)	AA (%)	κ
1%	88.70	92.90	87.41
5%	90.55	95.20	89.50
10%	91.27	95.36	90.07

Table 5
Per-class classification metrics (Train 5%)

№	Class	Precision (%)	Recall (%)	F1 (%)	Train	Test
1	Brocoli green weeds 1	100	99.63	99.82	100	1909
2	Brocoli green weeds 2	99.63	99.89	99.76	186	3540
3	Fallow	94.59	99.63	97.04	98	1878
4	Fallow rough plow	98.13	99.17	98.65	69	1325
5	Fallow smooth	98.43	98.35	98.39	133	2545
6	Stubble	99.92	99.79	99.85	197	3762
7	Celery	98.69	99.41	99.05	178	3401
8	Grapes untrained	82.33	75.25	78.63	563	10708
9	Soil vinyard develop	98.89	99.90	99.39	310	5893
10	Corn senesced green weeds	96.06	94.70	95.38	163	3115
11	Lettuce romaine 4wk	98.01	92.41	95.13	53	1015
12	Lettuce romaine 5wk	95.86	99.78	97.78	96	1831
13	Lettuce romaine 6wk	98.39	98.28	98.33	45	871
14	Lettuce romaine 7wk	97.35	93.90	95.60	53	1017
15	Vinyard untrained	66.81	74.58	70.49	363	6905
16	Vinyard vertical trellis	99.24	98.54	98.89	90	1717

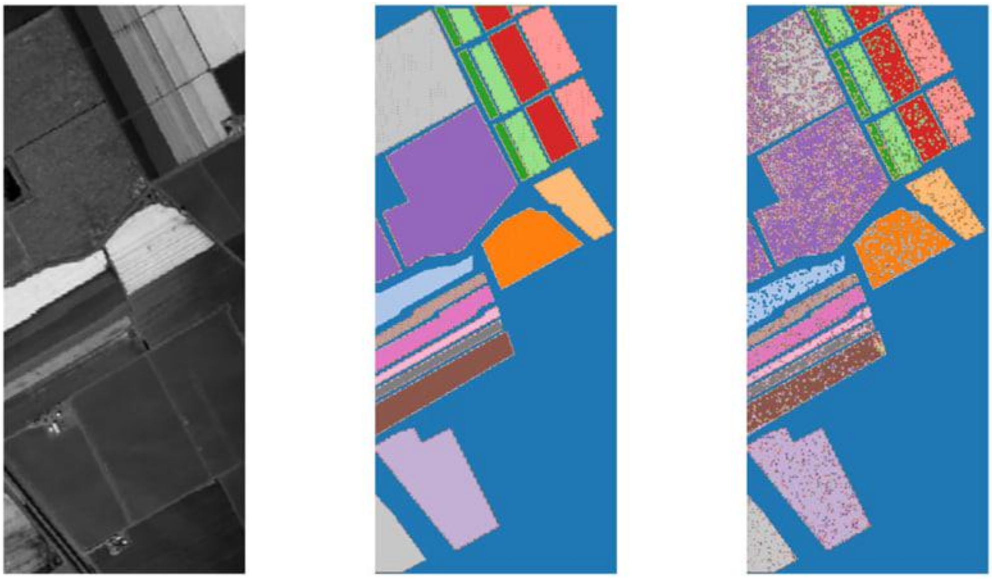

Figure 2: Prediction map for the Salinas dataset (OA = 91.09%). Left to right: original spectral band, ground truth (GT), classification result.

6.3.2. Classification Results Analysis: Indian Pines

The Indian Pines dataset is one of the most challenging due to strong spectral overlap between classes and significant class imbalance. Table 6 shows a steady improvement in accuracy as the training set size increases: overall accuracy (OA) rises from 54.15% to 77.54%, average accuracy (AA) from 43.24% to 71.15%, and the kappa coefficient (κ) from 46.78 to 74.35.

Table 7 shows per-class classification performance for Indian Pines with 5% training data. High F1-scores were achieved for classes with well-defined spectral profiles: Hay-windrowed (92.05%), Wheat (85.97%), Woods (88.80%), and Grass-trees (83.52%). In contrast, classes with a low number of training samples, such as Oats, Alfalfa, and Grass-pasture-mowed, showed lower F1 performance, ranging from 15% to 65%. Figure 3 demonstrates the prediction performance on the Indian Pines dataset, where spectral overlap and class imbalance make classification particularly challenging.

Particularly difficult were the Corn and Soybean-clean classes, where F1 did not exceed 40–50% due to spectral similarity with nearby crop types. At 5% of training data, the improved model achieves OA = 68.60%, AA = 58.09%, demonstrating stable classification for well-separated classes but with limitations on spectrally overlapping ones.

Table 6
Overall classification results (Indian Pines)

Train/Test	OA (%)	AA (%)	κ
1%	54.15	43.24	46.78
5%	68.60	58.09	63.63
10%	74.07	63.82	70.24
15%	74.88	65.14	71.13
20%	77.54	71.15	74.35

Table 7
Per-class classification metrics (Train 5%)

№	Class	Precision (%)	Recall (%)	F1 (%)	Train	Test
1	Alfalfa	50.00	9.09	15.38	2	44
2	Corn-notill	58.40	62.27	60.27	71	1357
3	Corn-mintill	59.20	44.87	51.05	41	789
4	Corn	57.14	21.24	30.97	11	226
5	Grass-pasture	88.30	69.06	77.51	24	459
6	Grass-trees	77.76	90.20	83.52	36	694
7	Grass-pasture-mowed	87.50	51.85	65.12	1	27
8	Hay-windrowed	86.77	98.02	92.05	23	455
9	Oats	50.00	10.53	17.39	1	19
10	Soybean-notill	69.44	54.11	60.83	48	924
11	Soybean-mintill	62.15	78.83	69.50	122	2333
12	Soybean-clean	53.00	29.79	38.14	29	564
13	Wheat	75.98	98.97	85.97	10	195
14	Woods	82.85	95.67	88.80	63	1202
15	Buildings-Grass-Trees-Drives	54.64	27.25	36.36	19	367
16	Stone-Steel-Towers	88.64	87.64	88.14	4	89

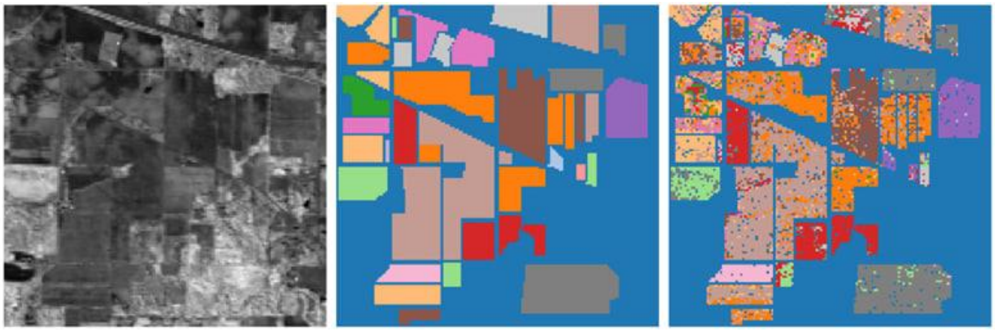

Figure 3: Prediction map for the Indian Pines dataset (OA = 68.96%). From left to right: original spectral channel, ground truth (GT), classification result.

6.3.3. Classification Results Analysis: KSC

The KSC dataset is characterized by clearly defined spectral differences between classes, which facilitates high classification performance. As shown in Table 8, even with only 5 % of training samples, the overall accuracy (OA) reaches 90.62%, the average accuracy (AA) is 86.30%, and the Cohen's kappa coefficient (κ) is 89.55.

Table 7 shows per-class classification performance for KSC with 5% training data. The highest F1-scores are observed for classes with stable spectral characteristics: Salt Marsh (98.02%), Bare Soil (90.52%), Reed Swamp (94.61%), and Water (98.44%).

Lower classification performance is observed for Hardwood Forest (66.19%) and Oak Forest (70.46%), which is partly due to the limited number of training samples (8–11) and the spectral similarity to other forest types.

The results demonstrate that the improved model is capable of accurately classifying most classes even with minimal training data, achieving F1-scores above 85% for 10 out of 13 classes.

As shown in Figure 4, the KSC dataset classification map illustrates high accuracy for classes with well-separated spectral features.

Table 8

Overall classification results (KSC)

Train/Test	OA (%)	AA (%)	κ
1%	80.71	70.73	78.50
5%	90.62	86.30	89.55
10%	93.06	89.46	92.27

Table 9

Per-class classification metrics (Train 5%)

№	Class	Precision (%)	Recall (%)	F1 (%)	Train	Test
1	Scrub	94.76	92.53	93.63	38	723
2	Willow Swamp	85.71	93.51	89.44	12	231
3	Evergreen Forest	82.31	93.44	87.52	12	244
4	Pine Forest	70.16	75.42	72.69	12	240
5	Oak Forest	77.34	64.71	70.46	8	153
6	Hardwood Forest	69.35	63.30	66.19	11	218
7	Marsh	82.61	76.00	79.17	5	100
8	Grass Marsh	85.65	88.78	87.19	21	410
9	Salt Marsh	93.40	97.37	95.34	26	494
10	Reed Swamp	95.49	93.75	94.61	20	384
11	Saline Marsh	96.82	99.25	98.02	20	399
12	Bare Soil	98.28	83.89	90.52	25	478
13	Water	96.92	100	98.44	46	881

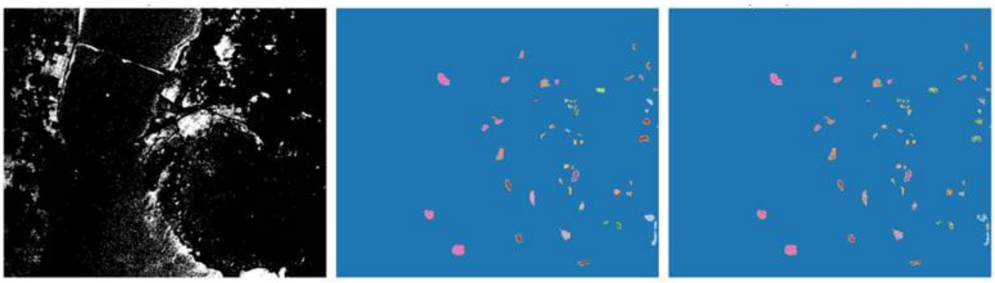

Figure 4: Prediction map for the KSC dataset (OA = 91 02%). From left to right: original spectral channel, ground truth (GT), and classification result.

7. Conclusions

In this work, we proposed an intelligent system for hyperspectral image classification based on the AC-WGAN-GP architecture, aimed at improving classification under conditions of data imbalance, spectral overlap between classes, and unstable training. The proposed improvements encompass all major components of the model: generator, discriminator, and classifier.

The generator was enhanced through the use of class-aware sampling and label embeddings, allowing better representation of minority classes. Its architecture is based on ResNet-inspired deconvolutional blocks with cross-attention mechanisms and spectral dropout, ensuring greater sample diversity and reduced risk of mode collapse. The generator input includes latent noise, a PCA-transformed spectral vector, and a dense class embedding.

The discriminator was adapted for stable WGAN-GP training by replacing Batch Normalization with Layer Normalization and introducing Minibatch Discrimination. This enables the detection of sample duplications and increases robustness against repetitive patterns in synthetic data. Additionally, the gradient penalty mechanism was implemented to enforce the 1-Lipschitz continuity requirement.

The classifier was modernized by replacing one-hot labels with dense embeddings and incorporating several loss functions: weighted categorical cross-entropy, contrastive loss, and embedding divergence. This enhanced the structure of the feature space, improved sensitivity to spectrally similar classes, and enabled better adaptation to rare cases.

Special attention was paid to fair evaluation: clustering and synthetic sample selectionwere performed strictly on the training set without access to test data. This eliminates any possibility of data leakage and guarantees the reliability of the reported metrics, even under stricter conditions than commonly used in related works.

The results confirm the effectiveness of the proposed intelligent system, particularly in classifying rare categories and under conditions of limited training data. Future research will aim to extend the system to 3D hyperspectral objects, segmentation tasks, and multisensor fusion, as well as to explore its potential in practical applications such as vegetation health monitoring and environmental assessment.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, J. Chanussot, Hyperspectral remote sensing data analysis and future challenges, IEEE Geoscience and Remote Sensing Magazine 1 (2013). doi:10.1109/MGRS.2013.2244672.

[2] P. Ghamisi, J. Plaza, Y. Chen, J. Li, A. Plaza, Advanced spectral classifiers for hyperspectral images: A review, IEEE Geoscience and Remote Sensing Magazine 5 5 1 (2017) 8–32. doi: 10.1109/MGRS.2016.2616418

[3] X. X. Zhu, D. Tuia, L. Mou, G.-S. Xia, L. Zhang, F. Xu, F. Fraundorfer, Deep learning in remote sensing: A comprehensive review and list of resources, IEEE Geoscience and Remote Sensing Magazine 5 (2017). doi:10.1109/MGRS.2017.2762307.

[4] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, J. A. Benediktsson, Deep learning for hyperspectral image classification: An overview, IEEE Transactions on Geoscience and Remote Sensing 57 (2019) 6690–6709. doi:10.1109/TGRS.2019.2907932.

[5] B. Lu, P. D. Dao, J. Liu, Y. He, J. Shang, Recent advances of hyperspectral imaging technology and applications in agriculture, Remote Sensing 12 (2020) 2659. doi:10.3390/rs12162659.

[6] B. G. Ram, P. Oduor, C. Igathinathane, K. Howatt, X. Sun, A systematic review of hyperspectral imaging in precision agriculture: Analysis of its current state and future prospects, Computers and Electronics in Agriculture 222 (2024). doi:10.1016/j.compag.2024.109037.

[7] H. Yu, B. Kong, Y. Hou, X. Xu, T. Chen, X. Liu, A critical review on applications of hyperspectral remote sensing in crop monitoring, Experimental Agriculture 58 (2022). doi:10.1017/S0014479722000278.

[8] V. Sineglazov, A. Kot, Design of hybrid neural networks of the ensemble structure, Eastern-European Journal of Enterprise Technologies 1 (2021) 31–45. doi:10.15587/1729-4061.2021.225301.

[9] M. Zgurovsky, V. Sineglazov, E. Chumachenko, Classification and analysis topologies known artificial neurons and neural networks, in: Artificial Intelligence Systems Based on Hybrid Neural Networks, volume 904 of Studies in Computational Intelligence, Springer, Cham, 2021.doi:10.1007/978-3-030-48453-8_1.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Advances in Neural Information Processing Systems, volume 27, 2014. doi:10.48550/arXiv.1406.2661.

[11] A. Odena, C. Olah, J. Shlens, Conditional image synthesis with auxiliary classifier gans, arXiv preprint arXiv:1610.09585 (2016). doi:10.48550/arXiv.1610.09585.

[12] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, A. C. Courville, Improved training of wasserstein gans, in: Advances in Neural Information Processing Systems, vol. 30, 2017, pp. 5767–5777. doi:10.48550/arXiv.1704.00028.

[13] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein GAN, arXiv preprint arXiv:1701.07875 (2017). doi:10.48550/arXiv.1701.07875.

[14] A. Abbas, S. Jain, M. Gour, S. Vankudothu, Tomato plant disease detection using transfer learnin with C-GAN synthetic images, Computers and Electronics in Agriculture 187 (2021). doi:10.1016/j.compag.2021.106279.

[15] C. Sun, X. Zhang, H. Meng, X. Cao, J. Zhang, AC-WGAN-GP: Generating labeled samples for improving hyperspectral image classification with small-samples, Remote Sensing 14 (2022) 4910. doi:10.3390/rs14194910.

[16] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, arXiv preprint arXiv:1511.06434 (2016). doi:10.48550/arXiv.1511.06434.

[17] Y. Zhan, D. Hu, Y. Wang, X. Yu, Semisupervised Hyperspectral image classification based on generative adversarial networks, IEEE Geoscience and Remote Sensing Letters 15 (2018) 212–216. doi:10.1109/LGRS.2017.2780890.

[18] Z. He, K. Xia, P. Ghamisi, Y. Hu, S. Fan, B. Zu, Hypervitgan: Semisupervised GAN with transformer for HSI classification, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 15 (2022). doi:10.1109/JSTARS.2022.3192127.

[19] M. Zgurovsky, V. Sineglazov, E. Chumachenko, Classification and analysis of multicriteria optimization methods, in: Artificial Intelligence Systems Based on Hybrid Neural Networks, volume 904 of Studies in Computational Intelligence, Springer, Cham, 2021. doi:10.1007/978-3-030-48453-8_2.

[20] Y. Huang, Z. Chen, J. Liu, Limited agricultural spectral dataset expansion based on generative adversarial networks, Computers and Electronics in Agriculture 215 (2023). doi:10.1016/j.Compag.2023.108385.

[21] L. Zhu, Y. Chen, P. Ghamisi, J. A. Benediktsson, Generative adversarial networks for hyperspectral image classification, IEEE Transactions on Geoscience and Remote Sensing 56 (2018) 5046–5063. doi:10.1109/TGRS.2018.2805286.

[22] J. Feng, N. Zhao, R. Shang, X. Zhang, L. Jiao, Self-supervised divide-and-conquer generative adversarial network for classification of hyperspectral images, IEEE Transactions on Geoscience and Remote Sensing 60 (2022) 1–17. doi:10.1109/TGRS.2022.3202908.

[23] Y. Lu, D. Chen, E. Olaniyi, Y. Huang, Gans for image augmentation in agriculture: A systematic review, Computers and Electronics in Agriculture 200 (2022). doi:10.1016/j.compag.2022.107208.

[24] M. Zgurovsky, V. Sineglazov, E. Chumachenko, Formation of hybrid artificial neural networks topologies, in: Artificial Intelligence Systems Based on Hybrid Neural Networks, volume 904 of Studies in Computational Intelligence, Springer, Cham, 2021. doi:10.1007/978-3-030-48453-8_3.

[25] V. M. Sineglazov, K. D. Riazanovskiy, O. I. Chumachenko, Multicriteria conditional optimization based on genetic algorithms, System Research and Information Technologies (2020). doi:10.20535/SRIT.2308-8893.2020.3.07.