Chapter 5

Lightweight Framework for HSI Classification via Convolution Kaiming-Gaussian Focused Linear Transformer Network

5.1 Introduction

HS has become a cornerstone technology in diverse fields such as remote sensing, precision agriculture, environmental monitoring, and medical diagnostics [164–166]. Unlike multispectral systems, HSI captures hundreds of contiguous spectral bands. This enables precise material differentiation [167, 168]. However, high spectral resolution leads to data redundancy and increased computational complexity. It also requires advanced feature extraction techniques [169, 170]. Over the years, DL has driven substantial advancements in HSI classification. Initially, 1D-CNNs [171] were employed for spectral feature extraction, while 2D-CNNs [137] captured spatial characteristics. Later, hybrid models such as MDCNN, as discussed in Chapter 3, integrated 3D-2D convolutions and mathematical morphology to improve classification accuracy. Despite these advancements, CNN-based models remain inherently limited by their local receptive fields, preventing them from effectively capturing long-range dependencies. As a result, their generalization capability remains suboptimal for complex HSI scenarios.

To capture long-range dependencies, researchers introduced ViT-based architectures like LogGroupFormer in Chapter 4. It integrates CNN with ViT to model both local and global contexts. However, traditional ViTs rely on self-attention with quadratic complexity, making them computationally expensive for large-scale HSI analysis. This

trade-off between accuracy and efficiency requires novel optimization techniques.
This chapter presents the Convolution-Kaiming-Gaussian Focused Linear Network (CKGFLNet), a hybrid framework for efficient HSI classification. Unlike conventional transformers, CKGFLNet introduces Kaiming-Gaussian Focused Linear Attention (KGFLA). It reduces computational complexity from $\mathcal{O}(N^2d)$ to $\mathcal{O}(Nd^2)$, where $d \ll N$, ensuring faster processing with high accuracy. KGFLA optimizes attention computation while preserving classification performance. Additionally, Kaiming and Gaussian initialization enhance gradient stability, prevent overfitting, and improve feature diversity. This makes CKGFLNet both efficient and robust. The primary contributions of this research are as follows:

1. **Efficient Hybrid Feature Learning:** CKGFLNet integrates CNN-based local feature extraction with KGFLA for effective global contextual modeling.
2. **Optimized Transformer Complexity:** KGFLA module reduces the computational burden associated with traditional self-attention, reduces time complexity from $O(N^2d)$ to $O(Nd^2)$, where $d << N$, ensuring scalability for large datasets.
3. **Enhanced Model Stability:** We propose Kaiming-Gaussian initialization that improves gradient flow, accelerates convergence and enhances generalization.

In this chapter, we have provided details of the our proposed architecture, optimization strategies, and experimental validation. The remainder of this chapter is structured as follows: Section 5.2 explores recent advancements in HSI classification, analyzing key methodologies and their associated challenges. Section 5.3 presents the problem statement, highlighting the limitations of existing approaches and motivating the need for an efficient hybrid framework. Section 5.4 introduces the proposed CKGFLNet, detailing its architectural innovations and computational optimizations. Section 5.5 presents the experimental setup, comparative analysis, and performance evaluation against existing models. Finally, Section 5.6 summarizes key findings, discusses their implications, and outlines potential future directions for improving HSI classification.

5.2 Related Work

5.2.1 Convolution-Based Approaches

Existing DL models for HSI classification mainly use CNN-based architectures. Early methods adopt 1D-CNNs to extract spectral features [172]. Later, 2D-CNNs target spatial information [145]. These models fail to capture spectral-spatial correlations jointly. To solve this, 3D-CNNs are proposed [95], allowing integrated spectral-spatial feature extraction. Hybrid models like MorphConvHyperNet [173] and MDCNN [174]

combine 2D and 3D convolutions to enhance spatial learning. However, CNNs still suffer from limited receptive fields. They struggle to model long-range dependencies, which affects classification performance.

5.2.2 Vision Transformer-Based Approaches

ViT models gain popularity in HSI classification due to their strength in capturing long-range dependencies and global context. These models overcome the local receptive field limitations of CNNs. ViTs use self-attention to model spectral-spatial correlations effectively. For example, [175] applies transfer learning and label smoothing to improve robustness on small datasets. Expanding the above concepts, [119] proposes a transformer with spectral embeddings and hierarchical skip connections to enhance accuracy. Later models like SSFTT [124], IFormer [163], and LGSA-ViT [125] use advanced attention strategies to improve spectral-spatial representation. Hybrid frameworks such as LogGroupFormer (Chapter 4) combine CNNs for local features with transformers for global modeling. These models achieve higher accuracy. However, most suffer from high computational costs due to the quadratic self-attention complexity, $\mathcal{O}(N^2d)$. This limits their scalability. To address this, Han et al. [176] propose the Flatten Transformer. It uses Focused Linear Attention to reduce complexity to $\mathcal{O}(Nd^2)$, where $N \gg d$. This makes processing more efficient, with only a slight drop in accuracy.

Motivated by these advancements, we propose CKGFLNet. It is a lightweight transformer-based framework. CKGFLNet integrates a novel KGFLA module. KGFLA reduces attention complexity to $\mathcal{O}(Nd^2)$. This makes the model scalable and suitable for real-time HSI classification. It also maintains high accuracy while improving efficiency.

5.2.3 Initialization Techniques

Effective weight initialization plays a crucial role in stabilizing training and speeding up convergence. Gaussian initialization [177,178] adds controlled randomness. It enhances feature diversity and reduces overfitting. Kaiming initialization [179] ensures stable gradient flow and supports efficient learning. [124] apply Gaussian feature weighting. It improves global feature extraction and reduces computational cost. Inspired by these approaches, we designed CKGFLNet with KGFLA. This module combines the benefits of Gaussian and Kaiming initialization. As a result, CKGFLNet achieves better training stability, generalization, and efficient HSI classification.

5.3 Problem Statement

Given a hyperspectral image (HSI) cube $\mathbf{I} \in \mathbb{R}^{[H \times W \times B]}$, where H, W, and B denote the spatial height, width, and the number of spectral bands, respectively, the goal is to design a classification framework that efficiently captures both local and global spectral-spatial dependencies with reduced computational cost. To achieve this, we define two convolutional feature extraction functions: $F_s : \mathbf{I} \rightarrow \mathbb{R}^{[H \times W \times \mathfrak{K}]}$ for capturing spectral-spatial features via 3D convolution, and $F_{sp} : \mathbf{I} \rightarrow \mathbb{R}^{[H \times W \times \mathfrak{K}]}$ for capturing spatial textures using 2D convolution. These modules extract hierarchical and fine-grained features from the raw HSI cube.

The combined feature map is then tokenized using a patch-based tokenizer function $F_{\text{Token}} : \mathbb{R}^{[H \times W \times \mathfrak{K}]} \rightarrow \mathbb{R}^{[N \times d]}$, where N is the number of tokens and d is the token dimension. To preserve spatial structure, we apply learnable positional encoding $F_{\text{Pos}} : \mathbb{R}^{[N \times d]} \rightarrow \mathbb{R}^{[N \times d]}$ to the tokens. For capturing global contextual dependencies efficiently, we introduce the KGFLA module $F_{\text{KGFLA}} : \mathbb{R}^{[N \times d]} \rightarrow \mathbb{R}^{[N \times d]}$, which is a low-complexity alternative to standard self-attention, reducing computation to $\mathcal{O}(Nd^2)$. The encoded tokens are then passed through the classification head $F_{\text{classify}} : \mathbb{R}^{[N \times d]} \rightarrow \mathbb{R}^{[C]}$, where C is the number of output classes.

Our objective is to maximize the classification accuracy while maintaining computational efficiency. The problem is formulated as:

$$\begin{aligned} \underset{F_s, F_{sp}, F_{\text{Token}}, F_{\text{Pos}}, F_{\text{KGFLA}}, F_{\text{classify}}}{\text{maximize}} \quad & \hat{\mathcal{C}}(F_{\text{classify}}(F_{\text{KGFLA}}(F_{\text{Pos}}(F_{\text{Token}}(F_{\text{integrate}}(F_s(\mathbf{I}), \\ & F_{sp}(\mathbf{I})))))) - \lambda \cdot \hat{\mathcal{L}}_{\text{comp}}(\mathbf{I}), \end{aligned} \tag{5.1}$$

where $\hat{\mathcal{C}}$ denotes the classification accuracy, and λ is a trade-off parameter that balances accuracy and computational cost $\hat{\mathcal{L}}_{\text{comp}}$.

5.4 Proposed Methodology

The proposed Convolution-Kaiming-Gaussian Focused Linear Network (CKGFLNet) is a four-stage architecture designed for efficient and accurate hyperspectral image (HSI) classification. Figure 5.1 illustrates the overall workflow of the framework, which consists of two main modules: (a) a Hybrid Spectral–Spatial Feature Extractor and Tokenizer, and (b) a KGFLA-Transformer Block. In CKGFLNet, feature selection begins with PCA-based dimensionality reduction to eliminate spectral redundancy. The reduced cube is divided into spectral–spatial patches, which are processed by a hybrid 3D–2D convolutional extractor. Here, 3D convolutions capture joint spectral–spatial

features, while 2D convolutions refine spatial details. The resulting feature maps are then tokenized into compact representations, forming the selected features for subsequent Transformer-based global modeling. The methodology can be summarized in the following steps:

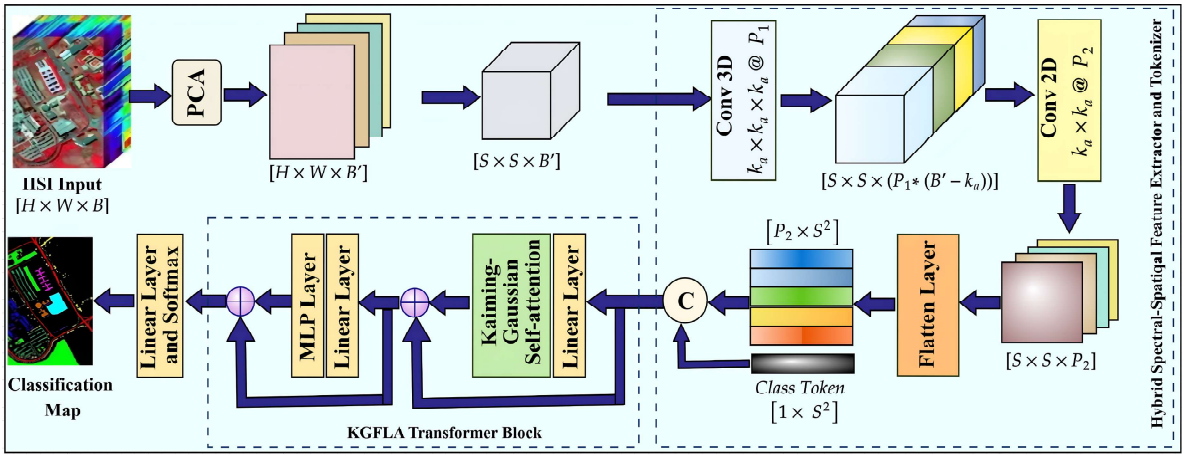


Figure 5.1: Illustration of the Convolution-Kaiming-Gaussian Focused Linear Network (CKGFLNet) architecture for HSI classification consists of two main modules: Hybrid Spectral-Spatial Feature Extractor module and a KGFLA-Transformer Block.

1. **HSI Representation:** Represent HSI as a hypercube $\mathbf{I} \in \mathbb{R}^{[H \times W \times B]}$, where H, W, and B denote height, width, and number of spectral bands.
2. **Dimensionality Reduction and Patch Extraction:** Apply PCA to reduce spectral redundancy while retaining essential features. Generate spectral–spatial patches from the reduced cube to preserve local contextual information.
3. **Hybrid Spectral–Spatial Feature Extractor and Tokenizer:** Use a 3D–2D convolutional structure to jointly learn spectral and spatial representations. The resulting feature maps are tokenized for attention-based processing.
4. **KGFLA Transformer Module:** At the heart of CKGFLNet lies the KGFLA Transformer module, which introduces a Flattened Linear Focused Multi-Head Attention mechanism [176]. This module effectively models long-range dependencies while significantly reducing computational overhead from $\mathcal{O}(N^2d)$ to $\mathcal{O}(Nd^2)$.
5. **Initialization Strategy:** Integrate Kaiming and Gaussian initialization within the attention module to promote stable training, improve feature diversity, and mitigate overfitting.
6. **Classification:** Feed the refined feature representations into a Softmax classifier to map them into class probabilities for final HSI classification.

5.4.1 Hybrid Spectral-Spatial Feature Extractor and Tokenizer Module

The proposed module is illustrated in Algorithm 5.1, which combines 3D convolution for joint spectral-spatial extraction and 2D convolution for spatial refinement. The 3D convolution preserves spectral continuity. The 2D convolution enhances spatial context. The process starts with an HSI cube $I \in \mathbb{R}^{[H\times W\times B]}$. PCA reduces the spectral dimension to obtain $I_{\text{pca}} \in \mathbb{R}^{[H\times W\times B']}$. From this, spatial patches of size $S \times S$ are extracted to form $I_{\text{patch}} \in \mathbb{R}^{[S\times S\times B']}$. These patches pass through a 3D convolutional layer, producing feature maps $I_{\text{3D}} \in \mathbb{R}^{[S\times S\times (B'-K_a+1)\times P_1]}$. The output is reshaped to $I_{\text{3Dn}} \in \mathbb{R}^{[S\times S\times ((B'-K_a+1)\cdot P_1)]}$. A 2D convolution is then applied to get $I_{\text{2D}} \in \mathbb{R}^{[S\times S\times P_2]}$. The features are flattened into a token sequence $T_1 \in \mathbb{R}^{[S^2\times P_2]}$. A learnable class token $C_T \in \mathbb{R}^{[1\times P_2]}$ is appended. The final token matrix becomes $T_2 \in \mathbb{R}^{[(S^2+1)\times P_2]}$ which is now ready for Transformer encoding.

Algorithm 5.1: Hybrid Spectral-Spatial Feature Extractor and Tokenizer

Input: HSI Cube $I \in \mathbb{R}^{[H\times W\times B]}$
Output: Token Matrix $T_2 \in \mathbb{R}^{[(S^2+1)\times P_2]}$

Step 1: Extract spatial patches of size $S \times S$
  $I_{pca} \leftarrow PCA(I), \quad I_{pca} \in \mathbb{R}^{[H\times W\times B']}$
  $I_{patch} \leftarrow ExtractPatches(I_{pca}, S \times S), \quad I_{patch} \in \mathbb{R}^{[S\times S\times B']}$
Step 2: Spectral-Spatial Feature Extraction using 3D Convolution Layer
  $I_{3D} \leftarrow \textbf{3DConv}(I_{patch}, K_a, P_1), \quad I_{3D} \in \mathbb{R}^{[S\times S\times (B'-K_a+1)\times P_1]}$
  $I_{3Dn} \leftarrow \textbf{Reshape}(I_{3D}), \quad I_{3Dn} \in \mathbb{R}^{[S\times S\times ((B'-K_a+1)\cdot P_1)]}$
Step 3: Contextual Spatial Features using 2D Convolution Layer
  $I_{2D} \leftarrow \textbf{2DConv}(I_{3Dn}, K_b, P_2), \quad I_{2D} \in \mathbb{R}^{[S\times S\times P_2]}$
Step 4: Tokenizer
  $T_1 \leftarrow Flatten(I_{2D}), \quad Tokens \in \mathbb{R}^{S^2\times P_2}$
  Class Token Concatenation
  $T_2 \leftarrow \textbf{Concat}[T_1, C_T], \quad C_T \in \mathbb{R}^{[1\times P_2]}, \quad T_2 \in \mathbb{R}^{[(S^2+1)\times P_2]}$
Return: Token Matrix T_2

5.4.2 Kaiming-Gaussian Focused Linear Attention Transformer Block

Algorithm 5.2 presents the KGFLA Transformer Block. It outlines the steps to generate class predictions from the token matrix $T_2 \in \mathbb{R}^{[(S^2+1)\times P_2]}$. First, the input tokens pass through Layer Normalization to stabilize training and normalize feature distributions. Next, the normalized tokens are divided into multiple heads. These heads are processed in parallel within the MHSA module. MHSA module is the KGFLA sub-module, as shown in Figure 5.2. Each head computes its respective (Q), (K), and (V) matrices, enhanced by Kaiming He initialization (Km) and Gaussian noise injection for better

convergence and regularization. The refined query $\hat{Q}_i$ is passed through a Softmax

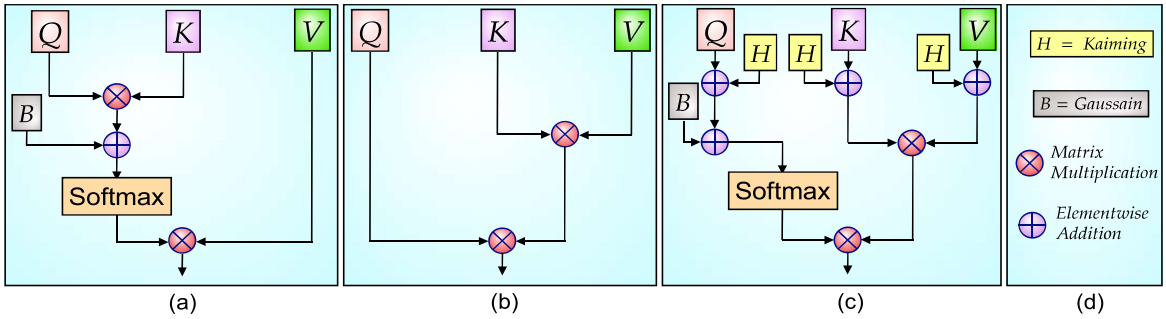


Figure 5.2: Visualization of different transformers for HSI classification. (a) Traditional model with Gaussian. (b) Flatten linear transformer. (c) CKGFLNet incorporates both Kaiming and Gaussian initialization for enhanced performance.

Algorithm 5.2: KGFLA Transformer Block

Input: Token Matrix $T_2 \in \mathbb{R}^{[(S^2+1)\times P_2]}$
Output: Predicted Class Probabilities $\hat{Y}$

Step 1: Apply Layer Normalization to Input
$T' \leftarrow LayerNormalization(T_2)$
Step 2: Multi-Head Kaiming-Gaussian Attention
$T' = [T'^{(1)}, T'^{(2)}, \ldots, T'^{(h)}], \quad T'^{(i)} \in \mathbb{R}^{[(S^2+1)\times(P_2/h)]}$
for $i = 1$ *to* h **do**
  $Q_i \leftarrow T'^{(i)}W_q^{(i)} + Km(X_q^{(i)}), \quad K_i \leftarrow T'^{(i)}W_k^{(i)} + Km(X_k^{(i)}), \quad V_i \leftarrow T'^{(i)}W_v^{(i)} + Km(X_v^{(i)})$
  $\hat{Q}_i \leftarrow Q_i + GuInit(W_g^{(i)}), \quad Q'_i \leftarrow Softmax(\hat{Q}_i)$
  $Z_i \leftarrow K_i^T V_i$
  $head_i \leftarrow Q'_i Z_i$
end
$MHSA\_out \leftarrow$ **Concat**$[head_1, head_2, \ldots, head_h]W_O$
Step 3: Residual Connection + Feedforward MLP block
$\hat{O} \leftarrow LayerNormalization(MHSA\_out + T_2)$
$F_1 \leftarrow GeLU(\hat{O}W_1 + b_1)$
Step 4: Classification Layer
$O_{fc} \leftarrow LayerNormalization(F_2 + \hat{O})$
$\hat{Y} \leftarrow Softmax(O_{fc})$
Return: Predicted Class Probabilities $\hat{Y}$

to obtain the attention scores Q'_i, which are then used to weight the intermediate representation $Z_i = K_i^T V_i$, ensuring efficient attention computation. The outputs from all attention heads are concatenated and linearly projected using W_O to form the final attention output $MHSA\_out$. This is followed by a residual connection that merges the MHSA output with the original input T_2, after which another Layer Normalization is applied. MLP with GeLU activation generates the intermediate feature F_1, which is

again refined via a residual connection. The resulting output is passed through a final FC layer and a Softmax function to yield the predicted class probabilities $\hat{Y}$.

Computation Analysis: Table 5.1 shows the computational differences between stan-

Table 5.1: Computational Analysis for Conventional and KGFLA Transformer.

Aspect	Conventional Transformer	KGFLA Transformer
Step 1	Compute $Z = QK^T$	Compute $Z = K^TV$
Size of Z:	$[(S^2+1) \times (P_2/h)] \times [(P_2/h) \times (S^2+1)]$ $= [(S^2+1) \times (S^2+1)]$	$[(P_2/h) \times (S^2+1)] \times [(S^2+1) \times (P_2/h)]$ $= [(P_2/h) \times (P_2/h)]$
No. of Multiplications	$(S^2+1)^2 \cdot (P_2/h)$	$(S^2+1) \cdot (P_2/h)^2$
Step 2	Compute $O = ZV$	Compute $O = QZ$
Size of O:	$[(S^2+1) \times (S^2+1)] \times [(S^2+1) \times (P_2/h)]$ $= [(S^2+1) \times (P_2/h)]$	$[(S^2+1) \times (P_2/h)] \times [(P_2/h) \times (P_2/h)]$ $[(S^2+1) \times (P_2/h)]$
No. of Multiplications	$(S^2+1)^2 \cdot (P_2/h)$	$(S^2+1) \cdot (P_2/h)^2$
Total Multiplications	$2(S^2+1)^2 \cdot (P_2/h)$	$2(S^2+1) \cdot (P_2/h)^2$
Time Complexity	$\mathcal{O}((S^2+1)^2 \cdot (P_2/h))$	$\mathcal{O}((S^2+1) \cdot (P_2/h)^2)$

dard and KGFLA Transformer modules. In standard Transformers, attention is computed as $Z = QK^T$, followed by $O = ZV$. This results in a quadratic cost in the number of tokens (S^2+1). KGFLA first computes a compact kernel $Z = K^TV$, then uses $O = QZ$ to get the output. This avoids forming the large token-token interaction matrix. Although output dimensions stay the same, the number of multiplications drops. Standard attention requires $2(S^2+1)^2 \cdot (P_2/h)$ operations. KGFLA reduces this to $2(S^2+1) \cdot (P_2/h)^2$. The time complexity also drops from $\mathcal{O}((S^2+1)^2 \cdot (P_2/h))$ to $\mathcal{O}((S^2+1) \cdot (P_2/h)^2)$. This is a major gain, especially due to $(S^2+1) \gg (P_2/h)$.

5.5 Experiments: Implementations and Results

In this section, we evaluate the classification performance of the proposed method on three widely used HSI datasets. We compare our method with seven well-known and recent HSI classification models. These include 2D-CNN [41], 3D-CNN [180], DBDA [154], MDCNN [174], SSFTT [124], IFormer [163], and JGSA-VIT [125]. Classification effectiveness has been measured through OA, AA, and κ scores, complemented by qualitative insights from classification map visualizations. At the same time, efficiency has been assessed by recording Tr and Te. These metrics help provide a complete and fair comparison of model performance across datasets.

5.5.1 Dataset Description and Training Details

The proposed CKGFLNet uses an embedding dimension of $P_2 = 120$ with $h = 12$ attention heads. We train the model for 70 epochs with a mini-batch size of 128. The

optimizer is Adam with a learning rate of 1×10^{-3}. All experiments run on an NVIDIA RTX A5000 GPU with 24.5 GB RAM. The input consists of spatial patches of size $[S \times S]$, where $S = 9$, extracted from the IP, PU, and SA datasets. Section 1.3 and Table 1.1 summarize the dataset based on sensor specifications, wavelengths, spatial sizes, spectral bands, and number of classes.

Following standard protocols, we use 10% of samples per class for training in IP. For PU and SA, only 1% of samples per class are used for training. The remaining samples are used for testing. Table 5.2 shows the class-wise distribution of training and testing samples, class names, and color codes used in classification maps.

Table 5.2: IP, PU, and SA Dataset description with a number of training, and test samples per class and overall used within the model.

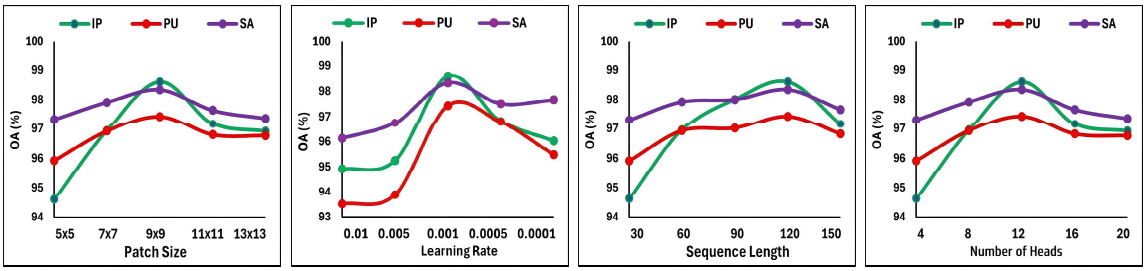
Dataset	IP				PU				SA			
Class	Color	Name	Train	Test	Color	Name	Train	Test	Color	Name	Train	Test
C01		Alfalfa	5	41		Asphalt	66	6565		Brocoli-gw1	20	1989
C02		Corn-NT	143	1285		Meadows	186	15463		Brocoli-gw2	37	3689
C03		Corn-MT	83	747		Gravel	21	2078		Fallow	20	1956
C04		Corn	24	213		Trees	31	3033		Fallow-RP	14	1380
C05		Grass-P	48	435		Painted-MS	13	1332		Fallow-S	27	2651
C06		Grass-T	73	657		Bare-S	50	4979		Stubble	40	3919
C07		Grass-PM	3	25		Bitumen	14	1316		Celery	36	3543
C08		Hay-W	48	430		SB-Bricks	37	3645		Grapes-U	83	6188
C09		Oats	2	18		Shadows	10	937		Soil-VD	62	6141
C10		Soybean-NT	98	874						Corn-SGW	33	3245
C11		Soybean-MT	246	2209						Lettuce-4wk	11	1057
C12		Soybean-C	59	534						Lettuce-5wk	19	1908
C13		Wheat	21	184						Lettuce-6wk	9	907
C14		Woods	127	1138						Lettuce-7wk	10	1060
C15		Buildings-GTD	39	347						Vinyard-U	73	7195
C16		SS-Towers	9	84						Vinyard-VT	17	1772

5.5.2 Hyperparameter Sensitivity Analysis

To optimize the performance of CKGFLNet, we analyze five critical hyperparameters. These are patch size, learning rate, sequence length, the number of Transformer heads, and KGFLA module initialization strategies. These parameters are key for extracting spectral-spatial features from HSI data. We evaluate their influence on OA across the IP, PU, and SA datasets. This helps in selecting optimal configurations for both accuracy and efficiency.

Patch Size: PS is a crucial factor influencing the spatial feature extraction capability of CKGFLNet. As shown in Figure 5.3(a), a $[9 \times 9]$ patch yields the highest OA. It achieves 98.60% for IP, 97.44% for PU, and 98.34% for SA. Smaller patches, like $[5 \times 5]$

and $[7 \times 7]$, miss the broader context, resulting in lower accuracy. Larger patches, like $[11 \times 11]$ and $[13 \times 13]$, introduce irrelevant background, slightly degrading performance.



(a) Patch Size (b) Learning Rate (c) Sequence Length (d) Transformer Heads

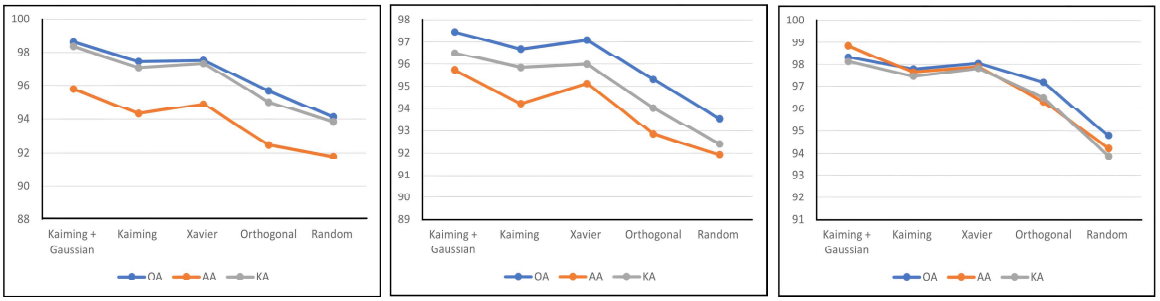
Figure 5.3: Presents a comparative analysis of hyperparameter sensitivity of CK-GFLNet's performance across IP, PU, and SA datasets.

Learning Rate: The LR governs the convergence speed and stability of the model. As shown in Figure 5.3(b), a rate of 0.001 consistently yields the best OA. Smaller values, like 0.0001, result in slow learning. Larger values 0.01, destabilize training and reduce accuracy by overshooting optimal solutions.

Sequence Length: Sequence length represents the number of spectral-spatial tokens processed by the Transformer. It significantly impacts the feature extraction capabilities of the model. Figure 5.3(c) shows that a length of 120 achieves optimal accuracy by capturing essential correlations. Longer sequences lead to redundancy and performance drops. Shorter sequences limit pattern extraction and hurt classification results.

Number of Transformer Heads: The number of self-attention heads in the Transformer is crucial for learning various spectral-spatial patterns. Figure 5.3(d) shows that 12 heads produce the highest OA. Fewer heads, such as 4 or 8, restrict learning capacity. Too many heads, like 16 or 20, increase the computational cost. They also risk overfitting due to unnecessary complexity.

Transformer Initialization Strategies: We assess various initialization methods to ensure stable weight updates and effective feature learning. Figure 5.4 shows the proposed Kaiming-Gaussian strategy. This approach consistently outperforms conventional techniques across all three datasets. For instance, it achieves the highest OA on IP with 98.6%, outperforming Kaiming (97.48%) and Xavier (97.55%). It also achieves 97.44% OA on PU and 98.34% OA, 98.84% AA, and 98.14% κ on SA. These results show that combining Kaiming and Gaussian initializations improves gradient flow and feature diversity. Orthogonal and Random methods perform poorly, especially on complex datasets.



(a) IP (b) PU (c) SA

Figure 5.4: Presents a comparative analysis of following initialization strategies: (i) Kaiming-Gaussian, (ii) Kaiming, (iii) Xavier, (iv) Orthogonal, and (v) Random across all datasets, demonstrating the model's performance over OA, AA, κ (KA).

5.5.3 Comparison With State-of-the-Art (SOTA) Methods

This section presents a comparative analysis of SOTA methods and proposed CK-GFLNet for HSI classification on three standard datasets: IP, PU, and SA. Performance is evaluated using OA, AA, κ, as detailed in Table 5.3. Additionally, the classification maps illustrated in Figures 5.5 visually confirm these performance findings.

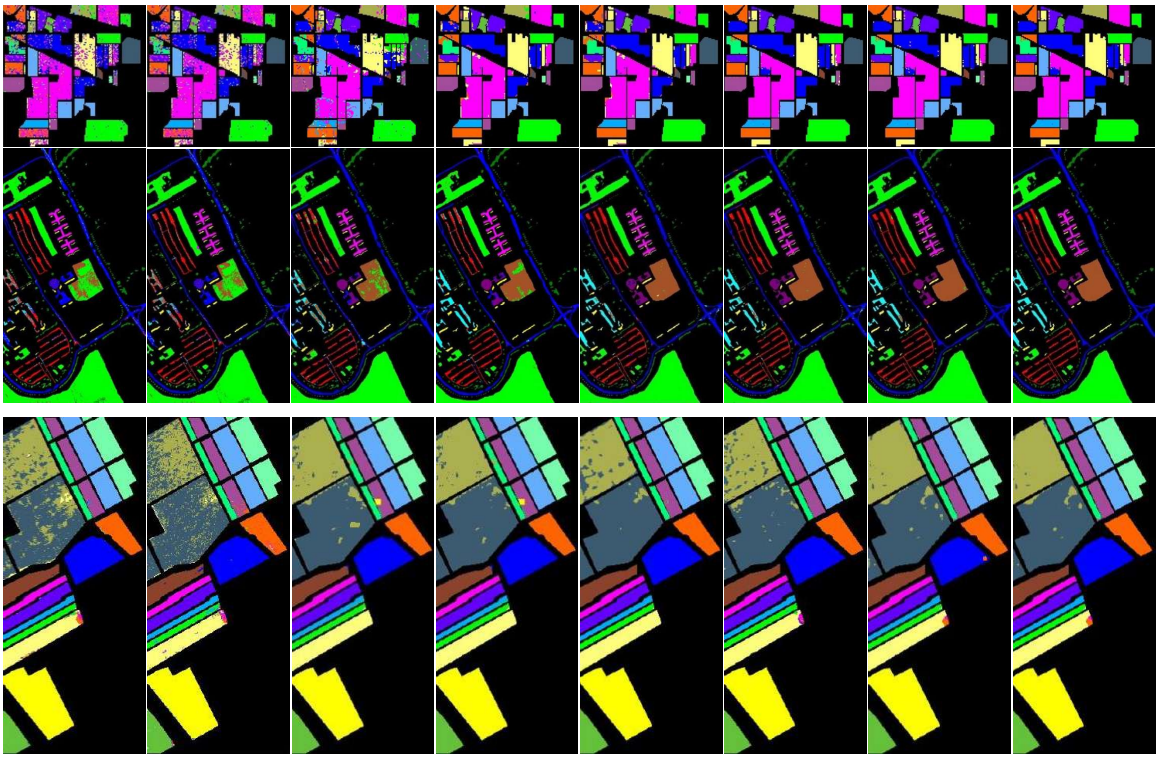
Performance Analysis on IP Dataset: The IP dataset presents moderate spectral complexity. CNN models like 2D-CNN and 3D-CNN achieve limited OAs of 85.07% and 80.46%, respectively. Enhanced CNNs such as DBDA and MDCNN show better results. However, transformer-based models like SSFTT and IFormer exceed 96% OA. CKGFLNet outperforms all, achieving 98.60% OA, 95.78% AA, and 98.35 κ. As illustrated in Fig. 5.5, Corn-NT (, C02) and Corn-MT (, C03) exhibit strong boundary distortion in CNN-based methods, often producing fragmented classification maps. Smaller classes such as Oats (, C09) are also prone to misclassification in earlier models. Our CKGFLNet minimizes these distortions, achieving sharper separations and better preservation of small and spectrally similar classes, resulting in a cleaner and more reliable IP classification map.

Performance Analysis on PU Dataset: PU dataset shows high spectral diversity. Traditional CNN models, like DBDA and MDCNN, achieve 92.84% and 94.88% OA, respectively. Transformer-based methods perform better. SSFTT and IFormer both cross the 96% OA mark. LGSA-VIT performs well with 97.32% OA. Still, CKGFLNet achieves the best results—97.44% OA and a κ of 96.50. Its AA of 95.74% also highlights its robustness across varied spectral classes.

As shown in Figure 5.5, Meadows (C02,) and Gravel (C03,) are particularly

Table 5.3: Classification Performance Comparison of Different Frameworks for IP, PU and SA datasets, Where OA (%), AA (%), and $\kappa \times 100$ Are Reported. The Best Results Are Shown Bold.

Methods	IP			PU			SA		
	OA	AA	$\kappa \times 100$	OA	AA	$\kappa \times 100$	OA	AA	$\kappa \times 100$
2D-CNN	85.07	91.97	82.43	85.86	92.1	83.74	91.03	93.22	90.18
3D-CNN	80.46	90.31	77.13	83.29	91.65	80.08	88.72	91.56	87.43
DBDA	93.28	91.92	93.04	92.84	94.22	91.25	95.19	97.34	94.81
MDCNN	94.62	92.24	95.28	94.88	93.94	94.03	96.03	97.29	95.76
SSFTT	96.19	93.04	95.82	96.11	94.07	95.30	97.82	98.45	97.36
IFormer	96.36	93.21	90.25	96.28	94.67	95.56	97.11	98.21	96.22
LGSA-VIT	98.52	95.42	98.25	97.32	**95.82**	96.43	**98.47**	98.72	**98.31**
CKGFLNet	**98.60**	**95.78**	**98.35**	**97.44**	95.74	**96.50**	98.34	**98.84**	98.14



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 5.5: Visual classification results for IP, PU, and SA datasets. The first row corresponds to IP, the second row to PU, and the third row to SA. The columns represent: (a) 2D-CNN, (b) 3D-CNN, (c) DBDA, (d) MDCNN, (e) SSFTT, (f) IFormer, (g) LGSA-VIT, and (h) CKGFLNet.

challenging, with earlier CNN and backbone models producing high overlap and noisy regions. Bitumen (C07,) is also inconsistently classified in prior methods. Our CKGFLNet significantly reduces these misclassifications, yielding more consistent and homogeneous PU classification maps.

Performance Analysis on SA Dataset: The SA dataset has a high spectral resolution. This increases classification difficulty. CNNs like DBDA and MDCNN achieve up to 96.03% OA. However, they lag behind transformer-based models. SSFTT and IFormer exceed 97% OA. LGSA-VIT achieves 98.47% OA and a κ of 98.31. CKGFLNet follows closely with 98.34% OA. It also records the highest AA of 98.84%. This shows that CKGFLNet balances spatial detail and spectral sensitivity effectively.
Overall, transformer models consistently show better performance than CNNs in HSI classification. LGSA-VIT performs best on the SA dataset. However, CKGFLNet leads on both the IP and PU datasets. It also achieves the highest AA on SA. Its hybrid design combines CNNs with attention mechanisms. Effective initialization strategies further boost its performance. As a result, CKGFLNet ensures high accuracy and strong generalization across datasets. As depicted in Fig. 5.5, Vineyards (, C15; , C16) and Lettuce classes (C11–C14, e.g.,) exhibit strong spectral similarity, leading to distortion and overlap in CNN-based models. Stubble (, C06) also shows scattered noise in earlier methods. Our CKGFLNet provides clearer separations, reducing boundary noise and preserving structural consistency in the SA map.

5.5.4 Computational Cost Analysis

Efficient HSI classification demands a balance between accuracy and computational efficiency. Table 5.4 compares overall accuracy, training time, and testing time across models, highlighting the trade-offs between performance and efficiency. The proposed

Table 5.4: Computation complexity analysis comparing overall accuracy (OA in %), training time (Tr in seconds: s), and testing time (Te in seconds: s) across different models. Red represents the highest OA and the lowest Tr and Te, while green indicates the second-highest OA and second-best Tr and Te.

Datasets	IP			PU			SA		
Methods	OA	Tr	Te	OA	Tr	Te	OA	Tr	Te
SSFTT	96.19	65.34	1.85	96.11	123.55	2.91	97.82	152.18	3.12
IFormer	96.36	75.21	1.92	96.28	130.92	3.06	97.11	160.45	3.35
LGSA-VIT	98.52	17.18	0.42	97.32	36.84	1.74	98.47	45.45	2.08
LGSA-VIT-G	98.40	17.10	0.45	96.82	36.31	1.72	98.40	45.10	1.99
CKGFLNet-K	98.42	15.06	0.38	96.66	32.71	1.49	98.11	40.31	1.82
CKGFLNet	98.60	15.10	0.39	97.44	31.44	1.47	98.34	40.76	1.85

CKGFLNet achieves the highest accuracy with 98.60% on the IP dataset, 97.44% on PU, and 98.34% on Salinas while maintaining relatively low training times of 15.10s, 31.44s, and 40.76s. Its variant, CKGFLNet-K, trains marginally faster but sacrifices accuracy, highlighting the balance CKGFLNet strikes between efficiency and performance. Compared to LGSA-VIT, which achieves similar accuracy, CKGFLNet trains

faster across all datasets. LGSA-VIT requires 17.18s, 36.84s, and 45.45s, indicating higher computational cost. LGSA-VIT-G, while slightly faster, performs worse, stressing the importance of initialization. Additionally, SSFTT and IFormer incur significantly longer training times, making them less practical for real-time or low-resource scenarios. Their performance does not justify the higher computational load compared to CKGFLNet. In conclusion, CKGFLNet achieves high accuracy with minimal training cost, making it ideal for real-world HSI classification.

5.6 Chapter Summary

This chapter presented CKGFLNet, a hybrid deep learning framework for HSI classification that combines 3D-2D CNNs with a KGFL Transformer. As described in Section 5.4, the model performs hierarchical feature extraction and global context modeling while mitigating the high computational cost of standard self-attention via a focused linear attention mechanism with complexity $\mathcal{O}(Nd^2)$.

Extensive experimental results in Section 5.5.3 demonstrate the effectiveness of CKGFLNet across three benchmark datasets. On the IP dataset, it achieved 98.60% OA, 95.78% AA, and 98.35 κ, outperforming CNN models like 2D-CNN (85.07% OA) and 3D-CNN (80.46% OA) as well as transformer-based approaches such as SSFTT (96.19% OA) and IFormer (96.36% OA). For the PU dataset, CKGFLNet attained 97.44% OA, 95.74% AA, and 96.50 κ, surpassing LGSA-ViT (97.32% OA) and other baselines. On the SA dataset, it reached 98.34% OA and 98.84% AA, demonstrating competitive performance relative to LGSA-ViT (98.47% OA) while achieving the highest average accuracy across classes.

Section 5.5.4 analyzed computational efficiency, showing that CKGFLNet maintains low training and inference times. Training times were 15.10s, 31.44s, and 40.76s for IP, PU, and SA, respectively, with testing times under 2s per dataset. Compared to SSFTT and IFormer, which incur significantly longer Tr, CKGFLNet achieves a superior balance of high accuracy and low computational cost. Its design enables efficient representation learning through 3D-2D convolutions and linear attention, making it suitable for real-time or resource-constrained scenarios.

Overall, CKGFLNet outperforms CNNs and transformer-based models in accuracy and efficiency. Its hybrid design enables rich spectral-spatial feature extraction, fast training, and robust generalization while avoiding the quadratic cost of self-attention. Future work may address class imbalance, adaptive loss, multi-modal fusion, and knowledge distillation to enhance performance.