SPIKESST: AN ENERGY-EFFICIENT SPIKING NEURAL NETWORK FOR HYPERSPECTRAL IMAGE CLASSIFICATION

Muhammad Hassaan Farooq Butt[1], *Chao Wang*[2]

[1]Faculty of Environmental Sciences and Natural Resource Management,
Norwegian University of Life Sciences (NMBU), Norway
[2]Department of Intelligent Systems, Hiroshima City University, Japan

ABSTRACT

Hyperspectral imaging (HSI) sensors aboard airborne and spaceborne platforms generate high-dimensional data, motivating the development of classifiers with low computational and memory demands. Spiking neural networks (SNNs) replace dense multiply-and-accumulate operations with sparse binary events; yet, many spiking attention mechanisms retain quadratic token interactions. We present SpikeSST, a compact 0.11M parameter SNN built around linear spike-driven attention. A convolutional encoder transforms PCA-reduced patches into temporal spike trains, while binary masking and summation efficiently aggregate global context. Temporal mixing and a binary channel gate further refine the resulting spike representations. Except for the input encoder and a single channel-gate projection, all weighted operations consume binary spikes, enabling conditional accumulation throughout most of the network. We evaluate SpikeSST on spatially disjoint splits of Pavia University (PU), WHU-Hi-HanChuan (HC), and QUH-Tangdaowan (QUHT) using identical split manifests across five paired runs. Compared with two SNN baselines, SpikeSST requires substantially fewer parameters and lower estimated arithmetic energy, with no consistent directional difference in classification accuracy. Under identical GPU conditions, it reduces training time by up to $6.9\times$ and inference time by approximately $16\times$. Layerwise analysis accounts for real-valued pooling operations in the energy estimate, further supporting the favorable accuracy–efficiency trade-off of SpikeSST. The code is available at `https://github.com/MHassaanButt/SpikeSST`.

***Index Terms*—** Hyperspectral image classification (HSIC), spiking neural network (SNN), spike-driven attention, energy-efficient deep learning, neuromorphic computing.

1. INTRODUCTION

The growing use of compact airborne hyperspectral platforms has increased the demand for classifiers with similarly modest computational and memory requirements. UAV-borne sensors can acquire centimeter-scale imagery across hundreds of spectral bands, producing high-dimensional data cubes that remain challenging to process under restricted computing resources [1]. Accurate HSIC requires modeling both subtle spectral variations and the spatial structure of neighboring pixels. 3D CNNs jointly learn spectral-spatial features [2], whereas hybrid (2D/3D) architectures organize this learning more efficiently across successive stages [3, 4]. Transformers extend the receptive field by tokenizing adjacent spectral bands [5] or applying self-attention to convolutionally generated tokens [6, 7]. Although these models provide strong representations, they rely largely on dense multiply-accumulate (MAC) operations, and standard self-attention additionally requires quadratic token interactions [8]. These costs can obstruct their use in resource-constrained HSI processing.

SNNs approach computational efficiency differently by representing activations as binary events over a small number of time steps. When event-driven execution is supported, a synaptic event can be processed through accumulation rather than a dense MAC, offering a lower arithmetic cost under the commonly used 45nm energy model [9]. Surrogate-gradient methods have made deep SNNs trainable directly [10], and frameworks such as SpikingJelly provide standardized multi-step leaky integrate-and-fire implementations [11]. For HSIC, however, spike encoding must retain informative spectral amplitudes while learning from relatively few labeled samples. Attention introduces a further efficiency challenge. Spikformer removes softmax from spiking self-attention but retains the $\mathcal{O}(N^2d)$ token score map [12]. The Spike-driven Transformer avoids this quadratic map by using elementwise interactions and accumulations that scale linearly with the number of tokens [13]. Such linear interaction is well-suited to HSI patches, where global spectral-spatial context is useful, but quadratic attention can be unnecessarily expensive.

Prior SNN-based HSIC differ mainly in how their spiking representations are obtained. Conversion-based methods map a trained CNN to a corresponding SNN; for example, HYPER-SNN quantizes the converted network and supplies analog pixel values directly to its input layer [14]. Other

This work was supported in part by JSPS KAKENHI under Grant Number JP26K21254.

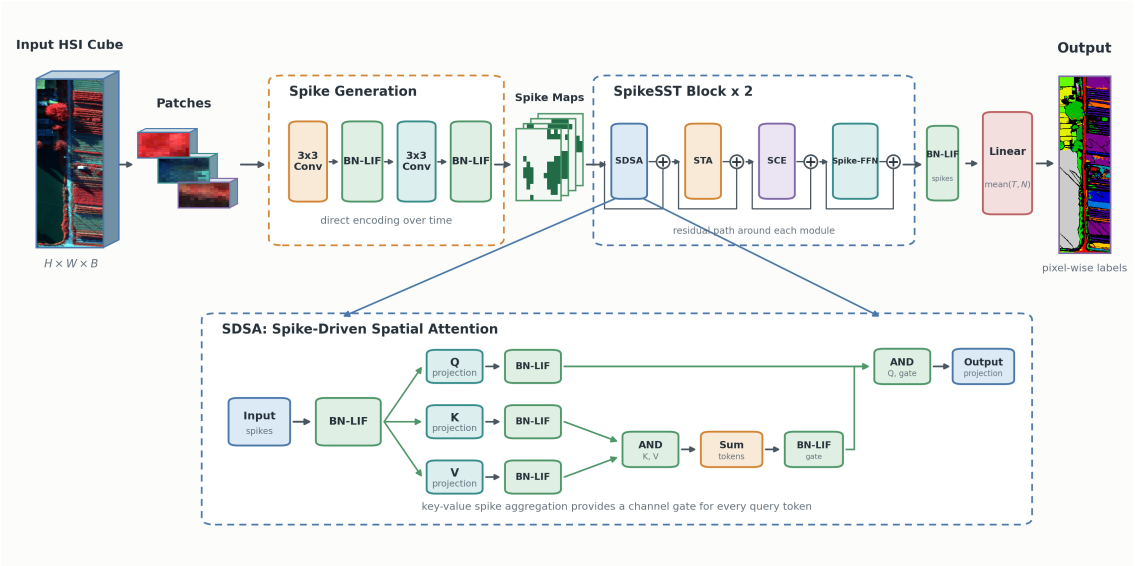


Fig. 1. An overview of the architecture of SpikeSST. A convolutional stem converts PCA-reduced HSI patches into temporal spike maps, which are processed by two residual SpikeSST blocks and decoded by a linear head with averaging across time steps and spatial tokens. The lower panel details SDSA, which forms linear global interactions from binary projections of the query, key, and value.

methods train spiking CNNs directly using surrogate gradients and direct coding [15, 16, 17], while recent work has investigated deeper residual and multi-branch spiking backbones [18]. These approaches are primarily based on convolutional mixing or ANN-to-SNN conversion. Linear spike-driven global interaction has received considerably less attention in HSIC. The evaluation protocol is equally important. Random pixel sampling can cause training and test patches to overlap, reducing their spatial independence and potentially producing overly optimistic results [19, 20, 21]. We therefore use fixed, non-overlapping spatial partitions and apply the same coordinate manifests across all reproduced methods.

In this work, we introduce SpikeSST, a compact SNN that combines linear spike-driven attention, temporal mixing, and binary channel gating. A convolutional encoder maps PCA-reduced HSI patches to temporal spike trains, after which linear spike-driven attention aggregates global context without constructing a quadratic token score map. Most weighted layers, including the classifier, operate on binary spike tensors and can therefore use conditional accumulations. The input convolution and the first squeeze projection of the channel gate operate on real-valued tensors and are explicitly counted as MAC operations. The complete four-timestep network contains approximately 0.11M parameters. We compare SpikeSST with Brain-inspired SNN [15] and FRM-SNN [18] using identical non-overlapping spatial partitions. For each scene, we conduct five paired runs with matched random seeds, enabling controlled comparisons of classification accuracy, computational cost, and runtime. We further perform a layerwise arithmetic-energy analysis that distinguishes dense MACs from spike-driven accumulations based on the input type and measured firing rate of each weighted layer, together with operation-count and runtime comparisons under identical hardware conditions.

2. PROPOSED METHOD

Given an HSI cube $\mathcal{X} \in \mathbb{R}^{H\times W\times B}$, we first reduce its spectral dimension to K principal components, with PCA fitted exclusively on the training pixels. A $w \times w$ patch centered at each labeled pixel is then extracted, yielding $\mathbf{x} \in \mathbb{R}^{K\times w\times w}$. As illustrated in Fig. 1, SpikeSST processes each patch over T discrete time steps. Under direct encoding, the real-valued patch is presented to the first stem convolution at every time step. The subsequent batch normalization, followed by a leaky integrate-and-fire (LIF) neuron (BN-LIF) unit, generates binary spikes that are consumed by the second stem convolution and most subsequent weighted layers, enabling their evaluation via conditional accumulation. Within each spiking channel enhancement (SCE) module, however, the first squeeze projection operates on real-valued pooled firing rates. Accordingly, the input convolution and the SCE squeeze projections are counted as MAC operations, whereas the remaining weighted layers are counted as spike-driven accumulations using their measured firing rates. The resulting spike maps pass through two SpikeSST blocks, each comprising spike-driven spatial attention (SDSA), spiking temporal aggregation (STA), SCE, and Spike-FFN modules with residual connections. Finally, a BN-LIF unit and a linear classifier produce class logits, which are averaged across time steps

and spatial tokens.

2.1. Spiking Neuron Model and Direct Encoding

We use the leaky integrate-and-fire (LIF) neuron in the discrete-time formulation of SpikingJelly [11]. At timestep t, with membrane state $H[t-1]$, input current $X[t]$, time constant τ, and threshold V_{th}:

$$U[t] = H[t-1] + X[t] - \tfrac{1}{\tau}\big(H[t-1] - V_{\text{reset}}\big), \quad (1)$$

$$S[t] = \Theta\big(U[t] - V_{th}\big), \quad (2)$$

$$H[t] = U[t]\,(1 - S[t]) + V_{\text{reset}}\,S[t], \quad (3)$$

where $\Theta(\cdot)$ is the Heaviside step, approximated in the backward pass by the arc-tangent surrogate gradient [10]. We denote a BN-LIF pair by $\mathcal{SN}(\cdot)$. The analog patch is broadcast to all T time steps and encoded by the first convolution (*direct encoding* [12]): the stem computes $\mathbf{s}^{(1)}[t] = \mathcal{SN}\big(\mathrm{Conv}_{3\times3}(\mathbf{x})\big)$, followed by a second spike consuming convolution. Because the stem input is identical across time steps, the encoding result can be cached; the LIF membranes nevertheless integrate across t and produce temporally structured spike trains. This retains spectral amplitude information that direct binarization of the input would discard.

2.2. Spike-Driven Spatial Attention (SDSA)

Tokens $\mathbf{X}[t] \in \mathbb{R}^{N\times D}$ ($N = w^2$) are first spiked, $\mathbf{s} = \mathcal{SN}(\mathbf{X})$, then projected to spiking queries, keys, and values:

$$\mathbf{q} = \mathcal{SN}(\mathbf{s}\mathbf{W}_Q), \quad \mathbf{k} = \mathcal{SN}(\mathbf{s}\mathbf{W}_K), \quad \mathbf{v} = \mathcal{SN}(\mathbf{s}\mathbf{W}_V), \quad (4)$$

with $\mathbf{q},\mathbf{k},\mathbf{v} \in \{0,1\}^{N\times D}$ split into h heads. Instead of the quadratic $\mathbf{q}\mathbf{k}^\top$ map, each head aggregates a global spiking context through an element-wise product and sum and gates the queries with it:

$$\mathbf{g} = \mathcal{SN}\Big(\textstyle\sum_{j=1}^{N} \mathbf{k}_j \odot \mathbf{v}_j\Big), \qquad \mathrm{SDSA}(\mathbf{X}) = \big(\mathbf{q}\odot\mathbf{g}\big)\mathbf{W}_O. \quad (5)$$

For binary operands, $\mathbf{k}_j \odot \mathbf{v}_j$ is a logical AND, and the sum is an accumulation. The token interaction in Eq. (5) has complexity $\mathcal{O}(ND)$ rather than $\mathcal{O}(N^2D)$ [13]. Including the four linear projections, the block has complexity $\mathcal{O}(ND^2 + ND)$ and remains linear in the number of tokens. The gate $\mathbf{g}$ aggregates the joint key and value activity before modulating each query.

2.3. Spiking Temporal Aggregation (STA)

Because LIF membranes integrate over time steps, the T spike maps carry complementary information. STA mixes them with a learnable matrix $\mathbf{M} \in \mathbb{R}^{T\times T}$ applied to spiked features,

$$\tilde{\mathbf{s}}[t] = \mathrm{BN}\Big(\textstyle\sum_{t'=1}^{T} M_{t,t'}\,\mathcal{SN}(\mathbf{X})[t']\Big), \quad (6)$$

adding only T^2 parameters (16 at T=4) while letting the network weight early (transient) against late (steady-state) firing patterns.

2.4. Spiking Channel Enhancement (SCE)

A squeeze and excitation gate is adapted to spikes by replacing the sigmoid with a binary spiking gate. Using firing rates pooled over tokens, $\mathbf{p}[t] = \frac{1}{N}\sum_n \mathcal{SN}(\mathbf{X})[t,n]$,

$$\mathbf{g}_c[t] = \mathcal{SN}\big(\mathbf{W}_2\,\mathcal{SN}(\mathbf{W}_1\mathbf{p}[t])\big) \in \{0,1\}^D, \quad (7)$$

and the block output is $\mathrm{BN}\big(\mathcal{SN}(\mathbf{X}) \odot \mathbf{g}_c\big)$. The first projection receives real-valued firing rates, whereas the second receives spikes. The output operation is a binary channel mask rather than a floating-point multiplication. Each spiking block applies SDSA, STA, SCE, and a two-layer spiking MLP (expansion ratio 2), all with residual connections. The classifier consumes the final spike tensor and applies rate decoding over timesteps and tokens: $\hat{\mathbf{y}} = \frac{1}{TN}\sum_{t,n} \mathbf{W}_{cls}\,\mathbf{s}^{(L)}[t,n]$.

2.5. Estimated Arithmetic Energy

Following [12, 13], we estimate the arithmetic cost of one patch inference in 45 nm CMOS [9] with $E_{MAC} = 4.6$ pJ and $E_{AC} = 0.9$ pJ. One dense operation denotes one multiply-and-accumulate. For a spike consuming layer ℓ with dense cost Ops_ℓ and measured input firing rate f_ℓ, the synaptic operations are $\mathrm{SOPs}_\ell = f_\ell T \mathrm{Ops}_\ell$, and

$$E_{\text{SpikeSST}} = E_{MAC}\sum_{\ell\in\mathcal{A}} \mathrm{Ops}_\ell + E_{AC}\sum_{\ell\in\mathcal{S}} \mathrm{SOPs}_\ell, \quad (8)$$

where $\mathcal{S}$ is the spike-consuming layers and $\mathcal{A}$ the real-valued ones: the direct coding stem convolution and the first SCE squeeze projection of each block. These are charged at the full E_{MAC} rate. Membership of $\mathcal{A}$ is not assumed from the layer's name: every layer is classified at run time by whether the tensor it consumes is verifiably binary. The dense reference is $E_{MAC}\sum_\ell \mathrm{Ops}_\ell$ for the identical one-step architecture, counted at dispatch level so that functional operations are included. Firing rates are measured over the entire test set of each scene, and the joint activity of the key and value AND in Eq. (5) is measured directly rather than approximated by the product of its marginals.

The model excludes memory access, neuron state updates, control logic, interconnect, PCA, and patch extraction, and is not a measurement of device energy. The estimate charges the analog stem convolution once because its input is identical at every timestep and its output can be cached. The current PyTorch implementation recomputes this convolution, so the estimate represents a cache-optimized arithmetic implementation rather than the literal GPU instruction stream. Per-layer firing rates and operation counts are retained with each result for independent verification.

Table 1. Accuracy on non-overlapping spatial splits (w=13, K=20). Values are mean±standard deviation (%) over 5 runs.

Method	PU			HC			QUHT		
	OA	AA	κ	OA	AA	κ	OA	AA	κ
Brain-inspired SNN‡	91.99±4.18	91.06±3.01	89.31±5.56	87.62±2.46	75.79±1.69	85.45±2.78	91.59±0.97	86.78±2.69	90.41±1.11
FRM-SNN‡	**94.33±2.45**	**91.12±4.72**	**92.32±3.43**	88.65±1.33	77.75±3.36	86.65±1.49	91.99±1.76	87.95±3.19	90.87±2.02
SpikeSST (ours)	91.41±4.33	90.75±4.67	88.54±5.83	**88.99±0.44**	**79.31±2.32**	**87.05±0.50**	**92.58±0.93**	**88.58±1.07**	**91.54±1.08**

3. EXPERIMENTS

3.1. Datasets and Protocol

We evaluate on three benchmarks: PU (ROSIS, 610×340, 103 bands, 9 classes), HC (1217×303, 274 bands, 16 classes) [22], and the UAV-borne QUHT (1740×860, 176 bands, 18 classes) [23]. 5% of the labeled pixels from each class are used for training and 5% for validation. Samples are selected from 16×16 spatial blocks. For patch radius $m = 6$, a test center is retained only if its Chebyshev distance from every training or validation center exceeds $2m = 12$. Thus, no 13×13 test patch shares a pixel with a training or validation patch. PCA is fitted on the training spectra and retains $K = 20$ components. SpikeSST uses $D = 64$, eight heads, two blocks, four timesteps, and a dropout of 0.2. We train with AdamW, a learning rate of 10^{-3}, a weight decay of 10^{-4}, a batch size of 64, label smoothing of 0.1, and cosine decay. Brain-inspired SNN [15] and FRM-SNN [18] are executed from their official repositories, with compatibility adapters used only to align data loading and evaluation. For a fair comparison, both baselines and SpikeSST use identical spatial partitions and the same five fixed random seeds on each dataset. We report mean and standard deviation across these five paired runs to account for stochastic variation.

3.2. Results Analysis

Table 1 shows comparable OA, AA, and κ across the three models. SpikeSST has the largest mean on HC and QUHT, whereas FRM-SNN has the largest mean on PU, exceeding SpikeSST by 2.92% in OA, 0.37% in AA, and 3.78% in κ. Relative to FRM-SNN, the strongest baseline on these datasets, SpikeSST improves OA, AA, and κ by 0.34, 1.56, and 0.40 points on HC, and by 0.59, 0.63, and 0.67 points on QUHT, respectively. Five of the six seed pairs had 95% t intervals for the OA difference spanning zero, including the largest mean difference, -2.92 points against FRM-SNN on PU ($[-6.22, 0.38]$). Thus, these runs do not establish formal equivalence or a general directional advantage. The only interval that excludes zero is against Brain-inspired SNN on QUHT (0.99, $[0.35, 1.63]$). Table 2 lists all six intervals. Fig. 2 shows classification maps for every scene and method at a common seed. The maps are consistent with the paired intervals: the three models produce visually comparable classifications, with the residual errors of the disjoint protocol concentrated at class boundaries.

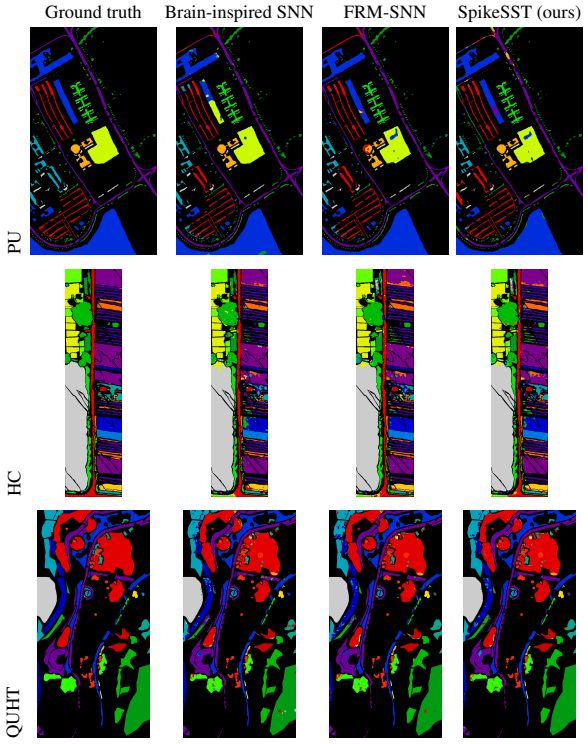


Fig. 2. Classification maps for the PU, HC, and QUH-Tangdaowan datasets.

Table 2. Seed paired OA difference (ours − baseline) on identical splits, with 95% t intervals.

Dataset	Baseline	ΔOA	95% CI
PU	Brain-inspired SNN‡	-0.58	[-2.38, +1.22]
	FRM-SNN‡	-2.92	[-6.22, +0.38]
HC	Brain-inspired SNN‡	+1.37	[-1.52, +4.26]
	FRM-SNN‡	+0.33	[-1.07, +1.74]
QUHT	Brain-inspired SNN‡	+0.99	[+0.35, +1.63]
	FRM-SNN‡	+0.58	[-0.69, +1.85]

3.3. Arithmetic Cost and Runtime

Table 3 compares operation counts, estimated arithmetic energy, and measured runtime. FLOPs denote the dense operation count of an equivalent one-step artificial network, whereas SOPs represent the spike-triggered synaptic operations accumulated over T time steps using the measured layer-wise firing rates. The primary estimate charges a layer

Table 3. Model cost and measured runtime on every scene.

Scene	Method	Params (M)	FLOPs (M)	SOPs (M)	Energy (μJ)		Train (s)	Infer (s)
					Primary	Optimistic		
PU	Brain-inspired SNN‡	0.86	1 572.0	178.8	541.9	235.4	359	7.2
	FRM-SNN‡	1.12	1 876.8	262.5	3 217.0	819.4	549	14.5
	SpikeSST (ours)	**0.11**	**15.4**	**10.5**	**14.0**	**10.4**	**103**	**0.9**
HC	Brain-inspired SNN‡	0.88	1 572.4	203.3	566.2	258.0	1 908	29.8
	FRM-SNN‡	1.13	1 877.4	245.2	3 204.3	804.4	3 803	46.8
	SpikeSST (ours)	**0.11**	**15.4**	**9.8**	**13.4**	**9.7**	**548**	**4.5**
QUHT	Brain-inspired SNN‡	0.88	1 572.6	239.5	599.4	290.6	3 428	48.0
	FRM-SNN‡	1.14	1 877.6	463.3	3 401.5	1 000.9	7 842	104.9
	SpikeSST (ours)	**0.11**	**15.4**	**10.5**	**14.0**	**10.3**	**1 208**	**10.0**

at MAC cost whenever its observed input is real-valued. The optimistic estimate charges those operations at accumulation cost and serves as a lower bound favorable to the baselines. Both estimates exclude memory traffic, neuron state, control, interconnect, and preprocessing. Pooling between the spiking blocks of the baselines produces multilevel activations. Consequently, real-valued layers contribute approximately 64% of the primary energy estimate for the Brain-inspired SNN and 88% for FRM-SNN on QUHT. For SpikeSST, the analog stem and pooled SCE projections account for about 9% of the charged operations and one-third of the primary estimate. Even when the primary estimate for SpikeSST is compared with the optimistic baseline estimates, it is $20.8\times$ lower than Brain-inspired SNN and $71.5\times$ lower than FRM-SNN on QUHT. All timings were obtained on one RTX PRO 6000 GPU using the same node and data splits. SpikeSST reduces total training time by 2.8–$6.9\times$ and full test set inference time by 4.8–$16.1\times$.

4. CONCLUSION

We introduced SpikeSST, a compact SNN that brings linear spike-driven attention to spectral-spatial HSIC while preserving binary computation across most weighted operations. Across spatially disjoint splits of three datasets, evaluated over five paired runs per dataset, the comparisons reveal no consistent directional difference in accuracy relative to the two reproduced SNN baselines, although the confidence intervals remain too broad to establish formal equivalence. The clearest advantage of SpikeSST lies in efficiency: the 0.11M parameter model requires lower estimated arithmetic energy and, under identical inference conditions, is $4.8\times$–$16.1\times$ faster. End-to-end training is $2.8\times$–$6.9\times$ shorter. Layerwise analysis further shows that pooling can produce real-valued inputs to downstream projections in nominally spiking models, underscoring the importance of determining arithmetic cost from the tensors actually consumed by each operation. These findings remain subject to several limitations: the energy analysis uses a 45 nm arithmetic model and excludes memory traffic, neuron-state updates, control flow, interconnect, and preprocessing; it therefore does not represent measured system power. Direct evaluation on neuromorphic hardware is needed to determine whether the estimated arithmetic savings translate into hardware-level energy reductions. Moreover, the tensor-aware accounting adopted here provides a practical basis for designing spiking backbones that remain accumulation-dominated throughout the network.

5. REFERENCES

[1] Etienne Ducasse, Karine Adeline, Audrey Hohmann, Véronique Achard, Anne Bourguignon, Gilles Grandjean, and Xavier Briottet, "Mapping of clay montmorillonite abundance in agricultural fields using unmixing methods at centimeter scale hyperspectral images," *Remote Sensing*, vol. 16, no. 17, pp. 3211, 2024.

[2] Yushi Chen, Hanlu Jiang, Chunyang Li, Xiuping Jia, and Pedram Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE transactions on geoscience and remote sensing*, vol. 54, no. 10, pp. 6232–6251, 2016.

[3] Swalpa Kumar Roy, Gopal Krishna, Shiv Ram Dubey, and Bidyut B Chaudhuri, "Hybridsn: Exploring 3-d–2-d cnn feature hierarchy for hyperspectral image classification," *IEEE geoscience and remote sensing letters*, vol. 17, no. 2, pp. 277–281, 2019.

[4] Muhammad Hassaan Farooq Butt, Hamail Ayaz, Muhammad Ahmad, Jian Ping Li, and Ramil Kuleev, "A fast and compact hybrid cnn for hyperspectral imaging-based bloodstain classification," in *2022 IEEE Congress on Evolutionary Computation (CEC)*, 2022, pp. 1–8.

[5] Danfeng Hong, Zhu Han, Jing Yao, Lianru Gao, Bing Zhang, Antonio Plaza, and Jocelyn Chanussot, "Spectralformer: Rethinking hyperspectral image classifica-

tion with transformers," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–15, 2021.

[6] Le Sun, Guangrui Zhao, Yuhui Zheng, and Zebin Wu, "Spectral–spatial feature tokenization transformer for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–14, 2022.

[7] Chao Wang and Daisuke Miyazaki, "Cross-scale spectral–spatial interaction with nonnegative positional encoding for hyperspectral image classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 18, pp. 28726–28742, 2025.

[8] Muhammad Hassaan Farooq Butt, Majid Manzoor, Muhammad Adnan Farooq Butt, Farhan Aadil, and Rehan Tariq Chohan, "A differential memory attention mamba for spatial-spectral representation learning toward hyperspectral image classification," *IEEE Access*, vol. 14, pp. 54380–54394, 2026.

[9] Mark Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC)*. IEEE, 2014, pp. 10–14.

[10] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke, "Surrogate gradient learning in spiking neural networks," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

[11] Wei Fang, Yanqi Chen, Jianhao Ding, Zhaofei Yu, Timothée Masquelier, Ding Chen, Liwei Huang, Huihui Zhou, Guoqi Li, and Yonghong Tian, "Spikingjelly: An open-source machine learning infrastructure platform for spike-based intelligence," *Science Advances*, vol. 9, no. 40, pp. eadi1480, 2023.

[12] Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng YAN, Yonghong Tian, and Li Yuan, "Spikformer: When spiking neural network meets transformer," in *The Eleventh International Conference on Learning Representations*, 2023.

[13] Man Yao, JiaKui Hu, Zhaokun Zhou, Li Yuan, Yonghong Tian, Bo XU, and Guoqi Li, "Spike-driven transformer," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[14] Gourav Datta, Souvik Kundu, Akhilesh R. Jaiswal, and Peter A. Beerel, "HYPER-SNN: Towards energy-efficient quantized deep spiking neural networks for hyperspectral image classification," *arXiv preprint arXiv:2107.11979*, 2021.

[15] Yang Liu, Kejing Cao, Rui Li, Hongxia Zhang, and Liming Zhou, "Hyperspectral image classification of brain-inspired spiking neural network based on approximate derivative algorithm," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–16, 2022.

[16] Jiangyun Li, Haoran Shen, Wenxuan Wang, Peixian Zhuang, Xi Liu, and Tianxiang Zhang, "Hyper-s3nn: Spatial–spectral spiking neural network for hyperspectral image classification," *Infrared Physics & Technology*, vol. 138, pp. 105251, 2024.

[17] Shuo Wang, Yuanxi Peng, Lei Wang, and Teng Li, "Boundary-aware deformable spiking neural network for hyperspectral image classification," *Remote Sensing*, vol. 15, no. 20, pp. 5020, 2023.

[18] Yahui Li, Yang Liu, Rui Li, Liming Zhou, Lanxue Dang, Huiyu Mu, and Qiang Ge, "Hyperspectral image classification based on faster residual multi-branch spiking neural network," *Computers & Geosciences*, vol. 197, pp. 105864, 2025.

[19] Jie Liang, Jun Zhou, Yuntao Qian, Lian Wen, Xiao Bai, and Yongsheng Gao, "On the sampling strategy for evaluation of spectral-spatial methods in hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, pp. 862–880, 2016.

[20] Muhammad Hassaan Farooq Butt, Jian Ping Li, and Muhammad Adnan Farooq Butt, "Mitigating hughes phenomenon: Improving hyperspectral imaging classification through active learning for generalization enhancement," in *2023 20th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*. IEEE, 2023, pp. 1–8.

[21] Muhammad Ahmad, Salvatore Distefano, Manuel Mazzara, and Adil Mehmood Khan, "Importance of disjoint sampling in conventional and transformer models for hyperspectral image classification," *arXiv preprint arXiv:2404.14944*, 2024.

[22] Yanfei Zhong, Xin Hu, Chang Luo, Xinyu Wang, Ji Zhao, and Liangpei Zhang, "Whu-hi: Uav-borne hyperspectral with high spatial resolution (h2) benchmark datasets and classifier for precise crop identification based on deep convolutional neural network with crf," *Remote Sensing of Environment*, vol. 250, pp. 112012, 2020.

[23] Hang Fu, Genyun Sun, Li Zhang, Aizhu Zhang, Jinchang Ren, Xiuping Jia, and Feng Li, "Three-dimensional singular spectrum analysis for precise land cover classification from uav-borne hyperspectral benchmark datasets," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 203, pp. 115–134, 2023.