HyTAS: A Hyperspectral Image Transformer Architecture Search Benchmark and Analysis

Fangqin Zhou[1], Mert Kilickaya[1], Joaquin Vanschoren[1], and Ran Piao[1]

Automated Machine Learning Group, Eindhoven University of Technology, the Netherlands
{f.zhou, j.vanschoren}@tue.nl
https://github.com/openml-labs

Abstract. Hyperspectral Imaging (HSI) plays an increasingly critical role in precise vision tasks within remote sensing, capturing a wide spectrum of visual data. Transformer architectures have significantly enhanced HSI task performance, while advancements in Transformer Architecture Search (TAS) have improved model discovery. To harness these advancements for HSI classification, we make the following contributions: *i)* We propose HyTAS, the first benchmark on transformer architecture search for Hyperspectral imaging, *ii)* We comprehensively evaluate 12 different methods to identify the optimal transformer over 5 different datasets, *iii)* We perform an extensive factor analysis on the Hyperspectral transformer search performance, greatly motivating future research in this direction. All benchmark materials are available at HyTAS.

Keywords: Hyperspectral Image Classification · Transformer Architecture Search · Zero-Cost Proxies · Benchmark and Analysis

1 Introduction

Hyperspectral Imaging (HSI) captures a broad spectrum of electromagnetic wavelengths, providing detailed spectral information beyond conventional RGB images. This rich spectral data has significantly impacted various industries, including agriculture for plant monitoring [6,8,21,32,36], remote sensing for Earth analysis [9, 27], and robotics for enhanced navigation and vision [13, 28].

Scientists in fields like agriculture or remote sensing may seek to analyze Hyperspectral images using transformer models, see Fig. 1. However, designing high-performing Hyperspectral image transformers requires significant expertise and computational resources. Transformer Architecture Search (TAS) methods like AutoFormer [5] or ViTAS [23] can help but demand extensive computational resources (e.g., 24 GPU days for AutoFormer).

An alternative is using training-free architecture search techniques, known as zero-cost proxies [14, 16, 25, 31, 39]. Hereafter, we will use the term "proxy" for brevity. Proxies quickly evaluate transformer architectures without training, offering two key advantages: *i) Efficiency:* Proxies identify high-performing transformers within minutes. *ii) Data Independence:* Many proxies do not require real data, reducing setup and data collection costs [3,20,25,39]. This characteristic is

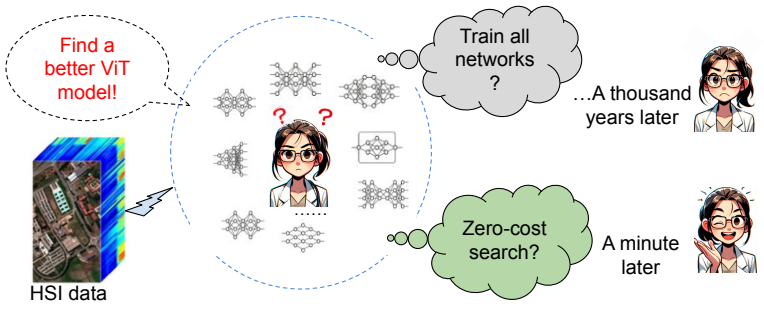

Fig. 1: A scientist outside AI domain may find it difficult to design a novel Hyperspectral image transformer tailored to their tasks and data. To automate the design of Hyperspectral transformers, we introduce **Hy**perspectral **T**ransformer **A**rchitecture **S**earch (HyTAS) in this paper. The scientist images are generated by DALL-E.

especially advantageous for hyperspectral imaging applications, where the setup and calibration of the camera, as well as the collection and annotation of data, incur significant costs.

To that end, in this paper, we introduce **Hy**perspectral **T**ransformer **A**rchitecture **S**earch (HyTAS), to automatically identify Hyperspectral transformers tailored for downstream tasks and datasets. Our motivations are three-fold: *i)* ***Democratization***: HyTAS enables researchers, especially in fields like biology or agriculture, to identify top-performing, lightweight models using minimal resources. *ii)* ***Attention***: We aim to highlight the unique challenges of HSI to the NAS community, which has focused more on RGB images. *iii)* ***Novelty***: We share novel findings to guide the development of better hyperspectral-specific search techniques and architectures.

Our approach begins by generating a substantial pool of candidate Hyperspectral transformers. Subsequently, we assess the performance of various proxies in their capacity to efficiently identify accurate Hyperspectral image transformers. Lastly, we conduct a comprehensive factor analysis to elucidate the factors influencing the performance of both Hyperspectral image transformers and proxy-based transformer search methods.

To sum up, this paper presents three primary contributions:

I. We introduce HyTAS, the pioneering benchmark for Transformer Architecture Search in HSI classification, comprising **2000** distinct Hyperspectral Image transformer architectures.
II. We conduct comprehensive benchmarking using **12** proxies across **5** datasets. Our benchmarking demonstrates the capability to achieve superior performance over a well-established, human-crafted counterpart within minutes.
III. We analyze various factors influencing the effectiveness of Transformer architecture search. Our findings include: *i)* Proxies demonstrate a bias towards larger models, *ii)* Proxies operate without necessitating real data input, *iii)* Proxies synergistically complement each other in predicting the accuracy of input transformers, suggesting promising avenues for future research.

2 Related Work

2.1 Transformer Architecture Search via Proxies

Neural Architecture Search (NAS) customizes optimal architectures for specific tasks [2,18,22,40]. Transformer Architecture Search (TAS) is gaining popularity alongside transformer-based model advancements [4, 5, 23, 39]. However, these methods are computationally intensive, often taking days or months to converge [15, 33]. Many researchers lack resources; for instance, this study employs only a single GPU. An alternative approach involves using proxy functions, which swiftly assess architecture fitness without training [3, 14–16, 25]. In this paper, we adapt proxy methods to quickly search over a large database of Hyperspectral transformers.

2.2 Hyperspectral Image Transformers

Hyperspectral image classification involves processing input cubes with significantly more channels (*e.g.*, 200 vs. 3) and lower spatial resolution than conventional RGB imaging [17]. With the emergence of Vision-Transformers (ViT) [7], manual design of transformer architectures for Hyperspectral image classification has become common practice. For instance, Hong et al. [12] introduce SpectralFormer, adapting ViT for HSI classification, while Sun et al. [24] and Zhou et al. [38] enhance it with spatial-local attention modules. However, manual network design is laborious and resource-intensive. Therefore, in this study, we propose HyTAS to automate Hyperspectral image transformer design.

3 HyTAS Benchmark

In practice, the optimal architecture is determined by its performance, often measured by the highest test accuracy for classification tasks. However, in TAS proxies, where training is not involved, a proxy measure is needed to rank architectures sampled from a search space based on their initial model weights or one-step gradients. This proxy score serves as the metric for estimating the performance of each architecture.

The process of TAS proxy for HSI classification is illustrated in Fig. 2, containing four main steps: 1). Patchify HSI image and tokenize each patch along its spectra, then randomly sample a batch of input. 2). Design a search space and sample architectures from it. 3). Select a proxy to estimate the sampled architectures and rank them by their proxy scores. 4). Evaluate the performance of search proxies. Formally, the objective of TAS proxy is formulated as follows:

$$a^* = \arg\max_{a \in \mathcal{A}} \mathrm{OA}\,(a) \equiv a^* = \arg\max_{a \in \mathcal{A}} \mathcal{F}_s\,(a) \tag{1}$$

where a is a transformer architecture sampled from the search space $\mathcal{A}$, OA and $\mathcal{F}_s$ denote the test overall accuracy and a scoring function, respectively. The objective aims to discover the optimal architecture a^* through a scoring function

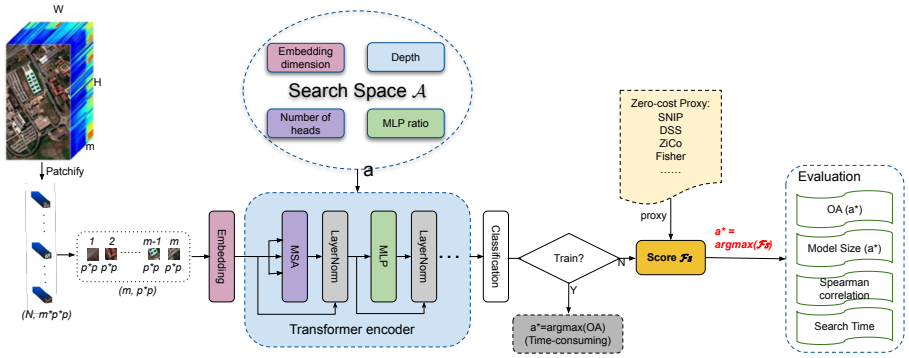

Fig. 2: Diagram of TAS proxies for HSI classification: (1) Patchify the image and tokenize each patch along its spectra. (2) Randomly sample architectures from a designed search space. (3) Select a proxy to compute scores for sampled architectures after passing a batch of input. (4) Use the scores to rank architectures and choose the one with the highest score to retrain. Proxy evaluation metrics include test overall accuracy, model size, Spearman correlation between proxy scores and test OA, and search time.

instead of training all models. Designing a good search space $\mathcal{A}$ and a suitable proxy scoring function that accurately estimates architectures' performance is crucial for TAS proxies.

3.1 Benchmark Proxies

Optimizing Equation 1 relies on employing an appropriate proxy scoring function $\mathcal{F}_s$. In this section, we provide a concise overview of various existing proxies commonly utilized within the NAS and TAS communities. The mathematical equation of each proxy as well as further details are presented in Section 7.1 in our supplementary material.

GradNorm [1]: aggregates the Euclidean norms of each layer's gradients after processing a single batch of training data.

SNIP [14]: calculates a saliency metric by incorporating both gradients and weights, facilitating parameter pruning in neural networks.

GraSP [31]: considers both the first-order and second-order derivatives of the network to preserve training dynamics instead of the loss at the beginning of training.

Synflow [25]: a modification of SNIP that eliminates the absolute expression to address layer collapse during parameter pruning.

LogSynflow [3]: addresses the issue of neglecting the significance of weights due to gradients in Synflow, by scaling down the gradients with a logarithmic function before summing up the contributions of their weights.

Fisher: initially presented by [26] to achieve more runtime-efficient neural architectures, and utilized as a pruning metric at initialization by [30]. More

recently, [1] applied this metric to aggregate all gradients of the activations for scoring a network.

JacobCov [19]: leverages gradients over the input data instead of parameters or activations.

NASWOT [20]: computes the network score according to its kernel matrix of the binary activation patterns between a batch of training inputs.

DSS [39]: specifically designed for transformer architectures via combining synaptic diversity scores of MSA and synaptic saliency scores of MLP modules.

CRoZe [11]: measures the consistency of features, parameters, and gradients between perturbed and clean samples.

T-CET [35]: incorporates the orthogonality of gradient-based (SSNR) and activation-based (NASWOT) compressibility to score an architecture.

ZiCo [16]: considers both absolute mean and standard deviation of gradients, emphasizing their influence on convergence rate and generalization capacity.

3.2 A New Proxy: ZiCo^{++}

Inspired by ZiCo's principle of favoring networks with high absolute mean and low standard deviation gradient values for faster convergence and improved generalization, we introduce ZiCo^{++}, a novel proxy. ZiCo^{++} enhances network expressive capacity by optimizing the mean and minimizing the standard deviation of Fisher information of activations across input samples. Additionally, to address the issue of the final score's heavy reliance on the number of layers in ZiCo, we incorporate a layer decay mechanism during score aggregation in ZiCo^{++}. The mathematical formulation is as follows:

$$\begin{aligned} ZiCo^{++} &\triangleq \sum_{i=1}^{n-1} ZiCo_i^{++} + \sum_{i=n}^{N-1} \frac{1}{i-n+1} ZiCo_i^{++} + ZiCo_N^{++} \quad \text{s.t.} \\ ZiCo_i^{++} &\triangleq \log\left(\sum_{\omega \in \theta_l} \frac{\mathbb{E}\left[\left(\nabla_\omega L\left(X_i, y_i; z_i\right) * z_i \right)^2 \right]}{\sqrt{\operatorname{Var}\left(\left(\nabla_\omega L\left(X_i, y_i; z_i\right) * z_i \right)^2 \right)}} \right) \end{aligned} \tag{2}$$

where ZiCo$_i^{++}$ and ZiCo$_N^{++}$ denote the ZiCo^{++} score of i_th and the last layer, respectively. n is a tunable parameter representing the starting layer for layer decay, such that n is constrained by $n \leq \text{min\_depth} * 4 + 1$, with min_depth denoting the minimal depth defined in the search space. In our experiments, we set min_depth $= 4$ and $n = 6$ to preserve the ZiCo^{++} score for the first embedding and the last projection layer, as well as for the first block containing two MSA layers and two MLP layers.

3.3 Search Space Design

A transformer architecture contains an encoder and decoder, but for our classification task, we focus on encoder-only transformers. Typically, a transformer encoder consists of a series of blocks, each containing a multi-head self-attention (MSA) and a multi-layer perceptron (MLP) module. Therefore, the search space

should account for four key elements: embedding dimension (*embed_dim*), number of blocks (*depth*), number of attention heads (*num_heads*), and MLP embedding ratio (*mlp_ratio*) for each block. These components significantly impact model performance.

Components	***depth***	***embed_dim***	***num_heads***	***mlp_ratio***
Choices	(4, 10, 1)	(32, 240, 16)	(3, 6, 1)	(1, 6, 1)

Table 1: The search space across all datasets. It includes *depth*, *embed_dim*, *num_heads*, and *mlp_ratio*, with values indicating initial, final, and intervals. Different numbers of heads and MLP ratios are utilized for each block. Notably, the total number of layers equals $depth * 4 + 2$.

Drawing inspiration from the effectiveness of manually crafted models proposed by [12, 38], we define our search space in Table 1. We randomly sample 2000 subnetworks from the search space for our experiments.

4 Experimental Setup

Datasets We conduct experiments on five well-established HSI datasets:

i. Indian Pines [10]: 224 spectral bands, 145×145 spatial resolution, 16 classes, 695 training, and $9k$ testing samples.
ii. Houston2013 [37]: 144 spectral bands, 349×1905 spatial resolution, 15 classes, $2k$ training, and $12k$ testing samples.
iii. Pavia University (PaviaU) [10]: 103 spectral bands, 610×340 spatial resolution, 9 classes, $3k$ training, and $40k$ testing samples.
iv. Kennedy Space Center (KSC) [10]: 176 spectral bands, 518×620 spatial resolution, 13 classes, 195 training, and $5k$ testing samples.
v. Salinas scene [10]: 224 spectral bands, 512×217 spatial resolution, 16 classes, 800 training, and $53k$ testing samples.

Indian Pines and Salinas contain mostly agricultural vegetation, and the remaining three datasets consist of non-agricultural vegetation, such as roads and buildings. More details are presented in Section 7.2 in our supplementary material. For data-dependent search proxies, we randomly sample a batch of training data with a batch size of 64, consistent with the batch size used for retraining.

Metrics We evaluate architecture performance with test Overall Accuracy (OA), indicating correctly predicted samples over all samples. Spearman correlation (ρ) measures the correlation between proxy scores and final test OA after training all sampled networks. We also compare Search Time (ST) and proposed Model Size (MS) for each proxy, calculated using the following equation from [29]:

$$MS(a,b,c,d) = 1,539d + a\left(4d + 256cd + 192c + 5d + 7,680 + 2bd^2 + bd + d\right) + 2d + \#\text{ class } \times (d+1)$$

Here, a, b, c, and d represent the depth, MLP ratio, number of heads, and embedding dimension, respectively. #class is the number of classes in the dataset.

5 HyTAS Analysis

This section presents experiments to answer the following research questions:

- RQ1: How do proxies perform on HSI classification?
- RQ2: What factors influence Hyperspectral image transformer performance?
- RQ3: What factors influence the proxy scores?
- RQ4: Are proxies complementary?

5.1 RQ1: How Do Proxies Perform on HSI Classification?

Previous studies have shown the varying performance of proxies under different search space constraints [15,33]. To address RQ1, we conduct experiments under full and constrained search space conditions.

Full Search Space We compare proxies across all (2000) subnetworks sampled from the entire search space on five datasets. We include two reference results: SpectralFormer as the human-crafted transformer, and Oracle as the upper bound. Results are presented in Table 2. We make four observations:

Dataset	**Indian Pines**			**Houston2013**			**PaviaU**			**KSC**			**Salinas**		
Proxy	OA	MS	ρ	OA	MS	ρ	OA	MS	ρ	OA	MS	ρ	OA	MS	ρ
#Flops	0.79	26.3	0.69	0.86	26.3	0.66	0.89	26.3	0.71	0.89	26.3	0.82	0.91	26.3	**0.94**
GradNorm	0.81	26.8	0.71	0.86	29.8	0.70	**0.90**	29.8	0.62	**0.90**	29.2	**0.84**	0.92	29.8	0.87
SNIP	0.81	26.8	0.70	0.87	29.9	0.66	0.89	26.3	0.69	0.89	28.1	0.83	0.93	26.8	**0.94**
GraSP	0.42	0.62	-0.67	0.78	0.94	-0.63	0.78	0.62	-0.67	0.47	0.62	-0.79	0.39	1.23	-0.88
Synflow	0.79	7.35	0.60	0.84	7.35	0.53	**0.90**	7.35	**0.73**	0.87	7.35	0.71	0.92	7.35	0.91
LogSynflow	0.79	7.35	0.60	0.84	7.35	0.52	**0.90**	7.35	**0.73**	0.87	7.35	0.68	0.92	7.35	0.89
Fisher	0.81	16.3	0.61	0.87	18.5	0.64	**0.90**	29.3	0.61	0.89	28.1	0.79	0.92	9.83	0.92
JacobCov	**0.82**	11.9	0.58	**0.88**	21.4	0.72	0.84	5.63	0.40	**0.90**	29.8	0.70	0.90	13.7	0.52
NASWOT	0.79	26.3	0.69	0.87	23.1	0.71	0.89	23.1	0.62	0.89	26.3	0.83	0.91	26.3	0.87
T-CET	0.79	26.3	0.63	0.86	26.3	0.71	0.89	26.3	0.38	0.89	26.3	0.74	**0.94**	29.9	0.65
CRoZe	0.79	26.3	0.46	0.87	26.7	0.54	0.84	26.4	0.18	0.89	29.3	0.55	0.91	28.1	0.42
DSS	0.80	25.9	0.60	0.84	7.35	0.53	0.89	9.89	**0.73**	0.89	10.6	0.72	0.92	29.8	0.91
ZiCo	0.81	26.8	0.52	0.86	26.3	0.62	0.84	25.6	0.26	0.89	26.3	0.63	0.91	26.3	0.50
ZiCo^{++}	0.81	25.6	**0.73**	0.86	29.3	**0.75**	0.84	25.6	0.62	0.89	29.3	0.83	**0.94**	29.3	0.86
SpectralFormer	0.79	0.48	-	0.85	0.48	-	0.85	0.48	-	0.86	0.48	-	0.89	0.48	-
Oracle	0.85	8.42	-	0.89	9.23	-	0.91	3.20	-	0.91	8.43	-	0.96	20.5	-

Table 2: Comparison of search results among proxies, presenting test Overall Accuracy (OA) and corresponding model size (MS) determined by each proxy, as well as their Spearman correlation (ρ) between proxy scores and final test OA after training. The evaluation is conducted over a search space of 2000 subnets across five HSI datasets. MS values are scaled by 10^6. T-CET is computed with SNIP and NASWOT [35].

- The effectiveness of proxies varies across datasets. JacobCov achieves the highest OA for Indian Pines, Houston2013, and KSC datasets, while GradNorm and T-CET demonstrate the highest OA of 0.90 and 0.94 for the PaviaU and Salinas datasets, respectively.
- Some proxies may yield high OA but exhibit low Spearman correlation. For instance, JacobCov achieves the highest OA for three datasets but with considerably lower Spearman correlations compared to the best ones, emphasizing the importance of considering both OA and Spearman correlation for comprehensive assessment.
- ZiCo^{++} does not outperform ZiCo in terms of OA across all datasets except for Salinas. However, it significantly exceeds ZiCo in Spearman correlation across all datasets, achieving the highest Spearman correlation among all proxies for the Indian Pines and Houston datasets.
- Apart from OA, model size, and ρ, searching efficiency is crucial for evaluating proxies. Table 3 compares search times of all proxies. Except for ZiCo and CRoZe, the search times for the other proxies are remarkably similar. ZiCo's extended search time can be attributed to the computation of scores across individual samples, requiring gradient calculations for each sample rather than utilizing batch mean gradients.

Proxy	*GradNorm*	*SNIP*	*Synflow*	*Fisher*	*JacobCov*	*NASWOT*	*CRoZe*	*DSS*	*ZiCo*	*ZiCo*$^{++}$
Search time (h)	0.12	0.12	0.12	0.20	0.12	0.14	0.38	0.17	0.84	0.14

Table 3: Comparison of search times for each proxy across a search space of 2000 subnets on the Indian Pines dataset.

Constrained Search Space We visualize Spearman correlations and proposed OA across varying model size constraints in Fig. 3. The depicted range represents the minimum and maximum model sizes within the search space. For instance, the range from 0 to 0.5e7 signifies model sizes from 0 to 5M, and from 0.5e7 to 1e7, it represents sizes from 5M to 10M. We make three observations:

- The Spearman correlation between proxies and OA decreases with increasing model size, indicating higher efficacy of proxies for smaller models.
- The Oracle OA increases for models smaller than 5M and remains relatively stable thereafter, with the optimal OA observed at 5M.
- Most proxies favor complex models and struggle when simpler models are more effective, suggesting their limited capability to filter out underperforming models.

Takeaway ①: Most proxies can identify a transformer with better accuracy than the human-crafted counterpart, SpectralFormer. However, the proposed architectures are more complex than necessary, when compared to the oracle.

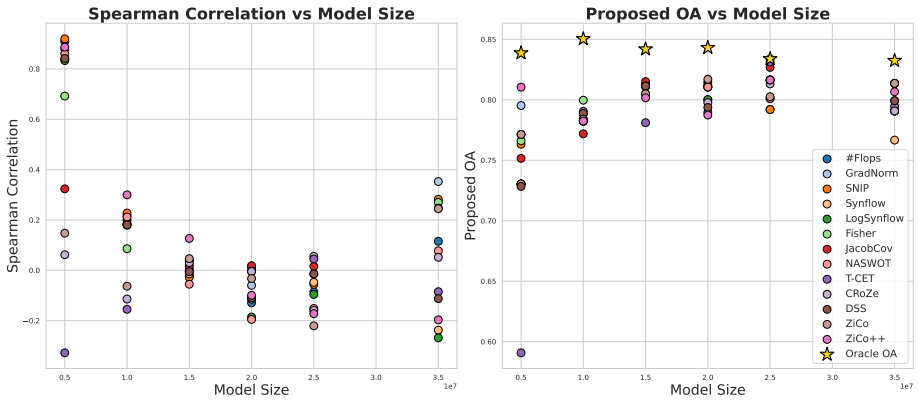

Fig. 3: Spearman correlation and proxy-proposed OA across different model size constraints on the Indian Pines dataset.

5.2 RQ2: What Factors Influence Hyperspectral Image Transformer Performance?

In RQ1, we show that proxies identify a highly accurate, yet complex model. However, is this an optimal Hyperspectral transformer? To answer this, in this section, we analyze the relationship between the performance and the architectural factors. We have three key observations:

- Fig. 4 presents the Spearman correlation between OA and: depth (*depth*), embedding dimension (*embed_dim*), number of heads (*num_heads*), and MLP ratio (*mlp_ratio*). Notably, factors such as the number of heads and MLP ratio show minimal correlation with overall model performance across all datasets. Conversely, the influence of depth and embedding dimension varies across datasets, with embedding dimension exhibiting particularly high correlations across all datasets, reaching 0.91 on Salinas dataset.
- Fig. 4 presents average head embedding dimension (*mean_head_dim*), average MLP dimension (*mean_mlp_dim*), sum of all head dimensions (*sum_head_dim*), sum of all MLP dimensions (*sum_mlp_dim*), and model size. Notably, average head dimension and average MLP ratio exhibit OA impacts similar to the embedding dimension, while total head dimension and total MLP ratio aggregate influence from both embedding dimension and depth. However, for datasets like PaviaU, where OA relies solely on embedding dimension, the impact of total head dimension diminishes. Conversely, for datasets like Indian Pines, Houston2013, and KSC, where OA correlates with both embedding dimension and depth, Spearman correlation of total head dimension increases. This suggests that if proxy scores correlate solely with one factor, they may not be robust indicators.
- In Fig. 5, we visualize the highest OA of different values of each component across all datasets. Optimal embedding dimension and depth vary across

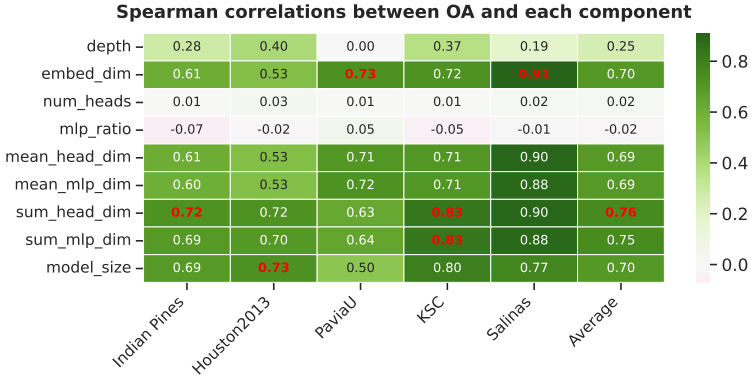

Fig. 4: Spearman correlations between final test OA and components of the search space across five datasets. *num_heads* and *mlp_ratio* exhibit minimal correlation, while *embed_dim*, *sum_head_dim*, and *sum_mlp_dim* show high correlation across all datasets.

datasets. As an example, in Indian Pines, OA increases with the embedding dimension, peaking at $embed\_dim = 128$ and remaining stable thereafter. Similarly, $depth = 8$ results in the highest OA, with no further improvements observed when increasing the depth to 9 or 10. Additionally, the impact of the average number of heads and the average MLP ratio on model performance is showcased in the right two subplots, where mean number of heads of $4-5$ and mean MLP ratio of $3-4$ yield relatively stable and high OA.

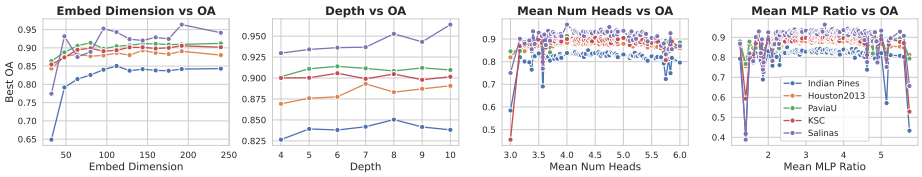

Fig. 5: The best OA corresponding to different values of diverse factors, including the embedding dimension (*embed_dim*), the depth (*depth*), the average number of heads (*mean_heads_num*), and the average MLP ratio (*mean_mlp_ratio*).

Takeaway ②: Across datasets, embedding dimension and depth exhibit consistent sensitivity, though with slight variations. Furthermore, while larger models generally achieve better performance, optimal networks do not necessarily have to be the largest or most complex ones.

5.3 RQ3: What Factors Influence the Proxy Scores?

To answer this question, we analyze the sensitivity of proxy scores to architecture components, input data, and module types separately.

Sensitivity to Architectural Components We examine Spearman correlations between proxy scores and each component on Indian Pines dataset in Fig. 6. Additionally, Spearman correlations between OA and each component are shown in the last column as a comparison. Depth and embedding dimension significantly impact scores of most proxies. Specifically, #Flops, SNIP, and GradNorm display a strong correlation (over 0.95) with the sum of heads' dimensions, surpassing the correlation with OA. Furthermore, Synflow, LogSynflow, and DSS show perfect correlation with embedding dimension and minimal correlation with depth due to taking sign of model parameters, which also exceeds the correlation with OA. Interestingly, removing sign operations improves the performance of Synflow, LogSynflow, and DSS. Details are shown in Section 7.3 in our supplementary material. These discrepancies in the correlation between OA and proxy scores result in a performance gap between proxies and oracle results.

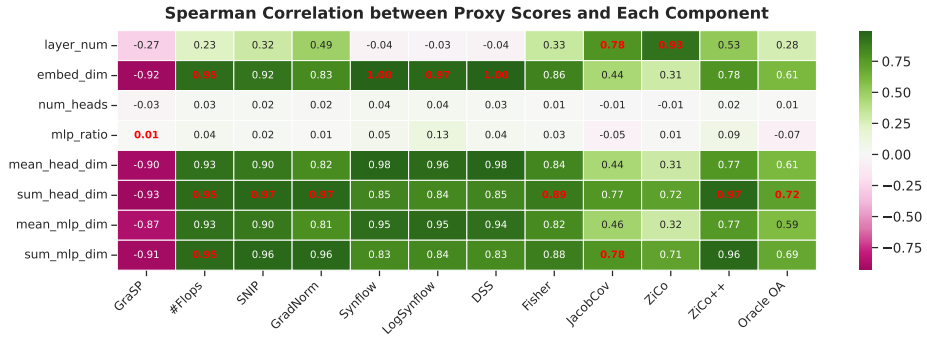

Fig. 6: Spearman correlations between proxy scores and architecture components on Indian Pines dataset.

Sensitivity to Input Data Considering Synflow, LogSynflow, NASWOT, and DSS are data-agnostic, we explore the sensitivity of remaining proxies to input data. To investigate this, we conduct search experiments using one batch of random input data across the remaining proxies. The results, presented in Table 4, reveal that performing the search with random inputs yields outcomes very similar to those obtained with the Indian Pines dataset. This suggests that the majority of the proxies exhibit negligible dependency on specific input data.

Furthermore, we compare OA, SNIP, and GradNorm scores from the same models between PaviaU and Indian Pines datasets in Fig. 7. The plots of SNIP and GradNorm indicate a perfect correlation between scores across different

Proxy	OA		ρ	
	Indian Pines	Random Input	Indian Pines	Random Input
GradNorm	0.81	0.81	0.71	0.71
SNIP	0.81	0.81	0.70	0.69 (-0.01)
GraSP	0.42	0.44 (+0.02)	-0.67	-0.70 (-0.03)
Fisher	0.81	0.82 (+0.01)	0.61	0.55 (-0.06)
JacobCov	0.82	0.79 (-0.03)	0.58	0.68 (+0.1)
ZiCo	0.81	0.79 (-0.02)	0.52	0.52

Table 4: Comparison between searching with a batch of Indian Pines inputs and random inputs.

datasets, suggesting that the proxies are largely independent of the input data. However, the plot of OA shows significant variation between the two datasets, indicating that the same architecture yields very different performances depending on the dataset. This finding further emphasizes the gap between proxies and model performance.

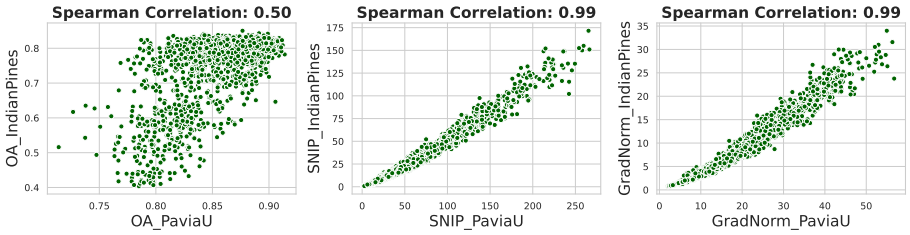

Fig. 7: Comparisons of OA, SNIP, GradNorm of same subnetworks between PaviaU and Indian Pines datasets.

Sensitivity to Module Type (MSA vs. MLP) Given the distinct structures of MSA and MLP modules within a transformer architecture, we hypothesize that the effectiveness of the two modules for proxy scores might differ significantly. To validate our assumption, we separately compute DSS, Synflow, and SNIP scores for MSA and MLP modules. Our analysis reveals a significant score scale difference between MSA and MLP. For instance, for DSS scores, MSA (2472 ± 259) is notably higher than MLP (268 ± 366), indicating the predominant contribution of MSA modules to final proxy scores, with minimal influence from MLP modules. To address this imbalance, we apply a logarithmic transformation to both module scores. The transformed results demonstrate comparable scores for MSA and MLP modules across all three proxies. Detailed scores are presented in Section 7.4 in our supplementary material.

Furthermore, we assess the effectiveness of scores with and without logarithmic transformation, as shown in Table 5. The origin score is represented as $origin\_score = msa\_score + mlp\_score$, and the new score is represented as

$logarithm\_score = \log(msa\_score \times mlp\_score)$, where msa_score represents the cumulative score of all MSA layers, and mlp_score denotes the cumulative score of all MLP layers. The results indicate improvements with the logarithmic transformation, particularly for Synflow and DSS proxies.

Dataset	*SNIP*		*GradNorm*		*Synflow*		*DSS*	
	origin	**logarithm**	**origin**	**logarithm**	**origin**	**logarithm**	**origin**	**logarithm**
Indian Pines	0.70	0.70	0.71	0.71	0.69	**0.72**	0.69	**0.70**
Houston2013	**0.66**	0.65	0.70	0.70	0.68	**0.73**	0.67	**0.69**
PaviaU	0.69	**0.70**	0.62	**0.64**	**0.69**	0.63	**0.70**	0.69
KSC	**0.83**	0.82	0.84	0.84	0.82	**0.86**	0.82	**0.84**
Salinas	0.94	0.94	0.87	**0.90**	**0.93**	0.90	0.94	0.94

Table 5: Comparison of Spearman correlations between proxy scores and OA, with and without logarithmic transformation.

Takeaway ③: Proxies exhibit a higher correlation with embedding dimension and depth than actual model performance, suggesting limited sensitivity to input data. Enhancing proxies' robustness may involve integrating more input information and decreasing sensitivity to individual architecture components. Furthermore, assessing scores independently for MSA and MLP modules, or employing distinct metrics for each, could improve proxy performance.

5.4 RQ4: Are Proxies Complementary?

Motivation. Our results indicate that TAS proxies are fast but may lack reliability, with search performance varying across datasets. Therefore, TAS proxies solely remain problematic. Some studies have employed proxies for pre-filtering or combined them with other search methods, such as one-shot search, evolutionary algorithms, or Bayesian optimization [33,34]. Here, we propose an alternative approach to leverage proxies for enhancing search performance.

Setup. We seek to forecast a network's performance using its architectural structure and proxy scores. By retraining all sampled networks and collecting their proxy scores, we explore the feasibility of predicting a model's OA, alongside exploring the number of training samples needed for accurate predictions. We apply a default Random Forest model, with inputs comprising 2000 samples. Each sample represents a subnetwork sampled from the search space, encompassing attributes such as $depth$, $embed\_dim$, num_heads, mlp_ratio, and various proxy scores including SNIP, GradNorm, Synflow, DSS, ZiCo, and Fisher scores. The target variable is OA, derived from the Indian Pines dataset. Detailed settings are presented in Section 7.5 in our supplementary material.

Results. Fig. 8 displays the test outcomes. The left subplot presents the predicted OA alongside the actual OA for 1950 networks trained on 50 networks.

The Spearman correlation between actual and predicted OA reaches 0.80, surpassing the best correlation obtained from proxies by 9%. Additionally, the right subplot demonstrates a correlation improvement with an increase in the number of training networks. Each result represents the mean and standard deviation across five runs with random seeds. Notably, the correlation rises to 0.78, a 6% increase over the untrained scenario, with only 20 networks trained. It is worth noting that training the Random Forest model takes less than one minute with a training size of 0.5x, making the cost almost negligible compared to training the original subnetworks with actual input data.

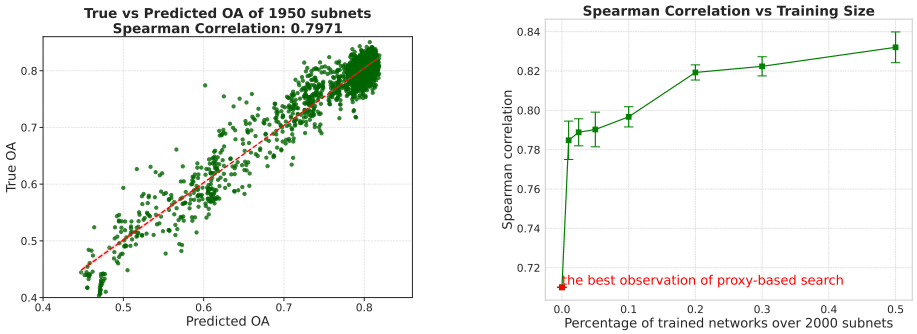

Fig. 8: Predicting subnetworks' performance using Random Forest model.

Takeaway ④: Proxies, while not entirely reliable on their own, can complement other architecture search methods to improve search efficiency or serve as training data for models predicting network performance to improve search accuracy.

6 Conclusion

In this study, we introduced HyTAS: a benchmark for Hyperspectral Image Transformer Architecture Search. We defined a search space comprising 2000 Hyperspectral image transformers and evaluated 12 proxies across five HSI datasets. Our key observations are as follows: Most proxies can identify a transformer surpassing the human-crafted architecture, SpectralFormer, within 10 minutes. However, there's a significant performance gap between proxies and optimal results, driven by preferences for larger and more complex models, minimal dependence on input data, and disparities between MSA and MLP modules. Our proposed proxy, ZiCo^{++}, demonstrates superior performance compared to ZiCo and other proxies. Furthermore, proxies can enhance the prediction of subnetworks' performance when used as input data. These findings encourage the TAS community to explore new methods for improving search quality and efficiency.

Acknowledgements

This work was funded by the Dutch Science Foundation (NWO) under grant 482.20.700. Furthermore, we deeply appreciate the invaluable and constructive feedback offered by our anonymous reviewers and the meta-reviewer. Their insights have greatly enriched our manuscript.

References

1. Abdelfattah, M.S., Mehrotra, A., Dudziak, Ł., Lane, N.D.: Zero-cost proxies for lightweight nas. arXiv preprint arXiv:2101.08134 (2021)
2. Baker, B., Gupta, O., Naik, N., Raskar, R.: Designing neural network architectures using reinforcement learning. arXiv preprint arXiv:1611.02167 (2016)
3. Cavagnero, N., Robbiano, L., Caputo, B., Averta, G.: Freerea: Training-free evolution-based architecture search. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1493–1502 (2023)
4. Chen, B., Li, P., Li, C., Li, B., Bai, L., Lin, C., Sun, M., Yan, J., Ouyang, W.: Glit: Neural architecture search for global and local image transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12–21 (2021)
5. Chen, M., Peng, H., Fu, J., Ling, H.: Autoformer: Searching transformers for visual recognition. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 12270–12280 (2021)
6. Dale, L.M., Thewis, A., Boudry, C., Rotar, I., Dardenne, P., Baeten, V., Pierna, J.A.F.: Hyperspectral imaging applications in agriculture and agro-food product quality and safety control: A review. Applied Spectroscopy Reviews **48**(2), 142–159 (2013)
7. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
8. Furbank, R.T., Silva-Perez, V., Evans, J.R., Condon, A.G., Estavillo, G.M., He, W., Newman, S., Poiré, R., Hall, A., He, Z.: Wheat physiology predictor: predicting physiological traits in wheat from hyperspectral reflectance measurements using deep learning. Plant Methods **17**, 1–15 (2021)
9. Goetz, A.F.: Three decades of hyperspectral remote sensing of the earth: A personal view. Remote sensing of environment **113**, S5–S16 (2009)
10. Graña, M., Veganzons, M., Ayerdi, B.: Hyperspectral remote sensing scenes. Hyperspectral Remote Sensing Scenes - Grupo de Inteligencia Computacional (GIC) `https://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes`
11. Ha, H., Kim, M., Hwang, S.J.: Generalizable lightweight proxy for robust nas against diverse perturbations. arXiv preprint arXiv:2306.05031 (2023)
12. Hong, D., Han, Z., Yao, J., Gao, L., Zhang, B., Plaza, A., Chanussot, J.: Spectralformer: Rethinking hyperspectral image classification with transformers. IEEE Transactions on Geoscience and Remote Sensing **60**, 1–15 (2021)
13. Jakubczyk, K., Siemiątkowska, B., Więckowski, R., Rapcewicz, J.: Hyperspectral imaging for mobile robot navigation. Sensors **23**(1), 383 (2022)
14. Lee, N., Ajanthan, T., Torr, P.H.: Snip: Single-shot network pruning based on connection sensitivity. arXiv preprint arXiv:1810.02340 (2018)

15. Li, G., Hoang, D., Bhardwaj, K., Lin, M., Wang, Z., Marculescu, R.: Zero-shot neural architecture search: Challenges, solutions, and opportunities. arXiv preprint arXiv:2307.01998 (2023)
16. Li, G., Yang, Y., Bhardwaj, K., Marculescu, R.: Zico: Zero-shot nas via inverse coefficient of variation on gradients. arXiv preprint arXiv:2301.11300 (2023)
17. Li, S., Song, W., Fang, L., Chen, Y., Ghamisi, P., Benediktsson, J.A.: Deep learning for hyperspectral image classification: An overview. IEEE Transactions on Geoscience and Remote Sensing **57**(9), 6690–6709 (2019)
18. Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.J., Fei-Fei, L., Yuille, A., Huang, J., Murphy, K.: Progressive neural architecture search. In: Proceedings of the European conference on computer vision (ECCV). pp. 19–34 (2018)
19. Lopes, V., Alirezazadeh, S., Alexandre, L.A.: Epe-nas: Efficient performance estimation without training for neural architecture search. In: International Conference on Artificial Neural Networks. pp. 552–563. Springer (2021)
20. Mellor, J., Turner, J., Storkey, A., Crowley, E.J.: Neural architecture search without training. In: International Conference on Machine Learning. pp. 7588–7598. PMLR (2021)
21. Nagasubramanian, K., Jones, S., Singh, A.K., Sarkar, S., Singh, A., Ganapathysubramanian, B.: Plant disease identification using explainable 3d deep learning on hyperspectral images. Plant methods **15**, 1–10 (2019)
22. Pham, H., Guan, M., Zoph, B., Le, Q., Dean, J.: Efficient neural architecture search via parameters sharing. In: International conference on machine learning. pp. 4095–4104. PMLR (2018)
23. Su, X., You, S., Xie, J., Zheng, M., Wang, F., Qian, C., Zhang, C., Wang, X., Xu, C.: Vitas: Vision transformer architecture search. In: European Conference on Computer Vision. pp. 139–157. Springer (2022)
24. Sun, L., Zhao, G., Zheng, Y., Wu, Z.: Spectral–spatial feature tokenization transformer for hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing **60**, 1–14 (2022)
25. Tanaka, H., Kunin, D., Yamins, D.L., Ganguli, S.: Pruning neural networks without any data by iteratively conserving synaptic flow. Advances in neural information processing systems **33**, 6377–6389 (2020)
26. Theis, L., Korshunova, I., Tejani, A., Huszár, F.: Faster gaze prediction with dense networks and fisher pruning. arXiv preprint arXiv:1801.05787 (2018)
27. Transon, J., d'Andrimont, R., Maugnard, A., Defourny, P.: Survey of hyperspectral earth observation applications from space in the sentinel-2 context. Remote Sensing **10**(2), 157 (2018)
28. Trierscheid, M., Pellenz, J., Paulus, D., Balthasar, D.: Hyperspectral imaging or victim detection with rescue robots. In: 2008 IEEE International Workshop on Safety, Security and Rescue Robotics. pp. 7–12. IEEE (2008)
29. Tsou, C.H., Lin, Y.C., Huang, Y.C., Li, W., Chen, J.C., Chen, C.S.: SEEKING THE SEARCH SPACE FOR SIZE-AWARE VISION TRANSFORMER ARCHITECTURE (2024), https://openreview.net/forum?id=iSdHl6qEs2
30. Turner, J., Crowley, E.J., O'Boyle, M., Storkey, A., Gray, G.: Blockswap: Fisher-guided block substitution for network compression on a budget. arXiv preprint arXiv:1906.04113 (2019)
31. Wang, C., Zhang, G., Grosse, R.: Picking winning tickets before training by preserving gradient flow. arXiv preprint arXiv:2002.07376 (2020)
32. Wang, D., Vinson, R., Holmes, M., Seibel, G., Bechar, A., Nof, S., Tao, Y.: Early detection of tomato spotted wilt virus by hyperspectral imaging and outlier removal

auxiliary classifier generative adversarial nets (or-ac-gan). Scientific reports **9**(1), 1–14 (2019)
33. White, C., Safari, M., Sukthanker, R., Ru, B., Elsken, T., Zela, A., Dey, D., Hutter, F.: Neural architecture search: Insights from 1000 papers. arXiv preprint arXiv:2301.08727 (2023)
34. Xiang, L., Dudziak, L., Abdelfattah, M.S., Chau, T., Lane, N.D., Wen, H.: Zero-cost operation scoring in differentiable architecture search. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 10453–10463 (2023)
35. Xiang, L., Hunter, R., Xu, M., Dudziak, Ł., Wen, H.: Exploiting network compressibility and topology in zero-cost nas. In: International Conference on Automated Machine Learning. pp. 18–1. PMLR (2023)
36. Xie, C., Shao, Y., Li, X., He, Y.: Detection of early blight and late blight diseases on tomato leaves using hyperspectral imaging. Scientific reports **5**(1), 1–11 (2015)
37. Xiong, Z., Minshan, C., Abhinav, S., Díaz, D.J.C.F.: 2013 IEEE GRSS Data Fusion Contest – Fusion of Hyperspectral and LiDAR Data (2013), `https://hyperspectral.ee.uh.edu/?page_id=459`
38. Zhou, F., Kilickaya, M., Vanschoren, J.: Locality-aware hyperspectral classification. arXiv preprint arXiv:2309.01561 (2023)
39. Zhou, Q., Sheng, K., Zheng, X., Li, K., Sun, X., Tian, Y., Chen, J., Ji, R.: Training-free transformer architecture search. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10894–10903 (2022)
40. Zoph, B., Le, Q.V.: Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578 (2016)