ChA-MAEViT: Unifying Channel-Aware Masked Autoencoders and Multi-Channel Vision Transformers for Improved Cross-Channel Learning

Chau Pham
Boston University
chaupham@bu.edu

Juan C. Caicedo
Morgridge Institute, UW–Madison
juan.caicedo@wisc.edu

Bryan A. Plummer
Boston University
bplum@bu.edu

Abstract

Prior work using Masked Autoencoders (MAEs) typically relies on random patch masking based on the assumption that images have significant redundancies across different channels, allowing for the reconstruction of masked content using cross-channel correlations. However, this assumption does not hold in Multi-Channel Imaging (MCI), where channels may provide complementary information with minimal feature overlap. Thus, these MAEs primarily learn local structures within individual channels from patch reconstruction, failing to fully leverage cross-channel interactions and limiting their MCI effectiveness. In this paper, we present ChA-MAEViT, an MAE-based method that enhances feature learning across MCI channels via four key strategies: (1) dynamic channel-patch masking, which compels the model to reconstruct missing channels in addition to masked patches, thereby enhancing cross-channel dependencies and improving robustness to varying channel configurations; (2) memory tokens, which serve as long-term memory aids to promote information sharing across channels, addressing the challenges of reconstructing structurally diverse channels; (3) hybrid token fusion module, which merges fine-grained patch tokens with a global class token to capture richer representations; and (4) Channel-Aware Decoder, a lightweight decoder utilizes channel tokens to effectively reconstruct image patches. Experiments on satellite and microscopy datasets, CHAMMI, JUMP-CP, and So2Sat, show that ChA-MAEViT significantly outperforms state-of-the-art MCI-ViTs by 3.0-21.5%, highlighting the importance of cross-channel interactions in MCI. Our code is publicly available at https://github.com/chaudatascience/cha_mae_vit.

1 Introduction

Visual encoders generally process fixed-channel inputs, such as RGB, during both training and testing phases [2–9]. However, in satellite imaging [10], robotic sensing [11], cell microscopy [12, 13], and medical imaging [14], the input data can vary in both the number and type of channels. These varying channel configurations arise from differences in sensor modalities, acquisition settings, or experimental conditions. Multi-Channel Imaging (MCI) models are designed to learn feature representations from heterogeneous channels whose type and number vary during both training and testing [12, 13]. This adaptability allows a single model to effectively support diverse channel configurations, thereby reducing computational costs and minimizing the risk of overfitting [12].

Prior MCI-Masked Autoencoder (MAE) models (*e.g.*, [1, 15]) demonstrated promise in capturing local spatial structures by learning to reconstruct randomly masked patches in multi-channel images. These approaches assume that images exhibit significant redundancy across channels, enabling the reconstruction of masked patches using unmasked patches from similar channels. While this

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

https://doi.org/10.52202/085713-0401

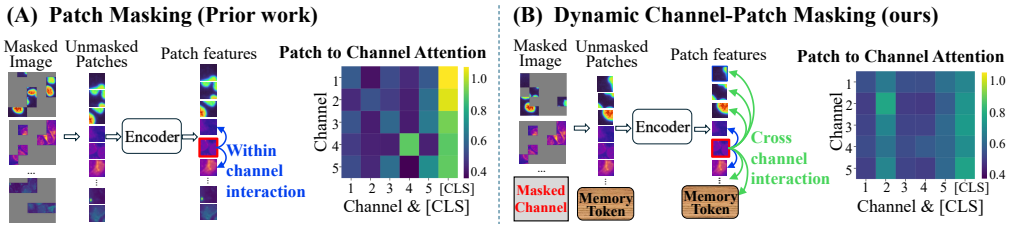

Figure 1: **Image patch interactions in MCI. (A) Prior Work on MCI-MAEs** (*e.g.*, CA-MAE [1]) employ random patch masking to train the model to reconstruct the masked patches. In the attention map, where each row represents the average attention score of all patches in a channel towards other channels and the [CLS] token (last column), we show each patch primarily attends to its own channel (the diagonal) and the [CLS] token. This suggests that patch masking may not effectively promote cross-channel interactions in MCI. **(B) In contrast, Dynamic Channel-Patch Masking (ours)** encourages more interactions between patches across different channels by using both channel and patch masking. Also, *memory tokens* serve as long-term memory to help information aggregation across channels. Its attention pattern demonstrates a more uniform distribution across channels, indicating that each image patch can learn more meaningful interactions.

assumption works well for natural images, where color channels typically show strong correlations, it presents challenges in MCI scenarios. In these situations, complementary sensors (*e.g.*, multispectral and LiDAR) may capture distinct physical properties with minimal feature overlap. As shown in Fig. 1(a), when using patch masking, the patch-to-channel attention concentrates on its own channel (the diagonal) and the [CLS] token. This indicates that prior MAE methods mainly focus on visible patches within the same channel, neglecting cross-channel feature interactions, restricting the model's ability to learn useful features that require information from multiple channels.

To address this challenge, we introduce **Ch**annel-**A**ware **MAE** – multichannel **Vi**sion **T**ransformer (ChA-MAEViT), which enhances cross-channel learning in MCI through four key improvements summarized in Fig. 2. First, we propose Dynamic Channel-Patch (DCP) Masking, which adaptively masks both patches and channels during training, compelling the model to reconstruct these missing patches and channels using the remaining patches (Fig. 2, left). DCP Masking adjusts the channel masking ratio to enhance feature learning and robustness to missing channels during inference. Fig. 1(b) shows our approach evenly redistributes patch attention scores across channels, highlighting improved cross-channel interactions. However, in MCI, reconstructing masked channels is challenging since each channel may encode unique features that are not easily inferred from others.

To enable the model to retain global information across all channels during reconstruction, we introduce the use of *memory tokens* in MCI (Fig. 2, middle). Inspired by register tokens [16], which are extra tokens added in the input to reduce artifacts in the feature maps of ViTs, these learnable embeddings serve as long-term memory to retain key cross-channel information in both the encoder and decoder. In addition, we use a hybrid token fusion module in the encoder to combine fine-grained patch tokens with the global class token for a richer representation (Fig. 2, middle).

Finally, we employ Channel-Aware Decoder that simultaneously processes patch tokens from all channels. Existing methods often rely on separate decoders for each channel [1, 17, 18], which do not scale well with many input channels. Our shared lightweight decoder utilizes channel tokens to provide channel-specific behavior and memory tokens to improve cross-channel feature reconstruction, enhancing performance while reducing computational costs.

Most of our experiments use both self-supervised (SSL) and supervised learning to boost performance, showing a similar complementary benefit from combining them (*e.g.*, [19–22]). However, Appendix F also evaluates SSL by itself, where we show our approach still outperforms prior work.

The most relevant work to ours focuses on enhancing cross-channel interactions [23–26]. However, these approaches mainly use single-modality images, such as RGB, where channel similarities facilitate strong correlations. As noted earlier, this often does not generalize to the more complex MCI domain where channels convey varying information, *e.g.*, a robot sensor can contain LiDAR, RGB, and thermal camera. Thus, MCI requires a balance between preserving unique information in each channel and effectively modeling the complex relationships between channels. MCI models must also be robust to missing channels to support varying channel configurations [12, 13]. This means that cross-channel interaction should not only improve the overall representation but also

https://doi.org/10.52202/085713-0401

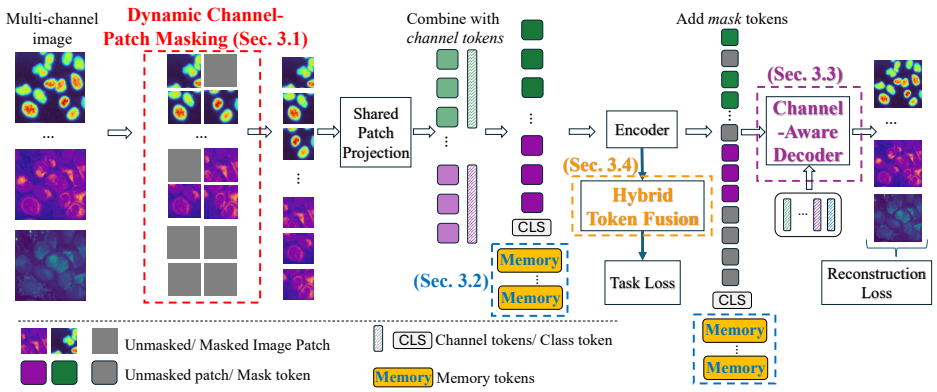

Figure 2: Our **ChA-MAEViT** approach enhances cross-channel learning via four key components: 1) **Dynamic Channel-Patch Masking**, which compels the model to reconstruct varying proportions of missing channels and patches, thus improving interactions across channels and robustness to the absence of some channels (Section 3.1). 2) **Memory Tokens**, which act as long-term memory to facilitate information sharing between channels (Section 3.2). 3) To reconstruct the masked patches and channels, we use a **Channel-Aware Decoder** that leverages channel tokens for image reconstruction, enhancing performance while minimizing computational costs (Section 3.3). 4) A **Hybrid Token Fusion** module, which combines fine-grained patch tokens with a global [CLS] token to improve feature representation (Section 3.4).

involve learning useful redundancy across channels. ChA-MAEViT addresses these challenges by introducing DCP Masking to enhance cross-channel interaction while allowing the model to learn important channel-specific features, resulting in a more robust and higher-performing model.

Our key contributions are as follows:

- We explore ChA-MAEViT, which integrates MAEs into MCI-ViTs for improved robustness that reports a gain of $3.0 - 21.5\%$ across three diverse MCI datasets, CHAMMI [12], JUMP-CP [27], So2Sat [10], and consistent gains on Cloud-38 [28].
- We propose Dynamic Channel-Patch Masking, which adaptively masks both channels and patches during training, requiring the model to reconstruct them to boost cross-channel interaction.
- We introduce a Channel-Aware Decoder that uses channel tokens to reconstruct image patches more efficiently, thereby enhancing performance for downstream tasks.
- We use *memory tokens* – learnable vectors to retain cross-channel information and aggregate channel-specific features, and *hybrid token fusion module* to merge the [CLS] token with fine-grained patch tokens for enhanced feature representation.

2 Related Work

Many MCI methods do not use self-supervised objectives [13, 29, 30], but rather focus on aspects like diversity and robustness to missing channels. However, as noted in the introduction, self-supervised objectives like Masked Autoencoders (MAEs) can provide complementary information to further boost performance. MAEs themselves are often built for RGB images (*e.g.*, [19, 31, 32]) or those that are also strongly aligned like thermal images [18], based on assumptions of strong correlations between channels does not extend to many MCI images. Many methods in other domains followed suit, performing channel-wise masking on protein markers [33] or masked both of spatial and spectral channels [34, 35]. However, these methods rely on a fixed ratio of masked channels, which can be inadequate for the varying channel configurations found in MCI. In addition, prior work uses dual-branch architectures that treats spatial and channel representations independently, limiting meaningful interactions between channels, especially when spatial alignment is not guaranteed. In contrast, we dynamically mask a variable number of channels each time. By combining both channel and patch-level masking within a single encoder, we achieve better feature interaction.

The work most similar to ours is CA-MAE [1], which randomly masks a fixed proportion of patches from all channels and employs distinct decoders for reconstruction. In contrast, ChA-MAEViT employs both channel and patch-level masking with a shared decoder, and memory tokens serve as long-term memory to facilitate feature interactions.

https://doi.org/10.52202/085713-0401

There are other types of self-supervised learning methods beyond MAEs that have been explored in other settings (*e.g.*, [31, 32, 36–45]). However, similar to MAEs, they often make assumptions that do not generalize to MCI images. For example, self-distillation methods (*e.g.*, SimSiam [40], BYOL [41], DINO [42]) process two different views through two encoders, and then map one view to the other using a predictor network. These methods often rely on complex augmentations designed for natural images to generate dual views, such as color jittering and Gaussian blur. These are less effective for MCI, where heterogeneous channels (*e.g.*, LiDAR, thermal, RGB) carry distinct physical properties. Such augmentations can distort critical features like depth or intensity, hindering the model's ability to capture cross-channel dependencies essential for MCI [46, 47]. This makes masked modeling techniques attractive (*e.g.*, MAE [31], SimMim [32]), as they do not rely on augmentations that may be challenging to apply with more complex and varied images.

3 Channel-Aware MAE for Improved Cross-Channel Learning

Given a multi-channel image (MCI) denoted as X, which consists of various channels $c_i \in C$, our objective is to train a model M that takes X as input to generate representations and/or predictions. Following [12, 13, 29, 30], we focus on the MCI setting where the model M is trained using all available channels C but tested with a subset of those channels $C_{\text{test}} \subseteq C$. We describe the details of the four main components of our proposed framework, which is illustrated in Fig. 2.

3.1 Dynamic Channel-Patch Masking

As shown in Fig. 1(a), image patches in prior work seem to attend mostly to their own channels while neglecting the others, potentially missing rich interactions among channels. To mitigate this issue, we propose Dynamic Channel-Patch (DCP) Masking, which integrates both patch and channel-level masking strategies. We start creating image patches (Appendix B), which results in n tokens per channel, each with d-dimensional embeddings, for c channels.

Our masking strategy consists of two components: *random patch masking* and *dynamic channel masking*. First, *random patch masking*, aims to generate a mask $\mathbf{p\_mask} \in \{0,1\}^{n\times c}$ that applies a fixed masking ratio r_p (*e.g.*, 75%) to mask patches independently across each channel. Specifically, for each channel j, we uniformly sample a set of $\lfloor n \cdot r_p \rfloor$ positions from the n available patch positions, denoted $S_j \subset \{1, 2, \ldots, n\}$ with $|S_j| = \lfloor n \cdot r_p \rfloor$. We then mask these patches as $\mathbf{p\_mask}_{i,j} = \begin{cases} 1 & \text{if } i \in S_j \\ 0 & \text{otherwise} \end{cases}$, where $\mathbf{p\_mask}_{i,j} = 1$ indicates the patch at position i of channel j is masked, and $\mathbf{p\_mask}_{i,j} = 0$ indicates unmasked. All channels have the same number of masked patches, but locations vary per channel because they are independently sampled, as opposed to prior work that first generates the mask for one channel and then replicates it to all other channels [35, 48].

The second component, *dynamic channel masking*, aims to generate a mask $\mathbf{c\_mask} \in \{0,1\}^{n\times c}$ that adaptively mask out some channels. Here, the term "dynamic" refers to the varying number of channels masked during training, rather than adaptation based on the specific input data. We start by uniformly sampling the number of channels to be masked, denoted as $k \sim \mathcal{U}\{0, 1, \ldots, c-1\}$. Then, we uniformly sample a set of k channels, denoted as $\mathcal{C}' \subset \{1, \ldots, c\}$ with $|\mathcal{C}'| = k$, and mask these channels out as $\mathbf{c\_mask}_{\star,j} = \begin{cases} \mathbf{1}^n & \text{if } j \in \mathcal{C}' \\ \mathbf{0}^n & \text{otherwise} \end{cases}$. This masking approach is inspired by Hierarchical Channel Sampling (HCS) [30]. However, unlike HCS, which serves as a channel dropout technique that completely removes selected channels, our approach employs masked channels as supervisory signals, designating them as labels for the reconstruction process. This enables the model to directly learn inter-channel relationships, thus enhancing greater cross-channel feature interaction.

Finally, we integrate these two components for training images in our DCP Masking strategy by introducing hyperparameters $\mathrm{p_{patch}}, \mathrm{p_{channel}} \in (0,1)$. These values divide the unit interval into three sections to randomly determine how to use one or the other type of mask, as follows:

$$\mathbf{mask} = \begin{cases} \mathbf{p\_mask} & \text{if } s < \mathrm{p_{patch}} \\ \mathbf{c\_mask} & \text{if } \mathrm{p_{patch}} \le s < \mathrm{p_{patch}} + \mathrm{p_{channel}} \\ \mathbf{p\_mask} \vee \mathbf{c\_mask} & \text{otherwise} \end{cases} \tag{1}$$

where s is a selection value chosen uniformly at random. In practice, we found that optimizing the model with these two masking strategies simultaneously can be difficult, as it may lead to excessive information loss, making image reconstruction challenging. To address this, we adjust the hyperparameters to alternate between the two masking strategies separately. Specifically, setting $\mathrm{p_{patch}} = \mathrm{p_{channel}} = 0$ merges both patch and channel masks into a unified mask, and setting $\mathrm{p_{patch}} = \mathrm{p_{channel}} = 0.5$ allows the model to switch between patch and channel masks. We adopt these two straightforward configurations for all our experiments. Refer to Appendix D for the procedure of DCP Masking, and Appendix F.5 for more analysis on the hyper-parameters.

3.2 Memory Tokens

Reconstructing masked channels is challenging due to their inherent differences, since each channel encodes unique features not easily inferred from others, *e.g.*, reconstructing LIDAR from RGB. Inspired by the concept of register tokens [16], which are used to reduce artifacts in the feature maps of ViTs, we introduce *memory tokens* for MCI (Fig. 2, middle). These memory tokens are learnable embeddings that serve as long-term memory, allowing for the storage of global information across all channels. During training, these tokens gather channel-specific features using self-attention mechanisms, helping the model retain and propagate information across layers that might otherwise be lost due to masking. Additionally, these tokens assist in decoding image patches to facilitate the reconstruction process. During inference, memory tokens enable the model to retrieve stored context, effectively addressing the issue of missing channels. Similar to the $[\mathrm{CLS}]$ token, memory tokens are incorporated into the input during both training and inference. However, while the $[\mathrm{CLS}]$ token is utilized as the final representation, the memory tokens are excluded.

Formally, we prepend l memory tokens $\{\mathbf{M_i}\}_{i\in[l]}$ into the input sequence, resulting in $[\mathbf{t}_{\mathrm{CLS}}; \mathbf{M}_1; \ldots; \mathbf{M}_l; \mathbf{t}_{1,1}; \mathbf{t}_{2,1}; \ldots; \mathbf{t}_{n,c}]$, where $\mathbf{t}_{\mathrm{CLS}}$ and $\mathbf{t}_{i,j}$ are the class token and patch token at i-th location of j-th channel, respectively. After masking some patches (Section 3.1), the remaining patches are fed into a transformer encoder. Following [29, 30], spatial information is incorporated through learnable positional embeddings, while channel-specific properties are captured by special *channel tokens*, each represented as a learnable embedding. These channel tokens are concatenated with the patch tokens and jointly processed by the transformer encoder and decoder.

3.3 Channel-Aware Decoder

After processing the unmasked tokens with the encoder, we reconstruct the masked patches using the unmasked ones with a $\mathrm{Channel\text{-}Aware\ Decoder}$. Unlike prior MCI-MAE methods that use separate decoders for each channel or modality [1, 17, 18], we employ a single decoder to process tokens from all channels simultaneously (Fig. 2, right). This scales better to the number of input channels (up to 18 in our experiments), while also boosting performance.

Let the output sequence from the encoder be $[\hat{\mathbf{t}}_{\mathrm{CLS}}; \hat{\mathbf{M}}_1; \ldots; \hat{\mathbf{M}}_l; \hat{\mathbf{t}}_1; \hat{\mathbf{t}}_2; \ldots; \hat{\mathbf{t}}_v]$, where v denotes the number of visible (*i.e.*, unmasked) patches fed into the encoder. This sequence is shorter than the original image patch length due to the missing masked patches. To reconstruct these masked patches, we utilize $u = (n \cdot c - v)$ mask tokens $\mathbf{m}_i$, where each mask token is a shared, learned vector representing the missing patch.

Incorporating Channel-Specific Information. To reconstruct channel-specific information, we combine patch tokens, whether visible ($\hat{\mathbf{t}}_i$) or masked ($\mathbf{m}_i$), with their corresponding *channel tokens*, which are also optimized by the encoder. We define $f(\cdot)$ as a function that returns the corresponding *channel token* for a given patch token. The input to our Channel-Aware Decoder is thus:

$$\hat{\mathbf{T}} = [\hat{\mathbf{t}}_{\mathrm{CLS}}; \hat{\mathbf{M}}_1; \ldots; \hat{\mathbf{M}}_l; \hat{\mathbf{t}}_1 + f(\hat{\mathbf{t}}_1); \hat{\mathbf{t}}_2 + f(\hat{\mathbf{t}}_2); \ldots; \hat{\mathbf{t}}_v + f(\hat{\mathbf{t}}_v); \mathbf{m}_1 + f(\mathbf{m}_1), \ldots, \mathbf{m}_u + f(\mathbf{m}_u)]$$

Incorporating channel tokens provides a more channel-specific context, which enhances the reconstruction process. Additionally, we incorporate learnable positional embeddings, which are shared with the encoder, to provide information about the position of each patch in the image.

Patch Reconstruction. We pass the token sequence $\hat{\mathbf{T}}$ into several Transformer Blocks, followed by a Decoder Head to reconstruct the pixels of each image patch. Let $\mathbf{T} \in \mathbb{R}^{(n\cdot c)\times d}$ represent the output of the patch tokens from the final Transformer Block. The Decoder Head is a linear layer $\mathbf{W}_{\mathrm{decoder}} \in \mathbb{R}^{d\times p^2}$, where p is the image patch size. The pixel reconstruction of the i-th patch of the

 https://doi.org/10.52202/085713-0401

j-th channel, is computed as $\hat{\mathbf{x}}_{i,j} = (\mathbf{T}_{i,j} \cdot \mathbf{W}_{\text{decoder}}) \in \mathbb{R}^{p^2}$. We found that a lightweight decoder with just $1-2$ Transformer Blocks is sufficient, aligning with findings from prior work [19, 31].

Reconstruction Loss. We use standard $L2$ loss on the image pixel, and the Fourier space $L1$ loss introduced in [1], both computed only on the masked patches. Let $P = \sum_{i=1}^{n}\sum_{j=1}^{c} \mathbf{mask}_{i,j}$ be the number of masked patches of the input image. Given $\mathbf{x}_{i,j}, \hat{\mathbf{x}}_{i,j} \in \mathbb{R}^{p^2}$, the original pixels of patch i in channel j, and its reconstruction, respectively, the reconstruction loss is as follows: $\mathcal{L}_{\text{pixel}} = \frac{1}{P}\sum_{i=1}^{n}\sum_{j=1}^{c} \mathbf{mask}_{i,j} \cdot L_2(\mathbf{x}_{i,j}, \hat{\mathbf{x}}_{i,j})$; $\mathcal{L}_{\text{fourier}} = \frac{1}{P}\sum_{i=1}^{n}\sum_{j=1}^{c} \mathbf{mask}_{i,j} \cdot L_1(|\mathcal{F}(\mathbf{x}_{i,j})|, |\mathcal{F}(\hat{\mathbf{x}}_{i,j})|)$

$$\mathcal{L}_{\text{recon}} = (1-\lambda_f)\cdot \mathcal{L}_{\text{pixel}} + \lambda_f \cdot \mathcal{L}_{\text{fourier}}, \tag{2}$$

where $|\mathcal{F}|$ is the amplitudes of Fast Fourier Transform. Following [1], we set a fixed λ_f to 0.01.

3.4 Hybrid Token Fusion

Prior work either utilizes the $[\text{CLS}]$ token [16, 49] or average pooling of patch tokens [1, 19] for downstream tasks. We found that incorporating both in our framework yields better performance (Fig. 2, middle). Given the output sequence from the encoder, we use a learnable query vector $\mathbf{q}_{\text{patch}}$ to interact with the patch tokens. The query attends to the patch tokens through cross-attention, and then combines with the $[\text{CLS}]$ token to generate a fused representation $f_{\text{fusion}} = \hat{t}_{\text{CLS}} \odot \sigma(\text{CrossAttention}(\mathbf{q}_{\text{patch}}, \mathbf{T}_p))$, where $\mathbf{T}_p = [\hat{\mathbf{t}}_1; \hat{\mathbf{t}}_2; \ldots; \hat{\mathbf{t}}_v]$, $\odot$ represents the elementwise product, and σ the sigmoid function. The fused representation, f_{fusion}, integrates global context from the $[\text{CLS}]$ token while also being refined by detailed spatial information from the patch tokens. We then use a multi-layer perceptron (MLP) with GELU activation [50] to enhance the fusion representation and tailor it for downstream tasks: $f_{\text{final}} = \text{Linear}(\text{GELU}(\text{Linear}(f_{\text{fusion}})))$.

Training Objective. The output of Section 3.4 is then used to compute the task loss, *e.g.*, cross-entropy for classification. Our final loss consists of three components: the primary loss for the specific task $\mathcal{L}_{\text{task}}$, regularization term $\mathcal{L}_{\text{d}}$, and reconstruction loss $\mathcal{L}_{\text{recon}}$ (Eq. (2)) as follows:

$$\mathcal{L}_{\text{final}} = (1-\lambda_{\text{recon}})\cdot(\mathcal{L}_{\text{task}} + \lambda_{\text{d}} \cdot \mathcal{L}_{\text{d}}) + \lambda_{\text{recon}} \cdot \mathcal{L}_{\text{recon}}, \tag{3}$$

where λ_{d} is to balance the task loss and the regularization term, and λ_{recon} is to balance the reconstruction loss with other losses. For regularization $\mathcal{L}_{\text{d}}$, we use the losses introduced in [29], in which $\text{Channel Diversification Loss}$ applies to the channel tokens, and $\text{Token Diversification Loss}$ applies to the patch tokens to encourage diversity learning. We set a small λ_{d} (*e.g.*, 0.001) as suggested by [29], and a fixed $\lambda_{\text{recon}} = 0.99$ for all experiments.

4 Experiments

Datasets. We evaluate on three cell microscopy and satellite datasets. *(i)* **CHAMMI** [12], a channel-adaptive benchmark consists of varying-channel images sourced from WTC-11, HPA and CP, with 3, 4, and 5 channels respectively. Together, these three datasets contain 220K microscopy images, in which, 100K images are for training, while the rest for testing across domain generalization tasks. *(ii)* **JUMP-CP** [27] is a cellular imaging dataset, where each image has 8 channels, consisting of 5 fluorescence and 3 brightfield channels. We use compound perturbation plate BR00116991, which contains 127K training, 45K validation, and 45K test images, across 161 classes. *(iii)* **So2Sat** [10] contains synthetic aperture radar and multispectral optical image patches from remote sensing satellites, with 18 channels (8 from Sentinel-1, 10 from Sentinel-2) and 17 climate zone classes. We use the city-split version, consisting of 352K training, 24K validation, and 24K test images.

Baseline methods. We compare to the following methods:
- **DepthwiseViT** [12] processes each input channel through a depthwise convolution layer, averages the filtered features, and feeds them into a ViT backbone.
- **TemplateMixingViT** [52–54] learns channel weights via shared parameter templates, forming a patch projection layer for a ViT backbone.
- **HyperNetViT** [51] uses a neural network to generate channel-specific weights, concatenating them into a patch projection layer for a ViT backbone.
- **CA-MAE** [1] extends MAE for multi-channel imaging. We add task loss to adapt this baseline.
- **ChAda-ViT** [13] employs a shared projection for channel-wise feature extraction, combining the tokens with positional and channel embeddings for ViT processing.

Table 1: **Comparison of channel adaptive models**. We report the mean accuracy with standard deviation on the test set of three runs. "Full" refers to inference on all channels, while "Partial" means inference on a subset of channels. ChA-MAEViT outperforms other baselines consistently across three datasets. Note: all models contain the same number of parameters, except CA-MAE due to its separate decoders (*e.g.*, $4X$ parameters on So2Sat).

	CHAMMI [12]	JUMP-CP [27]		So2Sat [10]	
Model	Avg score	Full	Partial	Full	Partial
HyperNetViT [51]	56.08±0.41	47.07±0.47	42.43±0.65	60.73±0.24	41.88±0.85
DepthwiseViT [12]	61.80±0.43	49.86±0.45	44.98±0.71	60.41±0.22	43.41±1.10
TemplateMixingViT [52–54]	58.16±0.42	52.48±0.27	43.85±0.73	55.86±0.10	37.28±0.34
CA-MAE [1] + Sup. loss	59.15±0.28	69.54±0.12	20.93±0.25	64.21±0.41	15.75±0.83
ChAda-ViT [13]	63.93±0.42	65.03±0.98	42.15±2.33	56.98±0.46	12.38±2.03
ChannelViT [30]	64.97±0.58	67.51±0.35	56.49±0.53	61.03±0.17	46.16±0.40
DiChaViT [29]	69.77±0.44	69.19±0.47	57.98±0.41	63.36±0.11	47.76±0.23
ChA-MAEViT (ours)	**74.63**±0.54	**90.73**±0.14	**68.05**±0.21	**67.44**±0.38	**52.11**±0.49

- **ChannelViT** [30] Similar to ChAda-ViT, while incorporating Hierarchical Channel Sampling.
- **DiChaViT** [29] improves ChannelViT with regularization terms and diverse channel sampling.

Additionally, we investigate the effect of combining the best supervised baseline, DiChaViT, with the following SSL to evaluate the importance of incorporating self-supervision: MAE [1, 31], SimCLR [37, 38], SimSiam [40], iBOT [43], and DINOv2 [44]. More details in Appendix C.

Implementation details. All baselines employ ViT-S (21M) as the backbone. In our method, we remove one transformer block from the encoder to accommodate the decoder, ensuring that our approach maintains the same number of parameters as the baselines. Refer to Appendix E for details.

Metrics. We reported top-1 classification accuracy for the classification tasks on So2Sat [10] and JUMP-CP [27]. For the representation learning tasks on CHAMMI [12], following [29], we report the average F1 scores across WTC-11 and HPA.

4.1 Results

Table 1 compares ChA-MAEViT with different MCI-ViTs. ChA-MAEViT significantly outperforms the baseline models by an average of 10.0% across three datasets. Specifically, ChA-MAEViT exceeds the state-of-the-art model DiChaViT by 5.0% on CHAMMI, a channel-adaptive benchmark. For JUMP-CP and So2Sat, we conducted evaluations in both *Full* (using all channels) and *Partial* (using subsets of channels) settings. In the full setting, our model surpasses the next best models by 21.5% and 3.0%, respectively. Additionally, ChA-MAEViT demonstrates its robustness in the partial settings, where we test JUMP-CP using five fluorescence channels and So2Sat using eight Sentinel-1 channels, showing improvement of 10.0% and 4.5% points respectively over prior work.

Table 2 presents the impact of integrating various SSL methods with the top-performing supervised approach, DiChaViT [29]. In the last row, we train DiChaViT to reconstruct the masked patches using only our DCP Masking, without incorporating *memory tokens*, Hybrid Token Fusion, or Channel-Aware Decoder. While the combination with MAE gives the highest performance, DCP still significantly outperforms the best SSL-enhanced variant by $0.6 - 5.6\%$ across three datasets. The performance gap highlights the effectiveness of DCP, which substantially exceeds the improvements gained from solely combining with SSL objectives. Additionally, by only processing the unmasked patches, our method achieves significantly faster runtime compared to other SSL methods, *e.g.*, $6X$ faster than DINOv2, making it both performant and computationally efficient for MCI tasks. Refer to Appendix C and Appendix Fig. 6 for more discussion on runtime and FLOPs comparisons.

To further demonstrate our approach's ability to generalize, Table 3 reports segmentation performance on 38-Cloud [28], complementing our results on the classification (So2Sat [10], JUMP-CP [27]) and representation learning tasks (CHAMMI [12]) reported earlier. ChA-MAEViT outperforms all baselines across evaluation metrics, demonstrating its ability to boost multi-channel learning.

https://doi.org/10.52202/085713-0401

Table 2: **SSL methods when combined with the best supervised baseline DiChaViT [29]**. The last row shows the combination of DiChaViT with Dynamic Channel-Patch Masking. Incorporating SSL in DiChaViT results in improvements. Best result with our masking strategy. **Boldfaces** and underlines indicates best and second best numbers, respectively.

DiChaViT [29] +	CHAMMI	JUMP-CP		So2Sat	
		Full	Partial	Full	Partial
SimCLR [37, 38]	70.72±0.40	67.12±0.39	56.96±0.68	64.44±0.58	49.42±0.63
SimSiam [40]	70.44±0.38	68.64±0.56	57.72±0.69	64.07±0.31	48.52±0.71
iBOT [43]	70.71±0.44	68.87±0.51	57.81±0.47	63.11±0.32	47.84±0.54
DINOv2 [44]	70.03±0.28	66.91±0.52	56.18±0.66	63.42±0.27	49.20±0.72
MAE [1, 31]	70.27±0.78	78.62±0.78	64.21±0.73	62.88±0.49	47.76±0.62
DCP Masking	**71.47**±0.53	**84.02**±0.49	**65.72**±0.43	**66.02**±0.22	**50.52**±0.45

Table 3: **Comparing segmentation Performance on 38-Cloud [28].** We report the average and standard deviation from three runs. ChA-MAEViT outperforms all baselines in all metrics.

Model	Accuracy	IoU	Precision	Recall	F1
ChannelViT [30]	0.945±0.003	0.843±0.012	0.919±0.015	0.911±0.004	0.915±0.003
DiChaViT [29]	0.951±0.004	0.857±0.011	0.924±0.007	0.922±0.006	0.923±0.003
CA-MAE [1]	0.946±0.002	0.845±0.003	0.917±0.005	0.916±0.003	0.916±0.002
DiChaViT [29] + CA-MAE [1]	0.959±0.001	0.886±0.004	0.939±0.005	0.943±0.004	0.941±0.002
ChA-MAEViT (ours)	**0.964**±0.001	**0.894**±0.002	**0.943**±0.002	**0.945**±0.001	**0.944**±0.001

Table 4: **ChA-MAEViT ablation study.** Replacing any component from ChA-MAEViT, such as substituting our *DCP Masking* with the random patch masking from CA-MAE [1] ("w/o DCP Masking"), results in a performance drop. The decline is most significant when *DCP Masking* is excluded. Incorporating all components consistently enhances performance across all three datasets.

	CHAMMI [12]	JUMP-CP [27]		So2Sat [10]	
Model	Avg score	Full	Partial	Full	Partial
ChA-MAEViT	**74.63**	**90.73**	**68.05**	**67.44**	**52.11**
w/o DCP Masking	70.51	88.01	52.33	64.50	28.70
w/o Hybrid Token Fusion	73.84	88.25	66.23	65.48	51.40
w/o Memory Tokens	73.62	87.81	67.21	65.18	50.46
w/o Channel-Aware Decoder	72.95	87.52	67.05	65.78	49.88

Ablation study of ChA-MAEViT. Table 4 presents the model's performance when a component is replaced. Specifically, "w/o DCP Masking" indicates that we replace our DCP Masking with random patch masking in CA-MAE [1], "w/o Hybrid Token Fusion" uses [CLS] token instead of Hybrid Token Fusion module, "w/o Memory Tokens" means no memory token is being used, and "w/o Channel-Aware Decoder" indicates replacing our Channel-Aware Decoder with CA-MAE's Separate Decoders. The results highlight the critical role of each of the components, especially DCP Masking, as its removal has the most detrimental effect on performance.

Inference under varying channel configurations. Table 5 evaluates ChA-MAEViT and the best baseline DiChaViT [29] when tested on varying numbers of channels of JUMP-CP. We train the model using all eight channels and assess performance after removing channels, *e.g.*, testing with seven channels, as shown in column "7," we averaged all $C_8^7 = 8$ possible combinations (refer to Appendix Table 9 for detailed results). ChA-MAEViT demonstrates enhanced robustness, particularly when some channels are missing during inference.

Comparing masking strategies. Table 6 compares various masking strategies applied to ChA-MAEViT. For fixed ratio approaches, we test several ratios (*e.g.*, 50%, 75%) and report the best results. DCP Masking consistently outperforms others, achieving highest scores in both *Full* and *Partial* settings. Notably, while random patch masking is effective in *Full*, it experiences a significant performance drop in *Partial*. In contrast, dynamic channel masking methods adapted from [29, 30] provide better adaptability for partial settings, and our DCP Masking approach improves even more.

Table 5: **Performance on varying channel configurations during inference**. Columns report the *mean*(std) across all channel combinations on JUMP-CP [27], *e.g.*, "7" indicates testing on 7 out of 8 channels ($C_8^7 = 8$ combinations). ChA-MAEViT consistently shows improved robustness with missing channels at test time. Note that the reported std reflects variation across channel combinations, not model training variance. Refer to Table 9 in the Appendix for model variance report.

	Number of channels at inference							
Method	8	7	6	5	4	3	2	1
DiChaViT [29]	69.19	61.91(9.3)	54.49(12.4)	46.35(13.4)	38.00(12.4)	30.09(9.3)	23.97(4.9)	20.90(1.6)
ChA-MAEViT	**90.73**	**83.36**(8.3)	**74.55**(11.7)	**63.46**(13.8)	**50.85**(13.9)	**38.13**(11.2)	**27.62**(6.4)	**21.59**(2.1)

Table 6: **Mask strategies in MCI when using with ChA-MAEViT**. While *Random Patch* and *Channel Sampling* perform similarly on Full channels, *Random Patch* experiences a significant drop in Partial channel settings. Dynamic Channel-Patch (DCP) Masking demonstrates its effectiveness in both Full and Partial channel settings, outperforming all strategies across three datasets.

	CHAMMI	JUMP-CP		So2Sat	
Mask Strategy	Avg score	Full	Partial	Full	Partial
Random Patch (fixed ratio) (*e.g.*, [1, 17, 19, 55, 56])	70.51	88.01	52.33	64.50	28.70
Random Patch (dynamic)	68.23	85.54	55.86	64.84	31.92
Channel (fixed ratio) [18]	47.97	73.00	65.25	65.22	38.59
Hierarchical Channel Sampling (dynamic, adapted [30])	69.86	83.71	67.75	65.20	48.77
Diverse Channel Sampling (dynamic, adapted [29])	71.57	84.69	67.86	65.49	51.05
Channel + Patch (fixed ratio) [35, 48]	48.46	69.41	62.19	62.20	32.18
DCP Combination ($\mathbf{p_{channel} = p_{patch} = 0}$) **(ours)**	73.75	85.95	**68.36**	**67.44**	**52.11**
DCP Alternate ($\mathbf{p_{channel} = p_{patch} = 0.5}$) **(ours)**	**74.63**	**90.73**	68.05	66.47	50.69

In addition, we analyze two variants of DCP that we use in all our experiments: Combination and Alternate. The DCP Combination unifies both patch and channel masks, while the DCP Alternate switches between the two for each training iteration. Both variants outperform traditional patch- and channel-based masking, highlighting the effectiveness of joint channel-patch masking. Refer to Appendix F.5 for more analysis and hyperparameter settings for the DCP Masking.

4.2 Model ablations and sensitivity analysis

Impact of Memory tokens. Fig. 3(a) & (b) show the accuracy achieved by numbers of *memory tokens*. Using memory tokens enhances performance, but beyond a certain point yields diminishing returns. This suggests that while memory tokens can improve performance, excessive reliance on them may negatively impact the interaction of patch features. We found that a default of 4 memory tokens works well across the three datasets.

Attention patterns of Memory tokens. Fig. 4 shows the attention patterns between image patches and memory tokens of the encoder. Each group of channels displays distinct preferences for specific memory tokens. Fig. 4(a) shows that the *VH* channels primarily focus on memory token 8, while the *Lee-filtered* channels have stronger attention toward memory token 1. Similarly, for JUMP-CP in Fig. 4(b), the *Brightfield* channels allocate more attention to memory token 3, whereas *Fluorescence* channels show a slight preference for memory token 1. This suggests that each type of channel utilizes different memory tokens to store the global information necessary for feature extraction.

Token pooling strategies. Table 7 compares the performance of various token pooling strategies on CHAMMI and So2Sat. In general, combining the [CLS] token with the average of the patch tokens gets better performance than either alone. By utilizing both the global [CLS] token and the fine-grained patch tokens, Hybrid Token Fusion consistently outperforms others across all settings.

Reconstruction loss lambda. Fig. 3(c) & (d) analyze the effect of the reconstruction weight λ_{recon} (Eq. (3)) on model performance. Consistent with prior work [19, 22], a large value of $\mathcal{L}_{\text{recon}}$ gives the best results. However, performance drops significantly using just the MAE loss (*i.e.*, $\lambda_{\text{recon}} = 1$), noting the contributions of both losses. In all our experiments, we set a fixed $\mathcal{L}_{\text{recon}}$ to 0.99.

https://doi.org/10.52202/085713-0401

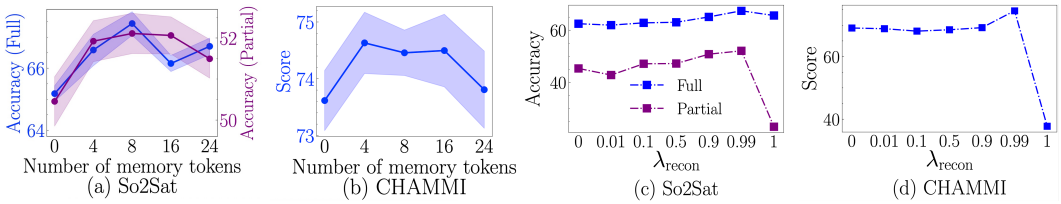

Figure 3: **Impact of the number memory tokens and reconstruction lambda λ_{recon} (Eq. (3)). (a) & (b)** Using $4-8$ tokens improves performance, however, using more memory tokens (*e.g.*, 24) may reduce the effectiveness. **(c) & (d)** $\lambda_{\text{recon}} = 0$ means without the reconstruction loss, while $\lambda_{\text{recon}} = 1$ indicates only using the reconstruction loss. For $\lambda_{\text{recon}} = 1$ on So2Sat, we run linear probing. $\lambda_{\text{recon}} = 0.99$ works best for ChA-MAEViT on both datasets.

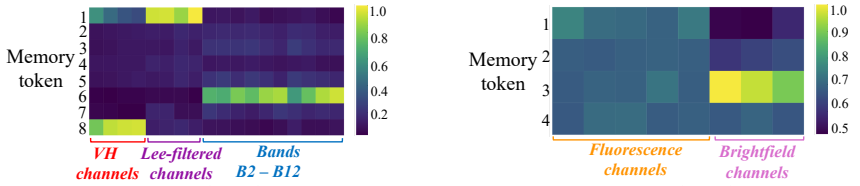

Figure 4: **Attention between image patches and memory tokens of the encoder**. Each channel group focuses on different memory tokens. **(a) So2Sat:** *VH* channels utilize memory token 8, whereas *Lee-filtered* channels attend more to memory token 1. **(b) JUMP-CP:** *Brightfield* channels focus on memory token 3, while Fluorescence channels favor memory token 1.

Table 7: **Comparison of token pooling methods.** Hybrid Token Fusion achieves the best performance, demonstrating the benefit of leveraging both global and fine-grained local tokens.

Token Pooling Method	CHAMMI	So2Sat	
		Full	Partial
Avg Patch tokens	71.41	64.86	51.87
[CLS] token	73.84	65.48	51.40
[CLS] + Avg Patch tokens	73.96	66.23	49.45
Hybrid Token Fusion (ours)	**74.63**	**67.44**	**52.11**

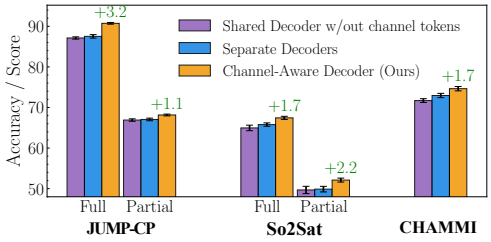

Figure 5: **Different Decoders when using with ChA-MAEViT**. Our Channel-Aware Decoder outperforms the best baseline by $1.1-3.2\%$ on all three datasets.

Channel-Aware Decoder analysis. Fig. 5 shows Channel-Aware Decoder in ChA-MAEViT outperforms both Separate Decoders [1] and Shared Decoder W/out Channel Tokens by $1.1-3.2\%$ across three datasets in full and partial settings. Channel-Aware Decoder is also more efficient than Separate Decoders thanks to its shared parameters, *e.g.*, $18X$ fewer parameters on So2Sat.

5 Conclusion

In this paper, we introduce ChA-MAEViT, a novel MAE-based method to enhance feature learning in Multi-Channel Imaging (MCI) ViTs. First, we introduce Dynamic Channel-Patch Masking, in which we adaptively mask both image patches and channels and train the model to reconstruct the masked patches. Additionally, we incorporate *memory tokens* to preserve global context across channels and Hybrid Token Fusion module to combine features from local patch tokens and global class tokens. Furthermore, we propose Channel-Aware Decoder to efficiently reconstruct channel-specific details. Experiments conducted on the CHAMMI, JUMP-CP, and So2Sat datasets demonstrate that ChA-MAEViT outperforms prior MCI-ViTs by $3.0-21.5\%$, highlighting the significance of enhanced cross-channel interactions. Future research could explore the adaptation of the channel-aware framework to additional complex modalities, such as volumetric medical imaging, which is composed of sequential two-dimensional slices.

https://doi.org/10.52202/085713-0401

Acknowledgments and Disclosure of Funding

This study was supported, in part, by the National Science Foundation under NSF-DBI awards 2134695 and 2134696. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the supporting agencies.

References

[1] Oren Kraus, Kian Kenyon-Dean, Saber Saberian, Maryam Fallah, Peter McLean, Jess Leung, Vasudev Sharma, Ayla Khan, Jia Balakrishnan, Safiye Celik, et al. Masked autoencoders for microscopy are scalable learners of cellular biology. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11757–11768, 2024.

[2] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, and Scott Reed. Dragomiranguelov, dumitru erhan, vincent vanhoucke, and andrew rabinovich. 2015. going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[3] Xuelei Li, Liangkui Ding, Li Wang, and Fang Cao. Fpga accelerates deep residual learning for image recognition. In *2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, pages 837–840. IEEE, 2017.

[4] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[5] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[6] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[7] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[8] Chaitanya Ryali, Yuan-Ting Hu, Daniel Bolya, Chen Wei, Haoqi Fan, Po-Yao Huang, Vaibhav Aggarwal, Arkabandhu Chowdhury, Omid Poursaeed, Judy Hoffman, et al. Hiera: A hierarchical vision transformer without the bells-and-whistles. *arXiv preprint arXiv:2306.00989*, 2023.

[9] Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. Vmamba: Visual state space model. *Advances in neural information processing systems*, 37:103031–103063, 2025.

[10] Xiaoxiang Zhu, Jingliang Hu, Chunping Qiu, Yilei Shi, Hossein Bagheri, Jian Kang, Hao Li, Lichao Mou, Guicheng Zhang, Matthias Häberle, Shiyao Han, Yuansheng Hua, Rong Huang, Lloyd Hughes, Yao Sun, Michael Schmitt, and Yuanyuan Wang. New: So2sat lcz42, 2019. URL `https://mediatum.ub.tum.de/1483140`.

[11] Jian Zou, Tianyu Huang, Guanglei Yang, Zhenhua Guo, Tao Luo, Chun-Mei Feng, and Wangmeng Zuo. Unim 2 ae: Multi-modal masked autoencoders with unified 3d representation for 3d perception in autonomous driving. In *European Conference on Computer Vision*, pages 296–313. Springer, 2024.

[12] Zitong Chen, Chau Pham, Siqi Wang, Michael Doron, Nikita Moshkov, Bryan A. Plummer, and Juan C Caicedo. CHAMMI: A benchmark for channel-adaptive models in microscopy imaging. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL `https://openreview.net/forum?id=Luc1bZLeMY`.

[13] Nicolas Bourriez, Ihab Bendidi, Cohen Ethan, Gabriel Watkinson, Maxime Sanchez, Guillaume Bollot, and Auguste Genovesio. Chada-vit : Channel adaptive attention for joint representation learning of heterogeneous microscopy images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[14] Min Chen, Aaron Carass, Amod Jog, Junghoon Lee, Snehashis Roy, and Jerry L Prince. Cross contrast multi-channel image registration using image synthesis for mr brain images. *Medical image analysis*, 36: 2–14, 2017.

 https://doi.org/10.52202/085713-0401

[15] Kian Kenyon-Dean, Zitong Jerry Wang, John Urbanik, Konstantin Donhauser, Jason Hartford, Saber Saberian, Nil Sahin, Ihab Bendidi, Safiye Celik, Juan Sebastián Rodríguez Vera, Marta Fay, Imran S Haque, and Oren Kraus. Vitally consistent: Scaling biological representation learning for cell microscopy, 2025. URL https://openreview.net/forum?id=niywLsa54R.

[16] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=2dnO3LLiJ1.

[17] Roman Bachmann, David Mizrahi, Andrei Atanov, and Amir Zamir. Multimae: Multi-modal multi-task masked autoencoders. In *European Conference on Computer Vision*, pages 348–367. Springer, 2022.

[18] Xiang Zhang, Huiyuan Yang, Taoyue Wang, Xiaotian Li, and Lijun Yin. Multimodal channel-mixing: Channel and spatial masked autoencoder on facial action unit detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6077–6086, 2024.

[19] Feng Liang, Yangguang Li, and Diana Marculescu. Supmae: Supervised masked autoencoders are efficient vision learners. *arXiv preprint arXiv:2205.14540*, 2022.

[20] Yong Cheng, Wei Wang, Lu Jiang, and Wolfgang Macherey. Self-supervised and supervised joint training for resource-rich machine translation. In *International Conference on Machine Learning*, pages 1825–1835. PMLR, 2021.

[21] Junwen Bai, Bo Li, Yu Zhang, Ankur Bapna, Nikhil Siddhartha, Khe Chai Sim, and Tara N Sainath. Joint unsupervised and supervised training for multilingual asr. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6402–6406. IEEE, 2022.

[22] Yifei Xin, Xiulian Peng, and Yan Lu. Masked audio modeling with clap and multi-objective learning. In *INTERSPEECH*, pages 2763–2767, 2023. URL https://doi.org/10.21437/Interspeech.2023-2488.

[23] Yu Gao, Xintong Han, Xun Wang, Weilin Huang, and Matthew Scott. Channel interaction networks for fine-grained image categorization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 10818–10825, 2020.

[24] Daquan Zhou, Zhiding Yu, Enze Xie, Chaowei Xiao, Animashree Anandkumar, Jiashi Feng, and Jose M Alvarez. Understanding the robustness in vision transformers. In *International Conference on Machine Learning*, pages 27378–27394. PMLR, 2022.

[25] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[26] Jianwei Yang, Zhile Ren, Chuang Gan, Hongyuan Zhu, and Devi Parikh. Cross-channel communication networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[27] Srinivas Niranj Chandrasekaran, Beth A. Cimini, Amy Goodale, Lisa Miller, Maria Kost-Alimova, Nasim Jamali, John Doench, Briana Fritchman, Adam Skepner, Michelle Melanson, John Arevalo, Juan C. Caicedo, Daniel Kuhn, Desiree Hernandez, Jim Berstler, Hamdah Shafqat-Abbasi, David Root, Sussane Swalley, Shantanu Singh, and Anne E. Carpenter. Three million images and morphological profiles of cells treated with matched chemical and genetic perturbations. *bioRxiv*, 2022. doi: 10.1101/2022.01.05.475090. URL https://www.biorxiv.org/content/early/2022/01/05/2022.01.05.475090.

[28] Sorour Mohajerani and Parvaneh Saeedi. Cloud-net: An end-to-end cloud detection algorithm for landsat 8 imagery. In *IGARSS 2019-2019 IEEE international geoscience and remote sensing symposium*, pages 1029–1032. IEEE, 2019.

[29] Chau Pham and Bryan A. Plummer. Enhancing feature diversity boosts channel-adaptive vision transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[30] Yujia Bao, Srinivasan Sivanandan, and Theofanis Karaletsos. Channel vision transformers: An image is worth 1 x 16 x 16 words. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=CK5Hfb5hBG.

[31] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[32] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9653–9663, 2022.

[33] Zachary Sims, Gordon B Mills, and Young Hwan Chang. Mim-cycif: masked imaging modeling for enhancing cyclic immunofluorescence (cycif) with panel reduction and imputation. *bioRxiv*, 2023.

[34] Junyan Lin, Feng Gao, Xiaochen Shi, Junyu Dong, and Qian Du. Ss-mae: Spatial–spectral masked autoencoder for multisource remote sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 61:1–14, 2023.

[35] Yue Wang, Ming Wen, Hailiang Zhang, Jinyu Sun, Qiong Yang, Zhimin Zhang, and Hongmei Lu. Hsimae: A unified masked autoencoder with large-scale pre-training for hyperspectral image classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2024.

[36] Jonas Geiping, Quentin Garrido, Pierre Fernandez, Amir Bar, Hamed Pirsiavash, Yann LeCun, and Micah Goldblum. A cookbook of self-supervised learning. *arXiv preprint arXiv:2304.12210*, 2023.

[37] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[38] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

[39] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9588–9597, 2021.

[40] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15750–15758, 2021.

[41] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[42] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[43] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. *International Conference on Learning Representations (ICLR)*, 2022.

[44] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL `https://openreview.net/forum?id=a68SUt6zFt`. Featured Certification.

[45] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Florian Bordes, Pascal Vincent, Armand Joulin, Mike Rabbat, and Nicolas Ballas. Masked siamese networks for label-efficient learning. In *European Conference on Computer Vision*, pages 456–473. Springer, 2022.

[46] Alfie Roddan, Tobias Czempiel, Daniel S Elson, and Stamatia Giannarou. Calibration-jitter: Augmentation of hyperspectral data for improved surgical scene segmentation. *Healthcare Technology Letters*, 11(6): 345–354, 2024.

[47] Ankit Patnala, Scarlet Stadtler, Martin G Schultz, and Juergen Gall. Generating views using atmospheric correction for contrastive self-supervised learning of multispectral images. *IEEE Geoscience and Remote Sensing Letters*, 20:1–5, 2023.

[48] Vishal Nedungadi, Ankit Kariryaa, Stefan Oehmcke, Serge Belongie, Christian Igel, and Nico Lang. Mmearth: Exploring multi-modal pretext tasks for geospatial representation learning. In *European Conference on Computer Vision*, pages 164–182. Springer, 2024.

 https://doi.org/10.52202/085713-0401

[49] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

[50] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[51] David Ha, Andrew Dai, and Quoc Le. Hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2016.

[52] Bryan A. Plummer, Nikoli Dryden, Julius Frost, Torsten Hoefler, and Kate Saenko. Neural parameter allocation search. In *International Conference on Learning Representations (ICLR)*, 2022.

[53] Pedro Savarese and Michael Maire. Learning implicitly recurrent CNNs through parameter sharing. In *International Conference on Learning Representations (ICLR)*, 2019.

[54] Chau Pham, Piotr Teterwak, Soren Nelson, and Bryan A. Plummer. Mixturegrowth: Growing neural networks by recombining learned parameters. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2024.

[55] Kian Kenyon-Dean, Zitong Jerry Wang, John Urbanik, Konstantin Donhauser, Jason Hartford, Saber Saberian, Nil Sahin, Ihab Bendidi, Safiye Celik, Juan Sebastián Rodríguez Vera, Marta Fay, Imran S Haque, and Oren Kraus. Vitally consistent: Scaling biological representation learning for cell microscopy, 2025. URL `https://openreview.net/forum?id=niywLsa54R`.

[56] Alexandre Eymaël, Renaud Vandeghen, Anthony Cioppa, Silvio Giancola, Bernard Ghanem, and Marc Van Droogenbroeck. Efficient image pre-training with siamese cropped masked autoencoders. In *European Conference on Computer Vision*, pages 348–366. Springer, 2024.

[57] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEit: BERT pre-training of image transformers. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=p-BhZSz59o4`.

[58] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Hervé Jégou. Spreading vectors for similarity search. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=SkGuG2R5tm`.

[59] Gianluca Donato and Serge Belongie. Approximate thin plate spline mappings. In *Computer Vision—ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28–31, 2002 Proceedings, Part III 7*, pages 21–31. Springer, 2002.

[60] Eu Wern Teh, Terrance DeVries, and Graham W Taylor. Proxynca++: Revisiting and revitalizing proxy neighborhood component analysis. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*, pages 448–464. Springer, 2020.

[61] Mubashir Noman, Muzammal Naseer, Hisham Cholakkal, Rao Muhammad Anwer, Salman Khan, and Fahad Shahbaz Khan. Rethinking transformers pre-training for multi-spectral satellite imagery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27811–27819, 2024.

[62] C Reed, Ritwik Gupta, Shufan Li, Sarah Brockman, Christopher Funk, Brian Clipp, S Candido, M UyttenDAele, and T Darrell. Scale-mae: A scale-aware masked autoencoder for multiscale geospatial representation learning. 2023 ieee. In *CVF International Conference on Computer Vision (ICCV)*, pages 4065–4076, 2022.

[63] Aamir Mustafa, Aliaksei Mikhailiuk, Dan Andrei Iliescu, Varun Babbar, and Rafał K Mantiuk. Training a task-specific image reconstruction loss. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2319–2328, 2022.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, the abstract and introduction reflect our main contributions and scope of the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: While we demonstrate that ChA-MAEViT can handle novel channels never seen during training in a preliminary experiment, we leave a systematic investigation of this setting for future work, as mentioned in the *Broader Impacts and limitations*.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

https://doi.org/10.52202/085713-0401

Answer: [NA]

Justification: We do not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Our method is described clearly in Fig. 2 and Section 3. We also provide the implementation details in Appendix E, and release the code at https://github.com/chaudatascience/cha_mae_vit.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We release the code with instructions to download the datasets.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the implementation details in Appendix E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide error bars in our resulting tables, such as in Table 1 and Table 9.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

 https://doi.org/10.52202/085713-0401

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the implementation details with computer resources in Appendix E

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, the research conducted conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We include a Broader Impacts and limitations.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the original papers and provide a link to the codes. The license and terms are properly respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

 https://doi.org/10.52202/085713-0401

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release the code with a document for people to use.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: In this paper, we do not use crowdsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: We only use LLMs for editing the writing.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

 https://doi.org/10.52202/085713-0401

A Broader Impacts and limitations

The development of ChA-MAEViT represents a significant advancement in Multichannel Imaging, with benefits such as improved medical diagnostics and faster healthcare research. Its applications in satellite imaging also hold potential for environmental monitoring. However, there are risks, including the possibility of misuse for invasive surveillance systems, highlighting the need for ethical considerations and responsible deployment.

While our model is capable of handling unseen channels by leveraging relationships learned from known ones (Appendix F.10), we have not thoroughly evaluated its performance in such scenarios. Adapting to novel channels requires mapping them to existing ones, which becomes more challenging under domain shifts. We leave a systematic investigation of this setting to future work. Additionally, our approach requires some extra hyper-parameter tuning, which can increase computational resource demands.

B Image patching

Formally, let $I \in \mathbb{R}^{h \times w \times c}$ represent a multi-channel input image, where (h, w) denotes the dimensions of the image and c is the number of channels. We begin by patchifying each channel into n 2-D patches of size $p \times p$ in pixels. This process yields $n \times c$ patches, denoted as $\mathbf{x}_{i,j} \in \mathbb{R}^{p \times p}$, where $n = \frac{h \cdot w}{p^2}$ is the number of patches per channel, i indicates the spatial location of a given patch, and j denotes the j-th channel. Next, we flatten each patch $\mathbf{x}_{i,j} \in \mathbb{R}^{p^2}$, and then pass it into a shared linear projection $\mathbf{W} \in \mathbb{R}^{p^2 \times d}$. This results in patch tokens $\mathbf{t}_{i,j} = (\mathbf{x}_{i,j} \cdot \mathbf{W}) \in \mathbb{R}^d$, leading to a sequence of $n \times c$ patch tokens $[\mathbf{t}_{1,1}; \mathbf{t}_{2,1}; \ldots; \mathbf{t}_{n,c}]$. Note that we utilize a shared projection across all channels, thus the number of parameters remains constant regardless of the number of input channels.

C Combination of DiChaViT with SSL

We adopt the following SSL methods with DiChaViT [29] as baselines: SimCLR [37, 38], SimSiam [40], MAE [31], iBOT [43], and DINOv2 [44].

SimCLR [37, 38] utilizes augmentation strategies that include random cropping, color distortion, and Gaussian blur to generate the dual views of an input image. This approach requires twice the computational resources per sample compared to single-path methods. Since hue and saturation are not well defined in multi-channel images, we apply the augmentation by only adjusting brightness and contrast.

SimSiam [40] employs similar augmentations as SimCLR, but it eliminates the use of negative pairs by stop-gradient. This operation helps prevent representation collapse by decoupling the optimization of the twin network branches, which simulates an alternating optimization process similar to the Expectation-Maximization (EM) approach. Additionally, SimSiam does not require momentum encoders.

MAE [31] involves randomly masking a large portion of the input image and training the model to reconstruct the missing content. Unlike contrastive methods, MAE does not depend on augmentation pipelines, which can be ineffective for heterogeneous channels in MCI [46, 47]. Additionally, by processing only a small fraction of the image, this method significantly lowers computational demands, thereby enhancing efficiency. Note that combining MAE with DiChaViT [29] results in a model very similar to CA-MAE [1], an extension of MAE for multi-channel imaging. However, key components from DiChaViT$-$ such as diverse regularizers, channel tokens, and the diverse channel sampling strategy $-$ are not present in CA-MAE, making the combined model more expressive for handling multi-channel imaging.

iBOT [43] combines masked image modeling with self-distillation by utilizing dual augmentation streams that include both masked and full views. This encourages the model to align representations across different views while reconstructing missing content. Particularly, iBOT uses clockwise masking introduced in BEiT [57], where a block of image patches is masked each time. To adapt this clockwise masking for multi-channel images, we generate a mask for each channel, ensuring that the masked blocks only cover the areas in the channel. Compared to MAE, iBOT introduces

additional computational overhead due to its use of teacher-student distillation. Furthermore, the dual-stream processing increases the per-sample training cost, making iBOT more computationally demanding than single-view methods like MAEs.

DINOv2 [44] employs multi-cropping alongside various augmentations, such as color jittering, Gaussian blur, and random solarization, as well as self-distillation techniques. Notably, DINOv2 also incorporates additional loss functions, such as iBOT loss [43] and KoLeo loss [58]. During training, DINOv2 generates multiple crops from each image (*e.g.*, 2 global views and 8 local views) which leads to significantly higher computational demands compared to other SSL methods. For example, training DINOv2 on JUMP-CP dataset using two GPUs requires approximately 6 days, whereas SimCLR takes around 2 days and MAE only requires 1 day. Refer to Fig. 6 for FLOPs counts.

To integrate these SSL methods with DiChaViT [29], we add the task loss and optimize it along with the SSL loss. Additionally, we explored another variant that included an additional branch utilizing standard augmentations recommended by the authors of the dataset to optimize the task loss and return the predictions at test time (*e.g.*, only using random cropping, horizontal flipping, and thin-plate-spline transformation [59] for CHAMMI). We observed that relying exclusively on complex augmentation pipelines, like those used in SimCLR, negatively impacts the performance of the combined model, while incorporating at least one view with regular augmentation enhances the model's learning capability. For self-distillation methods such as DINOv2, we found that the performance of the teacher network is slightly better than that of the student network, thus we report the performance on the teacher network.

D Dynamic Channel-Patch (DCP) Masking Algorithm

Algorithm 1 outlines the procedure of Dynamic Channel-Patch (DCP) Masking in Section 3.1 of the main paper, where we combine both *patch* and *channel* masking strategies. In practice, setting $\mathrm{p_{patch}} = \mathrm{p_{channel}} = 0$ merges both patch and channel masks into a unified mask, and setting $\mathrm{p_{patch}} = \mathrm{p_{channel}} = 0.5$ allows the model to switch between patch and channel masks. We adopt these two straightforward configurations for all our experiments. For $\mathrm{p_{patch}} = \mathrm{p_{channel}} = 0$ (*i.e.*, *DCP Combination*), we found that it works well with a small value of patch mask ratio r_p, and simply set $r_p = 0.25$ for all the experiments. For $\mathrm{p_{patch}} = \mathrm{p_{channel}} = 0.5$ (*i.e.*, *DCP Alternate*), we used a larger value of $r_p = 0.75$. Refer to Appendix F.5 for more analysis on these hyperparameters.

E Implementation details

For most of the baseline models, such as HyperNetViT, ChannelViT, ChAda-ViT, and DiChaViT, we utilize the implementation from [29] [1]. For iBOT and DINOv2, we adapt the implementation from [44] [2]. We employ a ViT small architecture with 21M parameters as the backbone for all methods. To ensure that we maintain the same number of parameters as the baselines, we removed one layer from the encoder to incorporate it into the decoder for our method. For both CHAMMI and JUMP-CP, we use a patch size of 16, while for the So2Sat dataset, we opt for a patch size of 8. We trained the models using the AdamW optimizer, aiming to minimize the cross-entropy loss for the JUMP-CP and So2Sat datasets, while employing proxy loss for CHAMMI.

CHAMMI dataset [12]. For the baselines, we utilize the [CLS] token from the final layer as the feature representation and train the model to minimize the proxy loss [60]. We then evaluate the model on various tasks using the evaluation code provided by [12], incorporating a 1-Nearest Neighbour classifier to calculate the macro-average F1-score for each task individually [3]. The channel-adaptive interfaces are adapted from the authors' implementation code [4]. In addition to the model, we apply the same data augmentation techniques recommended by the authors, including thin-plate-spline (TPS) transformations [59]. Each model is trained for 100 epochs with a learning rate of 0.0004 and a batch size of 64.

[1] https://github.com/chaudatascience/diverse_channel_vit

[2] https://github.com/facebookresearch/dinov2

[3] https://github.com/broadinstitute/MorphEm

[4] https://github.com/chaudatascience/channel_adaptive_models

https://doi.org/10.52202/085713-0401

Algorithm 1: Dynamic Channel-Patch Masking

Input : Number of patches per channel n;
Number of channels c;
Patch mask ratio r_p;
Probability of using patch masking $\mathrm{p}_{\mathrm{patch}}$;
Probability of using channel masking $\mathrm{p}_{\mathrm{channel}}$;
$(0 \leq \mathrm{p}_{\mathrm{patch}} + \mathrm{p}_{\mathrm{channel}} \leq 1)$

// Random Patch Mask
$\mathbf{p\_mask} = \mathbf{0}^{n \times c}$ *// initialize patch mask*
For $j = 1, 2, \ldots, c$:
  $\mathbf{p\_mask}[:, j] = \mathrm{generate\_random\_patch\_mask}(n, r_p)$
EndFor
// Dynamic Channel Mask
Uniformly sample a number of channels to mask $k \sim \mathcal{U}\{0, 1, \ldots, c-1\}$
Uniformly sample k masked channels $\mathcal{C}' = \mathrm{Sample}(\{1, \ldots, c\}, k)$
$\mathbf{c\_mask} = \mathbf{0}^{n \times c}$ *// initialize channel mask*
For $j \in \mathcal{C}'$:
  $\mathbf{c\_mask}[:, j] = \mathbf{1}^n$ *// mask whole channel j*
EndFor
// Masks For Current Iteration
Sample $s \sim \mathcal{U}(0, 1) \in \mathbb{R}$
If $s < \mathrm{p}_{\mathrm{patch}}$:
  $\mathbf{mask} = \mathbf{p\_mask}$ // Using patch mask
Else If $s < \mathrm{p}_{\mathrm{patch}} + \mathrm{p}_{\mathrm{channel}}$:
  $\mathbf{mask} = \mathbf{c\_mask}$ // Using channel mask
Else: // Combining both masks
  $\mathbf{mask} = \mathbf{p\_mask} \vee \mathbf{c\_mask}$

Output : $\mathbf{mask} \in \{0, 1\}^{n \times c}$, where 1 denotes mask

JUMP-CP [27] and So2Sat [10] datasets. Following [29, 30], we warm up the learning rate for the first 10 epochs, reaching a peak of 0.0004. After this initial period, the learning rate gradually decays to 10^{-6} according to a cosine scheduler. To mitigate overfitting, we apply a weight decay of 0.04 to the weight parameters while excluding the bias and normalization terms. We use the same data augmentation techniques as outlined in [29]. For the final prediction, we utilize $[\mathrm{CLS}]$ token from the Transformer encoder into a classifier head to predict the probability of each class. The final model checkpoint is selected based on validation performance. Each model is trained for 100 epochs, with a batch size of 64 on JUMP-CP and 128 on So2Sat.

Compute resources. Experiments conducted on So2Sat and CHAMMI utilize one NVIDIA RTX GPU with 48GB RAM, alongside three Intel(R) Xeon(R) Gold 6226R CPUs. For JUMP-CP experiments, two NVIDIA RTX GPUs and six Intel(R) Xeon(R) Gold 6226R CPUs were used.

F Additional Results and Analyses

F.1 FLOPs Count for Combination of DiChaViT with SSL

In Fig. 6, we present the FLOPs for each of the SSL methods when combined with DiChaViT, shown in Table 2 of the main paper. ChA-MAEViT achieves the lowest FLOPs, approximately one-sixth that of DINOv2.

F.2 CHAMMI Benchmark Results

Table 8 presents the F1 scores of various MCI-ViT models (ViT-S backbone) evaluated on the CHAMMI benchmark [12]. The evaluation consists of nine tasks, including six out-of-distribution (OOD) tasks. ChA-MAEViT demonstrates superior overall performance, achieving the highest scores in six out of the nine tasks across different settings.

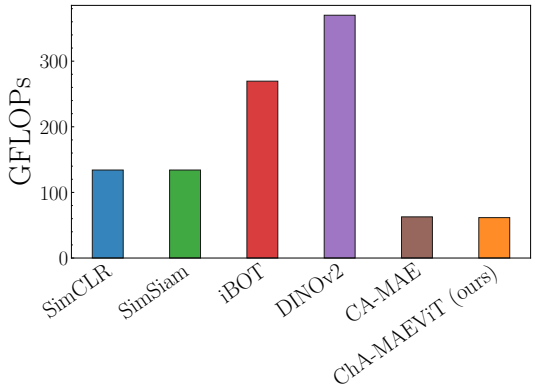

Figure 6: **FLOPs Count for Combination of DiChaViT with SSL**. We show FLOPs for each method when trained on JUMP-CP. ChA-MAEViT achieves the lowest FLOPs, $\approx 1/6$ that of DINOv2.

Table 8: **F1 Scores of MCI-ViT models on CHAMMI benchmark** [12]. ChA-MAEViT demonstrates better overall performance on CHAMMI, achieving the highest scores in 6 out of 9 tasks across various settings. "OOD" refers to out-of-distribution tasks. All models have the same number of parameters, except CA-MAE with $3X$ more parameters due to its use of separate decoders.

	Average OOD				WTC		HPA			CP			
Model	Mean	WTC	HPA	CP	Task1	Task2	Task1	Task2	Task3	Task1	Task2	Task3	Task4
HyperNetViT [51]	47.17	45.78	67.61	28.11	58.83	45.78	88.78	82.70	52.52	82.13	53.74	23.16	7.42
DepthwiseViT [12]	50.44	52.19	71.41	27.72	69.81	52.19	91.65	**88.04**	54.78	81.24	54.08	**23.21**	5.87
TempMixingViT [52–54]	47.33	51.52	64.80	25.66	61.66	51.52	85.01	79.91	49.69	77.45	48.83	22.56	5.60
CA-MAE [1] + Sup. Loss	48.82	61.85	56.45	**28.15**	77.57	61.85	84.45	71.89	41.01	76.49	56.52	19.43	**8.49**
ChAda-ViT [13]	50.82	67.18	60.67	24.60	77.58	67.18	87.49	75.94	45.41	83.92	45.58	21.94	6.28
ChannelViT [30]	52.54	67.58	62.35	27.81	78.36	67.58	83.93	76.73	47.97	77.70	55.16	21.89	6.38
DiChaViT [29]	55.36	75.18	64.36	26.53	80.87	75.18	88.08	79.26	49.45	84.08	53.03	20.95	5.60
ChA-MAEViT (ours)	**58.02**	**77.15**	**72.11**	24.81	**84.52**	**77.15**	**94.14**	87.47	**56.75**	**90.89**	**56.68**	10.25	7.50

F.3 Performance when only using SSL Objectives

We evaluate the effectiveness of ChA-MAEViT in the SSL setting by comparing the performance of different SSL methods. Specifically, all models were trained solely using SSL objectives, *i.e.*, no task-specific loss was incorporated. For the So2Sat [10] and JUMP-CP [27] datasets, we reported the accuracy from linear probing over 20 epochs. As shown in Fig. 7, ChA-MAEViT achieves the highest score of 37.7%, surpassing other approaches by $1.0 - 8.5\%$ on CHAMMI. In Fig. 8, our method consistently outperforms baseline models in both Full and Partial settings on So2Sat. A similar trend is observed for JUMP-CP in Fig. 9. This demonstrates the effectiveness of our method in SSL settings for MCI, highlighting its ability to extract meaningful representations in scenarios where supervised labels are unavailable.

F.4 Leave-One-Channel-Out at Test Time

In Table 9, we present the results of training the model using all eight channels of JUMP-CP and then testing it with various combinations of seven channels. This provides a detailed result for column "7" of Table 5 in the main paper, which represents $C_8^7 = 8$ different channel combinations. For each combination, we report the mean and standard deviation of the model's performance based on three runs. Our results demonstrate that ChA-MAEViT achieves an improvement of $17 - 23\%$ for each combination compared to the baseline models DiChaViT [29] and ChannelViT [30].

https://doi.org/10.52202/085713-0401

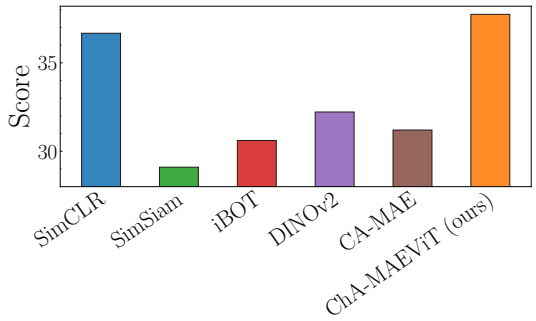

Figure 7: **Comparison of SSL methods on CHAMMI [12]**. All models were trained solely using SSL objectives, *i.e.*, without any task loss. ChA-MAEViT achieves the highest score, outperforming other baselines by $1.0 - 8.5\%$, demonstrating the effectiveness of our approach over other methods in SSL settings for MCI.

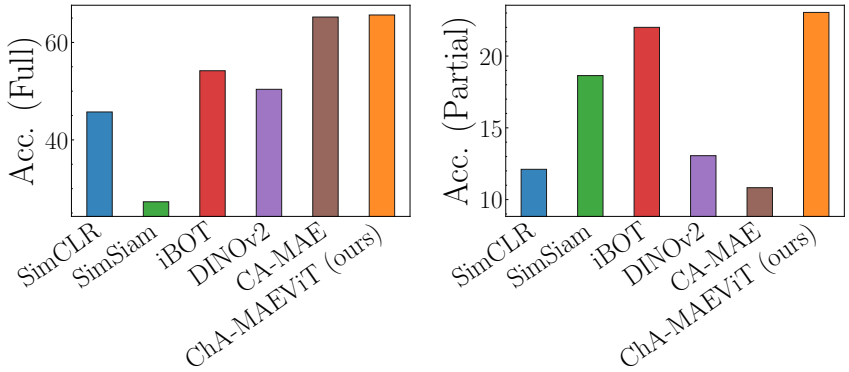

Figure 8: **Comparison of SSL methods on So2Sat [10]**. All models were trained solely using SSL objectives, *i.e.*, without any task loss, then trained with linear probing for another 20 epochs. ChA-MAEViT outperforms other baselines in both Full (left) and Partial channel settings (right), demonstrating its effectiveness for SSL in both scenarios.

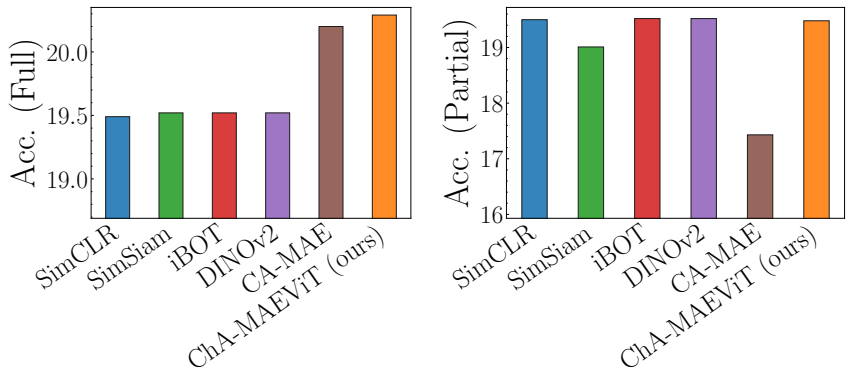

Figure 9: **Comparison of SSL methods on JUMP-CP [27]**. All models were trained solely using SSL objectives, *i.e.*, without any task loss, then trained with linear probing for another 20 epochs. ChA-MAEViT achieves the highest score when tested on Full channels (left) and performs comparably to the top-performing baselines on Partial channel settings (right).

https://doi.org/10.52202/085713-0401

Table 9: **Analysis of Leave-One-Channel-Out Testing.** We present the detailed results for column "7" in Table 5 of the main paper. Each row shows the results obtained by leaving out one channel and testing the model on the remaining seven channels. For all experiments conducted, we provide the *mean* $\pm$ *standard deviation* based on three runs. Our method, ChA-MAEViT, consistently outperforms the best two baselines ChannelViT [30] and DiChaViT [29] by $17 - 23\%$ across all tested combinations.

Missing channel at inference	ChannelViT [30]	DiChaViT [29]	**ChA-MAEViT (ours)**
0	61.72 ± 0.48	63.48 ± 0.20	**86.29** ± 0.37
1	61.21 ± 0.41	62.72 ± 0.28	**86.16** ± 0.32
2	61.90 ± 0.48	63.28 ± 0.31	**87.13** ± 0.24
3	37.70 ± 0.60	38.83 ± 0.46	**61.79** ± 0.58
4	58.52 ± 0.63	59.61 ± 0.17	**82.67** ± 0.53
5	67.28 ± 0.53	69.12 ± 0.16	**89.24** ± 0.35
6	67.20 ± 0.59	69.06 ± 0.20	**87.54** ± 0.20
7	67.37 ± 0.60	69.21 ± 0.19	**86.03** ± 0.32

F.5 Dynamic Channel-Patch (DCP) Masking Analysis

***DCP Combination* with varying masking ratios r_p.** Fig. 10 shows the impact of varying *random patch masking ratios* (r_p) on the performance of ChA-MAEViT when trained with DCP Combination ($\mathrm{p_{patch}} = \mathrm{p_{channel}} = 0$). The blue and purple lines indicate the performance for full and partial channels on JUMP-CP, respectively. As the masking ratio r_p increases, accuracy for both full and partial channels decreases due to excessive information loss, making reconstruction more difficult. Since DCP Combination masks both patches and channels, it is reasonable to use a small r_p, in contrast to previous studies that often utilize a high ratio (*e.g.*, 0.75). For example, a r_p of 0.25, when combined with dynamic channel masking, together mask ≈ 0.65 patches in the original images. We found *DCP Combination* works well with a small value of r_p, thus we use r_p of 0.25 as default value for all the datasets for *DCP Combination*.

***DCP Alternate* with varying proportions of $\mathbf{p_{channel}}$ and $\mathbf{p_{patch}}$.** Fig. 11 evaluates the performance of *DCP Alternate* in ChA-MAEViT by varying the proportions of *channel*- and *patch*-level masking. We adjust $\mathrm{p_{channel}}$, while setting $\mathrm{p_{patch}} = 1 - \mathrm{p_{channel}}$ to control the contribution of each masking strategy. For example, $\mathrm{p_{channel}} = 1$ indicates exclusive channel-level masking, and $\mathrm{p_{channel}} = 0$ denotes only using patch-level masking. The blue and purple lines represent accuracy on JUMP-CP for Full and Partial channel settings, respectively, across different $\mathrm{p_{channel}}$ values. We observed that using both channel and patch masks together improves performance. Additionally, a higher channel-level masking proportion (*i.e.*, larger $\mathrm{p_{channel}}$) improves accuracy in the Partial setting (purple) but leads to a decline in the Full setting (blue), highlighting the trade-off between these masking strategies. For *DCP Alternate*, we found that $r_p = 0.75$ works well across experiments.

Patch Masking Strategies in DCP. Fig. 12 compares two patch masking strategies with DCP. The first strategy, *Duplicate-Spatial Mask*, creates a patch mask for one channel and then duplicates the mask across all other channels (*e.g.*, [35, 48]). The second approach, *Independent-Spatial Mask*, generates a patch mask for each channel independently, resulting in varied patch positions. We set the same masking ratio for both strategies. Our findings indicate that using a different mask for each channel performs better than the duplicating one, yielding an improvement of $1.2 - 5.5\%$ across three datasets.

F.6 Impact of memory tokens on reconstruction loss

Fig. 13 shows the effect of memory tokens on reconstruction loss during training. Evaluated on the So2Sat dataset, the model using memory tokens (blue line) achieves lower and smoother loss compared to those without them (red). This indicates that memory tokens may improve the model's ability to capture global information, helping to ease the learning process.

https://doi.org/10.52202/085713-0401

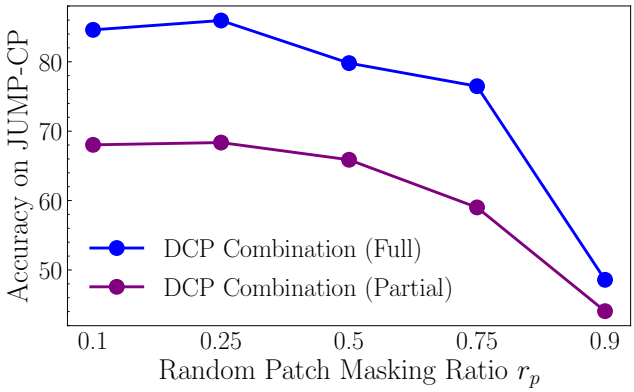

Figure 10: ***DCP Combination* with varying masking ratios** r_p. We train ChA-MAEViT using a combined mask integrating both channel- and patch-level masking, *i.e.*, *DCP Combination* ($\mathrm{p_{channel}} = \mathrm{p_{patch}} = 0$). The blue and purple lines depict accuracy on JUMP-CP for Full and Partial channel settings, respectively, across different patch masking ratios r_p.

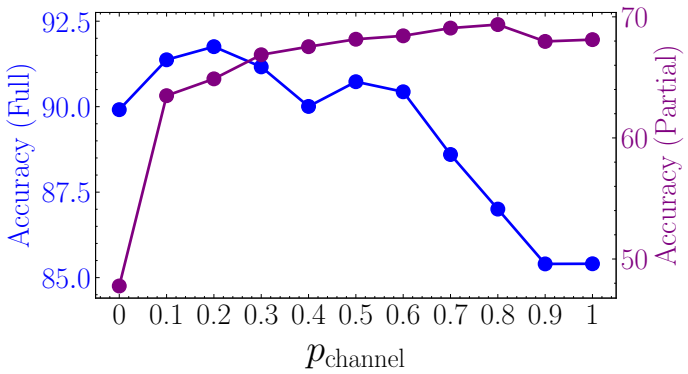

Figure 11: ***DCP Alternate* with different proportions of $\mathbf{p_{channel}}$ and $\mathbf{p_{patch}}$.** To evaluate the effects of *channel*- and *patch*- level masks in ChA-MAEViT during alternating masking, we vary $\mathrm{p_{channel}}$ while setting $\mathrm{p_{patch}} = 1 - \mathrm{p_{channel}}$. $\mathrm{p_{channel}} = 1$ indicates only using channel-level masking, whereas $\mathrm{p_{channel}} = 0$ indicates only using patch-level masking. The blue and purple lines represent accuracy on JUMP-CP for Full and Partial channel settings respectively, across different values of $\mathrm{p_{channel}}$.

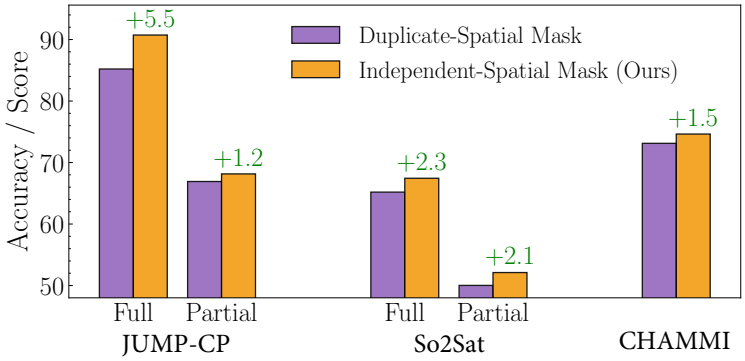

Figure 12: **Patch Masking Strategies in DCP**. We observed that using a different mask for each channel outperforms duplicating the same mask across all channels (purple), resulting in improvements of $1.2 - 5.5\%$ across three datasets.

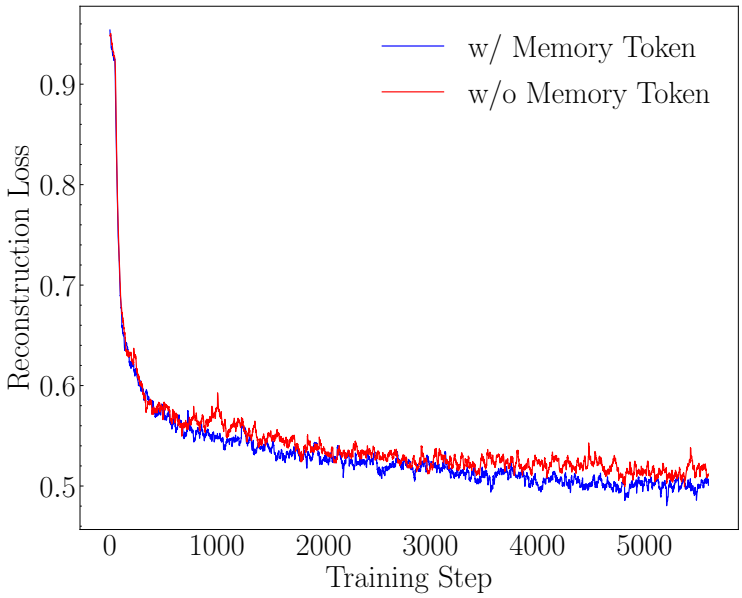

Figure 13: **Impact of memory tokens on the reconstruction loss**. Using memory tokens helps the reconstruction task, as demonstrated by the lower reconstruction loss observed on So2Sat.

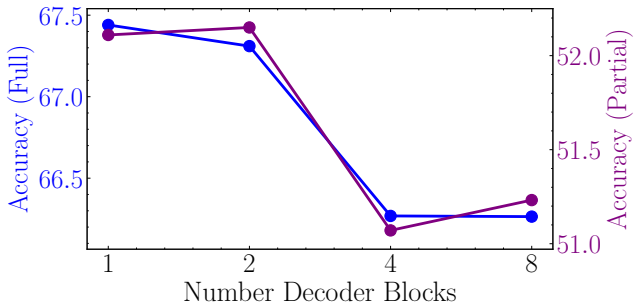

Figure 14: **Number of Decoder Blocks of ChA-MAEViT**. We show the performance of ChA-MAEViT when trained on So2Sat with different numbers of blocks in the decoder. We observed that using $1-2$ blocks gives the best performance.

F.7 Number of Decoder Blocks of ChA-MAEViT

Fig. 14 illustrates the performance of ChA-MAEViT when trained on the So2Sat dataset with varying numbers of decoder blocks. We found that a configuration of 1 to 2 blocks yields the best performance.

F.8 Scaling Model Size

Table 10 compares the performance of ChA-MAEViT with the two strongest supervised baselines: ChannelViT [30] and DiChaViT [29]. To evaluate the impact of model scaling, each model is evaluated using two backbone architectures: ViT-S (21M parameters) and ViT-B (85M parameters). Larger models exhibit improved performance, particularly on CHAMMI and JUMP-CP. Our approach consistently outperforms the baselines across all three datasets, regardless of the backbone used. In ChA-MAEViT, we reallocate one Transformer block from the encoder to the decoder, ensuring that all models maintain the same parameter count.

https://doi.org/10.52202/085713-0401

Table 10: **Scaling Model Size.** We compare ChA-MAEViT with the two best baseline models, ChannelViT [30] and DiChaViT [29]. Each model is evaluated using two backbones: ViT-S (21M parameters) and ViT-B (85M parameters). Increasing the model size boosts performance, particularly on CHAMMI and JUMP-CP. Our method outperforms others across three datasets with both backbones.

		CHAMMI [12]	JUMP-CP [27]		So2Sat [10]	
Model	Backbone	Avg score	Full	Partial	Full	Partial
ChannelViT [30]	ViT-S	64.97	67.51	56.49	61.03	46.16
DiChaViT [29]	ViT-S	69.77	69.19	57.98	63.36	47.76
ChA-MAEViT (ours)	ViT-S	74.63	90.73	68.05	67.44	52.11
ChannelViT [30]	ViT-B	67.72	67.92	56.97	62.19	46.20
DiChaViT [29]	ViT-B	70.56	69.47	58.33	63.93	47.92
ChA-MAEViT (ours)	ViT-B	**77.09**	**92.22**	**70.24**	**67.67**	**52.27**

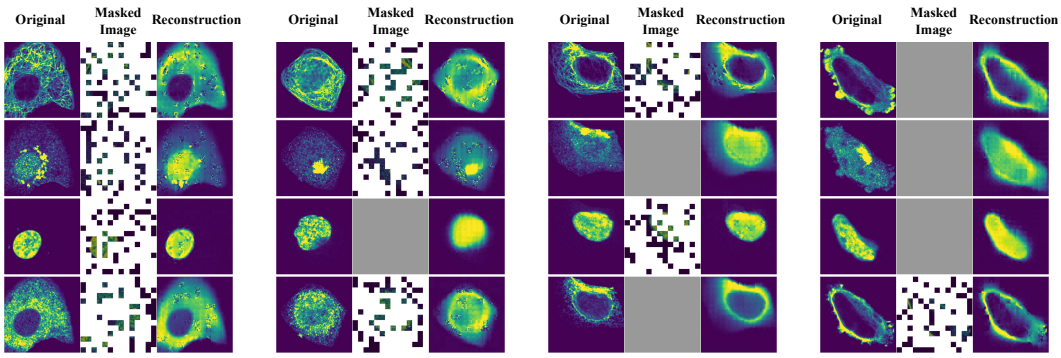

Figure 15: **Visualization of original, masked, and reconstructed images on CHAMMI-HPA [12], which contains 4 channels**. We show the reconstruction of ChA-MAEViT, utilizing a ViT-S/16 backbone. ChA-MAEViT demonstrates its capability to reconstruct masked channels and patches under different channel masking settings ($0-3$ masked channels with a fixed *patch masking ratio* $r_p = 75\%$). Gray blocks indicate fully masked channels.

F.9 Reconstruction Images

Fig. 15 shows the *original*, *masked*, and *reconstructed* images from the CHAMMI-HPA dataset [12], which contains four channels. Each set of images shows the original image on the left, followed by the masked image created using Dynamic Channel-Patch Masking (with patch masking ratio $r_p = 75\%$), and finally, the reconstructed image produced by ChA-MAEViT (utilizing a ViT-S/16 backbone) on the right. The gray blocks indicate the entire channel has been masked. ChA-MAEViT demonstrates its ability to leverage cross-channel dependencies as it can reconstruct the entire channels using available patches from the other channels.

Note that the reconstructions lack some high-frequency details, as the primary goal of ChA-MAEViT is to learn robust, high-quality representations rather than producing detailed reconstructions. Our approach prioritizes the encoder's ability to capture high-level semantic and cross-modal relationships, with a deliberately lightweight decoder designed to validate the encoding of essential latent information. Additionally, following prior MAEs (*e.g.*, [1, 31, 61, 62]), we employ the mean squared error (MSE) loss for reconstruction, which often results in blurry outputs by averaging plausible solutions [62, 63]. This leads to reconstructions dominated by low-frequency components, an expected outcome of the design choice.

F.10 Novel Channels at Inference

For the novel channels setting, we trained ChA-MAEViT using the first three channels (*i.e.*, channels $0, 1, 2$) and then tested it on two other channels (*i.e.*, channels $3, 4$) of JUMP-CP. One of the main challenges is that new channels do not have learned *channel tokens*. To address this, we assume that

https://doi.org/10.52202/085713-0401

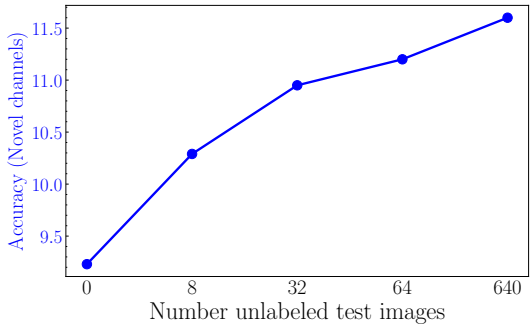

Figure 16: **Novel Channels at Inference**. Fine-tuning the *channel tokens* of novel channels with unlabeled test images boosted performance from 9.2% (without fine-tuning, denoted as "0") to 11.6% (with fine-tuning on 640 unlabeled test images).

some unlabeled test images are available at test time. We initialize the channel tokens for the new channels randomly and then fine-tuned these channel tokens on the unlabeled test images using the reconstruction loss in Eq. (2). Throughout this process, we kept the entire model frozen and only fine-tuned the newly initialized channel tokens.

In Fig. 16, we illustrate the accuracy on novel channels with varying numbers of unlabeled test images. For example, "0" indicates that no fine-tuning was performed, while "8" represents that eight test images (each containing only the two novel channels) were used to fine-tune the channel tokens before testing the model on the remaining test set. We observed that fine-tuning the channel tokens with some unlabeled test images improved performance from 9.2% (without fine-tuning) to 11.6% (with fine-tuning on 640 unlabeled test images). Additionally, to make an upper performance bound, we trained ChA-MAEViT on the training set of the novel channels (*i.e.*, channels 3 and 4). This model achieved an accuracy of 46.1%, indicating there is still a significant performance gap, which we leave for future work.

https://doi.org/10.52202/085713-0401