Vision-TTT: Efficient and Expressive Visual Representation Learning with Test-Time Training

Quan Kong[§], Yanru Xiao[◇], Yuhao Shen[§], and Cong Wang*[§]

[§] Zhejiang University, Hangzhou, China
[◇] Amazon Web Services, USA
{qkv, riven, cwang85}@zju.edu.cn
yrxiao@amazon.com

Abstract. Learning efficient and expressive visual representation has long been the pursuit of computer vision research. While Vision Transformers (ViTs) gradually replace traditional Convolutional Neural Networks (CNNs) as more scalable vision learners, their applications are plagued by the quadratic complexity of the self-attention mechanism. To address the challenge, we introduce a new linear-time sequence modeling method Test-Time Training (TTT) into vision and propose Vision-TTT, which treats visual sequences as datasets and compresses the visual token sequences in a novel self-supervised learning manner. By incorporating the dual-dataset strategy and Conv2d-based dataset preprocessing, Vision-TTT effectively extends vanilla TTT to model 2D visual correlations with global receptive fields. Extensive experiments show that `Vittt-T/S/B` achieve $77.7\%, 81.8\%, 82.7\%$ Top-1 accuracy on ImageNet classification and also greatly outperform their counterparts on downstream tasks. At 1280×1280 resolution, `Vittt-T` reduces FLOPs by 79.4% and runs $4.72\times$ faster with 88.9% less memory than DeiT-T. These results demonstrate the expressiveness and efficiency of Vision-TTT as an alternative for the next-generation visual backbone. Codes are available at https://github.com/imKQv/Vision-TTT.

Keywords: Visual representation learning · Linear Complexity Visual Sequence Modeling · Test-Time Training

1 Introduction

Learning efficient and expressive visual representation learning has long been a fundamental task in computer vision research. Convolutional Neural Networks (CNNs) [26, 28, 45, 49, 50, 55, 61] represent classic architectures in this domain. They can efficiently capture spatial hierarchies, yet are limited by the static nature of convolutional kernels, which restricts their performance scalability. In recent years, Vision Transformers (ViTs) [3, 16, 20, 36, 53, 65] have explored the self-attention mechanism for visual sequence modeling. With superior scalability demonstrated on large-scale experiments [41, 44], ViTs have gradually become a standard architectural paradigm in both academia and industry.

*Corresponding Author.

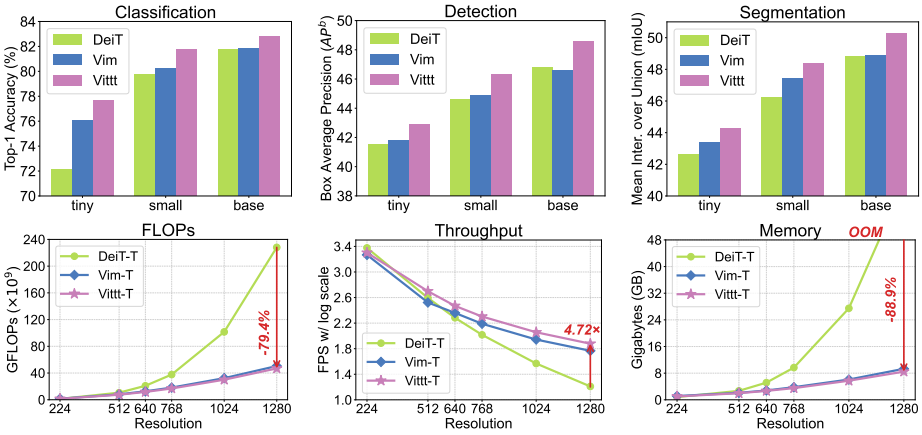

Fig. 1: Performance and efficiency comparison between DeiT [53], Vim [69] and our `Vittt` model. Results show that `Vittt` not only achieves superior performance on ImageNet classification and downstream detection and segmentation tasks, but also is more computation and memory efficient in dealing with high-resolution images.

However, as the community's demand for high-resolution image understanding continues to grow, ViTs are severely plagued by their quadratic computational complexity [54] when processing long-sequence tasks. To tackle this bottleneck, researchers have been striving to pursue a Pareto-optimal architecture that balances expressiveness and efficiency. Among the emerging visual sequence models [17,24,32,35,69], Vision-Mambas [35,69] represent the most popular and representative approaches that build upon the State Space Models (SSMs) with selective scan design and hardware-aware parallelism [10,21].

In this paper, we build on a new linear-time sequence modeling paradigm Test-Time Training (TTT) [47] and propose the Vision-TTT architecture, which establishes a new Pareto front between the expressiveness and efficiency in visual representation. In the context of visual sequence modeling [16, 17, 32, 69], the core idea of TTT is to treat the visual token sequence of an image as a dataset, along which self-supervised learning is performed to compress visual semantics into a hidden state via gradient updates. As a result, the token semantics are explicitly guided by the gradients to form an expressive and explainable visual representation. Furthermore, linear-complexity efficiency can also be achieved through hardware-aware matrix multiplication parallelism [47].

Despite this, vanilla TTT is originally designed for unidirectional modeling with inherent temporal dependence. This renders its global perception advantages in NLP inadequate for fully global modeling in visual tasks. To introduce spatial locality, we construct dual datasets [42] for the self-supervised learning and incorporate Conv2d-based [7] dataset preprocessing mechanism. Benefiting from these architectural designs, Vision-TTT exhibits a globally radial Effective Receptive Field (ERF) [38] to construct representations with explicit 2D correlations. As shown in Fig. 1, `Vittt` greatly outperforms the strong baselines of DeiT [53] and Vim [69] in both classification and downstream tasks, while keeping linear computational and memory complexity for high-resolution images.

The main contributions of this paper are summarized as follows:

- ✧ We propose Vision-TTT, the first generic visual backbone to leverage Test-Time Training RNNs with gradient-driven state adaptation to capture visual semantics and build expressive visual representations.
- ✧ By integrating hardware-aware kernel implementation, Vision-TTT alleviates the quadratic complexity bottleneck of Vision Transformers and achieves linear complexity modeling. At 1280×1280 resolution, `Vittt-T` saves 79.4% FLOPs and runs $4.72\times$ faster with 88.9% less memory than DeiT-T.
- ✧ With a dedicated 2D architectural design, Vision-TTT expands vanilla TTT to visual representation tasks with spatial locality. Extensive experiments show that `Vittt-T/S/B` achieve $77.7\%, 81.8\%, 82.7\%$ Top-1 accuracy on ImageNet classification, and significantly outperform their counterparts on downstream COCO detection and ADE20K segmentation tasks.

2 Related Work

Visual Representation Learning. Visual representation Learning serves as the foundation for a wide range of computer vision tasks [12, 33, 68]. It has evolved into two mainstream paradigms, convolution-based method and visual sequence modeling. Convolutional Neural Networks (CNNs) [14,26,28,31,34,37, 45,49,50,55,57,62] have long been the dominant architecture in this field due to their efficiency in processing spatial vision data. However, CNNs are limited by the static nature of convolution kernels [18], which restricts their scalability in performance. In recent years, Vision Transformer (ViT) [16] has pioneered the paradigm of visual sequence modeling, and DeiT [53] further refines ViT with advanced training techniques and distillation. They ignite a series of follow-up works that incorporate hierarchical designs [13, 15, 51, 58, 65, 67] or dedicated convolution operations [8,20]. Despite their remarkable performance, ViTs suffer from the quadratic complexity of the self-attention mechanism [54] when dealing with high-resolution images.

Linear Complexity Visual Sequence Modeling. To pursue linear complexity representation, various RNN-like sequence models [21, 40, 47, 64] have attracted growing interest. Building upon State Space Models (SSMs) [10, 21, 22], Vim [69] and VMamba [35] focus on employing 2D scanning strategies to gain global perception. Their variants [4, 29, 39, 63] mainly improve the efficiency-expressiveness trade-off with visually-adapted SSMs, while MambaVision [24] adopts a hybrid architecture of SSM and SwinTransformer [36] blocks for local-to-global perception. In addition, Vision-RWKV [17] introduces temporal decay to capture long-range dependency and ViG [32] adopts gated linear attention [64] to enhance expressiveness. While ViT^3 [23] performs single-step gradient descent, which is equivalent to gated linear attention, our work for the first time proposes Test-Time Training (TTT) as a multi-step visual learner.

Test-Time Training. The early multi-task learning version of TTT [48] introduces an auxiliary self-supervised task parallel to the main task, aiming to address the distribution shift problem in test images [19, 56]. Recently, TTT [2,

46,47] has been revisited in a meta-learning [1] perspective for adaptive sequence modeling, which involves an inner loop for self-supervised reconstruction and an outer loop for different main tasks. The machanism has been applied to the domain of video generation [9,66] and 3D reconstruction [5]. **Our work focuses on the meta-learning formulation of TTT [47]** as an RNN-like visual sequence modeling approach to learn efficient and expressive visual representations.

3 Preliminary

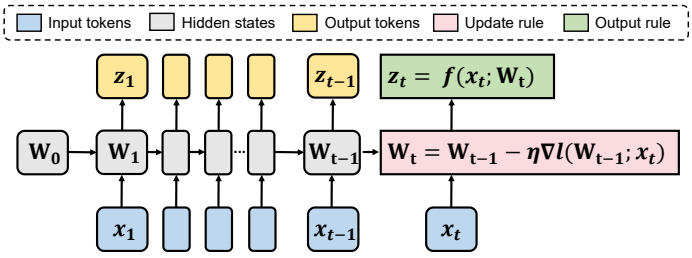

Fig. 2: Basics of TTT. The core idea is to update the hidden state W with steps of self-supervised gradient descent and then forward the output to the next layer.

Similar to most recurrent models like LSTM [27], Mamba [21] and RWKV [40], TTT is also a special RNN that maintains a hidden state $\mathrm{W} \in \mathbb{R}^{D \times D}$ and follows an update rule and an output rule. As illustrated in Fig. 2, the core idea is to treat the token sequence $X = [x_1, \cdots, x_T]$ of a sample as a dataset, where self-supervised learning is performed to compress visual semantics into the hidden state W. **The update rule** is written as the following gradient descent form,

$$\mathrm{W}_t = \mathrm{W}_{t-1} - \eta \nabla_{\mathrm{W}_{t-1}} \ell(\mathrm{W}_{t-1}; x_t), \tag{1}$$

and **the output rule** for the output sequence $Z = [z_1, \cdots, z_T]$ is given by,

$$z_t = f(x_t; \mathrm{W}_t) = \mathrm{W}_t x_t, \tag{2}$$

where ℓ is the self-supervised loss, η is the learning rate and f represents the mapping from X to Z. This adaptation occurs continuously during both training and testing, enabling the model to refine its internal representations in response to observed data patterns, thus called "Test-Time Training" [47].

4 Vision-TTT: Vision Test-Time Training

4.1 TTT as a Vision Learner

Formulation of Vision-TTT. Analogous to the attention mechanism [54], given the input visual sequence $X = [x_1, \cdots, x_T]$, we first project it to the key, query and value vectors,

$$X^K, X^Q, X^V = \theta_K X, \theta_Q X, \theta_V X, \tag{3}$$

where $\theta_K, \theta_Q, \theta_V$ are linear projection matrices. The self-supervised task can be formulated as a reconstruction problem of the original input and the target is set to minimize the $L2$ norm loss,

$$\ell(\mathrm{W}_{t-1}; x_t) = \|f(x_t^K; \mathrm{W}_{t-1}) - x_t^V\|^2 = \|\mathrm{W}_{t-1} x_t^K - x_t^V\|^2, \tag{4}$$

with the output token generated as,

$$z_t = f(x_t^Q; \mathrm{W}_t) = \mathrm{W}_t x_t^Q, \tag{5}$$

where x_t^K, x_t^Q, x_t^V are slices of X^K, X^Q, X^V at timestep t, respectively. By treating $(X^K, X^V) = \{(x_t^K, x_t^V)\}, t = 1, ..., T$ as the training dataset and $(X^Q) = \{x_t^Q\}$ as the test dataset, this represents reconstructing the x_t^V (label view) with the projected x_t^K (training view) and querying with x_t^Q (test view) [47].

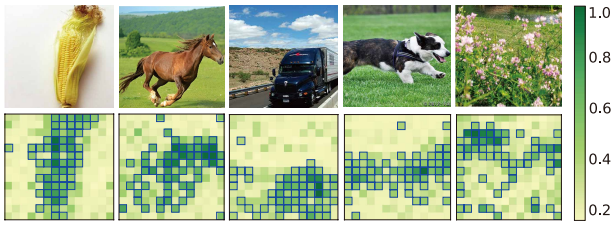

Fig. 3: Gradient Magnitude Map (illustrated by Eq. (6)). Vision-TTT employs gradients as the explicit indicators to quantify visual token semantics.

Interpretability of Vision-TTT. By taking the derivative of W_{t-1} in Eq. (4) with respect to the self-supervised loss ℓ, we can obtain the gradient G_t to quantify the token importance at timestep t,

$$G_t = \nabla_{\mathrm{W}_{t-1}} \ell(\mathrm{W}_{t-1}; x_t) = 2(\mathrm{W}_{t-1} x_t^K - x_t^V)(x_t^K)^T. \tag{6}$$

As shown in Fig. 3, we visualize the G_t distribution on the 2D patch tokens. We observe that tokens with significant visual semantics exhibit larger gradient signal values, which means Vision-TTT has a clear understanding of different regions after training. This provides us with an inherent explainability tool, just as the attention map visualization for vision transformers.

Efficiency of Vision-TTT. The original token-wise TTT relies on sequential computation, and thus suffers from insufficient efficiency. To fully exploit the parallelism of modern GPUs, we reduce the hidden size from $\mathrm{W} \in \mathbb{R}^{D \times D}$ to $\mathrm{W} \in \mathbb{R}^{nh \times d \times d}$ with multi-head mechanism [54], where nh is the number of heads, d is the head dimension, and $D = nh \times d$. In addition, the granularity of gradient descent is changed from 1 to 16. In other words, we perform the mini-batch gradient descent along the visual token sequence with mini-batch size $b = 16$. The update rule within the mini-batch b then follows a causal form [30]. For instance, starting from the initial values of W_0, the hidden state at timestep

t $(0 < t \leq b)$ within the first mini-batch is expressed as,

$$\mathrm{W}_t = \mathrm{W}_{t-1} - \eta G_t = \mathrm{W}_0 - \eta \sum_{s=1}^{t} G_s. \tag{7}$$

By leveraging the small matrix parallel computing capability of Tensor Cores (16×16), we achieve linear-time throughput. Referring to the great work of TTT in NLP [47], we encapsulate the forward and backward kernels with Triton [52] to transform the linear modeling capability of TTT from theory into reality. Due to space, the detailed kernel implementation is deferred to Appendix A.

4.2 Theoretical Comparison

Causal attention has proved to be an RNN with the kernel function of softmax operation [30]. To elaborate on why TTT-style update is more suitable for visual representation than attention and SSMs, we present a comparison of their encoding paradigms in Tab. 1 from a unified perspective of token recurrence.

Table 1: Encoding logic comparison of three methods. ϕ is the kernel function of softmax operation. $\bar{A}$ and $\bar{B}$ correspond to V and K in Mamba [21], respectively.

Method	Encoding Logic of Hidden States
Causal Atten.	$s_t = \sum_{s=1}^{t} \phi(K_s) V_s^T$
SSM	$h_t = \sum_{s=0}^{t} \bar{A}_{t:s} \bar{B}_s x_s$
TTT	$W_t = W_0 + \sum_{s=1}^{t} (V_s - W_{s-1} K_s) K_s^T$

Specifically, causal attention encodes each token merely with input selectivity via $K/V = \mathrm{Linear(X)}$. SSM further introduces a structured matrix $\bar{A}$ (typically a sparse diagonal matrix) for multiplicative decay. By contrast, TTT leverages a subtractive discrepancy $(V_t - W_{t-1}K_t)$ for the encoding of online tokens, which **selectively captures the differential features between the local token V_t and global representation W_{t-1}**. The set of $W_{t-1}K_t$ terms actually structures a string of *relative position anchors* to retain visual locality, which is inherently suitable for visual tasks with 2D correlations. TTT is not simply a switch to the gradient viewpoint, but a theoretically position-aware vision learner that distinguishes itself from previous visual modeling methods. The detailed derivation of the recurrence formula can be found in Appendix C.

4.3 Overall Architecture

As illustrated in Fig. 4, Vision-TTT consists of three stages as follows.
❶ **Patchification.** The input image $I \in \mathbb{R}^{H \times W \times C}$ is split into a sequence of 1D patches $x_p \in \mathbb{R}^{T \times (P^2 \cdot C)}$ [16, 69], where (H, W) is the resolution of the original image, C is the channel, P^2 is the size of each rectangle patch and the sequence length is $T = \frac{HW}{P^2}$. Then a learnable projection $E \in \mathbb{R}^{(P^2 \cdot C) \times D}$ is appended and

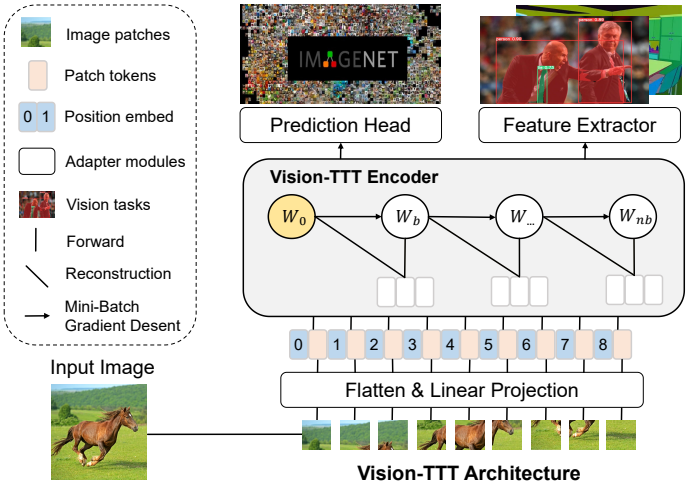

Fig. 4: Overall architecture of Vision-TTT. It consists of three stages: Patchification, Vision-TTT Encoder, and Task Adapters to learn representation for vision tasks.

a position embedding $E_{\text{pos}} \in \mathbb{R}^{(T+1)\times D}$ is added to form the processed input y_0 of feature embedding D,

$$y_0 = [x_p^1 E, x_p^2 E, \cdots, x_p^T E] + E_{\text{pos}}, \tag{8}$$

❷ **Vision-TTT Encoder.** The processed input is then fed into L hybrid blocks. Each of them consists of a `Vittt` block and a successive `SwiGluMLP` [43], a variant of Gated Linear Units [11].

$$\begin{aligned} y_l' &= \texttt{Vittt}\big(\text{LN}(y_{l-1})\big) + y_{l-1}, \\ y_l &= \texttt{SwiGluMLP}\big(\text{LN}(y_l')\big) + y_l', \end{aligned} \tag{9}$$

where LN denotes layer normalization, $y_l{}'$ is the intermediate activation.
❸ **Task Adapters.** During the ImageNet [12] pretraining stage, we adopt mean pool and a linear detection head to extract the class possibility $\hat{p}$ as follows,

$$\hat{p} = \texttt{Linear}\Big(\texttt{MeanPool}\big(\text{LN}(y_L)\big)\Big). \tag{10}$$

When finetuning for downstream COCO [33] detection and ADE20K [68] segmentation tasks, the common practice is to freeze the encoder backbone and only train the feature extractors for adaptation.

4.4 Design of `Vittt` Block

The design route from the original TTT layer [47] to our `Vittt` block is illustrated in Fig. 5. The projection matrices Q / K of the TTT layer are shared to save parameters, and the vanilla TTT block [47] is essentially a gated linear version. From the view of self-supervised learning in Sec. 4.1, the input sequence X

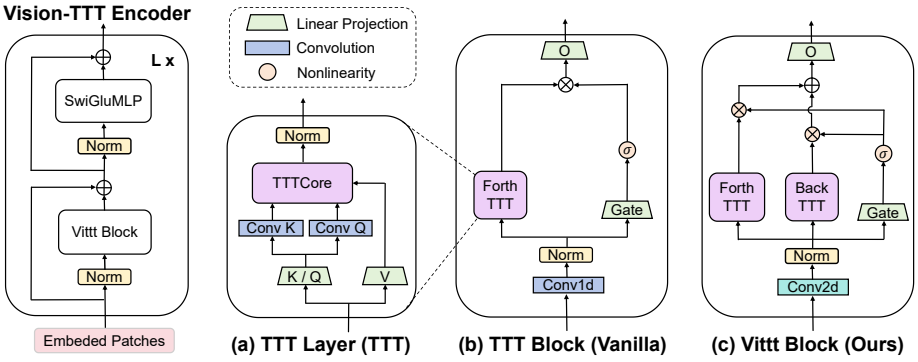

Fig. 5: `Vittt` Block. It evolves from vanilla TTT with the dual dataset strategy and the Conv2d-based dataset preprocessing mechanism for inner self-supervised learning.

gives one dataset $(X^K, X^V) = \{(x_t^K, x_t^V)\}, t = 1, ..., T$. The vanilla TTT (Fig. 5 (b)) is formulated with Conv1d-based dataset preprocessing mechanism,

$$Z_{\text{Gated}} = \sigma(\texttt{Linear}(X_{\text{Conv1d}})) \odot Z. \tag{11}$$

Although the vanilla TTT block is a promising vision learner, it is originally designed for unidirectional modeling, which conflicts with 2D spatial data. To introduce spatial locality, we further integrate two 2D architectural designs.

Dual Dataset Strategy. We construct another dataset in the backward direction $(X^{K'}, X^{V'}) = \{(x_t^{K'}, x_t^{V'})\}, t = T, ..., 1$ to alleviate the unidirectional bias of 1D global sequence modeling [42, 69], where the self-supervised learning performs simultaneously in the forward (Z_{forth}) and backward (Z_{back}) routes,

$$Z_{\text{Dual}} = \sigma(\texttt{Linear}(X_{\text{Conv2d}})) \odot (Z_{\text{forth}} + \texttt{flip}(Z_{\text{back}})). \tag{12}$$

Conv2d-based Dataset Preprocessing. The $X_{\text{Conv2d}} = \texttt{DWConv}(X)$ is the pre-processed result of the 1D input sequence X for 2D data augmentation of the self-supervised learning. We use depth-wise convolution [7] to reduce the parameters introduced (0.02M, 0.04M, 0.08M for `Vittt-T/S/B`).

For TTT to perform self-supervised learning along the visual sequence datasets, the dual dataset strategy and the Conv2d-based dataset preprocessing mechanism diversify the datasets to 2D, enabling effective adaptation to visual tasks.

5 Experiments

We conduct extensive experiments to validate the expressiveness of Vision-TTT, including ImageNet [12] classification and downstream COCO [33] detection and ADE20K [68] segmentation tasks. We also demonstrate its linear efficiency by measuring the FLOPs, throughput and memory metrics across resolutions.

Table 2: Performance comparison on ImageNet-1K classification. We report the parameters (#P.), FLOPs and Top-1 Accuracy (%) metrics. *Left*: hierarchical architectures; *Right*: non-hierarchical architectures.

Method	size	#P.	FLOPs	Acc.
CNNs				
RegNetY-4G	224^2	12M	4G	80.0
RegNetY-8G	224^2	25M	8G	81.7
RegNetY-16G	224^2	45M	16G	82.9
EffNet-B4	380^2	19M	4.2G	82.9
EffNet-B5	456^2	30M	9.9G	83.6
EffNet-B6	528^2	43M	19.0G	84.0
ConvNeXt-T	224^2	29M	4.5G	82.1
ConvNeXt-S	224^2	50M	8.7G	83.1
ConvNeXt-B	224^2	89M	15.4G	83.8
Transformers				
Swin-T	224^2	28M	4.5G	81.3
Swin-S	224^2	50M	8.7G	83.2
Swin-B	224^2	88M	15.4G	83.5
SSMs				
VMamba-T	224^2	30M	4.9G	82.5
VMamba-S	224^2	50M	8.7G	83.6
VMamba-B	224^2	89M	15.4G	83.9
MambaVision-T	224^2	32M	4.4G	82.3
MambaVision-S	224^2	50M	7.5G	83.3
MambaVision-B	224^2	98M	15.0G	84.2

Method	size	#P.	FLOPs	Acc.
Transformers				
ViT-B/16	384^2	86M	55.4G	77.9
ViT-L/16	384^2	307M	190.7G	76.5
DeiT-T	224^2	6M	1.3G	72.2
DeiT-S	224^2	22M	4.6G	79.8
DeiT-B	224^2	86M	17.6G	81.8
RWKVs				
VRWKV-T	224^2	6M	1.2G	75.1
VRWKV-S	224^2	24M	4.6G	80.1
VRWKV-B	224^2	94M	18.2G	82.0
SSMs				
Vim-T	224^2	7M	1.5G	76.1
Vim-S	224^2	26M	5.2G	80.3
Vim-B	224^2	98M	18.8G	81.9
TTTs				
ViT^3-T	224^2	6M	1.2G	76.5
ViT^3-S	224^2	24M	4.8G	81.6
ViT^3-B	224^2	90M	18.0G	82.6
`Vittt-T`	224^2	7M	1.4G	**77.7**
`Vittt-S`	224^2	26M	5.3G	**81.8**
`Vittt-B`	224^2	102M	20.3G	**82.7**

5.1 Image Classification

Settings. We first evaluate `Vittt` on the ImgeNet-1K dataset [12] and compare its performance against the benchmarks from ViT [16], DeiT [53], and linear-complexity Vision-RWKV [17], Vim [69], ViT^3 [23]. Despite using longer sequence lengths, hierarchical baselines [24, 35–37, 49, 62] are still included as references. For fair comparison, we mainly follow the training recipe of DeiT [53] and SwinTransformer [36]. The training details are provided in Appendix D.

Main Results. As shown in Tab. 2, `Vittt-T/S/B` achieves 77.7%, 81.8%, 82.7% Top-1 accuracy on ImageNet classification, respectively. Compared with other methods, `Vittt-T/S` have significant improvements, which are +1.6% and +1.5% better than Vim-T/S. Even for the base size model, `Vittt-B` surpasses its counterparts DeiT-B, VRWKV-B and Vim-B by +0.7% $\sim$ +0.9%, demonstrating the strong expressiveness of Vision-TTT at scale. Notably, `Vittt` even surpasses the strongest baseline ViT^3. The reason is that ViT^3 performs only one batch gradient descent step, but `Vittt` replaces it with sequence-aware (i.e. T/b) gradient descent steps, enabling continuous adaptation along the visual sequences.

Table 3: Performance comparison on downstream COCO2017 detection (*Left*) and ADE20K segmentation (*Right*) tasks. We report the parameters (#Param.), FLOPs and AP^b, AP^m, mIoU metrics, respectively. FLOPs are calculated with 1333×800 resolution for detection and 512×512 resolution for segmentation tasks.

Method	#Param.	FLOPs	AP^b	AP^m
ViT-T	8M	147.1G	41.6	37.9
VRWKV-T	8M	67.9G	41.7	38.0
Vim-T	9M	76.7G	41.8	38.2
ViT^3-T	8M	69.2G	42.0	38.3
`Vittt-T`	9M	74.3G	**42.9**	**38.7**
ViT-S	28M	344.5G	44.9	40.1
VRWKV-S	29M	189.9G	44.8	40.2
Vim-S	32M	203.6G	44.9	40.2
ViT^3-S	30M	195.6G	45.4	40.7
`Vittt-S`	32M	206.3G	**46.3**	**41.4**
ViT-B	100M	893.3G	46.8	41.8
VRWKV-B	107M	599.0G	46.8	41.7
Vim-B	111M	611.5G	46.6	41.6
ViT^3-B	103M	597.1G	47.4	42.1
`Vittt-B`	115M	646.7G	**48.6**	**43.0**

Method	#Param.	FLOPs	mIoU
ViT-T	8M	20.9G	42.6
VRWKV-T	8M	16.6G	43.3
Vim-T	9M	18.4G	43.4
ViT^3-T	8M	16.7G	43.6
`Vittt-T`	9M	17.8G	**44.3**
ViT-S	28M	54.0G	46.2
VRWKV-S	29M	46.3G	47.2
Vim-S	32M	49.7G	47.0
ViT^3-S	30M	47.7G	47.6
`Vittt-S`	32M	50.3G	**48.4**
ViT-B	100M	157.9G	48.8
VRWKV-B	107M	146.0G	49.2
Vim-B	111M	149.2G	48.9
ViT^3-B	103M	145.7G	49.4
`Vittt-B`	115M	157.7G	**50.3**

5.2 Downstream Tasks

Settings. We benchmark `Vittt` model against other popular non-hierarchical visual sequence modeling methods, including ViT [16], Vision-RWKV [17], Vim [69] and ViT^3 [23]. For object detection and instance segmentation, we evaluate `Vittt` on the COCO2017 dataset [33]. The procedure mainly follows the Vision-RWKV recipe [17] to integrate the ViT-Adapter [6] and Mask R-CNN [25] detection head on our plain `Vittt` models. The image resolution is scaled up to 1333×800. In addition, we perform semantic segmentation on the ADE20K dataset [68]. For a fair comparison, we also follow prior works Vim [69] and Vision-RWKV [17] to use UperNet [60] as the segmentation head. All images are cropped into 512×512. Further details are listed in Appendix D.

Main Results. Tab. 3 demonstrates the remarkable superiority of Vision-TTT on dense prediction tasks. `Vittt-T` outperforms the popular Vim-T by +1.1% AP^b, +0.5% AP^m and +0.9% mIoU. The advantage is further enlarged to +1.4% AP^b, +1.2% AP^m and +1.4% mIoU for `Vittt-S` against Vim-S. For the base size model, where Vim-B fails to scale effectively, `Vittt-B` still maintains consistent performance gains of +2.0% AP^b, +1.3% AP^m and +1.1% mIoU over its VRWKV-B counterpart. Although ViT^3 achieves comparable performance with `Vittt` in classification, it exhibits a considerable performance gap in downstream dense prediction tasks due to limited gradient descent steps for adaptation. In contrast, the maintenance of `Vittt`'s performance superiority from 512×512 (1024 image tokens) to 1333×800 (4200 image tokens) effectively demonstrates the scalable representation capability of Vision-TTT's adaptive gradient descent steps, particularly in addressing more challenging long sequence modeling tasks.

5.3 Efficiency Analysis

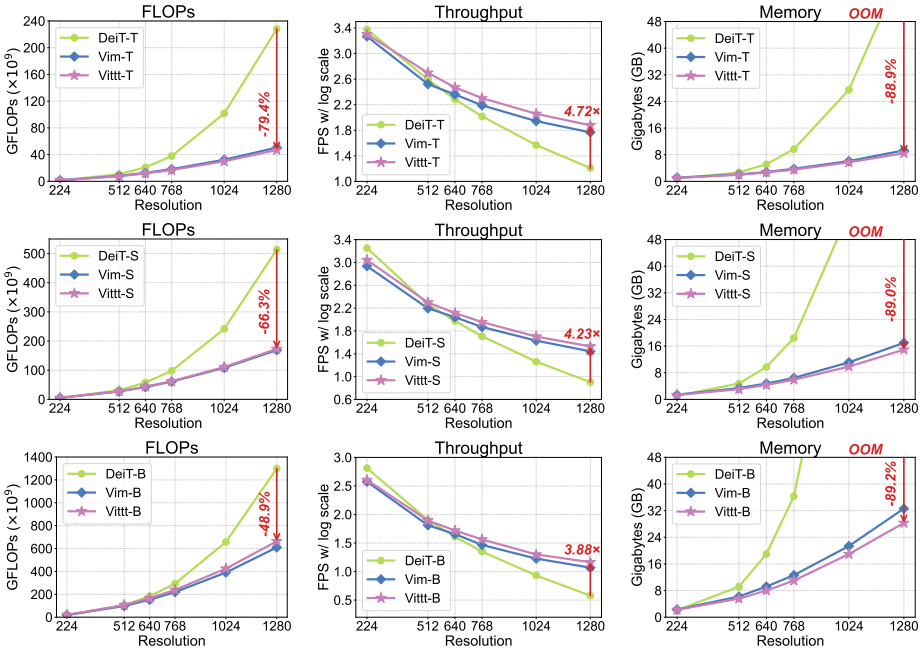

Fig. 6: Efficiency comparison between DeiT [53], Vim [69] and our `Vittt` model. We sketch the FLOPs, throughput, and memory footprint curves for different resolutions ranging from 224×224 (i.e. 196 tokens) to 1280×1280 (i.e. 6400 tokens).

Settings. We compare the theoretical complexity and runtime throughput of `Vittt` with ViT [16] and Vim [69]. Following [36,59], we set the batch size to 64 to test the FPS (frames per second) and memory footprint with an L40S GPU.
Computational Efficiency. Given an input sequence $y \in \mathbb{R}^{1\times T\times D}$, the theoretical computation complexity is given by,

$$\begin{aligned}
\Omega(\text{ViT}) &= 4TD^2 + 2T^2D \\
\Omega(\text{Vim}) &= 6TD^2 + 18T(2D)N \\
\Omega(\texttt{Vittt}) &= 6TD^2 + 6TDd + 4bTD,
\end{aligned} \tag{13}$$

where $b = 16$ denotes the mini-batch size, $d = 64$ is the head dimension, and $N = 16$ is the state expansion factor of Mamba [21]. By referring to Vim [69], we include the input/output projections, the dicretization steps, the gating operations and then multiply the bidirectional factor to compute the overall complexity of Vim blocks. Obviously, ViT exhibits quadratic complexity w.r.t. the sequence length T. In contrast, both Vim and `Vittt` enjoy linear complexity via RNN-style updates, enabling efficient long-sequence modeling. As shown in the left two columns of Fig. 6, the linearly increasing FLOPs and throughput of `Vittt` align well with the theoretical complexity analysis. Although DeiT achieves slightly higher throughput in processing 224×224 images, it slows down

quickly with the growth of resolution. At 1280×1280 resolution, `Vittt-T/S/B` saves $79.4\%, 66.3\%, 48.9\%$ FLOPs and achieves $4.72\times, 4.23\times, 3.88\times$ higher FPS than DeiT-T/S/B, respectively. For the linear models `Vittt` and Vim, `Vittt-T` has fewer FLOPs than Vim-T, but `Vittt-S/B` yield slightly more FLOPs than Vim-S/B, implying that `Vittt` has a larger constant coefficient related to the embedding size D. Nevertheless, we observe that `Vittt` consistently outperforms Vim in speed across different model sizes and input resolutions. This stems from their distinct kernel utilization strategies during execution. While Vim relies on the selective scan mechanism [21] parallelized over CUDA Cores, `Vittt` directly operates on Tensor Cores, which are 16×16 specialized matrix multiplication units that runs several to tens of times faster.

Memory Efficiency. With the global batch size of B, the asymptotic memory complexity is as follows,

$$\begin{aligned} \Omega(\text{ViT}) &= \mathcal{O}(BTD + BT^2) \\ \Omega(\text{Vim}) &= \mathcal{O}(BTD + BTDN) \\ \Omega(\texttt{Vittt}) &= \mathcal{O}\left(BTD + BTd + BTDd/b\right). \end{aligned} \tag{14}$$

Linear models of Vim and `Vittt` alleviate the quadratic memory cost w.r.t. T that ViT demands. Our implementation follows Mamba [21] to incorporate kernel fusion and recomputation techniques, which further reduces the memory footprint to $\mathcal{O}\left(BTD\right)$. As shown in the right column of Fig. 6, while the memory consumption of DeiT explodes at high resolutions, the footprint only grows linearly for Vim and `Vittt`. At 1280×1280 resolution, `Vittt-T/S/B` consume approximately 89.0% less memory than DeiT-T/S/B. We also observe a considerable memory reduction of `Vittt-S/B` compared to Vim-S/B beyond 1024×1024 resolution, indicating that `Vittt` enjoys a smaller constant factor of memory usage than Vim.

The above experiments consistently demonstrate the linear-complexity computation and memory consumption of Vision-TTT across diverse model scales and sequence lengths. This effectively bypasses the scalability bottleneck of Vision Transformers and renders it highly favorable for high-resolution modeling. For readers with further interest in Vision-TTT's efficiency, we provide the derivation of the computational complexity and the details of hardware-aware implementation in Appendix B.

5.4 Ablation Study

To verify the effectiveness of our design, we systematically explore the design space to identify favorable configurations. Specifically, we conduct ablation studies on 1) the inner modules of the `Vittt` block to validate the rationality of our architecture design; 2) different classification strategies to obtain the most compatible supervised learning scheme; 3) mini-batch size b to determine the suitable steps of gradient descent; 4) number of learnable initial states $\mathrm{W}_0 \in \mathbb{R}^{nh \times d \times d}$ to enable effective dual dataset learning. The results demonstrate that our design for Vision-TTT is modularly effective, friendly to the inner supervised learning, and achieves a favorable trade-off between model expressiveness and efficiency.

Table 4: Performance comparison for different snapshots on the design route for `Vittt-T`. Throughput metric FPS (frames per second) is tested with batch size 64.

Design Route	#Param.	FLOPs	FPS	Acc.	AP^b	AP^m	mIoU
TTT layer	5.87M	1.14G	3607	74.2	40.4	36.8	41.5
+ Share Q / K	5.43M	1.07G	3892	74.0	40.1	36.5	41.2
+ Gating	5.89M	1.15G	3565	75.2	41.2	37.1	42.0
+ Conv1d (vanilla TTT)	5.90M	1.16G	3346	75.6	41.6	37.6	42.6
+ Dual dataset strategy	6.96M	1.42G	2068	77.5	42.7	38.6	44.1
+ Conv2d dataset preprocessing	6.98M	1.44G	2029	77.7	42.9	38.7	44.3

Design Route of the Vittt Block. The design route presented in Tab. 4 follows the illustration in Sec. 4.4. Starting from the TTT layer, the shared Q / K projection first achieves a parameter reduction of 0.44M with no discernible performance degradation. Subsequently, the gating mechanism and Conv1d module are incorporated, endowing TTT with nonlinearity and unidirectional token perception. This integration results in a 14.1% decrease in model throughput, while yielding preliminary gains of +1.6% in classification accuracy, +1.5% AP^b, +1.1% AP^m for detection and +1.4% mIoU for segmentation tasks. To construct the proposed `Vittt` block, we diversify the visual sequence dataset in two directions. This dual dataset strategy substantially enhances performance by +1.9% in classification, +1.1% AP^b, +1.0% AP^m in detection and +1.5% mIoU in segmentation. The significant improvement justifies the sacrifice of 38.2% in FPS. Finally, the Conv2d dataset preprocessing mechanism further contributes incremental performance gains of $+0.1\% \sim 0.2\%$ for different metrics.

Table 5: Performance comparison for different classification strategies on `Vittt-T`.

Method	#Param.	FLOPs	Acc.	AP^b	AP^m	mIoU
HeadClassTok	6.98M	1.44G	76.6	41.6	37.8	43.0
MidClassTok	6.98M	1.44G	76.8	41.7	37.9	43.0
DoubleClassTok	6.98M	1.45G	75.4	41.2	37.2	42.5
MaxPool	6.98M	1.44G	77.2	42.2	38.4	43.8
MeanPool	6.98M	1.44G	77.7	42.9	38.7	44.3

Effects of the Classification Strategy. Following Vim [69], we experiment with five different ways to obtain the classification feature to identify the optimal scheme for class-supervised learning. The head class token [16] refers to insert an extra token before the input patch sequence and using only this token after the output layer to compute the class possibilities. The "MidClassTok" and "DoubleClassTok" are two variants to insert one class token in the middle or two class tokens before and after the input patch sequence [69], respectively. "MaxPool" and "MeanPool" are two pooling strategies that extract the $L2$-norm maximum or the average of the output feature sequence. As shown in Tab. 5, the two pooling strategies achieve significantly better performance than the other three class token strategies, and the "MeanPool" strategy is the most suitable for the gradient-driven representation of Vision-TTT.

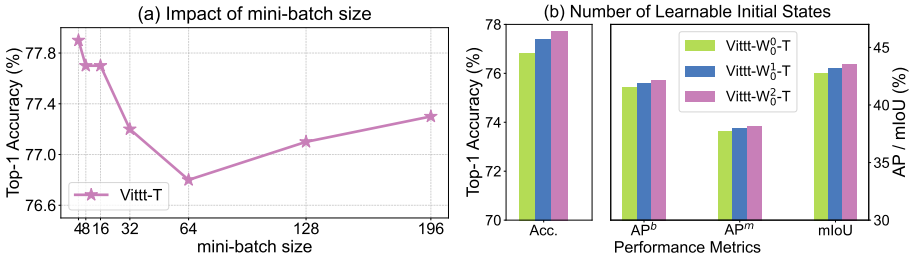

Fig. 7: Ablation study of (a) mini-batch size b to perform gradient descent along the sequence length T, and (b) number of learnable initial states W_0 (indicated by the superscript number ^{012}) for the self-supervised learning of dual datasets.

Impact of the Mini-Batch Size b. Mini-batch size b along the sequence length T is the crucial hyper-parameter to decide the steps of mini-batch gradient descent (i.e. $\lceil T/b \rceil$ steps). As shown in Fig. 7 (a), with the increase of b from 4 (49 steps of mini-batch gradient descent) to 196 (only one batch gradient descent), the Top-1 classification accuracy first decreases sharply and then grows slightly. The worst case appears in the middle of $b = 64$, as there are neither enough gradient descent steps nor a sufficiently global gradient perspective. Fortunately, our kernel implementation of mini-batch size $b = 16$ lies in the sweet region from $4 \sim 16$ to balance the expressiveness and achieved efficiency.

Number of Learnable Initial States. Recall that in Sec. 4.4 we adopt dual dataset strategy to achieve bidirectional diversity of 2D spatial data. In the process, there are two initial states $\mathrm{W}_0 \in \mathbb{R}^{nh \times d \times d}$ in `Forth TTT` and `Back TTT` modules. In such a scenario, the initialization of the hidden states is a noteworthy problem [47]. Unlike general parameter initialization, these states can either be initialized once before training, or set as learnable parameters and trained during supervised learning. We denote the one-time initialization before training as `Vittt-`$\mathtt{W}_0^0$, where no hidden states are learnable. The setting `Vittt-`$\mathtt{W}_0^2$ uses two separate learnable initial states for `Forth TTT` and `Back TTT` and `Vittt-`$\mathtt{W}_0^1$ shares one copy of learnable initial state for the dual datasets. As shown in Fig. 7 (b), more learnable initial states generally reduces interference between the self-supervised learning of dual datasets and leads to improved performance.

6 Visualization of Interpretability

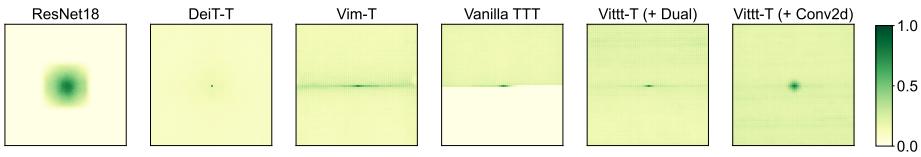

Fig. 8: Comparison of Effective Receptive Field (ERF) [38]. Pixels with higher intensity indicate larger responses related to the central pixel.

Effective Receptive Field (ERF). The Effective Receptive Field (ERF) [14, 38] refers to the region in the input space that contributes to the activations of a specific output unit. As shown in Fig. 8 , we conduct a comparative analysis of

the central pixel's ERF across various visual backbones, including ResNet [26], DeiT [53], Vim [69] and three snapshots of `Vittt` illustrated in Sec. 5.4. From the vanilla TTT to the "+ Dual" dataset strategy version, `Vittt` overcomes the limitations of unidirectional modeling. Further with 2D dataset preprocessing, the "+ Conv2d" version ultimately exhibits the globally radial pattern, demonstrating the effectiveness of our design route in Sec. 4.4. In addition, we observe that DeiT, Vim, `Vittt` all achieve global coverage of the pixels, but the 2D-aware distribution of `Vittt` may intuitively explain its superior expressiveness.

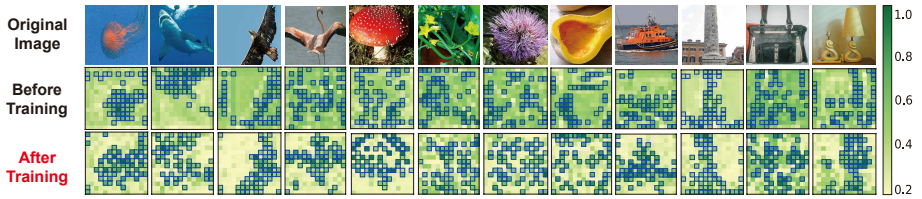

Fig. 9: Gradient Magnitude Map of `Vittt-T` before and after training. Visual tokens with higher intensity indicate steeper gradients (Eq. (6))and greater importance.

Gradient Magnitude Map. Similar to the attention maps [54] for Vision Transformers and the activation maps [35] for Vision Mambas, there are also intermediate features to gain an in-depth understanding of Vision-TTT's representation. As shown in Fig. 9, we visualize the Gradient Magnitude Map (GMM) of Vision-TTT before and after training and observe that Vision-TTT develops a discriminative understanding of different input regions. The GMM serves as a unique interpretive tool for Vision-TTT, offering patch-level explanatory insights that may lay the foundation for future academic advancements of Vision-TTT.

7 Conclusion

In this paper, we propose Vision-TTT that introduces a novel sequence modeling method Test-Time Training to the visual representation learning domain. Our method addresses the inherent limitations of Vision Transformers by maintaining a globally radial receptive field meanwhile keeping linear complexity. To adapt the unidirectional TTT model for 2D vision tasks, we propose to utilize the dual dataset strategy for dataset diversity and Conv2d-based module for dataset preprocessing in `Vittt` blocks. The significant performance of `Vittt` across various vision tasks at different model scales, and the verified efficiency for high-resolution images, together demonstrate it as an alternative for the next-generation visual backbone.

Acknowledgements

This work is supported by National Science Foundation of China under grants 62576310, Zhejiang Provincial National Science Foundation of China under Grant No. LZ25F020007 and ICT2026A20.

References

1. Andrychowicz, M., Denil, M., Colmenarejo, S.G., Hoffman, M.W., Pfau, D., Schaul, T., de Freitas, N.: Learning to learn by gradient descent by gradient descent. In: Neural Information Processing Systems (2016)
2. Behrouz, A., Li, Z., Kacham, P., Daliri, M., Deng, Y., Zhong, P., Razaviyayn, M., Mirrokni, V.: Atlas: Learning to optimally memorize the context at test time. arXiv preprint arXiv:2505.23735 (2025)
3. Chen, C.F.R., Fan, Q., Panda, R.: Crossvit: Cross-attention multi-scale vision transformer for image classification. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 347–356 (2021)
4. Chen, T., Ye, Z., Tan, Z., Gong, T., Wu, Y., Chu, Q., Liu, B., Yu, N., Ye, J.: Mim-istd: Mamba-in-mamba for efficient infrared small-target detection. IEEE Transactions on Geoscience and Remote Sensing (2024)
5. Chen, X., Chen, Y., Xiu, Y., Geiger, A., Chen, A.: Ttt3r: 3d reconstruction as test-time training. arXiv preprint arXiv:2509.26645 (2025)
6. Chen, Z., Duan, Y., Wang, W., He, J., Lu, T., Dai, J., Qiao, Y.: Vision transformer adapter for dense predictions. arXiv preprint arXiv:2205.08534 (2022)
7. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1800–1807 (2017)
8. Dai, Z., Liu, H., Le, Q.V., Tan, M.: Coatnet: Marrying convolution and attention for all data sizes. Advances in neural information processing systems pp. 3965–3977 (2021)
9. Dalal, K., Koceja, D., Xu, J., Zhao, Y., Han, S., Cheung, K.C., Kautz, J., Choi, Y., Sun, Y., Wang, X.: One-minute video generation with test-time training. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 17702–17711 (2025)
10. Dao, T., Gu, A.: Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In: International Conference on Machine Learning (ICML) (2024)
11. Dauphin, Y.N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks (2017)
12. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition pp. 248–255 (2009)
13. Ding, M., Xiao, B., Codella, N.C.F., Luo, P., Wang, J., Yuan, L.: Davit: Dual attention vision transformers. In: European Conference on Computer Vision (2022)
14. Ding, X., Zhang, X., Han, J., Ding, G.: Scaling up your kernels to 31×31: Revisiting large kernel design in cnns. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11953–11965 (2022)
15. Dong, X., Bao, J., Chen, D., Zhang, W., Yu, N., Yuan, L., Chen, D., Guo, B.: Cswin transformer: A general vision transformer backbone with cross-shaped windows. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 12114–12124 (2021)
16. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. ICLR (2021)

17. Duan, Y., Wang, W., Chen, Z., Zhu, X., Lu, L., Lu, T., Qiao, Y., Li, H., Dai, J., Wang, W.: Vision-rwkv: Efficient and scalable visual perception with rwkv-like architectures. arXiv preprint arXiv:2403.02308 (2024)
18. Fukushima, K.: Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics (1980)
19. Gandelsman, Y., Sun, Y., Chen, X., Efros, A.: Test-time training with masked autoencoders. Advances in Neural Information Processing Systems (2022)
20. Gao, P., Lu, J., Li, H., Mottaghi, R., Kembhavi, A.: Container: Context aggregation network. arXiv preprint arXiv:2106.01401 (2021)
21. Gu, A., Dao, T.: Mamba: Linear-time sequence modeling with selective state spaces. In: First conference on language modeling (2024)
22. Gu, A., Goel, K., Ré, C.: Efficiently modeling long sequences with structured state spaces. arXiv preprint arXiv:2111.00396 (2021)
23. Han, D., Li, Y., Li, T., Cao, Z., Wang, Z., Song, J., Cheng, Y., Zheng, B., Huang, G.: Vit^3: Unlocking test-time training in vision. arXiv preprint arXiv:2512.01643 (2025)
24. Hatamizadeh, A., Kautz, J.: Mambavision: A hybrid mamba-transformer vision backbone. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 25261–25270 (2025)
25. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask r-cnn (2017)
26. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)
27. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**, 1735–1780 (1997)
28. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2261–2269 (2017)
29. Huang, T., Pei, X., You, S., Wang, F., Qian, C., Xu, C.: Localmamba: Visual state space model with windowed selective scan. In: ECCV Workshops (2024)
30. Katharopoulos, A., Vyas, A., Pappas, N., Fleuret, F.: Transformers are rnns: Fast autoregressive transformers with linear attention. In: International Conference on Machine Learning (2020)
31. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Communications of the ACM (2012)
32. Liao, B., Wang, X., Zhu, L., Zhang, Q., Huang, C.: Vig: Linear-complexity visual sequence learning with gated linear attention. In: Proceedings of the AAAI Conference on Artificial Intelligence (2025)
33. Lin, T.Y., Maire, M., Belongie, S.J., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European Conference on Computer Vision (2014)
34. Liu, S., Chen, T., Chen, X., Chen, X., Xiao, Q., Wu, B., Kärkkäinen, T., Pechenizkiy, M., Mocanu, D., Wang, Z.: More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. arXiv preprint arXiv:2207.03620 (2022)
35. Liu, Y., Tian, Y., Zhao, Y., Yu, H., Xie, L., Wang, Y., Ye, Q., Jiao, J., Liu, Y.: Vmamba: Visual state space model. Advances in neural information processing systems (2024)
36. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. 2021 IEEE/CVF International Conference on Computer Vision (ICCV) pp. 9992–10002 (2021)

37. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11966–11976 (2022)
38. Luo, W., Li, Y., Urtasun, R., Zemel, R.: Understanding the effective receptive field in deep convolutional neural networks. Advances in neural information processing systems (2016)
39. Pei, X., Huang, T., Xu, C.: Efficientvmamba: Atrous selective scan for light weight visual mamba. In: Proceedings of the AAAI conference on artificial intelligence (2025)
40. Peng, B., Alcaide, E., Anthony, Q., Albalak, A., Arcadinho, S., Biderman, S., Cao, H., Cheng, X., Chung, M., Derczynski, L., et al.: Rwkv: Reinventing rnns for the transformer era. In: Findings of the association for computational linguistics: EMNLP 2023. pp. 14048–14077 (2023)
41. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning (2021)
42. Schuster, M., Paliwal, K.: Bidirectional recurrent neural networks. Signal Processing, IEEE Transactions on pp. 2673 – 2681 (12 1997)
43. Shazeer, N.: Glu variants improve transformer. arXiv preprint arXiv:2002.05202 (2020)
44. Sim'eoni, O., Vo, H.V., Seitzer, M., Baldassarre, F., Oquab, M., Jose, C., Khalidov, V., Szafraniec, M., Yi, S., Ramamonjisoa, M., Massa, F., Haziza, D., Wehrstedt, L., Wang, J., Darcet, T., Moutakanni, T., Sentana, L., Roberts, C., Vedaldi, A., Tolan, J., Brandt, J., Couprie, C., Mairal, J., J'egou, H., Labatut, P., Bojanowski, P.: Dinov3 (2025)
45. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
46. Sun, Y., Li, X., Dalal, K., Hsu, C., Koyejo, S., Guestrin, C., Wang, X., Hashimoto, T., Chen, X.: Learning to (learn at test time). arXiv preprint arXiv:2310.13807 (2023)
47. Sun, Y., Li, X., Dalal, K., Xu, J., Vikram, A., Zhang, G., Dubois, Y., Chen, X., Wang, X., Koyejo, S., et al.: Learning to (learn at test time): Rnns with expressive hidden states. arXiv preprint arXiv:2407.04620 (2024)
48. Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A.A., Hardt, M.: Test-time training with self-supervision for generalization under distribution shifts. In: International Conference on Machine Learning (2019)
49. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
50. Tan, M., Le, Q.V.: Efficientnetv2: Smaller models and faster training. In: International Conference on Machine Learning (2021)
51. Tian, Y., Xie, L., Wang, Z., Wei, L., Zhang, X., Jiao, J., Wang, Y., Tian, Q., Ye, Q.: Integrally pre-trained transformer pyramid networks. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 18610–18620 (2023)
52. Tillet, P., Kung, H.T., Cox, D.D.: Triton: an intermediate language and compiler for tiled neural network computations. Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages (2019)
53. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., J'egou, H.: Training data-efficient image transformers & distillation through attention. In: International Conference on Machine Learning (2020)

54. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. p. 6000–6010. NIPS'17, Curran Associates Inc., Red Hook, NY, USA (2017)
55. Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., Liu, W., Xiao, B.: Deep high-resolution representation learning for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)
56. Wang, R., Sun, Y., Tandon, A., Gandelsman, Y., Chen, X., Efros, A.A., Wang, X.: Test-time training on video streams. Journal of Machine Learning Research (2025)
57. Wang, W., Dai, J., Chen, Z., Huang, Z., Li, Z., Zhu, X., Hu, X., Lu, T., Lu, L., Li, H., Wang, X., Qiao, Y.: Internimage: Exploring large-scale vision foundation models with deformable convolutions. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 14408–14419 (2022)
58. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. 2021 IEEE/CVF International Conference on Computer Vision (ICCV) pp. 548–558 (2021)
59. Wightman, R.: Pytorch image models. `https://github.com/rwightman/pytorch-image-models` (2019), accessed: August 24, 2025
60. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: Proceedings of the European conference on computer vision (ECCV). pp. 418–434 (2018)
61. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5987–5995 (2017)
62. Xu, J., Pan, Y., Pan, X., Hoi, S.C.H., Yi, Z., Xu, Z.: Regnet: Self-regulated network for image classification. IEEE Transactions on Neural Networks and Learning Systems (2021)
63. Yang, C., Chen, Z., Espinosa, M., Ericsson, L., Wang, Z., Liu, J., Crowley, E.J.: Plainmamba: Improving non-hierarchical mamba in visual recognition. arXiv preprint arXiv:2403.17695 (2024)
64. Yang, S., Wang, B., Shen, Y., Panda, R., Kim, Y.: Gated linear attention transformers with hardware-efficient training. arXiv preprint arXiv:2312.06635 (2023)
65. Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Tay, F.E.H., Feng, J., Yan, S.: Tokens-to-token vit: Training vision transformers from scratch on imagenet. 2021 IEEE/CVF International Conference on Computer Vision (ICCV) pp. 538–547 (2021)
66. Zhang, T., Bi, S., Hong, Y., Zhang, K., Luan, F., Yang, S., Sunkavalli, K., Freeman, W.T., Tan, H.: Test-time training done right. arXiv preprint arXiv:2505.23884 (2025)
67. Zhang, X., Tian, Y., Xie, L., Huang, W., Dai, Q., Ye, Q., Tian, Q.: Hivit: A simpler and more efficient design of hierarchical vision transformer. In: International Conference on Learning Representations (2023)
68. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Semantic understanding of scenes through the ade20k dataset. International Journal of Computer Vision (2016)
69. Zhu, L., Liao, B., Zhang, Q., Wang, X., Liu, W., Wang, X.: Vision mamba: Efficient visual representation learning with bidirectional state space model. In: International Conference on Machine Learning (2024)