

UniFormer: Unifying Convolution and Self-attention for Visual Recognition

Kunchang Li, Yali Wang, Junhao Zhang, Peng Gao, Guanglu Song,
Yu Liu, Hongsheng Li and Yu Qiao

Abstract—It is a challenging task to learn discriminative representation from images and videos, due to large local redundancy and complex global dependency in these visual data. Convolution neural networks (CNNs) and vision transformers (ViTs) have been two dominant frameworks in the past few years. Though CNNs can efficiently decrease local redundancy by convolution within a small neighborhood, the limited receptive field makes it hard to capture global dependency. Alternatively, ViTs can effectively capture long-range dependency via self-attention, while blind similarity comparisons among all the tokens lead to high redundancy. To resolve these problems, we propose a novel Unified transFormer (UniFormer), which can seamlessly integrate the merits of convolution and self-attention in a concise transformer format. Different from the typical transformer blocks, the relation aggregators in our UniFormer block are equipped with local and global token affinity respectively in shallow and deep layers, allowing tackling both redundancy and dependency for efficient and effective representation learning. Finally, we flexibly stack our blocks into a new powerful backbone, and adopt it for various vision tasks from image to video domain, from classification to dense prediction. Without any extra training data, our UniFormer achieves **86.3** top-1 accuracy on ImageNet-1K classification task. With only ImageNet-1K pre-training, it can simply achieve state-of-the-art performance in a broad range of downstream tasks. It obtains **82.9/84.8** top-1 accuracy on Kinetics-400/600, **60.9/71.2** top-1 accuracy on Something-Something V1/V2 video classification tasks, **53.8** box AP and **46.4** mask AP on COCO object detection task, **50.8** mIoU on ADE20K semantic segmentation task, and **77.4** AP on COCO pose estimation task. Moreover, we build an efficient UniFormer with a concise hourglass design of token shrinking and recovering, which achieves **2-4x** higher throughput than the recent lightweight models. Code is available at <https://github.com/Sense-X/UniFormer>.

Index Terms—UniFormer, Convolution Neural Network, Transformer, Self-Attention, Visual Recognition.

1 INTRODUCTION

REPRESENTATION learning is a fundamental research topic for visual recognition [24], [61]. Basically, we confront two distinct challenges that exist in visual data such as images and videos. On one hand, the local redundancy is large, e.g., visual content in a local region (space, time or space-time) tends to be similar. Such locality often introduces inefficient computation. On the other hand, the global dependency is complex, e.g., targets in different regions have dynamic relations. Such long-range interaction often causes ineffective learning.

To tackle such difficulties, researchers have proposed a number of powerful models in visual recognition [28], [107], [116], [117]. In particular, the mainstream backbones are Convolution

• This work was supported in part by the National Key R&D Program of China 2022ZD0160505, the Joint Lab of CAS-HK, the National Natural Science Foundation of China under Grant 62272450, the Shenzhen Research Program RCJC20200714114557087, and in part by the Youth Innovation Promotion Association of Chinese Academy of Sciences 2020355. (Kunchang Li and Yali Wang are equally-contributed authors. Yu Qiao is the corresponding author.)

• Kunchang Li is with Shenzhen Key Lab of Computer Vision and Pattern Recognition, Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China. University of Chinese Academy of Sciences, Beijing 100049, China. (e-mail: kc.li@siat.ac.cn)

• Yali Wang and Yu Qiao are with Shenzhen Key Lab of Computer Vision and Pattern Recognition, Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China. Shanghai Artificial Intelligence Laboratory, Shanghai 200232, China. (e-mail: yl.wang@siat.ac.cn, qiaoyu@pjlab.org.cn)

• Junhao Zhang is with National University of Singapore, Peng Gao is with Shanghai Artificial Intelligence Laboratory, Guanglu Song and Yu Liu are with SenseTime Research and Hongsheng Li is with the Chinese University of Hong Kong.

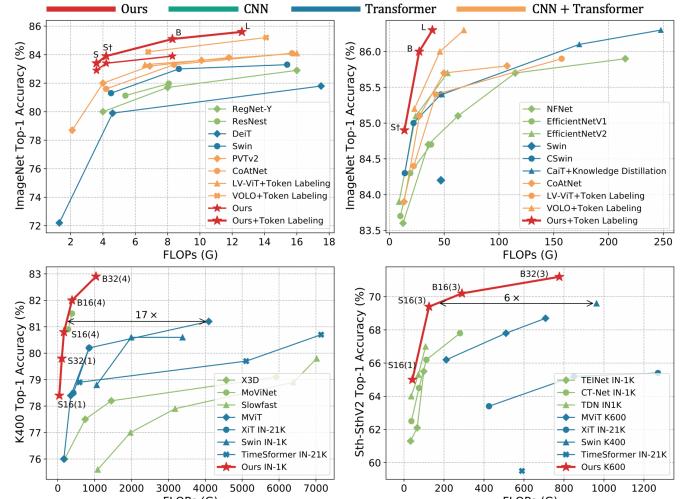


Fig. 1. Accuracy vs. GFLOPs per input. Top: Image Classification (Left/Right: ImageNet with resolution of $224 \times 224/384 \times 384$). Bottom: Video Classification (Left/Right: K400/Sth-SthV2). '(4)' and '(3)' mean we test UniFormer with 4 clips and 3 crops respectively (more testing details can be found in Section 6.7). Our UniFormer achieves the best balance between accuracy and computation on all the datasets.

Neural Networks (CNNs) [30], [37], [40] and Vision Transformers (ViTs) [24], [88], where convolution and self-attention are the key operations in these two structures. Unfortunately, each of these operations mainly addresses one aforementioned challenge while ignoring the other. For example, the convolution operation is good at reducing local redundancy and avoiding unnecessary computation, by aggregating each pixel with context from a small

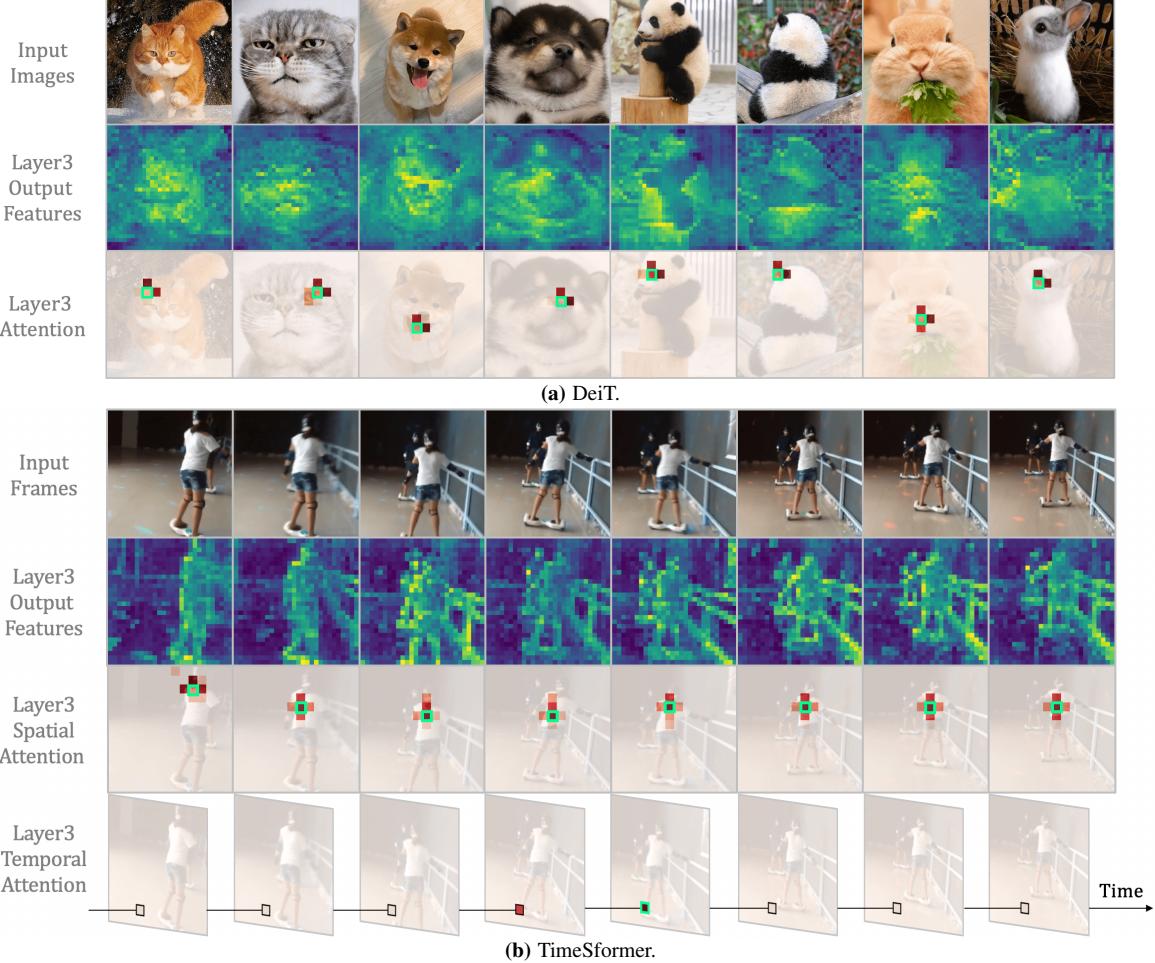


Fig. 2. Visualization of vision transformers. We take the well-known Vision Transformers (ViTs) in both image and video domains (i.e., DeiT [87] and TimeSformer [3]) for illustration, where we respectively show the feature maps, spatial and temporal attention maps from the 3rd layer of these ViTs. We find that, such ViTs learn local representations with redundant global attention. For an anchor token (green box), spatial/temporal attention compares it with all the contextual tokens for aggregation, while only its neighboring tokens (boxes filled with red color) actually work. Hence, ViTs spend large computation on encoding very local visual representations with global self-attention.

neighborhood (e.g., 3×3 or $3 \times 3 \times 3$). However, the limited receptive field makes convolution suffer from difficulty in learning global dependency [53], [101]. Alternatively, self-attention has been recently highlighted in the ViTs. By similarity comparison among visual tokens, it exhibits the strong capacity of learning global dependency in both images [24], [61] and videos [1], [3], [62]. Nevertheless, we observe that ViTs are often inefficient to encode local features in the shallow layers.

We take the well-known ViTs in the image and video domains (i.e., DeiT [87] and TimeSformer [3]) as examples, and visualize their attention maps in the shallow layer. As shown in Figure 2, both ViTs indeed capture detailed visual features in the shallow layer, while spatial and temporal attention are redundant. One can easily see that, given an anchor token, spatial attention largely concentrates on the tokens in the local region (mostly 3×3), and learns little from the rest tokens in this image. Similarly, temporal attention mainly aggregates the tokens in the adjacent frames, while losing sight of the rest tokens in the distant frames. However, such local focus is obtained by global comparison among all the tokens in space and time. Clearly, this redundant attention manner brings large and unnecessary computation burden, thus deteriorating the computation-accuracy balance in ViTs (Figure 1).

Based on these discussions, we propose a novel Unified Transformer (UniFormer) in this work. It flexibly unifies convolution and

self-attention in a concise transformer format, which can tackle both local redundancy and global dependency for effective and efficient visual recognition. Specifically, our UniFormer block consists of three key modules, i.e., Dynamic Position Embedding (DPE), Multi-Head Relation Aggregator (MHRA), and Feed-Forward Network (FFN). The distinct design of the relation aggregator is the key difference between our UniFormer and the previous CNNs and ViTs. In the shallow layers, our relation aggregator captures local token affinity with a small learnable parameter matrix, which inherits the convolution style that can largely reduce computation redundancy by context aggregation in the local region. In the deep layers, our relation aggregator learns global token affinity with token similarity comparison, which inherits the self-attention style that can adaptively build long-range dependency from distant regions or frames. Via progressively stacking local and global UniFormer blocks in a hierarchical manner, we can flexibly integrate their cooperative power to promote representation learning. Finally, we provide a generic and powerful backbone for visual recognition and successfully address various downstream vision tasks with simple and elaborate adaptations. Additionally, we further introduce the lightweight design for UniFormer, which can achieve a preferable accuracy-throughout balance, by a concise hourglass style of token shrinking and recovering.

Extensive experiments demonstrate the strong performance

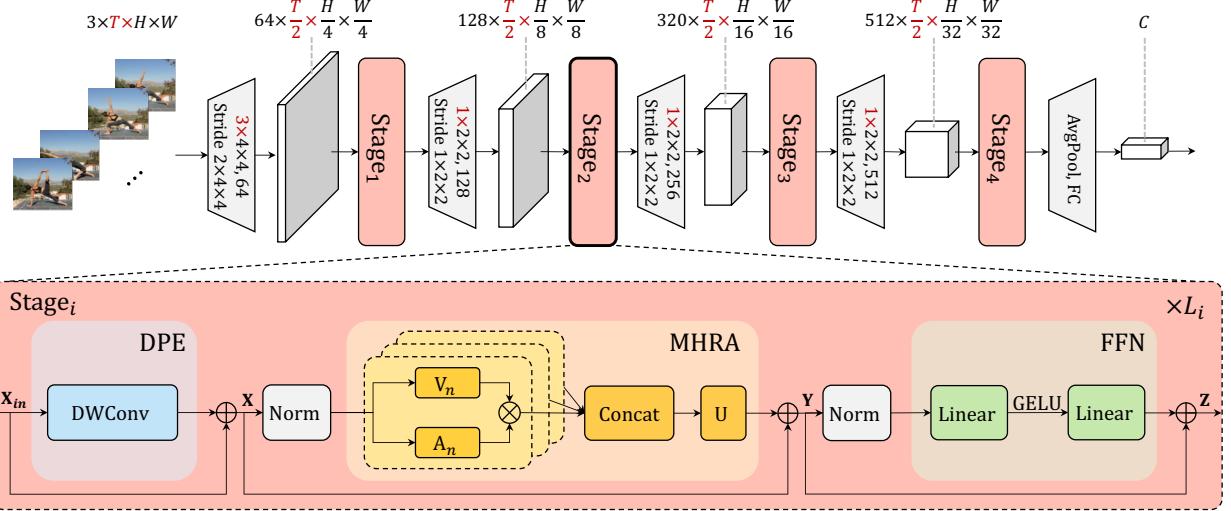


Fig. 3. Unified transFormer (UniFormer). A UniFormer block consists of three key modules, i.e., Dynamic Position Embedding (DPE), Multi-Head Relation Aggregator (MHRA), and Feed Forward Network (FFN). The dimensions highlighted in red only exist for the video input, while all of them are equal to one for image input. More detailed explanations can be found in Section 3.

of our UniFormer on a broad range of vision tasks, including image classification, video classification, object detection, instance segmentation, semantic segmentation and pose estimation. Without any extra training data, UniFormer-L achieves **86.3** top-1 accuracy on ImageNet-1K. Moreover, with only ImageNet-1K pre-training, UniFormer-B achieves **82.9/84.8** top-1 accuracy on Kinetics-400/Kinetics-600, **60.9** and **71.2** top-1 accuracy on Something-Something V1&V2, **53.8** box AP and **46.4** mask AP on the COCO detection task, **50.8** mIoU on the ADE20K semantic segmentation task, and **77.4** AP on the COCO pose estimation task. Finally, our efficient UniFormer with a concise hourglass design can achieve **2-4**× higher throughput than the recent lightweight models.

2 RELATED WORK

2.1 Convolution Neural Networks (CNNs)

In the past few years, the development of computer vision has been mainly driven by convolutional neural networks (CNNs). Beginning with the classical AlexNet [49], many powerful CNN networks have been proposed [37], [40], [41], [80], [84], [85], [107], [120] and achieved remarkable performance in various tasks of image understanding [6], [22], [36], [47], [60], [95], [104], [121]. Recently, due to the fact that video has gradually become one main data resource in many realistic applications, researchers have attempted to apply CNNs in the video domain. Naturally, one can adapt 2D convolution as 3D one, by temporal dimension extension [89]. However, 3D CNNs often suffer from difficult optimization problem and large computation cost. To resolve these issues, the prior works try to inflate the pre-trained 2D convolution kernels for better optimization [11] and factorize 3D convolution kernels in different dimensions to reduce complexity [29], [30], [52], [72], [90], [91]. Besides, many recent studies of video understanding [44], [53], [56], [59] focus on adapting vanilla 2D CNNs with elaborated temporal modeling modules, such as temporal shift [59], [67], motion enhancement [44], [56], [63], and spatiotemporal excitation [52], [53], etc. Unfortunately, due to the limited reception field, the traditional convolution struggles to capture long-range dependency even if they are stacked deeper.

2.2 Vision Transformers (ViTs)

To capture long-term dependencies, Vision Transformer (ViT) has been proposed [24]. With the inspiration of Transformer

architectures in NLP [92], ViT treats image as a number of visual tokens and leverages attention to encode token relations for representation learning. However, vanilla ViT depends on sufficient training data and careful data augmentation. To tackle these problems, several approaches have been developed by improved patch embedding [55], data-efficient training [87], efficient self-attention [23], [61], [109], and multi-scale architectures [28], [99], [100]. These works successfully boost the performance of ViT on various image tasks [8], [12], [15], [38], [46], [55], [57], [106], [110], [114], [123]. Recently, researchers have attempted to extend image ViTs for video modeling. The classical work is TimeSformer [3] by spatial-temporal attention. Starting from this, many works propose different variants for spatiotemporal representation learning [1], [3], [5], [28], [62], [71], and subsequently they are adapted to various video understanding tasks [7], [25], [26], [31], [198]. Although these works demonstrate the outstanding ability of ViTs to learn long-term token relations, the self-attention mechanism requires costly token-to-token comparisons [24]. Hence, it is often inefficient to encode low-level features, as shown in Figure 2. Though Video Swin [62] advocates an inductive bias of locality with shift window, window-based self-attention is still less efficient than local convolution [37] when encoding low-level features. Moreover, the shifted window should be carefully configured.

2.3 Combination of CNNs and ViTs

To bridge the gap between CNNs and ViTs, researchers have tried to take advantage of them to build stronger vision backbones for image understanding, by adding convolutional patch stem for fast convergence [105], [111], introducing convolutional position embedding [17], [23], inserting depthwise convolution into feed-forward network [111], [114], utilizing convolutional projection in self-attention [102], and combining MBConv [77] with Transformer [21]. As for video understanding, the combination is also straightforward, i.e., one can insert self-attention as global attention [101], and/or use convolution as patch stem [64]. However, all these approaches ignore inherent relations between convolution and self-attention, leading to inferior local and/or global token relation learning. Several recent works have demonstrated that self-attention operates similarly to convolution [20], [75]. But they suggest replacing convolution instead of combining them together. Differently, our UniFormer unifies convolution and self-attention in

the transformer style, which can effectively learn local and global token relation, and achieve better accuracy-computation trade-offs on all the vision tasks from image to the video domain.

2.4 Lightweight CNNs and ViTs

In many practical applications, the running platforms usually lack enough computability. Hence, a series of lightweight CNNs are proposed to satisfy such on-device requirements. For example, the classical MobileNets [39], [40], [77] adopt depthwise separable convolution in well-organized efficient ResNet. ShuffleNets [68], [120] leverage channel shuffle for computation reduction. EfficientNets [85], [86] further scale model with neural architecture search at depth, width, and resolution. However, such lightweight design has not been fully investigated in ViTs. Two recent works are proposed by designing transformers as convolutions (i.e., MobiViT [69]), and introducing a parallel architecture of MobileNet and ViT (i.e., MobileFormer [14]). But both works ignore the inference speed (e.g., throughout). Hence, the prior efficient CNNs are still the better choice. To bridge this gap, we build a lightweight UniFormer by token shrinking and recovering in Section 5.

3 METHOD

In this section, we introduce the proposed UniFormer in detail. First, we describe the overview of our UniFormer block. Then, we explain its key modules such as multi-head relation aggregator and dynamic position embedding. Moreover, we discuss the distinct relations between our UniFormer and existing CNNs/ViTs, showing its preferable design for accuracy-computation balance.

3.1 Overview

Figure 3 shows our Unified transFormer (UniFormer). For simple description, we take a video with T frames as an example and an image input can be seen as a video with a single frame. Hence, the dimensions highlighted in red only exit for the video input, while all of them are equal to one for image input. Our UniFormer is a basic transformer format, while we elaborately design it to tackle computational redundancy and capture complex dependency.

Specifically, our UniFormer block consists of three key modules: Dynamic Position Embedding (DPE), Multi-Head Relation Aggregator (MHRA) and Feed-Forward Network (FFN):

$$\mathbf{X} = \text{DPE}(\mathbf{X}_{in}) + \mathbf{X}_{in}, \quad (1)$$

$$\mathbf{Y} = \text{MHRA}(\text{Norm}(\mathbf{X})) + \mathbf{X}, \quad (2)$$

$$\mathbf{Z} = \text{FFN}(\text{Norm}(\mathbf{Y})) + \mathbf{Y}. \quad (3)$$

Considering the input token tensor $\mathbf{X}_{in} \in \mathbb{R}^{C \times T \times H \times W}$ ($T=1$ for an image input), we first introduce DPE to dynamically integrate position information into all the tokens (Eq. 1). It is friendly to arbitrary input resolution and makes good use of token order for better visual recognition. Then, we use MHRA to enhance each token by exploiting its contextual tokens with relation learning (Eq. 2). Via flexibly designing the token affinity in the shallow and deep layers, our MHRA can smartly unify convolution and self-attention to reduce local redundancy and learn global dependency. Finally, we add FFN like traditional ViTs [24], which consists of two linear layers and one non-linear function, i.e., GELU (Eq. 3). The channel number is first expanded by the ratio of 4 and then recovered, thus each token will be enhanced individually.

3.2 Multi-Head Relation Aggregator

As analyzed before, the traditional CNNs and ViTs focus on addressing either local redundancy or global dependency, leading

to unsatisfactory accuracy or/and unnecessary computation. To overcome these difficulties, we introduce a generic Relation Aggregator (RA), which elegantly unifies convolution and self-attention for token relation learning. It can achieve efficient and effective representation learning by designing local and global token affinity in the shallow and deep layers respectively. Specifically, MHRA exploits token relationships in a multi-head style:

$$R_n(\mathbf{X}) = A_n V_n(\mathbf{X}), \quad (4)$$

$$\text{MHRA}(\mathbf{X}) = \text{Concat}(R_1(\mathbf{X}); R_2(\mathbf{X}); \dots; R_N(\mathbf{X})) \mathbf{U}. \quad (5)$$

Given the input tensor $\mathbf{X} \in \mathbb{R}^{C \times T \times H \times W}$, we first reshape it to a sequence of tokens $\mathbf{X} \in \mathbb{R}^{L \times C}$ with the length of $L=T \times H \times W$. Moreover, $R_n(\cdot)$ refers to RA in the n -th head and $\mathbf{U} \in \mathbb{R}^{C \times C}$ is a learnable parameter matrix to integrate N heads. Each RA consists of token context encoding and token affinity learning. We apply a linear transformation to encode the original tokens into contextual tokens $V_n(\mathbf{X}) \in \mathbb{R}^{L \times \frac{C}{N}}$. Subsequently, RA can summarize context with the guidance of token affinity $A_n \in \mathbb{R}^{L \times L}$. We will describe how to learn the specific A_n in the following.

3.2.1 Local MHRA

As shown in Figure 2, though the previous ViTs compare similarities among all the tokens, they finally learn local representations. Such redundant self-attention design brings large computation cost in the shallow layers. Based on it, we suggest learning token affinity in a small neighborhood, which coincidentally shares a similar insight with the design of a convolution filter. Hence, we propose to represent local affinity as a learnable parameter matrix in the shallow layers. Concretely, given an anchor token \mathbf{X}_i , our local RA learns the affinity between this token and other tokens in the small neighborhood $\Omega_i^{t \times h \times w}$ ($t=1$ for an image input):

$$A_n^{\text{local}}(\mathbf{X}_i, \mathbf{X}_j) = a_n^{i-j}, \quad \text{where } j \in \Omega_i^{t \times h \times w}, \quad (6)$$

where $a_n \in \mathbb{R}^{t \times h \times w}$ is the learnable parameter, and \mathbf{X}_j refers to any neighbor token in $\Omega_i^{t \times h \times w}$. $(i-j)$ denotes the relative position between token i and j . Note that, visual content between adjacent tokens varies subtly in the shallow layers, since the receptive field of tokens is small. In this case, it is not necessary to make token affinity dynamic in these layers. Hence, we use a learnable parameter matrix to describe local token affinity, which simply depends on the relative position between tokens.

Comparison to Convolution Block. Interestingly, we find that our local MHRA can be interpreted as a generic extension of MobileNet block [29], [77], [90]. Firstly, the linear transformation $V(\cdot)$ in Eq. 4 is equivalent to a pointwise convolution (PWConv), where each head is corresponding to an output feature channel $V_n(\mathbf{X})$. Furthermore, our local token affinity A_n^{local} can be instantiated as the parameter matrix that operated on each output channel (or head) $V_n(\mathbf{X})$, thus the relation aggregator $R_n(\mathbf{X}) = A_n^{\text{local}} V_n(\mathbf{X})$ can be explained as a depthwise convolution (DWConv). Finally, the linear matrix \mathbf{U} , which concatenates and fuses all heads, can also be seen as a pointwise convolution. As a result, such local MHRA can be reformulated with a manner of PWConv-DWConv-PWConv in the MobileNet block. In our experiments, we instantiate our local MHRA as such channel-separated convolution, so that our UniFormer can boost computation efficiency for visual recognition. Moreover, different from the MobileNet block, our local UniFormer block is designed as a generic transformer format, i.e., it also contains dynamical position encoding (DPE) and feed-forward network (FFN), besides MHRA. This unique integration

Model	Type	#Blocks	#Channels	#Param.	FLOPs
Small	[L, L, G, G]	[3, 4, 8, 3]	[64, 128, 320, 512]	21.5M	3.6G
Base	[L, L, G, G]	[5, 8, 20, 7]	[64, 128, 320, 512]	50.3M	8.3G
Large	[L, L, G, G]	[5, 10, 24, 7]	[128, 192, 448, 640]	100M	12.6G

TABLE 1. Backbones for image classification. ‘L’ and ‘G’ refer to our local and global UniFormer blocks respectively. The FLOPs are measured at resolution 224×224.

can effectively enhance token representation, which has not been explored in the previous convolution blocks.

3.2.2 Global MHRA

In the deep layers, it is important to exploit long-range relation in the broader token space, which naturally shares a similar insight with the design of self-attention. Therefore, we design the token affinity via comparing content similarity among all the tokens:

$$A_n^{global}(\mathbf{X}_i, \mathbf{X}_j) = \frac{e^{Q_n(\mathbf{X}_i)^T K_n(\mathbf{X}_j)}}{\sum_{j' \in \Omega_{T \times H \times W}} e^{Q_n(\mathbf{X}_i)^T K_n(\mathbf{X}_{j'})}}, \quad (7)$$

where \mathbf{X}_j can be any token in the global tube with a size of $T \times H \times W$ ($T=1$ for an image input), while $Q_n(\cdot)$ and $K_n(\cdot)$ are two different linear transformations.

Comparison to Transformer Block. Our global MHRA A_n^{global} (Eq. 7) can be instantiated as a spatiotemporal self attention, where $Q_n(\cdot)$, $K_n(\cdot)$ and $V_n(\cdot)$ become Query, Key and Value in ViT [24]. Hence, it can effectively learn long-range dependency. However, our global UniFormer block is different from the previous ViT blocks. First, most video transformers divide spatial and temporal attention in the video domain [1], [3], [79], in order to reduce the dot-product computation in token similarity comparison. But such an operation inevitably deteriorates the spatiotemporal relation among tokens. In contrast, our global UniFormer block jointly encodes spatiotemporal token relation to generate more discriminative video representation for recognition. Since our local UniFormer block largely saves computation of token comparison in the shallow layers, the overall model can achieve a preferable computation-accuracy balance. Second, instead of absolute position embedding [24], [92], we adopt dynamic position embedding (DPE) in our UniFormer. It is in convolution style (see the next section), which can overcome permutation-invariance and be friendly to different input lengths of visual tokens.

3.3 Dynamic Position Embedding

The position information is an important clue to describe visual representation. Previously, most ViTs encode such information by absolute or relative position embedding [24], [61], [99]. However, absolute position embedding has to be interpolated for various input sizes with fine-tuning [87], [88], while relative position embedding does not work well due to the modification of self-attention [17]. To improve flexibility, convolutional position embedding has been recently proposed [16], [23]. In particular, conditional position encoding (CPE) [17] can implicitly encode position information via convolution operators, which unlocks Transformer to process arbitrary input size and promotes recognition performance. Due to its plug-and-play property, we flexibly adopt it as our Dynamical Position Embedding (DPE) in the UniFormer:

$$DPE(\mathbf{X}_{in}) = DWConv(\mathbf{X}_{in}), \quad (8)$$

where DWConv refers to depthwise convolution with zero paddings. We choose such a design as our DPE based on the following reasons. First, depthwise convolution is friendly

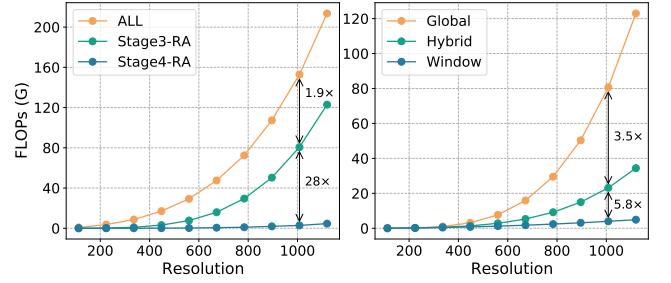


Fig. 4. FLOPs vs. Resolution. **Left:** Total FLOPs and the FLOPs of MatMul in RA in Stage3 and Stage4. RA in Stage3 requires more computation for large resolution. **Right:** The FLOPs of MatMul in different RA in Stage3. Window and hybrid blocks save computation.

to arbitrary input shapes, e.g., it is straightforward to use its spatiotemporal version to encode 3D position information in videos. Second, depthwise convolution is light-weight, which is an important factor for computation-accuracy balance. Finally, we add extra zero paddings, since it can help tokens be aware of their absolute positions by querying their neighbors progressively [17].

4 FRAMEWORK

In the section, we mainly develop visual frameworks for various downstream tasks. Specifically, we first develop a number of visual backbones for image classification, by hierarchically stacking our local and global UniFormer blocks with consideration of computation-accuracy balance. Then, we extend the above backbones to tackle other representative vision tasks, including video classification and dense prediction (i.e., object detection, semantic segmentation and human pose estimation). Such generality and flexibility of our UniFormer demonstrate its valuable potential for computer vision research and beyond.

4.1 Image Classification

It is important to progressively learn visual representation for capturing semantics in the image. Hence, we build up our backbone with four stages, as illustrated in Figure 3.

More specifically, we use the local UniFormer blocks in the first two stages to reduce computation redundancy, while the global UniFormer blocks are utilized in the last two stages to learn long-range token dependency. For the local UniFormer block, MHRA is instantiated as PWConv-DWConv-PWConv with local token affinity (Eq. 6), where the spatial size of DWConv is set to 5×5 for image classification. For the global UniFormer block, MHRA is instantiated as multi-head self-attention with global token affinity (Eq. 7), where the number of attention heads is set to 64. For both local and global UniFormer blocks, DPE is instantiated as DWConv with a spatial size of 3×3 , and the expand ratio of FFN is 4.

Additionally, as suggested in the CNN and ViT literatures [24], [37], we utilize BN [43] for convolution and LN [2] for self-attention. For feature downsampling, we use the 4×4 convolution with stride 4×4 before the first stage and the 2×2 convolution with stride 2×2 before other stages. Besides, an extra LN is added after each downsampling convolution. Finally, the global average pooling and fully connected layer are applied to output the predictions. When training models with Token Labeling [45], we add another fully connected layer for auxiliary loss. For various computation requirements, we design three model variants as shown in Table 1.

4.2 Video Classification

Given our image-based 2D backbones, one can easily adapt them as 3D backbones for video classification. Without loss of generality,

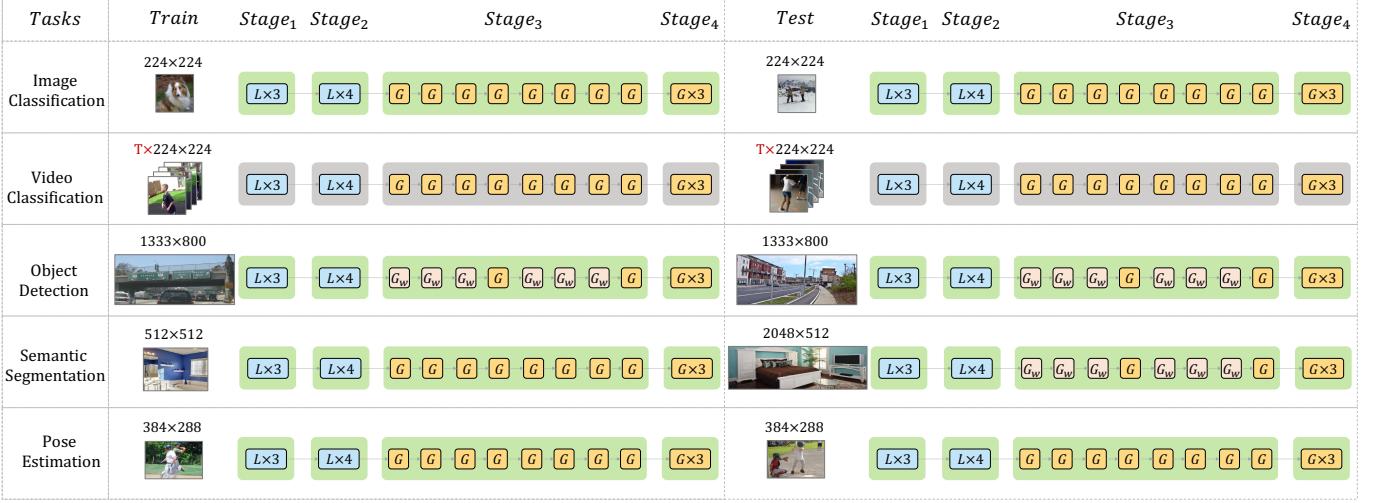


Fig. 5. Training and testing adaption for downstream tasks. For video classification, we inflate all the 2D convolution filters to 3D ones. For dense prediction, we modify RA in Stage3 for different downstream tasks. ‘ G_w ’ means we apply global MHRA in a predefined window.

we adjust Small and Base models for spatiotemporal modeling. Specifically, the model architectures keep the same with four stages, where we use the local UniFormer blocks in the first two stages and the global UniFormer blocks in the last two stages. But differently, all the 2D convolution filters are changed as 3D ones via filter inflation [11]. Concretely, the kernel size of DWConv in DPE and local MHRA are $3 \times 3 \times 3$ and $5 \times 5 \times 5$ respectively. Moreover, we downsample both spatial and temporal dimensions before the first stage. Hence, the convolution filter before this stage becomes $3 \times 4 \times 4$ with the stride of $2 \times 4 \times 4$. For the other stages, we just downsample the spatial dimension to decrease the computation cost and maintain high performance. Hence, the convolution filters before these stages are $1 \times 2 \times 2$ with stride of $1 \times 2 \times 2$.

Note that we use spatiotemporal attention in the global UniFormer blocks for learning token relation jointly in the 3D view. It is worth mentioning that, due to the large model sizes, the previous video transformers [1], [3] divide spatial and temporal attention to reduce computation and alleviate overfitting, but such factorization operation inevitably tears spatiotemporal token relations. In contrast, our joint spatiotemporal attention can avoid the issue. Besides, our local UniFormer blocks largely save computation via 3D DWconv. Hence, our model can achieve effective and efficient video representation learning.

4.3 Dense Prediction

Dense prediction tasks are necessary to verify the generality of our recognition backbones. Hence, we adopt our UniFormer backbones for a number of popular dense tasks such as object detection, instance segmentation, semantic segmentation, and human pose estimation. However, direct usage of our backbone is not suitable because of the high input resolution of most dense prediction tasks, e.g., the size of the image is 1333×800 in the COCO object detection dataset. Naturally, feeding such images into our classification backbones would inevitably lead to large computation, especially when operating self-attention of global UniFormer block in the last two stages. Taking $h \times w$ visual tokens as an example, the MatMul operation in token similarity comparison (Eq. 7) causes $O(w^2 h^2)$ complexity, which is prohibitive for most dense tasks.

We propose to adjust the global UniFormer block for different downstream tasks. First, we analyze the FLOPs of our UniFormer-S under different input resolutions. Figure 4 clearly indicates that Relation Aggregator (RA) in Stage3 occupies large computation.

For example, for a 1008×1008 image, the MatMul operation of RA in Stage3 even occupies over 50% of the total FLOPs, while the FLOPs in Stage4 is only 1/28 of that in Stage3. Thus we focus on modifying RA in Stage3 for computation reduction.

Inspired by [61], [75], we propose to apply our global MHRA in a predefined window (e.g., 14×14), instead of using it in the entire image with high resolution. Such operation can effectively cut down computation with the complexity of $\mathcal{O}(whp^2)$, where p is the window size. However, it undoubtedly drops model performance, due to insufficient token interaction. To bridge this gap, we integrate window and global UniFormer blocks together in Stage3, where a hybrid group consists of three window blocks and one global block. In this case, there are 2/5 hybrid groups in Stage3 of our UniFormer-Small/Base backbones.

Based on this design, we next introduce the specific backbone settings of various dense tasks, depending on the input resolution of training and testing images. For object detection and instance segmentation, the input images are usually large (e.g., 1333×800), thus we adopt the hybrid block style in Stage3. In contrast, the inputs are relatively small for pose estimation, such as 384×288 , hence global blocks are still applied in Stage3 for both training and testing. Specially, for semantic segmentation, the testing images are often larger than the training ones. Therefore, we utilize the global blocks in Stage3 for training, while adapting the hybrid blocks in Stage3 for testing. We use the following simple design. The window-based block in testing has the same receptive field as the global block in training, e.g., 32×32 . Such design can maintain training efficiency, and boost testing performance by keeping consistency with training as much as possible.

5 TOWARDS LIGHTWEIGHT UNIFORMER

Recently, researchers have tried to combine CNNs with ViTs to design lightweight models. For example, MobileFormer [14] proposes a parallel design of MobileNet [77] and ViT [24], and MobileViT [69] designs transformers as convolutions. However, the inference speeds of these works should be further improved. Hence, the prior efficient CNNs are still the better choice, such as EfficientNet [85] for image tasks and MoViNet [48] for video tasks. To bridge the gap, we propose a lightweight UniFormer architecture in Figure 6, by designing a distinct hourglass UniFormer block. In this block, we adaptively leverage token shrinking and recovering, to achieve a preferable accuracy-throughput balance.

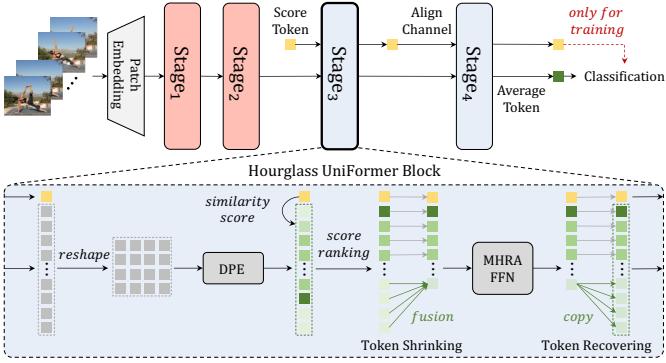


Fig. 6. Lightweight UniFormer. We propose an Hourglass UniFormer block, saving computation of MHRA and FFN by adaptive token shrinking and recovering.

5.1 Hourglass UniFormer Block

Since a large computation load lies in token similarity comparison in the global UniFormer block, we propose an Hourglass UniFormer (H-UniFormer) block to reduce the number of visual tokens involved in global MHRA. Note that, the existing token pruning methods [58], [76], [124] are infeasible for our UniFormer and those ViTs with convolution [21], [100], [111]. The main reason is that, after pruning, the rest tokens often maintain a broken spatiotemporal structure, which makes convolution inapplicable. To overcome such difficulty, we propose a concise integration of token shrinking and recovering in our H-UniFormer block.

Token Shrinking. We introduce a score token $\mathbf{s} \in \mathbb{R}^C$ to measure the importance of the visual tokens $\mathbf{X} \in \mathbb{R}^{C \times THW} = \{\mathbf{X}_1, \dots, \mathbf{X}_{THW}\}$ after DPE. Specifically, we compare similarity (using Eq. 7) between the score token \mathbf{s} and a visual token \mathbf{X}_j , and average the similarity values over N attention heads,

$$\mathbf{A}_j = \sum_{n=1}^N \mathbf{A}_n^{global}(\mathbf{s}, \mathbf{X}_j)/N, \quad (9)$$

where \mathbf{A}_j represents the importance of visual token \mathbf{X}_j , and we use $\mathbf{A} \in \mathbb{R}^{THW}$ to represent the importance vector of all the visual tokens. For tokens with high values in \mathbf{A} , we consider them to be crucial tokens and keep them. For tokens with low values, we consider them to be unimportant tokens. Hence, we fuse them as one representative token, where their score values are used as the fusing weights. By such unimportant token reduction, we can shrink visual tokens from $\mathbf{X} \in \mathbb{R}^{C \times THW}$ to $\mathbf{X}^S \in \mathbb{R}^{C \times M}$, where the number of tokens $M \ll THW$. Subsequently, the reduced visual tokens are fed into global MHRA and FFN, for computation saving. Additionally, we pass this score token through all the H-UniFormer blocks in Stage3 and Stage4, and use it to calculate the classification loss when training. In this case, the score token is effectively guided by the ground-truth label, and it thus becomes discriminative to weight token importance.

Token Recovering. After learning token interactions in global MHRA and FFN, we replicate the representative token to recover the unimportant tokens. Thus, we can maintain the spatiotemporal structure of all the visual tokens, for effective dynamic position encoding (i.e., convolution) in the next H-UniFormer block.

5.2 LightWeight UniFormer Architecture

To build our light-weight UniFormer, We follow most of the architectures in Section 4.1, except that we change to use the H-UniFormer block in Stage3 and Stage4, and adopt smaller depth, width or resolution (e.g., 128). Specifically, for UniFomrer-XS, the

Arch.	Method	#Param (M)	FLOPs (G)	Train Size	Test Size	ImageNet Top-1
CNN	RegNetY-4G [74]	21	4.0	224	224	80.0
	EffcientNet-B5 [85]	30	9.9	456	456	83.6
	EfficientNetV2-S [86]	22	8.5	384	384	83.9
Trans	DeiT-S [87]	22	4.6	224	224	79.9
	PVT-S [99]	25	3.8	224	224	79.8
	T2T-14 [112]	22	5.2	224	224	80.7
	Swin-T [61]	29	4.5	224	224	81.3
	Focal-T [109]	29	4.9	224	224	82.2
	CSwin-T [23]	23	4.3	224	224	82.7
	CSwin-T ↑384 [23]	23	14.0	224	384	84.3
CNN+Trans	CvT-13 [102]	20	4.5	224	224	81.6
	CvT-13 ↑384 [102]	20	16.3	224	384	83.0
	CoAtNet-0 [21]	25	4.2	224	224	81.6
	CoAtNet-0 ↑384 [21]	20	13.4	224	384	83.9
	Container [32]	22	8.1	224	224	82.7
	LV-ViT-S [45]	26	6.6	224	224	83.3
	LV-ViT-S ↑384 [45]	26	22.2	224	384	84.4
	UniFormer-S	22	3.6	224	224	82.9
	UniFormer-S*	22	3.6	224	224	83.4
	UniFormer-S* ↑384	22	11.9	224	384	84.6
CNN	UniFormer-S†	24	4.2	224	224	83.4
	UniFormer-S†*	24	4.2	224	224	83.9
	UniFormer-S†* ↑384	24	13.7	224	384	84.9
	RegNetY-8G [74]	39	8.0	224	224	81.7
	EffcientNet-B7 [85]	66	39.2	600	600	84.3
	EfficientNetV2-M [86]	54	25.0	480	480	85.1
	PVT-L [99]	61	9.8	224	224	81.7
Trans	T2T-24 [112]	64	13.2	224	224	82.2
	Swin-S [61]	50	8.7	224	224	83.0
	Focal-S [109]	51	9.1	224	224	83.5
	CSwin-S [23]	35	6.9	224	224	83.6
	CSwin-S ↑384 [23]	35	22.0	224	384	85.0
	CvT-21 [102]	32	7.1	224	224	82.5
CNN+Trans	CoAtNet-1 [21]	42	8.4	224	224	83.3
	CoAtNet-1 ↑384 [21]	42	27.4	224	384	85.1
	LV-ViT-M [45]	56	16.0	224	224	84.1
	LV-ViT-M ↑384 [45]	56	42.2	224	384	85.4
	UniFormer-B	50	8.3	224	224	83.9
	UniFormer-B*	50	8.3	224	224	85.1
CNN	UniFormer-B* ↑384	50	27.2	224	384	86.0
	RegNetY-16G [74]	84	16.0	224	224	82.9
	EfficientNetV2-L [86]	121	53	480	480	85.7
	NFNet-F4 [4]	316	215.3	384	512	85.9
Trans	DeiT-B [87]	86	17.5	224	224	81.8
	Swin-B [61]	88	15.4	224	224	83.3
	Swin-B ↑384	88	47.0	224	384	84.2
	Focal-B [61]	90	16.0	224	224	83.8
	CSwin-B [23]	78	15.0	224	224	84.2
	CSwin-B ↑384 [23]	78	47.0	224	384	85.4
	CaiT-S36 ↑384Υ [88]	68	48.0	224	384	85.4
	CaiT-M36 ↑448Υ [88]	271	247.8	224	448	86.3
CNN+Trans	BoTNet-T7 [82]	79	19.3	256	256	84.2
	CoAtNet-3 [21]	168	34.7	224	224	84.5
	CoAtNet-3 ↑384 [21]	168	107.4	224	384	85.8
	LV-ViT-L [45]	150	59.0	288	288	85.3
	LV-ViT-L ↑448 [45]	150	157.2	288	448	85.9
	VOLO-D3 [113]	86	20.6	224	224	85.4
	VOLO-D3 ↑448 [113]	86	67.9	224	448	86.3
	UniFormer-L*	100	12.6	224	224	85.6
Trans	UniFormer-L* ↑384	100	39.2	224	384	86.3

TABLE 2. Comparison with the state-of-the-art on ImageNet. * means Token Labeling proposed in LV-ViT [45]. ‘Υ’ means using RegNet-16GF [74] as teacher. For UniFormer-S†, we apply overlapped patch embedding and more blocks for fair comparison.

Method	Pretrain	#frame × #crop × #clip	FLOPs (G)	K400		K600	
				Top-1	Top-5	Top-1	Top-5
SmallBig _{EN} [53]	IN-1K	(8+32) × 3 × 4	5700	78.7	93.7	-	-
TDN _{EN} [97]	IN-1K	(8+16) × 3 × 10	5940	79.4	94.4	-	-
CT-Net _{EN} [52]	IN-1K	(16+16) × 3 × 4	2641	79.8	94.2	-	-
LGD [73]	IN-1K	128 × N/A	N/A	79.4	94.4	81.5	95.6
SlowFast [30]	-	8 × 3 × 10	3180	77.9	93.2	80.4	94.8
SlowFast+NL [30]	-	16 × 3 × 10	7020	79.8	93.9	81.8	95.1
ip-CSN [90]	Sports1M	32 × 3 × 10	3270	79.2	93.8	-	-
CorrNet [94]	Sports1M	32 × 3 × 10	6720	81.0	-	-	-
X3D-M [29]	-	16 × 3 × 10	186	76.0	92.3	78.8	94.5
X3D-XL [29]	-	16 × 3 × 10	1452	79.1	93.9	81.9	95.5
MoViNet-A5 [48]	-	120 × 1 × 1	281	80.9	94.9	82.7	95.7
MoViNet-A6 [48]	-	120 × 1 × 1	386	81.5	95.3	83.5	96.2
ViT-B-VTN [70]	IN-21K	250 × 1 × 1	3992	78.6	93.7	-	-
TimeSformer-HR [3]	IN-21K	16 × 3 × 1	5109	79.7	94.4	82.4	96.0
TimeSformer-L [3]	IN-21K	96 × 3 × 1	7140	80.7	94.7	82.2	95.5
STAM [79]	IN-21K	64 × 1 × 1	1040	79.2	-	-	-
X-ViT [5]	IN-21K	8 × 3 × 1	425	78.5	93.7	82.5	95.4
X-ViT [5]	IN-21K	16 × 3 × 1	850	80.2	94.7	84.5	96.3
Mformer-HR [71]	IN-21K	16 × 3 × 10	28764	81.1	95.2	82.7	96.1
MViT-B,16×4 [28]	-	16 × 1 × 5	353	78.4	93.5	82.1	95.7
MViT-B,32×3 [28]	-	32 × 1 × 5	850	80.2	94.4	83.4	96.3
ViViT-L [1]	IN-21K	16 × 3 × 4	17352	80.6	94.7	82.5	95.6
ViViT-L [1]	JFT-300M	16 × 3 × 4	17352	82.8	95.3	84.3	96.2
ViViT-H [1]	JFT-300M	16 × 3 × 4	99792	84.8	95.8	85.8	96.5
Swin-T [62]	IN-1K	32 × 3 × 4	1056	78.8	93.6	-	-
Swin-B [62]	IN-1K	32 × 3 × 4	3384	80.6	94.6	-	-
Swin-B [62]	IN-21K	32 × 3 × 4	3384	82.7	95.5	84.0	96.5
Swin-L-384↑ [62]	IN-21K	32 × 5 × 10	105350	84.9	96.7	86.1	97.3
UniFormer-S	IN-1K	16 × 1 × 4	167	80.8	94.7	82.8	95.8
UniFormer-B	IN-1K	16 × 1 × 4	389	82.0	95.1	84.0	96.4
UniFormer-B	IN-1K	32 × 1 × 4	1036	82.9	95.4	84.8	96.7
UniFormer-B	IN-1K	32 × 3 × 4	3108	83.0	95.4	84.9	96.7

TABLE 3. Comparison with the state-of-the-art on Kinetics-400&600. Our UniFormer outperforms most of the current methods with much fewer computation cost.

block number, channel number and head dimension are [3, 5, 9, 3], [64, 128, 256, 512] and 32. For UniFomrer-XXS, the block number, channel number and head dimension are [2, 5, 8, 2], [56, 112, 224, 448] and 28.

Beginning from the second layer in Stage3, we utilize the similarity score in the previous layer \mathbf{A}^{pre} to guide the token shrinking. Based on the phenomenon that the locations of the crucial tokens are basically the same among different layers [108], we update the similarity scores via mean, i.e., $\mathbf{A} = (\mathbf{A} + \mathbf{A}^{pre}) / 2$. Thus it can focus on the significant tokens consistently. By default, we keep half of the tokens and fuse the rest (i.e., the shrinking ratio is 0.5) in our light-weight UniFormer.

6 EXPERIMENTS

To verify the effectiveness and efficiency of our UniFormer for visual recognition, we conduct extensive experiments on ImageNet-1K [22] image classification, Kinetics-400 [10]/600 [9] and Something-Something V1&V2 [34] video classification, COCO [60] object detection, instance segmentation and pose estimation, and ADE20K [121] semantic segmentation. We also perform comprehensive ablation studies to analyze each design of our UniFormer.

6.1 Image Classification

Settings. We train our models from scratch on the ImageNet-1K dataset [22]. For a fair comparison, we follow the same training strategy proposed in DeiT [87] by default, including strong data augmentation and regularization. Additionally, we set the stochastic depth rate as 0.1/0.3/0.4 respectively for our UniFormer-S/B/L in Table 1. We train all models via AdamW [65] optimizer with cosine

Method	Pretrain	#frame × #crop × #clip	FLOPs (G)	SSV1		SSV2	
				Top-1	Top-5	Top-1	Top-5
TSN [96]	IN-1K	16 × 1 × 1	66	19.9	47.3	30.0	60.5
TSM [59]	IN-1K	16 × 1 × 1	66	47.2	77.1	-	-
GST [67]	IN-1K	16 × 1 × 1	59	48.6	77.9	62.6	87.9
TEINet [63]	IN-1K	16 × 1 × 1	66	49.9	-	62.1	-
TEA [56]	IN-1K	16 × 1 × 1	70	51.9	80.3	-	-
MSNet [50]	IN-1K	16 × 1 × 1	101	52.1	82.3	64.7	89.4
CT-Net [52]	IN-1K	16 × 1 × 1	75	52.5	80.9	64.5	89.3
CT-Net _{EN} [52]	IN-1K	8+12+16+24	280	56.6	83.9	67.8	91.1
TDN [97]	IN-1K	16 × 1 × 1	72	53.9	82.1	65.3	89.5
TDN _{EN} [97]	IN-1K	8+16	198	56.8	84.1	68.2	91.6
TimeSformer-HR [3]	IN-21K	16 × 3 × 1	5109	-	-	62.5	-
TimeSformer-L [3]	IN-21K	96 × 3 × 1	7140	-	-	62.3	-
X-ViT [5]	IN-21K	16 × 3 × 1	850	-	-	65.2	90.6
X-ViT [5]	IN-21K	32 × 3 × 1	1270	-	-	65.4	90.7
Mformer-HR [71]	K400	16 × 3 × 1	2876	-	-	67.1	90.6
Mformer-L [71]	K400	32 × 3 × 1	3555	-	-	68.1	91.2
ViViT-L [1]	K400	16 × 3 × 4	11892	-	-	65.4	89.8
MViT-B,64×3 [28]	K400	64 × 1 × 3	1365	-	-	67.7	90.9
MViT-B-24,32×3 [28]	K600	32 × 1 × 3	708	-	-	68.7	91.5
Swin-B [62]	K400	32 × 3 × 1	963	-	-	69.6	92.7
UniFormer-S	K400	16 × 1 × 1	42	53.8	81.9	63.5	88.5
UniFormer-S	K600	16 × 1 × 1	42	54.4	81.8	65.0	89.3
UniFormer-S	K400	16 × 3 × 1	125	57.2	84.9	67.7	91.4
UniFormer-S	K600	16 × 3 × 1	125	57.6	84.9	69.4	92.1
UniFormer-B	K400	16 × 3 × 1	290	59.1	86.2	70.4	92.8
UniFormer-B	K600	16 × 3 × 1	290	58.8	86.5	70.2	93.0
UniFormer-B	K400	32 × 3 × 1	777	60.9	87.3	71.2	92.8
UniFormer-B	K600	32 × 3 × 1	777	61.0	87.6	71.2	92.8
UniFormer-B	K400	32 × 3 × 2	1554	61.0	87.3	71.4	92.8
UniFormer-B	K600	32 × 3 × 2	1554	61.2	87.6	71.3	92.8

TABLE 4. Comparison with the state-of-the-art on Something-Something V1&V2. Our UniFormer achieves new state-of-the-art performances on both datasets.

learning rate schedule [66] for 300 epochs, while the first 5 epochs are utilized for linear warm-up [33]. The weight decay, learning rate and batch size are set to 0.05, 1e-3 and 1024 respectively. For UniFormer-S†, we follow state-of-the-art ViTs [23], [109] to apply overlapped patch embedding and blocks (3/5/9/3 blocks in each stage) for fair comparisons. As for UniFormer-B, we use the learning rate of 8e-4 for better convergence.

For training high-performance ViTs, hard distillation [87] and Token Labeling [45] are proposed, both of which are complementary to our backbones. Since Token Labeling is more efficient, we apply it with an extra fully connected layer and auxiliary loss, following the settings in LV-ViT [45]. Different from the training settings in DeiT, MixUp [118] and CutMix [115] are not used since they conflict with MixToken [45]. The base learning rate is 1.6e-3 for the batch size of 1024 by default. Specially, we adopt the base learning rate of 1.2e-3 and layer scale [88] for UniFormer-L to avoid NaN loss. When fine-tuning our models on larger resolution, i.e., 384 × 384, the weight decay, learning rate, batch size, warm-up epoch and total epoch are set to 1e-8, 5e-6, 512, 5 and 30.

Results. In Table 2, we compare our UniFormer with the state-of-the-art CNNs, ViTs and their combinations. It clearly shows that our UniFormer outperforms previous models under different computation restrictions. For example, our UniFormer-S† achieves 83.4% top-1 accuracy with only 4.2G FLOPs, surpassing RegNetY-4G [74], Swin-T [61], CSwin-T [23] and CoAtNet [21] by 3.4%, 2.1%, 0.7% and 1.8% respectively. Though EfficientNet [85] comes from extensive neural architecture search, our UniFormer outperforms it (83.9% vs. 83.6%) with less computation cost (8.3G vs. 9.9G). Furthermore, we enhance our models with Token Labeling [45], which is denoted by ‘*’. Compared with the models

Method	#Params (M)	FLOPs (G)	Mask R-CNN 1× schedule						Mask R-CNN 3× + MS schedule					
			AP ^b	AP ^b ₅₀	AP ^b ₇₅	AP ^m	AP ^m ₅₀	AP ^m ₇₅	AP ^b	AP ^b ₅₀	AP ^b ₇₅	AP ^m	AP ^m ₅₀	AP ^m ₇₅
Res50 [37]	44	260	38.0	58.6	41.4	34.4	55.1	36.7	41.0	61.7	44.9	37.1	58.4	40.1
PVT-S [99]	44	245	40.4	62.9	43.8	37.8	60.1	40.3	43.0	65.3	46.9	39.9	62.5	42.8
TwinsP-S [16]	44	245	42.9	65.8	47.1	40.0	62.7	42.9	46.8	69.3	51.8	42.6	66.3	46.0
Twins-S [16]	44	228	43.4	66.0	47.3	40.3	63.2	43.4	46.8	69.2	51.2	42.6	66.3	45.8
Swin-T [61]	48	264	42.2	64.6	46.2	39.1	61.6	42.0	46.0	68.2	50.2	41.6	65.1	44.8
VIL-S [119]	45	218	44.9	67.1	49.3	41.0	64.2	44.1	47.1	68.7	51.5	42.7	65.9	46.2
Focal-T [109]	49	291	44.8	67.7	49.2	41.0	64.7	44.2	47.2	69.4	51.9	42.7	66.5	45.9
UniFormer-S _{h14}	41	269	45.6	68.1	49.7	41.6	64.8	45.0	48.2	70.4	52.5	43.4	67.1	47.0
Res101 [37]	63	336	40.4	61.1	44.2	36.4	57.7	38.8	42.8	63.2	47.1	38.5	60.1	41.3
X101-32 [107]	63	340	41.9	62.5	45.9	37.5	59.4	40.2	44.0	64.4	48.0	39.2	61.4	41.9
PVT-M [99]	64	302	42.0	64.4	45.6	39.0	61.6	42.1	44.2	66.0	48.2	40.5	63.1	43.5
TwinsP-B [16]	64	302	44.6	66.7	48.9	40.9	63.8	44.2	47.9	70.1	52.5	43.2	67.2	46.3
Twins-B [16]	76	340	45.2	67.6	49.3	41.5	64.5	44.8	48.0	69.5	52.7	43.0	66.8	46.6
Swin-S [61]	69	354	44.8	66.6	48.9	40.9	63.4	44.2	48.5	70.2	53.5	43.3	67.3	46.6
Focal-S [109]	71	401	47.4	69.8	51.9	42.8	66.6	46.1	48.8	70.5	53.6	43.8	67.7	47.2
CSWin-S [23]	54	342	47.9	70.1	52.6	43.2	67.1	46.2	50.0	71.3	54.7	44.5	68.4	47.7
Swin-B [61]	107	496	46.9	-	-	42.3	-	-	48.5	69.8	53.2	43.4	66.8	46.9
Focal-B [109]	110	533	47.8	-	-	43.2	-	-	49.0	70.1	53.6	43.7	67.6	47.0
UniFormer-B _{h14}	69	399	47.4	69.7	52.1	43.1	66.0	46.5	50.3	72.7	55.3	44.8	69.0	48.3

TABLE 5. Object detection and instance segmentation with Mask R-CNN on COCO val2017. The FLOPs are measured at resolution 800×1280. All the models are pre-trained on ImageNet-1K [22]. ‘h14’ refers to hybrid UniFormer style with window size of 14 in Stage3.

Method	Pretrain	Backbone	UCF101	HMDB51
C3D [89]	Sports-1M	ResNet18	85.8	54.9
TSN [96]	IN1K+K400	InceptionV2	91.1	-
I3D [11]	IN1K+K400	InceptionV2	95.8	74.8
R(2+1)D [91]	K400	ResNet34	96.8	74.5
TSM [59]	IN1K+K400	ResNet50	94.5	70.7
STM [44]	IN1K+K400	ResNet50	96.2	72.2
TEA [56]	IN1K+K400	ResNet50	96.9	73.3
CT-Net [52]	IN1K+K400	ResNet50	96.2	73.2
TDN [97]	IN1K+K400	ResNet50	97.4	76.3
VidTr [54]	IN21K+K400	ViT-B	96.6	74.4
UniFormer	IN1K+K400	UniFormer-S	98.1	76.9

TABLE 6. Comparison with the state-of-the-art on UCF101 and HMDB51. Our UniFormer has strong generalization ability.

training with the same settings, our UniFormer-L achieves higher accuracy but only 21% FLOPs of LV-ViT-M [45] and 61% FLOPs of VOLO-D3 [113]. Moreover, when fine-tuned on 384×384 images, our UniFormer-L obtains 86.3% top-1 accuracy. It is even better than EfficientNetV2-L [86] with larger input, demonstrating the powerful learning capacity of our UniFormer.

6.2 Video Classification

Settings. We evaluate our UniFormer on the popular Kinetics-400 [10], Kinetics-600 [9], UCF101 [81] and HMDB51 [93], and we verify the transfer learning performance on temporal-related datasets Something-Something (SthSth) V1&V2 [34]. Our codes mainly rely on PySlowFast [27]. For training, we adopt the same training strategy as MViT [28] by default, but we do not apply random horizontal flip for SthSth. We utilize AdamW [65] optimizer with cosine learning rate schedule [66] to train our video backbones.. The first 5 or 10 epochs are used for warm-up [33] to overcome early optimization difficulty. For UniFormer-S, the warmup epoch, total epoch, stochastic depth rate, weight decay are set to 10, 110, 0.1 and 0.05 respectively for Kinetics, 5, 50, 0.3 and 0.05 respectively for SthSth, and 5, 20, 0.2 and 0.05 for UCF101 and HMDB51. For UniFormer-B, all the hyper-parameters are the same unless the stochastic depth rates are doubled. Moreover, We

linearly scale the base learning rates according to the batch size, which are 1e-4. $\frac{\text{batchsize}}{32}$ for Kinetics, 2e-4. $\frac{\text{batchsize}}{32}$ for SthSth, and 1e-5. $\frac{\text{batchsize}}{32}$ for UCF101 and HMDB51.

We utilize the dense sampling strategy [101] for Kinetics, UCF101 and HMDB51, and uniform sampling strategy [96] for Something-Something. To reduce the total training cost, we inflate the 2D convolution kernels pre-trained on ImageNet for Kinetics [11]. To obtain a better FLOPs-accuracy balance, Besides, we adopt multi-clip testing for Kinetics and multi-crop testing for Something-Something. All scores are averaged for the final prediction.

Results on Kinetics. In Table 3, we compare our UniFormer with the state-of-the-art methods on Kinetics-400 and Kinetics-600. The first part shows the prior works using CNN. Compared with SlowFast [30] equipped with non-local blocks [101], our UniFormer-S_{16f} requires 42× fewer GFLOPs but obtains 1.0% performance gain on both datasets (80.8% vs. 79.8% and 82.8% vs. 81.8%). Even compared with MoViNet [48], which is a strong CNN-based models via extensive neural architecture search, our model achieves slightly better results (82.0% vs. 81.5%) with fewer input frames (16f×4 vs. 120f). The second part lists the recent methods based on vision transformers. With only ImageNet-1K pre-training, UniFormer-B_{16f} surpasses most existing backbones with large dataset pre-training. For example, compared with ViViT-L [1] pre-trained from JFT-300M [83] and Swin-B [62] pre-trained from ImageNet-21K, UniFormer-B_{32f} obtains comparable performance (82.9% vs. 82.8% and 82.7%) with 16.7× and 3.3× fewer computation on both Kinetics-400 and Kinetics-600. These results demonstrate the effectiveness of our UniFormer for video.

Results on Something-Something. Table 4 presents the results on Something-Something (SthSth) V1&V2. Since these datasets require robust temporal relation modeling, it is difficult for the CNN-based methods to capture long-term dependencies, which leads to their worse results. On the contrary, video transformers are good at processing long sequential data and demonstrate better transfer learning capabilities [122], thus they achieve higher accuracy but with large computation costs. In contrast, our UniFormer-S_{16f} combines the advantages of both convolution and

Method	#Params (M)	FLOPs (G)	3× + MS schedule					
			AP^b	AP_{50}^b	AP_{75}^b	AP^m	AP_{50}^m	AP_{75}^m
Res50 [37]	82	739	46.3	64.3	50.5	40.1	61.7	43.4
DeiT [87]	80	889	48.0	67.2	51.7	41.4	64.2	44.3
Swin-T [61]	86	745	50.5	69.3	54.9	43.7	66.6	47.1
Shuffle-T [42]	86	746	50.8	69.6	55.1	44.1	66.9	48.0
Focal-T [109]	87	770	51.5	70.6	55.9	-	-	-
UniFormer-S _{h14}	79	747	52.1	71.1	56.6	45.2	68.3	48.9
X101-32 [107]	101	819	48.1	66.5	52.4	41.6	63.9	45.2
Swin-S [61]	107	838	51.8	70.4	56.3	44.7	67.9	48.5
Shuffle-S [42]	107	844	51.9	70.9	56.4	44.9	67.8	48.6
CSWin-S [23]	92	820	53.7	72.2	58.4	46.4	69.6	50.6
Swin-B [61]	145	972	51.9	70.9	57.0	45.3	68.5	48.9
Shuffle-B [42]	145	989	52.2	71.3	57.0	45.3	68.5	48.9
UniFormer-B _{h14}	107	878	53.8	72.8	58.5	46.4	69.9	50.4

TABLE 7. Object detection and instance segmentation with Cascade Mask R-CNN on COCO val2017. All the models are trained with 3× multi-scale schedule.

self-attention, obtaining **54.4% / 65.0%** in SthSth V1/V2 with only **42** GFLOPs. It also demonstrates that small model UniFormer-S benefit from larger dataset pre-training (Kinetics-400, 53.8% vs. Kinetics-600, 54.4%), but large model UniFormer-B do not (Kinetics-400, 59.1% vs. Kinetics-600, 58.8%). We argue that the large model is easy to converge better. Besides, it is worth noting that our UniFormer pre-trained from Kinetics-600 outperforms all the current methods under the same settings. In fact, our best model achieves the new state-of-the-art results: **61.0%** top-1 accuracy on SthSth V1 (**4.2%** higher than TDN_{EN}) [97] and **71.2%** top-1 accuracy on SthSth V2 (**1.6%** higher than Swin-B [62]). Such results verify its high capability of spatiotemporal learning.

Results on UCF101 and HMDB51. We further verify the generalization ability on UCF101 and HMDB51. Since these datasets are relatively small, the performances have already saturated. As shown in Table 6, our UniFormer significantly outperforms the previous SOTA methods, revealing its strong generalization ability to transfer to small datasets.

6.3 Object Detection and Instance Segmentation

Settings. We benchmark our models on object detection and instance segmentation with COCO2017 [60]. The ImageNet-1K pre-trained models are utilized as backbones and then armed with two representative frameworks: Mask R-CNN [6] and Cascade Mask R-CNN [36]. Our codes are mainly based on mmdetection [13], and the training strategies are the same as Swin Transformer [61]. We adopt two training schedules: 1× schedule with 12 epochs and 3× schedule with 36 epochs. For the 1× schedule, the shorter side of the image is resized to 800 while keeping the longer side no more than 1333. As for the 3× schedule, we apply the multi-scale training strategy to randomly resize the shorter side between 480 to 800. Besides, we use AdamW optimizer with the initial learning rate of 1e-4 and weight decay of 0.05. To regularize the training, we set the stochastic depth drop rates to 0.1/0.3 and 0.2/0.4 for our small/base models with Mask R-CNN and Cascade Mask R-CNN.

Results. Table 5 reports box mAP (AP^b) and mask mAP (AP^m) of the Mask R-CNN framework. It shows that our UniFormer variants outperform all the CNN and Transformer backbones. To reduce the training cost for object detection, we utilize a hybrid UniFomer style with a window size of 14 in Stage3 (denoted by $h14$). Specifically, with 1× schedule, our UniFormer brings 7.0-7.6 points of box mAP and 6.7-7.2 mask mAP against ResNet [37] at comparable settings. Compared with the popular

Method	Semantic FPN 80K		
	#Param(M)	FLOPs(G)	mIoU(%)
Res50 [37]	29	183	36.7
PVT-S [99]	28	161	39.8
TwinsP-S [16]	28	162	44.3
Twins-S [16]	28	144	43.2
Swin-T [61]	32	182	41.5
UniFormer-S _{h32}	25	199	46.2
UniFormer-S	25	247	46.6
Res101 [37]	48	260	38.8
PVT-M [99]	48	219	41.6
PVT-L [99]	65	283	42.1
TwinsP-B [16]	48	220	44.9
TwinsP-L [16]	65	283	46.4
Twins-B [16]	60	261	45.3
Swin-S [61]	53	274	45.2
Twins-L [16]	104	404	46.7
Swin-B [61]	91	422	46.0
UniFormer-B _{h32}	54	350	47.7
UniFormer-B	54	471	48.0

TABLE 8. Semantic segmentation with semantic FPN on ADE20K. The FLOPs are measured at resolution 512×2048. ‘h32’ means we utilize hybrid UniFormer blocks with window size of 32 in Stage3.

Method	Upernet 160K			
	#Param.(M)	FLOPs(G)	mIoU(%)	MS mIoU(%)
TwinsP-S [16]	55	919	46.2	47.5
Twins-S [16]	54	901	46.2	47.1
Swin-T [61]	60	945	44.5	45.8
Focal-T [109]	62	998	45.8	47.0
Shuffle-T [42]	60	949	46.6	47.8
UniFormer-S _{h32}	52	955	47.0	48.5
UniFormer-S	52	1008	47.6	48.5
Res101 [37]	86	1029	-	44.9
TwinsP-B [16]	74	977	47.1	48.4
Twins-B [16]	89	1020	47.7	48.9
Swin-S [61]	81	1038	47.6	49.5
Focal-T [109]	85	1130	48.0	50.0
Shuffle-S [42]	81	1044	48.4	49.6
Swin-B [61]	121	1188	48.1	49.7
Focal-B [109]	126	1354	49.0	50.5
Shuffle-B [42]	121	1196	49.0	50.5
UniFormer-B _{h32}	80	1106	49.5	50.7
UniFormer-B	80	1227	50.0	50.8

TABLE 9. Semantic segmentation with Upernet on ADE20K. The FLOPs are measured at resolution 512×2048. ‘h32’ means we utilize hybrid UniFormer blocks with window size of 32 in Stage3.

Swin Transformer [61], our UniFormer achieves 2.6-3.4 points of box mAP and 2.2-2.5 mask mAP improvement. Moreover, with 3× schedule and multi-scale training, our models still consistently surpass CNN and Transformer counterparts. For example, our UniFormer-B outperforms the powerful CSwin-S [23] by +0.3 box mAP and +0.3 mask mAP, and even better than larger backbones such as Swin-B and Focal-B [109]. Table 7 reports the results with the Cascade Mask R-CNN framework. The consistent improvement demonstrates our stronger context modeling capacity.

6.4 Semantic Segmentation

Settings. Our semantic segmentation experiments are conducted on the ADE20k [121] dataset and our codes are based on mmseg [19]. We adopt the popular Semantic FPN [47] and Upernet [104] as the basic framework. For a fair comparison, we follow the same

Arch.	Method	Input Size	#Param(M)	FLOPs(G)	<i>AP</i>	<i>AP⁵⁰</i>	<i>AP⁷⁵</i>	<i>AP^M</i>	<i>AP^L</i>	<i>AR</i>
CNN	SimpleBaseline-R101 [103]	256×192	53.0	12.4	71.4	89.3	79.3	68.1	78.1	77.1
	SimpleBaseline-R152 [103]	256×192	68.6	15.7	72.0	89.3	79.8	68.7	78.9	77.8
	HRNet-W32 [95]	256×192	28.5	7.1	74.4	90.5	81.9	70.8	81.0	78.9
	HRNet-W48 [95]	256×192	63.6	14.6	75.1	90.6	82.2	71.5	81.8	80.4
CNN+Trans	TransPose-H-A6 [110]	256×192	17.5	21.8	75.8	-	-	-	-	80.8
	TokenPose-L/D24 [55]	256×192	27.5	11.0	75.8	90.3	82.5	72.3	82.7	80.9
	HRFormer-S [114]	256×192	7.8	3.3	74.0	90.2	81.2	70.4	80.7	79.4
	HRFormer-B [114]	256×192	43.2	14.1	75.6	90.8	82.8	71.7	82.6	80.8
	UniFormer-S	256×192	25.2	4.7	74.0	90.3	82.2	66.8	76.7	79.5
	UniFormer-B	256×192	53.5	9.2	75.0	90.6	83.0	67.8	77.7	80.4
CNN	SimpleBaseline-R152 [103]	384×288	68.6	35.6	74.3	89.6	81.1	70.5	79.7	79.7
	HRNet-W32 [95]	384×288	28.5	16.0	75.8	90.6	82.7	71.9	82.8	81.0
	HRNet-W48 [95]	384×288	63.6	32.9	76.3	90.8	82.9	72.3	83.4	81.2
CNN+Trans	PRTR [51]	384×288	57.2	21.6	73.1	89.4	79.8	68.8	80.4	79.8
	PRTR [51]	512×384	57.2	37.8	73.3	89.2	79.9	69.0	80.9	80.2
	HRFormer-S [114]	384×288	7.8	7.2	75.6	90.3	82.2	71.6	82.5	80.7
	HRFormer-B [114]	384×288	43.2	30.7	77.2	91.0	83.6	73.2	84.2	82.0
	UniFormer-S	384×288	25.2	11.1	75.9	90.6	83.4	68.6	79.0	81.4
	UniFormer-B	384×288	53.5	22.1	76.7	90.8	84.0	69.3	79.7	81.9
	UniFormer-S	448×320	25.2	14.8	76.2	90.6	83.2	68.6	79.4	81.4
	UniFormer-B	448×320	53.5	29.6	77.4	91.1	84.4	70.2	80.6	82.5

TABLE 10. Human Pose estimation on COCO pose estimation val set. All the models are pre-trained on ImageNet-1K [22].

Method	Size	#Param (M)	FLOPs (G)	Throughput (imaegs/s) ↑	ImageNet Top-1
UniFormer-XXS	128	10.2	0.43	12886	76.8
MobileViT-XXS [69]	256	1.3	0.43	9742	68.9
EfficientNet-B0 [85]	224	5.3	0.42	9501	77.1
UniFormer-XXS	160	10.2	0.67	9382	79.1
PVTv2-B0 [100]	224	3.7	0.57	8737	70.7
EfficientNet-B1 [85]	240	7.8	0.74	5820	79.1
UniFormer-XXS	192	10.2	0.96	5766	79.9
MobileFormer [14]	224	14.0	0.51	4953	79.3
MobileViT-XS [69]	256	2.3	1.1	4822	74.6
PVTv2-B1 [100]	224	14.0	2.1	4812	78.7
UniFormer-XS	192	16.5	1.4	4492	81.5
UniFormer-XXS	224	10.2	1.3	4446	80.6
EfficientNet-B2 [85]	260	9.1	1.1	4247	80.1
UniFormer-XS	224	16.5	2.0	3506	82.0
MobileViT-S [69]	256	5.6	2.0	3360	78.3
EfficientNet-B3 [85]	300	12.2	1.9	2568	81.6

TABLE 11. Comparison with the SOTA lightweight models on ImageNet. Our efficient UniFormer achieves better accuracy-throughput trade-off. The throughput is measured on an NVIDIA A100 GPU using the largest possible batch size for each model.

setting of PVT [99] to train Semantic FPN for 80k iterations with cosine learning rate schedule [66]. As for Upernet, we apply the settings of Swin Transformer [61] with 160k iteration training. The stochastic depth drop rates are set to 0.1/0.2 and 0.25/0.4 for small/base variants with Semantic FPN and Upernet respectively.

Results. Table 8 and Table 9 report the results of different frameworks. It shows that with the Semantic FPN framework, our UniFormer-S_{h32/B_{h32}} achieve +4.7/+2.5 higher mIoU than the Swin Transformer [61] with similar model sizes. When equipped with the UperNet framework, they achieve +2.5/+1.9 mIoU and +2.7/+1.2 MS mIoU improvement. Furthermore, when utilizing the global MHRA, the results are consistently improved but with a larger computation cost. More results can be found in Table 21.

6.5 Pose Estimation

Settings. We evaluate the performance of UniFormer on the COCO2017 [60] human pose estimation benchmark. For a fair

Method	#frame	Size	#Param (M)	FLOPs (G)	Throughput (videos/s) ↑	K400 Top-1
UniFormer-XXS	4	128	10.4	1.0	1878	63.2
UniFormer-XXS	4	160	10.4	1.6	1384	65.8
UniFormer-XXS	8	128	10.4	2.0	1125	68.3
UniFormer-XXS	8	160	10.4	3.3	679	71.4
UniFormer-XXS	16	128	10.4	4.2	591	73.3
UniFormer-XXS	16	160	10.4	6.9	367	75.1
MoViNet-A0 [48]	50	172	3.1	2.7	315	65.8
MoViNet-A1 [48]	50	172	4.6	6.0	167	72.7
UniFormer-XXS	32	160	10.4	15.4	160	77.9
MoViNet-A2 [48]	50	224	4.8	10.3	103	75.0
UniFormer-XS	32	192	16.7	34.2	75	78.6
X3D-XS* [48]	4	182	3.8	27.4	39	69.1
MoViNet-A3 [48]	120	256	5.3	56.9	30	78.2
X3D-S* [48]	13	182	3.8	88.7	18	73.3

TABLE 12. Comparison with the SOTA lightweight models on Kinetics-400. Our efficient UniFormer achieves better accuracy-throughput trade-off. '*' indicates X3D is tested with 3 crops and 10 clips. The throughput is measured on an NVIDIA A100 GPU.

comparison with previous SOTA methods, we employ a single Top-Down head after our backbones. We follow the same training and evaluation setting of mmpose [18] as HRFormer [114]. In addition, the batch size and stochastic depth drop rates are set to 1024/256 and 0.2/0.5 for small/base variants during training.

Results. Table 10 reports results of different input resolutions on COCO validation set. Compared with previous SOTA CNN models, our UniFormer-B surpasses HRNet-W48 [95] by 0.4% AP with fewer parameters (53.5M vs. 63.6M) and FLOPs (22.1G vs. 32.9G). Moreover, our UniFormer-B can outperform the current best approach HRFormer [114] by 0.2% AP with smaller FLOPs (29.6G vs. 30.7G). It is worth noting that HRFormer [114], PRTR [51], TokenPose [55] and TransPose [110] are sophisticatedly designed for pose estimation task. On the contrary, our UniFormer can outperform all of them as a simple yet effective backbone.

6.6 Light-Weight UniFormer

Settings. For the light-weight UniFormer, we follow most of the previous settings. Differently, we train UniFormer-XSS and

Method	#Params (M)	Mask R-CNN 1× + MS schedule					
		AP ^b	AP ^b ₅₀	AP ^b ₇₅	AP ^m	AP ^m ₅₀	AP ^m ₇₅
PVTv2-B0 [100]	23.5	38.2	60.5	40.7	36.2	57.8	38.6
ResNet18 [37]	31.2	34.0	54.0	36.7	31.2	51.0	32.7
PVTv1-Tiny [99]	32.9	36.7	59.2	39.3	35.1	56.7	37.3
PVTv2-B1 [100]	33.7	41.8	64.3	45.9	38.8	61.2	41.6
UniFormer-XXS	29.4	42.8	65.0	47.0	39.2	61.7	42.0
UniFormer-XS	35.6	44.6	67.4	48.8	40.9	64.2	44.1

TABLE 13. Object detection and instance segmentation of SOTA lightweight models on COCO val2017.

Method	Semantic FPN 80K		
	#Param(M)	FLOPs(G)	mIoU(%)
PVTv2-B0 [100]	7.6	25.0	37.2
ResNet18 [37]	15.5	32.2	32.9
PVTv1-Tiny [99]	17.0	33.2	35.7
PVTv2-B1 [100]	17.8	34.2	42.5
UniFormer-XXS	13.5	29.2	42.3
UniFormer-XS	19.7	32.9	44.4

TABLE 14. Semantic segmentation of SOTA lightweight models on ADE20K. The FLOPs are measured at resolution 512×512.

UniFormer-XS for 600 epochs on ImageNet, since the lightweight models are difficult to converge [14], [35]

Results of classification. Table 11 represents the results on ImageNet. We roughly divide the models according to the FLOPs: <1G and 1G–2G. It clearly reveals that our efficient UniFormer achieves the best accuracy-throughput trade-off under similar FLOPs. For example, compared with the strong CNN method EfficientNet-B3 [85], our UniFormer-XS₁₉₂ obtains 1.7× higher throughput with similar performance. Compared with SOTA MobileFormer [14] that combines CNN and ViT, our UniFormer-XXS₁₉₂ obtains 0.6% higher accuracy with 16% higher throughput. We further fine-tune the above models with different frames on Kinetics-400. Results in Table 12 show that our efficient backbone surpasses the SOTA lightweight video backbones by a large margin. Compared with MoViNet-A0 [48], our UniFormer-XXS_{150×16f} achieves 9.3% higher performance with 16% higher throughput. While compared with X3D-S [29], our UniFormer-XS_{192×32f} runs 4.2× faster with 5.4% higher accuracy. Note that we do not apply complicated designs as in the recent lightweight methods [14], [48], [69]. Our concise extension already shows powerful performance, which further demonstrates the great potential of UniFormer.

Results of dense prediction. We also verify the efficient UniFormer for COCO object detection and instance segmentation in Table 13, and ADE20K semantic segmentation in Table 14. Our models obviously beat ResNet [37] and PVTv2 [100] on these dense prediction tasks. For example, our UniFormer-XS brings 2.7 points of box mAP and 2.1 mask mAP against PVTv2-B1 on COCO, and achieves +1.9 mIoU improvement on ADE20K.

6.7 Ablation Studies

To inspect the effectiveness of UniFormer as the backbone, we ablate each key structure design and evaluate the performance on image and video classification datasets. Furthermore, for video backbones, we explore the vital designs of pre-training, training and testing. Finally, we demonstrate the efficiency of our adaption on downstream tasks, and the effectiveness of H-UniFormer.

6.7.1 Model designs for image and video backbones

We conduct ablation studies of the vital components in Table 15.

FFN	DPE	MHRA		ImageNet		K400	
		Size	Type	#Param	Top-1	GFLOPs	Top-1
✓	✓	5	LLGG	21.5	82.9	41.8	79.3
✗	✓	5	LLGG	21.3	82.6	41.0	78.6
✓	✗	5	LLGG	21.5	82.4	41.4	77.6
✓	✓	3	LLGG	21.5	82.8	41.0	79.0
✓	✓	7	LLGG	21.6	82.9	43.5	79.1
✓	✓	9	LLGG	21.6	82.8	46.6	78.9
✓	✓	5	LLLL	23.3	81.9	31.6	77.2
✓	✓	5	LLLG	22.2	82.5	31.6	78.4
✓	✓	5	LGGG	21.6	82.7	39.0	79.0
✓	✓	5	GGGG	20.1	82.1	72.0	75.3

TABLE 15. Structure designs. All image models are trained for 300 epochs on ImageNet. All video models are trained for 50 epochs on Kinetics-400 with 16 frames. For fair comparisons, we guarantee the parameters and computation of all the models are similar. When modifying the stage types, we modify the stage numbers and reduce the computation of self-attention as MViT [28] for LGGG and GGGG.

Type	Joint	GFLOPs	Pretrain Dataset		SSV1	
			Top-1	Top-5	Top-1	Top-5
LLLL	✓	26.1	IN-1K 81.0	49.2	77.4	
			K400 77.4	49.2(+0.0)	77.6(+0.2)	
LLGG	✗	36.8	IN-1K 82.9	51.9	80.1	
			K400 80.1	51.8(-0.1)	80.1(+0.0)	
LLGG	✓	41.8	IN-1K 82.9	52.0	80.2	
			K400 80.8	53.8(+1.8)	81.9(+1.7)	

TABLE 16. Transfer learning. When the model is gradually pre-trained from ImageNet to Kinetics-400, joint manner performs better.

Dataset	Pretrain	#frame × #crop × #clip	2D		3D	
			Top-1	Top-5	Top-1	Top-5
K400	IN-1K	8×1×1	74.7	90.8	74.9	90.7
		8×1×4	78.5	93.2	78.4	93.3
SSV1	IN-1K	8×1×1	47.9	75.8	48.3	76.1
		8×3×1	51.4	79.6	51.3	79.7
K400		8×1×1	47.9	75.6	48.6	75.6
		8×3×1	51.3	79.4	51.5	79.5

TABLE 17. Inflating methods of convolutional filters. Inflating the filters to 3D performs better for different datasets.

FFN. As mentioned in Section 3.2, our UniFormer blocks in the shallow layers are instantiated as a transformer-style MobileNet block [90] with extra FFN as in ViT [24]. Hence, we first investigate its effectiveness by replacing our UniFormer blocks in the shallow layers with MobileNet blocks [77]. BN and GELU are added as the original paper, but the expand ratios are set to 3 for similar parameters. Note that the dynamic position embedding is kept for a fair comparison. As expected, our UniFormer outperforms such MobileNet block both in ImageNet (+0.3%) and Kinetics-400 (+0.7%). It shows that, FFN in our model can further mix token context at each position to boost classification accuracy.

DPE. With dynamic position embedding, our UniFormer obviously improves the top-1 accuracy by +0.5% on ImageNet, but +1.7% on Kinetics-400. It shows that via encoding the position information, our DPE can maintain spatial and temporal order, thus contributing to better representation learning, especially for video.

MHRA. In our local token affinity (Eq. 6), we aggregate the context from a small local tube. Hence, we investigate the influence of this tube by changing the size from 3 to 9. Results show that our network is robust to the tube size on both ImageNet and Kinetics-400. We simply choose the kernel size of 5 for better

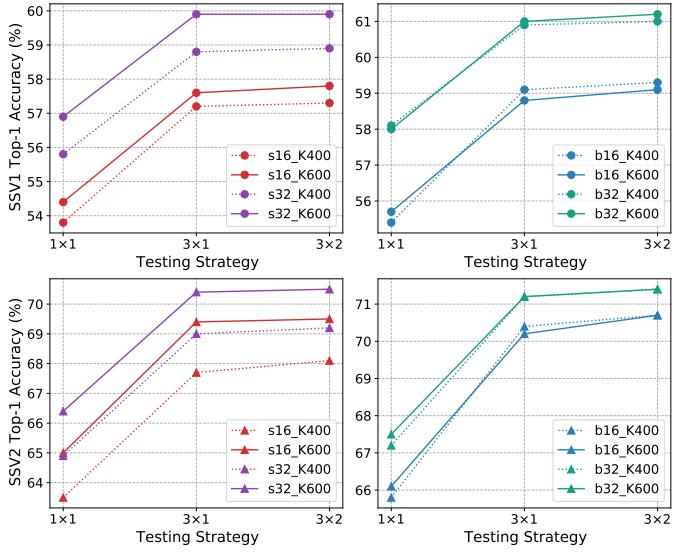


Fig. 7. Pre-trained dataset scales. Small models are eager for larger dataset pre-training on both Something-Something V1 and V2.

Model	#frame × #crop × #clip	FLOPs (G)	Sampling Stride	K400		K600	
				Top-1	Top-5	Top-1	Top-5
Small	16 × 1 × 1	41.8	4	76.2	92.2	79.0	93.6
			8	78.4	92.9	80.8	94.7
Base	16 × 1 × 4	167.2	4	80.8	94.7	82.8	95.8
			8	80.8	94.4	82.7	95.7
Small	16 × 1 × 1	96.7	4	78.1	92.8	80.3	94.5
			8	79.3	93.4	81.7	95.0
Base	16 × 1 × 4	386.8	4	82.0	95.1	84.0	96.4
			8	81.7	94.8	83.4	96.0
Small	32 × 1 × 1	109.6	2	77.3	92.4	-	-
			4	79.8	93.4	-	-
Base	32 × 1 × 4	438.4	2	81.2	94.7	-	-
			4	82.0	95.1	-	-

TABLE 18. Sampling stride of dense sampling. Larger sampling stride often achieves a higher single-clip result.

accuracy. More importantly, we investigate the configuration of local and global UniFormer block stage by stage, in order to verify the effectiveness of our network. As shown in row1, 7-10 in Table 15, when we only use local MHRA (LLLL), the computation cost is light. However, the accuracy is largely dropped (-1.0% and -2.1% on ImageNet and Kinetics-400) due to the lack of capacity for learning long-term dependency. When we gradually replace local MHRA with global MHRA, the accuracy becomes better as expected. Unfortunately, the accuracy is dramatically dropped with a heavy computation load when all the layers apply global MHRA (GGGG), i.e., -4.0% on Kinetics-400. It is mainly because that, without local MHRA, the network lacks the ability to extract detailed representations, leading to severe model overfitting with redundant attention for sequential video data.

6.7.2 Pre-training, training and testing for video backbone

In this section, we explore more designs for our video backbones. Firstly, to load 2D pre-trained backbones, it is essential to determine how to inherit self-attention and inflate convolution filters. Hence, we compare the transfer learning performance of different MHRA configurations and inflating methods of filters. Besides, since we use dense sampling [101] for Kinetics, we should confirm the appropriate sampling stride. Furthermore, as we utilize Kinetics pre-trained models for SthSth, it is interesting to explore the effect of sampling methods and dataset scales for pre-trained models.

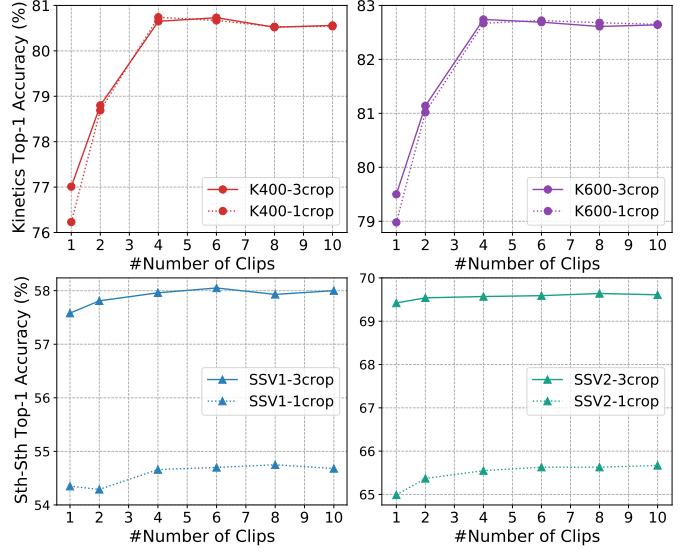


Fig. 8. Multi-clip/crop testing strategies. Multi-clip testing is better for Kinetics and multi-crop testing is better for Something-Something.

Model	Train #frame	Pre-train #frame × #stride	1crop × 1clip		3crops × 1clip	
			Top-1	Top-5	Top-1	Top-5
Small	16	16 × 4	53.8	81.9	57.2	84.9
		16 × 8	53.7	81.3	57.3	84.6
Base	16	16 × 4	55.4	82.9	59.1	86.2
		16 × 8	55.5	83.1	58.8	86.2
Small	32	16 × 4	55.8	83.6	58.8	86.4
		32 × 2	55.6	83.1	58.6	85.6
		32 × 4	55.9	82.9	58.9	86.0

TABLE 19. Sampling methods of Kinetics pre-trained model. All models are pre-trained on K400 and fine-tuned on SthSth V1.

Finally, we ablate the testing strategies for different datasets.

Transfer learning. Table 16 presents the results of transfer learning. All models share the same stage numbers but the stage types are different. For Kinetics-400, it clearly shows that the joint version is more powerful than the separate one, verifying that joint spatiotemporal attention can learn more discriminative video representations. As for SthSth V1, when the model is gradually pre-trained from ImageNet to Kinetics-400, the performance of our joint version becomes better. Compared with pre-training from ImageNet, pre-training from Kinetics-400 will further improve the top-1 accuracy by +1.8%. However, such distinct characteristic is not observed in the pure local MHRA structure (LLLL) and UniFormer with divided spatiotemporal attention. It demonstrates that the joint learning manner is preferable for transfer learning, thus we adopt it by default.

Inflating methods. As indicated in I3D [11], we can inflate the 2D convolutional filters for easier optimization. Here we consider whether or not to inflate the filters. Note that the first convolutional filter in the patch stem is always inflated for temporal downsampling. As shown in Table 17, inflating the filters to 3D achieves similar results on Kinetics-400, but obtains performance improvement on SthSth V1. We argue that Kinetics-400 is a scene-related dataset, thus 2D convolution is enough to recognize the action. In contrast, SthSth V1 is a temporal-related dataset, which requires powerful spatiotemporal modeling. Hence, we inflate all the convolutional filters to 3D for better generality by default.

Sampling stride. For dense sampling strategy, the basic hyperparameter is the sampling stride of frames. Intuitively, a larger sampling stride will cover a longer frame range, which is essential

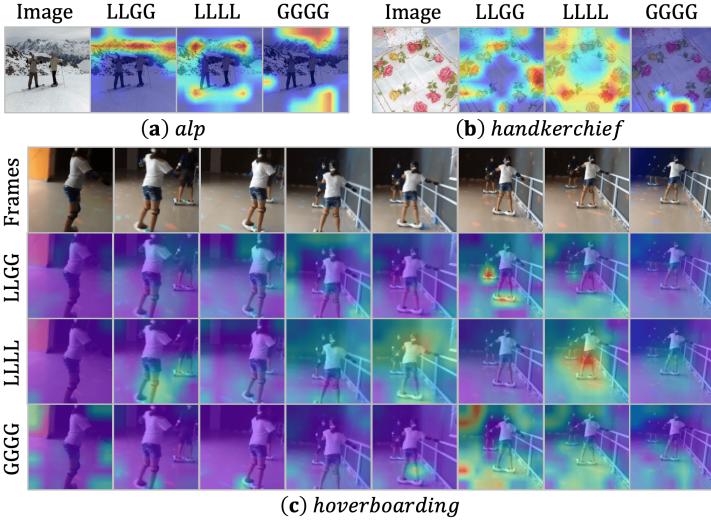


Fig. 9. Attention visualization of different structures.

Type	FLOPs (G)	1× + MS				3× + MS			
		AP ^b	AP ^b ₅₀	AP ^m	AP ^m ₅₀	AP ^b	AP ^b ₅₀	AP ^m	AP ^m ₅₀
W-14	250	45.0	67.8	40.8	64.7	47.5	69.8	43.0	66.7
H-14	269	45.4	68.2	41.4	64.9	48.2	70.4	43.4	67.1
G	326	45.8	68.7	41.5	50.5	48.1	70.1	43.4	67.1

TABLE 20. Adaption types for object detection.

Model	Type	Semantic FPN		UperNet	
		GFLOPs	mIoU(%)	GFLOPs	(MS)mIoU(%)
Small	W-32	183	45.2	939	(48.4)46.6
	H-32	199	46.2	955	(48.5) 47.0
	G	247	46.6	1004	(48.5) 47.6
Base	W-32	310	47.2	1066	(50.6)49.1
	H-32	350	47.7	1106	(50.7)49.5
	G	471	48.0	1227	(50.8) 50.0

TABLE 21. Adaption types for semantic segmentation.

Type	Input Size	FLOPs (G)	AP	AP ⁵⁰	AP ⁷⁵	AP ^M	AP ^L	AR
W-14	384×288	12.3	76.1	90.8	83.2	68.9	79.1	81.1
H-14	384×288	12.0	75.9	90.7	83.2	68.6	78.9	81.0
G	384×288	11.1	75.9	90.6	83.4	68.6	79.0	81.4

TABLE 22. Adaption types for pose estimation. Due to zero padding, the models with window UniFormer style require more computation.

for better video understanding. In Table 18, we show more results on Kinetics under different sampling strides. As expected, larger sampling stride (i.e. sparser sampling) often achieves higher single-clip results. However, when testing with multi clips, sampling with a frame stride of 4 always performs better.

Sampling methods of Kinetics pre-trained model. For SthSth, we uniformly sample frames as suggested in [52]. Since we load Kinetics pre-trained models for fast convergence, it is necessary to find out whether pre-trained models that cover more frames can help fine-tuning. Table 19 shows that, different pre-trained models achieve similar performances for fine-tuning. We apply 16×4 pre-training for better generalization.

Pre-trained dataset scales. In Figure 7, we show more results on SthSth with Kinetics-400/600 pre-training. For UniFormer-S, Kinetics-600 pre-training consistently performs better than Kinetics-400 pre-training, especially for large benchmark SthSth V2. However, both of them achieve comparable results for UniFormer-B. These results indicate that small models are harder to converge and eager for larger dataset pre-training, but big models are not.

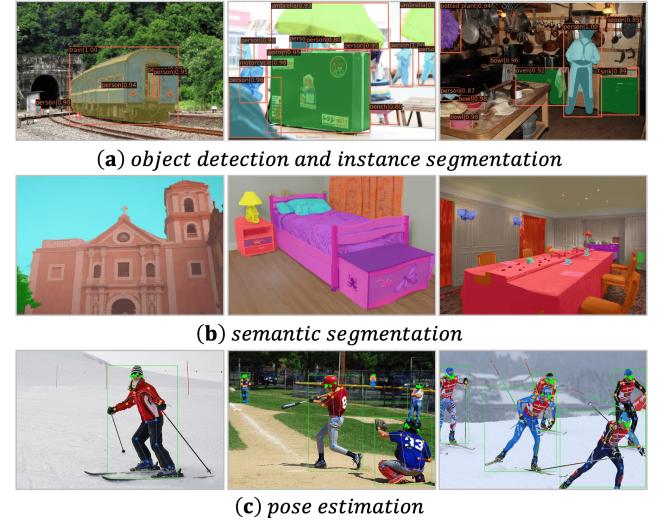


Fig. 10. Qualitative examples for different downstream tasks.

Score Token	Running Mean	Shrinking Ratio	GFLOPs	Throughput	Top-1
✓	✓	0.5	0.67	9382	79.1
✗	✓	0.5	0.67	9395(+0.1%)	78.7(-0.4)
✓	✗	0.5	0.67	9283(+0.0%)	78.8(-0.3)
✓	✓	1.0	0.91	8342(-11.1%)	79.9(+0.8)
✓	✓	0.8	0.82	8452(-9.9%)	79.8(+0.7)
✓	✓	0.6	0.72	9094(-3.1%)	79.3(+0.2)
✓	✓	0.4	0.62	9692(+3.3%)	78.4(-0.7)
✓	✓	0.25	0.55	10162(+8.3%)	76.8(-2.3)

TABLE 23. Lightweight designs for H-UniFormer.

Testing strategies. We evaluate our network with various numbers of clips and crops for the validation videos on different datasets. As shown in Figure 8, since Kinetics is a scene-related dataset and trained with dense sampling, multi-clip testing is preferable to cover more frames for boosting performance. Alternatively, Something-Something is a temporal-related dataset and trained with uniform sampling, so multi-crop testing is better for capturing the discriminative motion for boosting performance.

6.7.3 Adaption designs for downstream tasks

We verify the effectiveness of our adaption for dense prediction tasks in Table 20, Table 21 and Table 22. ‘W’, ‘H’ and ‘G’ refer to window, hybrid and global UniFormer style in Stage3 respectively. Note that the pre-trained global UniFormer block can be seen as a window UniFormer block with a large window size, thus the minimal window size in our experiments is 224/32=14.

Table 20 shows results on object detection. Though the hybrid style performs worse than the global style with the 1× schedule, it achieves comparable results with the 3× schedule, which indicates that training more epochs can narrow the performance gap. We further conduct experiments on semantic segmentation with different model variants in Table 21. As expected, large window size and global UniFormer blocks contribute to better performances, especially for big models. Moreover, when testing with multi-scale inputs, hybrid style with a window size of 32 obtains similar results to the global style. As for human pose estimation (Table 22), due to the small input resolution, i.e. 384×288, utilizing window style requires more computation for zero paddings. We simply apply global UniFormer blocks for better computation-accuracy balance.

6.7.4 Designs for H-UniFormer

We further explore the lightweight designs based on UniFormer-XXS₁₆₀ in Table 23. Firstly, we try to remove the score token (i.e., \mathbf{s}) and simply use the mean of global similarity $A_n^{global}(\mathbf{X}, \mathbf{X})$ to measure the token importance. Results show that the learnable score token is more helpful for token selection. Besides, the running mean of the similarity score (i.e., $\mathbf{A} = (\mathbf{A} + \mathbf{A}^{pre})/2$) will improve the top-1 accuracy, which verifies the effectiveness of consistent important tokens. Finally, we ablate different shrinking ratios, where we use the ratio of 0.5 for a better trade-off.

6.8 Visualizations

In Figure 9, We apply Grad-CAM [78] to show the areas of the greatest concern in the last layer. Images are sampled from ImageNet validation set [22] and the video is sampled from Kintics-400 validation set [10]. It reveals that GGGG struggles to focus on the key object, i.e., the mountain and the skateboard, as it blindly compares the similarity of all tokens in all layers. Alternatively, LLLL only performs local aggregation. Hence, its attention tends to be coarse and inaccurate without a global view. Different from both cases, our UniFormer with LLGG can cooperatively learn local and global contexts in a joint manner. As a result, it can effectively capture the most discriminative information, by paying precise attention to the mountain and the skateboard.

In Figure 10, we further conduct visualization on validation datasets for various downstream tasks. Such robust qualitative results demonstrate the effectiveness of our UniFormer backbones.

7 CONCLUSION

In this paper, we propose a novel UniFormer for efficient visual recognition, which can effectively unify convolution and self-attention in a concise transformer format to overcome redundancy and dependency. We adopt local MHRA in shallow layers to largely reduce computation burden and global MHRA in deep layers to learn global token relation. Extensive experiments demonstrate the powerful modeling capacity of our UniFormer. Via simple yet effective adaption, our UniFormer achieves state-of-the-art results on a broad range of vision tasks with less training cost.

REFERENCES

- [1] A. Arnab, M. Dehghani, G. Heigold, Chen Sun, Mario Lucic, and C. Schmid. Vivit: A video vision transformer. *ICCV*, 2021. 2, 3, 5, 6, 8, 9
- [2] Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *ArXiv*, abs/1607.06450, 2016. 5
- [3] Gedas Bertasius, Heng Wang, and L. Torresani. Is space-time attention all you need for video understanding? *ICML*, 2021. 2, 3, 5, 6, 8
- [4] Andrew Brock, Soham De, Samuel L. Smith, and K. Simonyan. High-performance large-scale image recognition without normalization. *ArXiv*, abs/2102.06171, 2021. 7
- [5] Adrian Bulat, Juan-Manuel Pérez-Rúa, Swathikiran Sudhakaran, Brais Martínez, and Georgios Tzimiropoulos. Space-time mixing attention for video transformer. *NIPS*, 2021. 3, 8
- [6] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: High quality object detection and instance segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 3, 10
- [7] Jie Cao, Yawei Li, K. Zhang, and Luc Van Gool. Video super-resolution transformer. *ArXiv*, abs/2106.06847, 2021. 3
- [8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *ECCV*, 2020. 3
- [9] João Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600. *ArXiv*, abs/1808.01340, 2018. 8, 9
- [10] João Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. *CVPR*, 2017. 8, 9, 15
- [11] João Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. *CVPR*, 2017. 3, 6, 9, 13
- [12] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. *CVPR*, 2021. 3
- [13] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 10
- [14] Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Xiaoyi Dong, Lu Yuan, and Zicheng Liu. Mobile-former: Bridging mobilenet and transformer. *CVPR*, 2022. 4, 6, 11, 12
- [15] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *ArXiv*, abs/2107.06278, 2021. 3
- [16] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. In *NIPS*, 2021. 5, 9, 10
- [17] Xiangxiang Chu, Bo Zhang, Zhi Tian, Xiaolin Wei, and Huaxia Xia. Do we really need explicit position encodings for vision transformers? *ArXiv*, abs/2102.10882, 2021. 3, 5
- [18] MPose Contributors. Openmmlab pose estimation toolbox and benchmark. <https://github.com/open-mmlab/mmpose>, 2020. 11
- [19] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. <https://github.com/open-mmlab/mmsegmentation>, 2020. 10
- [20] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. *ArXiv*, abs/1911.03584, 2020. 3
- [21] Zihang Dai, Hanxiao Liu, Quoc V. Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *NIPS*, 2021. 3, 7, 8
- [22] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 3, 8, 9, 11, 15
- [23] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and B. Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. *CVPR*, 2022. 3, 5, 7, 8, 9, 10
- [24] A. Dosovitskiy, L. Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, M. Dehghani, Matthias Minderer, G. Heigold, S. Gelly, Jakob Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 1, 2, 3, 4, 5, 6, 12
- [25] Brendan Duke, Abdalla Ahmed, Christian Wolf, Parham Aarabi, and Graham W. Taylor. Sstvos: Sparse spatiotemporal transformers for video object segmentation. *CVPR*, 2021. 3
- [26] Maksim Dzabraev, Maksim Kalashnikov, Stepan Alekseevich Komkov, and Aleksandr Petushko. Mdmm: Multidomain multimodal transformer for video retrieval. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021. 3
- [27] Haoqi Fan, Yanghao Li, Bo Xiong, Wan-Yen Lo, and Christoph Feichtenhofer. Pyslowfast. <https://github.com/facebookresearch/slowfast>, 2020. 9
- [28] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, J. Malik, and Christoph Feichtenhofer. Multiscale vision transformers. *ICCV*, 2021. 1, 3, 8, 9, 12
- [29] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. *CVPR*, 2020. 3, 4, 8, 12
- [30] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. *ICCV*, 2019. 1, 3, 8, 9
- [31] Valentin Gabeur, Chen Sun, Alahari Karttik, and Cordelia Schmid. Multi-modal transformer for video retrieval. In *ECCV*, 2020. 3
- [32] Peng Gao, Jiasen Lu, Hongsheng Li, R. Mottaghi, and Aniruddha Kembhavi. Container: Context aggregation network. *NIPS*, 2021. 7
- [33] Priya Goyal, Piotr Dollár, Ross B. Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *ArXiv*, abs/1706.02677, 2017. 8, 9
- [34] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fründ, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The “something something”

- video database for learning and evaluating visual common sense. *ICCV*, 2017. 8, 9
- [35] Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé J’egou, and M. Douze. Levit: a vision transformer in convnet’s clothing for faster inference. *ICCV*, 2021. 12
- [36] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 3, 10
- [37] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 2016. 1, 3, 5, 9, 10, 12
- [38] Shuteng He, Haowen Luo, Pichao Wang, F. Wang, Hao Li, and Wei Jiang. Transreid: Transformer-based object re-identification. *ArXiv*, abs/2102.04378, 2021. 3
- [39] Andrew G. Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for mobilenetv3. *International Conference on Computer Vision*, 2019. 4
- [40] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, M. Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *ArXiv*, abs/1704.04861, 2017. 1, 3, 4
- [41] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CVPR*, 2017. 3
- [42] Zilong Huang, Youcheng Ben, Guozhong Luo, Pei Cheng, Gang Yu, and Bin Fu. Shuffle transformer: Rethinking spatial shuffle for vision transformer. *ArXiv*, abs/2106.03650, 2021. 10
- [43] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ArXiv*, abs/1502.03167, 2015. 5
- [44] Boyuan Jiang, Mengmeng Wang, Weihao Gan, Wei Wu, and Junjie Yan. Stm: Spatiotemporal and motion encoding for action recognition. *ICCV*, 2019. 3, 9
- [45] Zihang Jiang, Qibin Hou, Li Yuan, Daquan Zhou, Yujun Shi, Xiaojie Jin, Anran Wang, and Jiashi Feng. All tokens matter: Token labeling for training better vision transformers. *NIPS*, 2021. 5, 7, 8, 9
- [46] Youngsaeng Jin, David K. Han, and Hanseok Ko. Trseg: Transformer for semantic segmentation. *Pattern Recognit. Lett.*, 148:29–35, 2021. 3
- [47] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, 2019. 3, 10
- [48] D. Kondratyuk, Liangzhe Yuan, Yandong Li, Li Zhang, Mingxing Tan, Matthew A. Brown, and Boqing Gong. Movinet: Mobile video networks for efficient video recognition. *ArXiv*, abs/2103.11511, 2021. 6, 8, 9, 11, 12
- [49] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 3
- [50] Heeseung Kwon, Manjin Kim, Suha Kwak, and Minsu Cho. Motion-squeeze: Neural motion feature learning for video understanding. In *ECCV*, 2020. 8
- [51] Ke Li, Shijie Wang, Xiang Zhang, Yifan Xu, Weijian Xu, and Zhiuwon Tu. Pose recognition with cascade transformers. *CVPR*, 2021. 11
- [52] Kunchang Li, Xianhang Li, Yali Wang, Jun Wang, and Y. Qiao. Ct-net: Channel tensorization network for video classification. *ICLR*, 2021. 3, 8, 9, 14
- [53] X. Li, Yali Wang, Zhipeng Zhou, and Yu Qiao. Smallbignet: Integrating core and contextual views for video classification. *CVPR*, 2020. 2, 3, 8
- [54] Xinyu Li, Yanyi Zhang, Chunhui Liu, Bing Shuai, Yi Zhu, Biagio Brattoli, Hao Chen, Ivan Marsic, and Joseph Tighe. Vidtr: Video transformer without convolutions. *ICCV*, 2021. 9
- [55] Yanjie Li, Shoukui Zhang, Zhicheng Wang, Sen Yang, Wankou Yang, Shutao Xia, and Erjin Zhou. Tokenpose: Learning keypoint tokens for human pose estimation. *ArXiv*, abs/2104.03516, 2021. 3, 11
- [56] Yinong Li, Bin Ji, Xintian Shi, Jianguo Zhang, Bin Kang, and Limin Wang. Tea: Temporal excitation and aggregation for action recognition. *CVPR*, 2020. 3, 8, 9
- [57] Jingyun Liang, Jie Cao, Guolei Sun, K. Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. 2021 *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 1833–1844, 2021. 3
- [58] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Evit: Expediting vision transformers via token reorganizations. In *International Conference on Learning Representations*, 2022. 7
- [59] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. *ICCV*, 2019. 3, 8, 9
- [60] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 3, 8, 10, 11
- [61] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *ICCV*, 2021. 1, 2, 3, 5, 6, 7, 8, 9, 10, 11
- [62] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, S. Lin, and Han Hu. Video swin transformer. *CVPR*, 2022. 2, 3, 8, 9, 10
- [63] Zhaoyang Liu, D. Luo, Yabiao Wang, L. Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Tong Lu. Teinet: Towards an efficient architecture for video recognition. *AAAI*, 2020. 3, 8
- [64] Zhouyong Liu, Shun Luo, Wubin Li, Jingben Lu, Yufan Wu, Chunguo Li, and Luxi Yang. Convtransformer: A convolutional transformer network for video frame synthesis. *ArXiv*, abs/2011.10185, 2020. 3
- [65] I. Loshchilov and F. Hutter. Fixing weight decay regularization in adam. *ArXiv*, abs/1711.05101, 2017. 8, 9
- [66] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv: Learning*, 2017. 8, 9, 11
- [67] Chenxu Luo and Alan L. Yuille. Grouped spatial-temporal aggregation for efficient action recognition. *ICCV*, 2019. 3, 8
- [68] Ningning Ma, Xiangyu Zhang, Haitao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *European Conference on Computer Vision*, 2018. 4
- [69] Sachin Mehta and Mohammad Rastegari. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. In *ICML*, 2021. 4, 6, 11, 12
- [70] Daniel Neimark, Omri Bar, Maya Zohar, and Dotan Asselmann. Video transformer network. *ICCV*, 2021. 8
- [71] Mandela Patrick, Dylan Campbell, Yuki M. Asano, Ishan Misra Florian Metze, Christoph Feichtenhofer, A. Vedaldi, and João F. Henriques. Keeping your eye on the ball: Trajectory attention in video transformers. *NIPS*, 2021. 3, 8
- [72] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. *ICCV*, 2017. 3
- [73] Zhaofan Qiu, Ting Yao, C. Ngo, Xinmei Tian, and Tao Mei. Learning spatio-temporal representation with local and global diffusion. *CVPR*, 2019. 8
- [74] Ilija Radosavovic, Raj Prateek Kosaraju, Ross B. Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. *CVPR*, 2020. 7, 8
- [75] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. In *NIPS*, 2019. 3, 6
- [76] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamiccvit: Efficient vision transformers with dynamic token sparsification. In *Neural Information Processing Systems*, 2021. 7
- [77] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018. 3, 4, 6, 12
- [78] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 2019. 15
- [79] Gilad Sharir, Asaf Noy, and Lihai Zelnik-Manor. An image is worth 16x16 words, what is a video worth? *ICLR*, 2021. 5, 8
- [80] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3
- [81] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *ArXiv*, abs/1212.0402, 2012. 9
- [82] A. Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, P. Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. *CVPR*, 2021. 7
- [83] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Kumar Gupta. Revisiting unreasonable effectiveness of data in deep learning era. *ICCV*, pages 843–852, 2017. 9
- [84] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, D. Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CVPR*, 2015. 3
- [85] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *ArXiv*, abs/1905.11946, 2019. 3, 4, 6, 7, 8, 11, 12
- [86] Mingxing Tan and Quoc V. Le. Efficientnetv2: Smaller models and faster training. *ArXiv*, abs/2104.00298, 2021. 4, 7, 9
- [87] Hugo Touvron, M. Cord, M. Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé J’egou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 2, 3, 5, 7, 8, 10
- [88] Hugo Touvron, M. Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé J’egou. Going deeper with image transformers. *ArXiv*, abs/2103.17239, 2021. 1, 5, 7, 8

- [89] Du Tran, Lubomir D. Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. *ICCV*, 2015. 3, 9
- [90] Du Tran, Heng Wang, L. Torresani, and Matt Feiszli. Video classification with channel-separated convolutional networks. *ICCV*, 2019. 3, 4, 8, 12
- [91] Du Tran, Hong xiu Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. *CVPR*, 2018. 3, 9
- [92] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762, 2017. 3, 5
- [93] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. *2013 IEEE International Conference on Computer Vision*, pages 3551–3558, 2013. 9
- [94] Heng Wang, Du Tran, L. Torresani, and Matt Feiszli. Video modeling with correlation networks. *CVPR*, 2020. 8
- [95] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, D. Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 3, 11
- [96] L. Wang, Yuanjun Xiong, Zhe Wang, Y. Qiao, D. Lin, X. Tang, and L. Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016. 8, 9
- [97] Limin Wang, Zhan Tong, Bin Ji, and Gangshan Wu. Tdn: Temporal difference networks for efficient action recognition. *CVPR*, 2021. 8, 9, 10
- [98] Ning Wang, Wen gang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. *CVPR*, 2021. 3
- [99] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, P. Luo, and L. Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *ICCV*, 2021. 3, 5, 7, 9, 10, 11, 12
- [100] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvtt2: Improved baselines with pyramid vision transformer. *ArXiv*, abs/2106.13797, 2021. 3, 7, 11, 12
- [101] X. Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. *CVPR*, 2018. 2, 3, 9, 13
- [102] Haiping Wu, Bin Xiao, N. Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *ICCV*, 2021. 3, 7
- [103] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *ECCV*, 2018. 11
- [104] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018. 3, 10
- [105] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross B. Girshick. Early convolutions help transformers see better. *ArXiv*, abs/2106.14881, 2021. 3
- [106] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *NIPS*, 2021. 3
- [107] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. *CVPR*, 2017. 1, 3, 9, 10
- [108] Yifan Xu, Zhijie Zhang, Mengdan Zhang, Kekai Sheng, Ke Li, Weiming Dong, Liqing Zhang, Changsheng Xu, and Xing Sun. Evo-vit: Slow-fast token evolution for dynamic vision transformer. In *AAAI*, 2021. 8
- [109] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. *arXiv preprint arXiv:2107.00641*, 2021. 3, 7, 8, 9, 10
- [110] Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. Transpose: Towards explainable human pose estimation by transformer. *ArXiv*, abs/2012.14214, 2020. 3, 11
- [111] Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers. *ArXiv*, abs/2103.11816, 2021. 3, 7
- [112] Li Yuan, Y. Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis E. H. Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *ArXiv*, abs/2101.11986, 2021. 7
- [113] Li Yuan, Qibin Hou, Zihang Jiang, Jiashi Feng, and Shuicheng Yan. Volo: Vision outlooker for visual recognition. *ArXiv*, abs/2106.13112, 2021. 7, 9
- [114] Yuhui Yuan, Rao Fu, Lang Huang, Weihong Lin, Chao Zhang, Xilin Chen, and Jingdong Wang. Hrformer: High-resolution transformer for dense prediction. *NIPS*, 2021. 3, 11
- [115] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Young Joon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *ICCV*, 2019. 8
- [116] David Junhao Zhang, Kunchang Li, Yunpeng Chen, Yali Wang, Shashwat Chandra, Yu Qiao, Luoqi Liu, and Mike Zheng Shou. Morphmlp: A self-attention free, mlp-like backbone for image and video. *arXiv preprint arXiv:2111.12527*, 2021. 1
- [117] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020. 1
- [118] Hongyi Zhang, Moustapha Cissé, Yann Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ArXiv*, abs/1710.09412, 2018. 8
- [119] Pengchuan Zhang, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. *ArXiv*, abs/2103.15358, 2021. 9
- [120] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *CVPR*, 2018. 3, 4
- [121] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 2019. 3, 8, 10
- [122] Hong-Yu Zhou, Chixiang Lu, Sibei Yang, and Yizhou Yu. Convnets vs. transformers: Whose visual representations are more transferable? *ArXiv*, abs/2108.05305, 2021. 9
- [123] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *ICLR*, 2021. 3
- [124] Zhuofan Zong, Kunchang Li, Guanglu Song, Yali Wang, Y. Qiao, Biao Leng, and Yu Liu. Self-slimmed vision transformer. In *European Conference on Computer Vision*, 2021. 7



Kunchang Li is currently a second-year Ph.D. student with Shenzhen Institutes of Advanced Technology (SIAT), Chinese Academy of Science. He received the B.Eng. degree from Beihang University, China, in 2020. His research interests focus on video understanding and efficient architecture design.



Yali Wang received the Ph.D. degree in computer science from Laval University, Quebec, QC, Canada, in 2014. He is currently an Associate Professor with the Shenzhen Institutes of Advanced Technology (SIAT), Chinese Academy of Sciences. His research interests are deep learning and computer vision, machine learning, and pattern recognition.



Junhao Zhang is currently a first-year Ph.D. student with the National University of Singapore, Singapore. He received the B.Eng. degree from Shandong University, China, in 2020. He was a Research Assistant with Shenzhen Institutes of Advanced Technology, Chinese Academy of Science. His research interests are deep learning, computer vision, and robotics.



Yu Qiao (Senior Member, IEEE) is a professor with the Shenzhen Institutes of Advanced Technology (SIAT), the Chinese Academy of Science and Shanghai AI Laboratory. His research interests include computer vision, deep learning, and bioinformation. He has published more than 240 papers in international journals and conferences, including T-PAMI, IJCV, T-IP, T-SP, CVPR, ICCV etc. His H-index is 69, with 31,000 citations in Google scholar. He is a recipient of the distinguished paper award in AAAI 2021. His group achieved the first runner-up at the ImageNet Large Scale Visual Recognition Challenge 2015 in scene recognition, and the winner at the ActivityNet Large Scale Activity Recognition Challenge 2016 in video classification. He served as the program chair of IEEE ICIST 2014.



Peng Gao received his Ph.D. degree from Chinese University of Hong Kong in 2021. Currently, he is a Young Research Scientist at Shanghai AI Lab. His research interest span from efficient neural architecture design, multimodality learning and representation learning.



Guanglu Song is a senior researcher at SenseTime Research. He received a master's degree in Computer Science and Technology from Beihang University. His current research interests lie in computer vision, efficient architecture design, and large-scale model optimization. Several papers are accepted by ECCV, CVPR, ICLR, and AAAI. He won the championships in various famous world AI competitions such as OpenImage 2019, ActivityNet 2020, and ICCV2021-MFR.



tyNet 2020.

Yu Liu received his Ph.D. from the Multimedia Lab of CUHK and was the only awardee of the Google Ph.D. Fellowship in Greater China. He previously worked as a researcher in Microsoft Research, Google AI, and SenseTime Research. His research interests lie in large-scale machine learning and decision intelligence, where he published more than 30 papers with around 2000 citations. He won the championships in various famous world AI competitions such as ImageNet 2016, MOT 2016, OpenImage 2019, and Activi-



Hongsheng Li received the bachelor's degree in automation from the East China University of Science and Technology, and the master's and doctorate degrees in computer science from Lehigh University, Pennsylvania, in 2006, 2010, and 2012, respectively. He is currently an assistant professor in the Department of Electronic Engineering at The Chinese University of Hong Kong. His research interests include computer vision, medical image analysis, and machine learning.