IA-RED^2: Interpretability-Aware Redundancy Reduction for Vision Transformers

Anonymous Author(s)
Affiliation
Address
email

Abstract

The self-attention-based model, transformer, is recently becoming the leading backbone in the field of computer vision. In spite of the impressive success made by transformers in a variety of vision tasks, it still suffers from heavy computation and intensive memory cost. To address this limitation, this paper presents an Interpretability-Aware REDundancy REDuction framework (IA-RED^2). We start by observing a large amount of redundant computation, mainly spent on uncorrelated input patches, and then introduce an interpretable module to dynamically and gracefully drop these redundant patches. This novel framework is then extended to a hierarchical structure, where uncorrelated tokens at different stages are gradually removed, resulting in a considerable shrinkage of computational cost. We include extensive experiments on both image and video tasks, where our method could deliver up to **1.4$\times$ speed-up** for state-of-the-art models like DeiT [47] and TimeSformer [3], by only sacrificing **less than 0.7%** accuracy. More importantly, contrary to other acceleration approaches, our method is **inherently interpretable** with substantial visual evidence, making vision transformer closer to a more human-understandable architecture while being lighter. We demonstrate that the interpretability that naturally emerged in our framework can outperform the raw attention learned by the original visual transformer, as well as those generated by off-the-shelf interpretation methods, with both qualitative and quantitative results.

1 Introduction

Transformer, a self-attention-based architecture processing sequential input without any recurrent or convolutional operations, has set off a storm in the computer vision literature recently. By dividing the input image into a series of patches and then tokenizing them with linear transformation, the transformer can effectively process the visual data in different modalities [12, 47, 48, 27, 3, 16, 59]. Despite its versatility, the transformer is always deeply troubled with inefficient computation and its vague interpretability. The vision transformer suffers heavy computational costs especially when the input sequence is long. As the attention module in the vision transformer computes the fully-connected relations among all of the input patches, the computational cost is then quadratic with regard to the length of the input sequence. On the other hand, previous works [6, 8] have already shown the vulnerable interpretability of the original vision transformer, where the raw attention coming from the architecture sometimes fails to perceive the informative region of the input images.

Recently, more designs of vision transformer architecture [33, 58, 17, 49, 13, 9, 3] are proposed to get higher accuracy with less computational cost. Although these methods anchor good trade-offs between efficiency and accuracy, their compression makes the vision transformer even more lack interpretability. Most of these methods assume that the input sequences are sampled from a regular visual input in a fixed shape rule, and thus their network architectures are not flexible as well, which makes the vision transformer (1) no longer able to process the input sequence with arbitrary length as the architecture is designed for a specific input shape; (2) neither model-agnostic nor task agnostic

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Figure 1: Two examples of redundancy reduction in vision transformers. Our proposed multi-head interpreters serve as a model-agnostic module which are built on top of the existing transformer-based backbones for different tasks, including image recognition and video action recognition.

anymore; or (3) neglect the fact that the model redundancy is also input-dependant. We yet argue that there is **no inherent tension** between efficiency and interpretability, and achieving them both does **not** have to pay design flexibility as a price. Indeed, starting from the philosophy of *Occam's razor*, the law of parsimony, or always pursuing more compact solutions when possible, is always treated as a rule-of-thumb for pursing interpretability, especially in complicated fitting problems [20].

This paper aims to seek the win-win between both efficiency and interpretability while keeping the flexibility and versatility of the original vision transformer. We propose a novel Interpretability-Aware REDundancy REDuction (IA-RED^2) framework for reducing the redundancy of vision transformers. The key mechanism that IA-RED^2 uses to increase efficiency is to dynamically drop some less informative patches in the original input sequence so that the length of the input sequence could be reduced. While the original vision transformer tokenizes all of the input patches, it neglects the fact that some of the input patches are redundant and such redundancy is input-dependant (see from Figure 1). As the computational complexity of the attention module is quadratically linear to the input sequence length, the effect of reducing input sequence length would be magnified in the amount of the computation. Motivated by this, we leverage the idea of dynamic inference [38, 36, 37, 54, 50], and adopt a policy network (referred to as *multi-head interpreter*) to decide which patches are uninformative and then discard them. Our proposed method is inherently interpretability-aware as the policy network learns to discriminate which region is crucial for the final prediction results.

To summarize, the main contributions of our work includes: (1) We propose the IA-RED^2 framework, the first **interpretability-aware** redundancy reduction framework for vision transformer. (2) Our IA-RED^2 framework is the **first input-dependent** dynamic inference framework for the vision transformer, which adaptively decides the patch tokens to compute per input instance. (3) Our IA-RED^2 framework is both **model-agnostic and task-agnostic**. We conduct experiments with our IA-RED^2 framework spanning different tasks, including image recognition and action recognition, and different models, including DeiT [47], TimeSformer [3]. (4) We attain **promising interpretable results** (shown in Figure 3) over baselines, with a 1.4$\times$ acceleration over DeiT on image recognition tasks, and a 4$\times$ acceleration over TimeSformer on video action recognition task while largely maintaining the accuracy. We also provide both qualitative results regarding interpretability with heatmaps by our method and those from other baseline methods like raw attention, MemNet [28]; as well as the quantitative comparison with current state-of-the-art model interpretability methods, such as GradCAM [40], on ImageNet-Segmentation [15] dataset with the weakly-supervised image segmentation task.

2 Related Work

Interpretability of Neural Networks. Deep networks has gained tremendous success on both computer vision and natural language processing tasks. However, the end-to-end training strategy makes it a black box to understand. Besides improving the discrimination power of neural networks, model interpretability raised another significant and popular research question. One of the important goals is to predict the heatmap visualization that precisely indicates the objects or contexts of relevance. Simonyan *et. al.* [41] attempts to maximize the class score that generate a saliency map for the given inputs. Dabkowski *et al.* [11] mask the salient parts of the inputs to manipulate the scores of the classifier, which generalises well to unseen images and enables fast saliency detection. Khosla *et al.* [28] provide the largest annotated image memorability dataset to benchmark the visualization and explanation of natural images. After that, gradient-based methods [45, 42, 43] are proposed to generate precise heatmaps, by computing the gradient with respect to input during the backpropagation. While all the above approaches are studying the interpretability of convolutional

neural networks (CNNs), only a few works contribute to the visualization of vision transformer. Chefer *et al.* shed some light on its visualization by assigning the local relevance on Transformer layers. Caron *et al.* [6] demonstrate that a self-supervised trained ViT produces explicit representation about the semantic location of a given object in natural images. Different from all of them, our approach starts with a novel multi-head interpreter which is supervised by an efficiency-driven signal, and then benefits from this powerful interpreter by reducing the redundancy of transformer, achieving a "win-win" between interpretability and efficiency.

Dynamic Networks. Neural networks are found as redundant regarding its huge computation cost [18, 22, 34]. To overcame this issue, many adaptive computation methods are explored during the inference stage [1, 2, 50, 14, 25, 51, 19]. These adaptive computation strategies help speed up the inference time of convolutional neural networks (CNNs) [32], recurrent neural network (RNNs) [14], and also self-attention based methods (BERT) [24]. Besides the model-level adaptation, others further extend this idea to data-level adaptation, by either reducing the spatial redundancy [56] or focusing on key area [52]. However, those methods are limited by the convolutional structure, where only 2D data can be taken as input. Different from those approaches, our methods naturally benefit from the unstructured input taken by vision transformer, and thus can provide a much more precise glance on target object with the affection of background being eliminated.

Vision Transformer. Transformer, as a self-attention based model, has been widely adopted in natural language processing area before. The recent advance [12] shows that the transformer can also achieve incredible performance on computer vision tasks. While vision transformer suffers from the necessity large-scaled dataset [44], many recent works try to encode strong inductive prior by either combining it with convolutional layer [53, 31, 57, 55] or introducing 2D-hierarchical structure to vision transformer [33, 49, 13, 9]. Besides, transformer also shows strong power in other vision tasks, including semantic segmentation [60], object detection [5, 61], image processing [10], and image generation [27, 26]. These successes further suggest the potential of transformer to become the universal model for general vision tasks.

3 Proposed Method

Our main goal is to reduce the redundancy in vision transformers by dynamically dropping less informative patches in the original input sequence while classifying it correctly with the minimum computation. Our method is built on top of vision transformer (ViT) [12], so we first present a brief overview of ViT including the computational complexity of each module regarding the input sequence length, and then describe our proposed IA-RED^2 framework for hierarchically reducing the redundant patch tokens at different layers of the vision transformer.

3.1 Overview of Vision Transformer

Vision transformer mainly consists of three main modules as follows: `(1) Multi-head Self Attention layer:` ViT uses MSA to learn relationships between every two different patches among all the input tokens. There are h self-attention heads inside the MSA. In each self-attention head, the input token X_i is first projected to a query Q_i, a key K_i, and a value V_i by three different linear transformations. Then, the query Q_i computes the dot products with all the keys K and these dot products will be scaled and normalized by the softmax layer to get the attention weights. After that, it outputs the token Y_i by weighted sum up all the values V with the obtained attention weights. Finally, the outputs from all heads are concatenated and re-projected by a linear layer into an output token. `(2) Feed-Forward Network:` The FFN consists of two linear layers which are connected by the GeLU activation [23] function. For each output token $Y_i \in R^D$ from the precedent MSA layer, FFN processes it individually. The first linear layer upgrades its dimension from D to $4D$, and the second linear layer downgrades its dimension from $4D$ to D. Both MSA and FFN are functioning as residual connection [21]. `(3) Linear Patch Embedding and Positional Encoding:` For an image or a video clip, ViT first splits it into several fixed-size patches and embeds them into input tokens with a linear layer. After transforming the original image and video into a series of tokens, the network is no longer capable to be aware of the positional information of the input tokens. Thus the positional embeddings are added to the input tokens right after the patch embedding to learn the positional information of each token.

Computational Complexity. For an input sequence $N \times D$, where N is the length of the input sequence and D is the embedding dimension of each input token. The computation complexity of the MSA is $\mathcal{O}(4ND^2 + 2N^2D)$. While for the FFN, the computational complexity is $\mathcal{O}(8ND^2)$. As

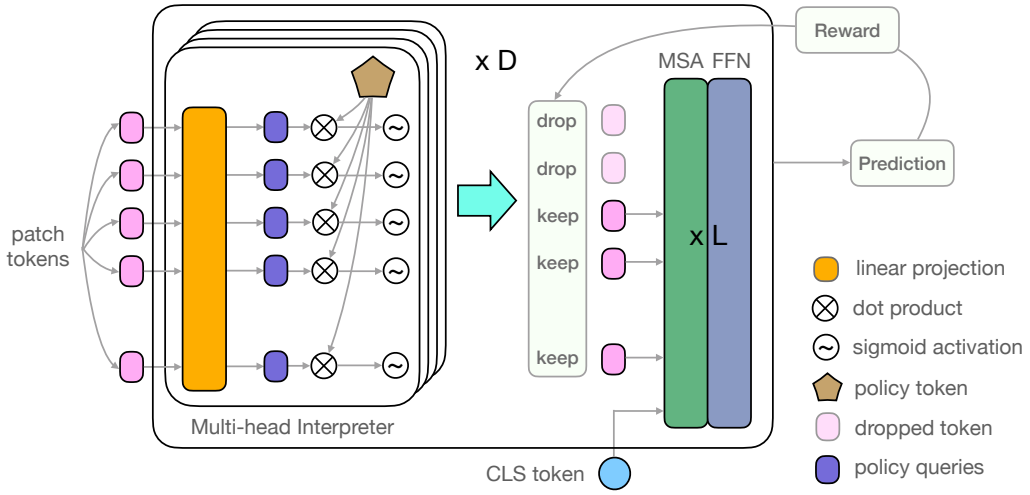

Figure 2: Illustration of our IA-RED^2 framework. We divide the transformer into D groups. Each group contains a multi-head interpreter and L combinations of the MSA and FFN. Before input to the MSA and FFN, the patch tokens will be evaluated by the multi-head interpreter to drop some uninformative patches. The multi-head interpreters are optimized by reward considering both the efficiency and accuracy.

the computational complexity of patch embedding can be neglected compared with the MSA and FFN, the total computational complexity of the ViT is $\mathcal{O}(12ND^2 + 2N^2D)$.

3.2 Interpretability-Aware Redundancy Reduction

In this section, we introduce our multi-head interpreter in detail which uses a policy token to estimate the importance of the input token. We also demonstrate how we hierarchically train the multi-head interpreter based on a pre-trained vision transformer. Finally, we illustrate that how the interpretability emerges in our IA-RED^2 framework.

Multi-head Interpreter. We borrow the idea from the architecture of the MSA layer to devise our policy module, named multi-head interpreter. Given a sequence of patch tokens $X \in R^{N \times d}$ which already contain the positional information, we drop the uninformative patch tokens by using the multi-head interpreter. We first divide the original ViT evenly into D groups, each group contains a multi-head interpreter and L blocks which consists of one MSA layer and one FFN. Inside each group, before inputting to the blocks, the patch tokens will first be evaluated by the multi-head interpreter for the informative score I_{ij}, where i and j represent the position of the input token and the group respectively. If I_{ij} is below the threshold 0.5, the patch X_i will be completely discarded at j^{th} group and will not be available in the subsequent groups. The I_{ij} is obtained by:

$$I_{ij} = \frac{1}{H} \sum_h \phi(\mathcal{F}_q^h(X_i) * \mathcal{F}_k^h(P_j)), \tag{1}$$

where P_j is the policy token in the jth multi-head interpreter, H is the number of the heads in the multi-head interpreter, $\mathcal{F}_q^h$ and $\mathcal{F}_k^h$ are the linear layer at h^{th} head for the patch tokens and the policy token respectively, $*$ represents the dot product and ϕ the sigmoid activation function.

Hierarchical Training Scheme. Our hierarchical training scheme is built on top of a well-trained ViT. In our IA-RED^2 framework, the ViT is divided into D groups, where each group contains one multi-head interpreter and L MSA-FFN blocks. We fix the parameters of the patch embedding layer, positional encoding, and the class token during the training, and only focus on the parameters inside each group. The network groups are optimized separately from the group right after the patch embedding layer to the group right before the prediction head. Intuitively, we hope the interpreter at the early stage could learn to select the patches containing all of the necessary contextual information for the correct final prediction, while the interpreter at later stages could focus more on the part-level information since now each token's information has already gone through global interaction and fusion. We optimize the multi-head interpreters by using the REINFORCE method where the

reward considers both the efficiency and accuracy, and finetune the MSA-FFN blocks with gradients computed based on cross-entropy loss.

Formally, during the training phase, given a sequence of patch tokens $X \in R^{N \times d}$ input to the jth multi-head interpreter, the multi-head interpreter will generate policies for each input token of dropping or keeping it as Bernoulli distribution by: $\pi_W(u_i|X_i) = I_{ij}^{u_i} * (1 - I_{ij})^{1-u_i}$, where $u_i = 1$ means to keep the token and $u_i = 0$ means to discard the token, I_{ij} is defined in the Eq. 1 and X_i denotes the ith token in the token sequence X. We associate these actions with the reward function:

$$R(u) = \begin{cases} 1 - (\frac{|u|_0}{N})^2 & \text{if correct} \\ -\tau & \text{otherwise} \end{cases}, \tag{2}$$

where $(\frac{|u|_0}{N})^2$ measures the percentage of the patches kept, and τ is the value of penalty for the error prediction which controls the trade-off between the efficiency and the accuracy of the network. This reward function encourages the multi-head interpreter to predict the correct results with as few patch tokens as possible. Then we optimize the multi-head interpreter individually by the expected gradient:

$$\nabla_{W_j} J = E_{u \sim \pi}[A \nabla_{W_j} \sum_{i=1}^{N} log[I_{ij} u_i + (1 - I_{ij})(1 - u_i)]], \quad A = R(u) - R(\hat{u}), \tag{3}$$

where $J = E_{u \sim \pi}[R(u)]$ is the expected reward to compute the policy gradient [46], W_j denotes the parameters of the jth multi-head interpreter. We use the self-critical baseline $R(\hat{u})$ in [39] to reduce the variance of optimization, where $\hat{u}$ denotes the maximally probable configuration under the current policy: i.e., $u_i = 1$ if $I_{ij} > 0.5$, and $u_i = 0$ otherwise. As the computation of the jth multi-head interpreter is based on the output tokens of $(j-1)^{th}$ group, we optimize the entire network in a curriculum learning manner. We first train the interpreter in the earlier layer, and then fix the interpreter and finetune all of the subsequent MSA-FFN blocks. Let's take the jth group for example. For the jth group, we first only train the multi-head interpreter and then fix it while optimizing the subsequent MSA-FFN modules in the jth, ... , D^{th} groups. When we optimize the jth group, the multi-head interpreter in the latter groups will be masked and keep all of the tokens.

Emergence of Interpretability. By visualizing the informative scores predicted by the multi-head interpreters in different network groups, we can see the redundancy of the input patches is hierarchically reduced at different levels clearly. For those patches that are removed in the precedent groups, we treat the informative score of them as zero. Thus we can obtain a sequence of the informative scores from each network group whose length equals the original input sequence length. We rearrange this score sequence and interpolate it back to the size of the input vision content (e.g. image or video). As the range of the informative score is from 0 to 1, we can draw a heatmap for each network group which interprets that what is redundant for this network group.

4 Experiments

Datasets and Metrics. We conduct image recognition experiments on the ImageNet-1k classification dataset [30]. The performance of our models on ImageNet-1k is measured with the metrics of top-1 and top-5 accuracy rates. For weakly-supervised image segmentation experiments, we adopt the ImageNet-Segmentation dataset [15] to evaluate the heatmaps we generate. We report three metrics: pixel accuracy, mean accuracy (mAcc), and mean IoU (mIoU) to reflect the segmentation performance. Finally, for video action recognition, we conduct our experiments on Kinetics-400 dataset [7], which contains 240k training videos and 10K videos for testing across 400 classes. We report the metrics of clip-1 and video-1 error of video models, which denotes the error rate of evaluating the model with the single clip and the Left-Center-Right three clips respectively.

Model Architectures. We build our image model on top of DeiT [47] which adopts the architecture of the ViT [12] by modifying the depth and width. Comparing to the original ViT [12], DeiT has a distillation token that is in charge of distilling the knowledge from the teacher CNN network. DeiT is trained and evaluated on ImageNet-1k [30], without large-scale pre-training. We choose DeiT-S and DeiT-B as our base models, where DeiT-B is 4× larger than DeiT-S in terms of FLOPs. For the video model, we construct our model based on TimeSformer [3]. There are several different attention mechanisms introduced in [3]. Here we adopt the TimeSformer with the JointST attention method,

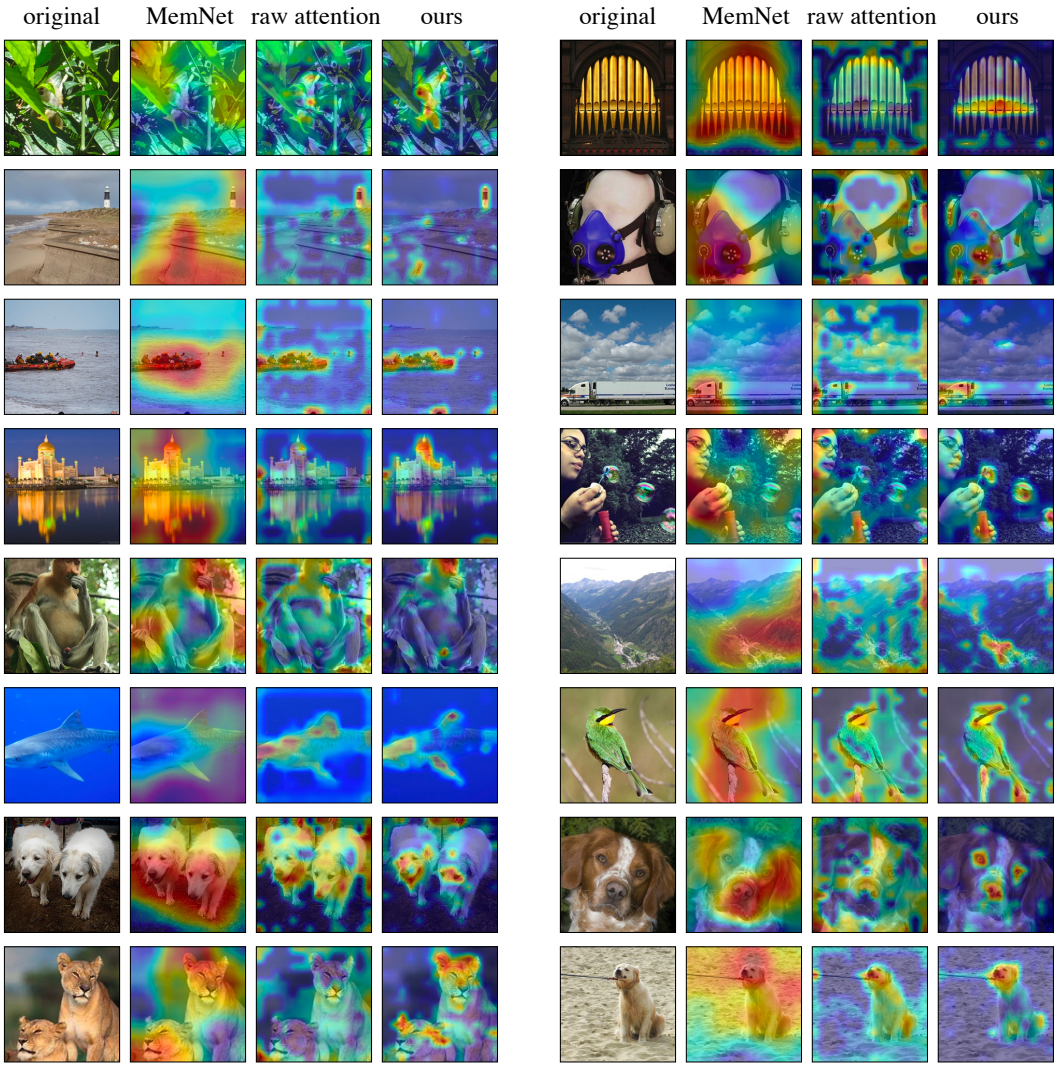

Figure 3: We visualize the heatmaps which hightlight the informative region of the input images of MemNet, raw attention at the second block, and our method with DeiT-S model. We find that our method can obviously better interpret the part-level stuff of the objects of interest.

which keeps the architecture of the vanilla ViT and takes all of the input patches as one sequence. Our model samples 8 frames in one video clip and splits them into 1568 frame patches. During inference, our model evenly crops 3 views from the video clip, each view of them has 8 frames.

Implementation Details. For the image recognition task, we divide the vision transformer backbone [47] into 3 ($D = 3$) groups, where each group contains 4 ($L = 4$) MSA-FFN modules and one multi-head interpreter. We optimize the entire framework for $D \times 30$ epochs. During every 30 epochs, we optimize the multi-head interpreter for 10 epochs and all of the subsequent MSA-FFN modules for 20 epochs. We use a mini-batch size of 32 images per GPU and adopt Adam [29] optimizer with an initial learning rate of 4e-5, which decays by cosine strategy [35] to train all our models. For the video understanding task, we set $D = 1$, i.e., we only select the informative patches at the input level. And we train the multi-head interpreter for 5 epochs and then finetune the backbone network for 1 epoch mainly following the settings listed in the original paper [3]. We use a mini-batch size of 8 video clips per GPU and adopt an SGD optimizer with an initial learning rate of 2.5e-3 in cosine strategy [35]. We train most of our models using 16 NVIDIA Tesla V100-32GB GPUs.

4.1 Emergence of Interpretability

In this section, we demonstrate the interpretability that emerges in our proposed method. We first show some qualitative results to show our method can better interpret what is the informative region for the correct prediction of classification. Then we provide the quantitative results of our weakly-

Table 1: Results of weakly-supervised image segmentation on ImageNet-segmentation [15]. We use our method based on the training with DeiT-S model. Higher is better.

Metrics	raw attention	MemNet [28]	GradCAM [40]	LRP [4]	Ours
pixel accuracy	67.87	52.81	65.91	50.72	**70.36**
mAcc	61.77	53.70	55.04	50.62	**64.86**
mIoU	46.37	34.66	41.31	32.62	**49.42**

Table 2: Redundancy reduction results of our IA-RED^2 with DeiT on ImageNet-1k [30].

Arch.	Method	speed (fps)	Top-1	Top-5	Arch.	Method	speed (fps)	Top-1	Top-5
DeiT-S	*original*	*≤930*	*79.8*	*95.0*	DeiT-B	*original*	*≤320*	*81.8*	*95.6*
	random	≥1360	78.4	94.2		random	≥440	80.2	94.6
	MemNet	≤350	77.6	93.6		MemNet	≤190	79.9	94.5
	attention	≥1360	78.4	94.1		attention	≥440	80.6	94.8
	ours	≥1360	**79.1**	**94.5**		ours	≥440	**80.9**	**95.0**

supervised image segmentation experiments on ImageNet which demonstrated that our method can better localize the salient object on the input images.

Qualitative Evaluation. We visualize the output of the multi-head interpreter in the second network group to be our results, where we choose the DeiT-S model to be the testbed. Then, we compare our method with other two baselines: (1) *Memorability map*, generated by MemNet [28], models how memorable is a certain region of the image with a concrete score ranging from zero to one. Intuitively, a memorability map highlights the region which stimulates our brains more dramatically. (2) *Raw attention*, coming inherently with the pre-trained vision transformer, highlights the regions on the image with more significant attention weight. We generate the raw attention map by extracting the attention weight at the multi-head self attention module in the `block 1`. We demonstrate the comparison of our method with the two baselines in Figure 3, from which we can see that our proposed method localizes the objects of interest, especially the part-level stuff, more accurately. From the third example in the second column, we can see that the heatmap generates by our method combines the pattern of memorability map, which detects the truck head while the attention map highlights more irrelevant regions, like the cloud in the sky. In the example of the shark at the sixth row of the first column, our method accurately localizes the fin and the head, the two most informative features of the shark, which indicates that our method can better interpret the part-level stuff comparing to the raw attention. Also, examples of dogs and tigers at the seventh and eighth rows demonstrate that our method can detect the features on the animal face, like eyes, noses, and ears.

Weakly-supervised Image Segmentation. To quantitatively compare our method with other model interpretability methods, we conduct the weakly-supervised image segmentation experiments on the ImageNet-Segmentation [15] dataset. Besides the memorability map and the raw attention map, we add another two baselines: gradient propagation method GradCAM [40] and Layer-wise Relevance Propagation method LRP [4]. The goal of weakly-supervised ImageNet-Segmentation task is to predict a precise mask of the objects of interest without pixel-level supervision, where a binary mask served as the ground-truth label. We list the segmentation results in Table 1, from which we observe that the proposed method IA-RED^2 significantly outperforms other methods. As a dedicated method for the interpretability of CNNs, we find that GradCAM does not perform well in this task. We guess this is because there is a significant gap between the interpretability of CNNs and vision transformers.

4.2 Redundancy Reduction on Recognition Tasks

In this section, we conduct our experiments on top of the image recognition task and video action recognition task. We demonstrate that our method can hierarchically reduce the redundancy of the input patch tokens with both qualitative results and quantitative results.

Results of Image Recognition. We try several different methods of dropping the input patch tokens, including our methods, to see how it harms the performance of the original network. The baselines we use are (1) *random baseline*, which randomly drops the path tokens at the input level, (2) *MemNet baseline*, which drops the patch tokens based on the memorability score of the corresponding patch, (3) *attention baseline*, which drops the patch tokens based on the raw attention map in `block 1`. For the baseline methods, we adjust the threshold to set the drop-out rate as 30%, thus there would 30%

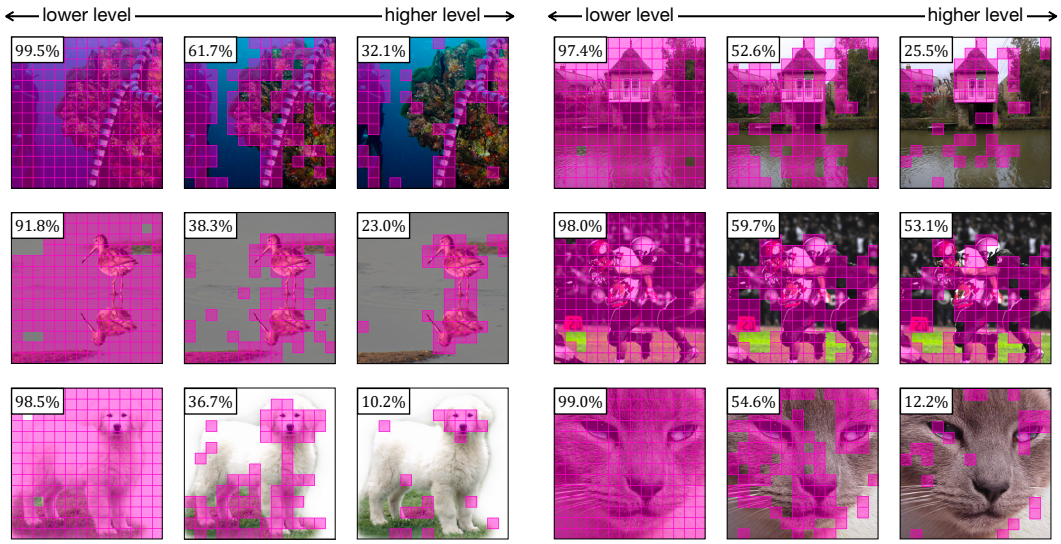

Figure 4: We visualize the hierarchical redundancy reduction process of our method with the DeiT-S model. The number on the upper-left corner of each image indicates the ratio of the remaining patches. From left to right, we can see that the network drops the redundant patches and focuses more on the high-level features of the objects.

Table 3: Redundancy reduction results of our IA-RED^2 with the DeiT-Base in different resolutions. To fairly compare the ratio of the redundancy patches, we keep the parameters of MSA-FFN modules the same as the original and only optimize the multi-head interpreters.

resolution	method	speed (fps)	FLOPs$_{avg}$	Top-1	speedup ratio	gap
224×224	*original*	*316.8*	*16.8 G*	*81.8*	1.42×	1.5%
	ours	452.5	11.8 G	80.3		
384×384	*original*	*87.0*	*49.4 G*	*82.8*	1.48×	0.9%
	ours	129.6	34.7 G	81.9		

of the patch tokens be discarded right after the positional embedding layer. We choose the model of our method with the similar inference speed of (1), which would be the baseline with the fastest speed as it does not need any pre-process to the patch tokens, to fairly compare the performance. We test the inference speed in terms of frames per second (fps) of each method on a single NVIDIA Tesla V100-32GB GPU with PyTorch 1.7 and CUDA 10.2. We list our quantitative results in Table 2. In Figure 4, we visualize several examples of hierarchically dropping the patch tokens at different groups, where we can see that our model takes almost all of the tokens at the lower level of the network (the first group), then drops the patches on the background and focuses on the entire object of interest at the middle level (the second group). And finally, it focuses on the part-level stuff of the object at the higher level (the third group).

Results of Video Action Recognition. We further explore the redundancy reduction strategies on the video action recognition task. Similar to the image task, we compare our method with *(1) random* baseline and *(2) attention map* baseline. Besides these two, we devise the *(3) temporal difference* baseline, which calculates the L_2 distance between the patch tokens at T and $(T-1)$ time step. For those tokens that have a longer distance to the previous one, we assume they have larger entropy thus need to be kept. For the patch tokens sampled from the initial frame, we set their previous tokens as zero. We list the results in Table 4. We can see that our method outperforms the attention baseline while worse than the random and temporal difference baseline. Although our method does not get the best results of the four redundancy reduction methods, it learns to identify the informative patches among thousands of input patch tokens, which will be further illustrated in the Appendix. We guess the reason why the random baseline performs better is that the input redundancy of video is significantly higher than that in the image, which makes the model quite robust to random patch dropping as the similar technique is applied in the training process.

Table 4: Redundancy reduction results of our IA-RED^2 with TimeSformer on Kinetics-400 [7]. We compare three baseline methods with our IA-RED^2: *(1) random*, *(2) attention map*, and *(3) temporal difference*. The speed (clip per second) of each method is measure on a single NVIDIA TESLA V100-32GB GPU. For the performance, we report the clip-1 error and video-1 error, where lower is better. To be consistent, we report our reproduced results of the original TimeSformer as our Kinetics-400 dataset does not include all of the original data.

Method	speed	clip-1	video-1
original	*≤24.0*	*28.2*	*23.8*
random	≥81.0	34.3	28.2
attention	≥81.0	38.1	31.4
temp. diff.	≥81.0	**32.3**	**26.9**
ours	≥81.0	35.3	29.1

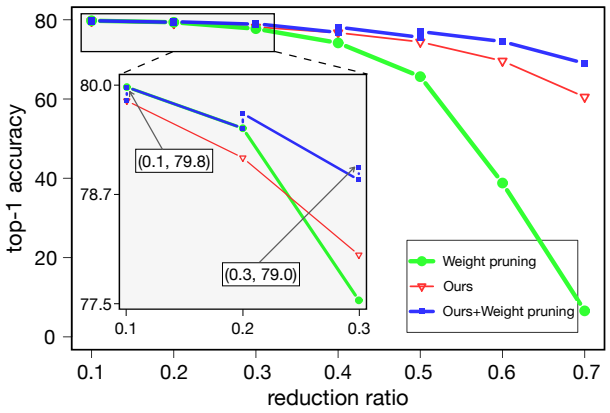

Figure 5: We plot the redundancy reduction results of weight pruning, our IA-RED^2, and the combination of them, where the X axis represents the FLOPs reduction ratio of all linear layers in the DeiT-S, and Y axis represents the top-1 accuracy on ImageNet. For a fair comparison, we do not finetune the network after we reduce the redundancy. With the combination of our IA-RED^2 and the weight pruning method, the model can be directly accelerated by 1.7× without finetuning while suffering only a 1.7% accuracy drop.

Discussion #1: Does the Higher Resolution has More Redundancy? From Figure 4, we can see the redundancy varies depending on the input instance: images with more background or small features which can identify the object tend to have more redundancy while the images with more complicated object always need more computation. Intuitively, as the redundancy is input-data-dependent, the input data with higher resolution would contain more redundancy. To validate this, we conduct the ablation study in Table 3, where we keep the reduction ratio of the computational cost the same and compare the accuracy loss of the models in different resolutions. We find that the model which takes higher resolution suffers a lower performance decrease than that with the lower resolution, which supports that input data in higher resolution tends to contain more redundancy.

Discussion #2: Is Data-level Redundancy Orthogonal to the Model-level? Contrary to those works [18, 22, 34] prune the model-level redundancy, our approach seeks to reduce the data-level redundancy. To further study these two counterparts, we start by choosing the magnitude-based weight pruning approach [34] as the subject. The pruning method is applied to all of the FC layers in the transformer. We first plot the trade-off curves between accuracy and efficiency of the weight pruning and our IA-RED^2 in Figure 5, where we can see that our IA-RED^2 outperforms the weight pruning especially when the FLOPs reduction ratio is high. Then we combine these two methods to see if they are complementary to each other: for each compression step, we choose the model with higher accuracy achieved by either increasing the weight pruning ratio or lifting the threshold of our multi-head interpreter to reduce more redundant tokens. We observe that the combined approach achieves the best trade-off, suggesting that the proposed data-level redundancy is orthogonal to the model-level redundancy, and our method is complementary to the weight pruning method.

5 Conclusions

In this work we propose a novel interpretability-aware redundancy reduction framework called IA-RED^2 for the recent vision transformer. We show that IA-RED^2 hierarchically reduces the computational cost and speeds up the vision transformer effectively with human-understandable trajectories. The experiments are conducted on image and video tasks to demonstrate that our IA-RED^2 is both model-agnostic and task-agnostic. We finally compare our IA-RED^2 with the model compression method, like weight pruning, to demonstrate the complementarity between them.

Broader Impact. Our work eases the suffering of heavy computational cost for vision transformer, which could save more energy and reduce the carbon emissions for the industry. The interpretability which emerges in our method help we human to understand what happening inside the vision transformer. However, the potential negative impact would be that, since our method makes neural networks easier to run and more understandable to everyone, it may cause the abuse of AI technology.

References

[1] Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. Conditional computation in neural networks for faster models. *arXiv preprint arXiv:1511.06297*, 2015.

[2] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[3] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? *arXiv preprint arXiv:2102.05095*, 2021.

[4] Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. Layer-wise relevance propagation for neural networks with local renormalization layers. In *International Conference on Artificial Neural Networks*, pages 63–71. Springer, 2016.

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

[7] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[8] Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. *arXiv preprint arXiv:2012.09838*, 2020.

[9] Chun-Fu Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. *arXiv preprint arXiv:2103.14899*, 2021.

[10] 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. *arXiv preprint arXiv:2012.00364*, 2020.

[11] Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. *arXiv preprint arXiv:1705.07857*, 2017.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[13] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. *arXiv preprint arXiv:2104.11227*, 2021.

[14] Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

[15] Matthieu Guillaumin, Daniel Küttel, and Vittorio Ferrari. Imagenet auto-annotation with segmentation propagation. *International Journal of Computer Vision*, 110(3):328–348, 2014.

[16] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *arXiv preprint arXiv:2012.09688*, 2020.

[17] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. *arXiv preprint arXiv:2103.00112*, 2021.

[18] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[19] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *arXiv preprint arXiv:2102.04906*, 2021.

[20] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical learning with sparsity: the lasso and generalizations*. Chapman and Hall/CRC, 2019.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[22] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1389–1397, 2017.

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

[24] Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. Dynabert: Dynamic bert with adaptive width and depth. *arXiv preprint arXiv:2004.04037*, 2020.

[25] Ting-Kuei Hu, Tianlong Chen, Haotao Wang, and Zhangyang Wang. Triple wins: Boosting accuracy, robustness and efficiency together by enabling input-adaptive inference. In *International Conference on Learning Representations*, 2020.

[26] Drew A Hudson and C Lawrence Zitnick. Generative adversarial transformers. *arXiv preprint arXiv:2103.01209*, 2021.

[27] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. Transgan: Two transformers can make one strong gan. *arXiv preprint arXiv:2102.07074*, 2021.

[28] Aditya Khosla, Akhil S Raju, Antonio Torralba, and Aude Oliva. Understanding and predicting image memorability at a large scale. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2390–2398, 2015.

[29] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[31] Changlin Li, Tao Tang, Guangrun Wang, Jiefeng Peng, Bing Wang, Xiaodan Liang, and Xiaojun Chang. Bossnas: Exploring hybrid cnn-transformers with block-wisely self-supervised neural architecture search. *arXiv preprint arXiv:2103.12424*, 2021.

[32] Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. Runtime neural pruning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 2178–2188, 2017.

[33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021.

[34] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017.

[35] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[36] Yue Meng, Chung-Ching Lin, Rameswar Panda, Prasanna Sattigeri, Leonid Karlinsky, Aude Oliva, Kate Saenko, and Rogerio Feris. Ar-net: Adaptive frame resolution for efficient action recognition. In *European Conference on Computer Vision*, pages 86–104. Springer, 2020.

[37] Yue Meng, Rameswar Panda, Chung-Ching Lin, Prasanna Sattigeri, Leonid Karlinsky, Kate Saenko, Aude Oliva, and Rogerio Feris. Adafuse: Adaptive temporal fusion network for efficient action recognition. *arXiv preprint arXiv:2102.05775*, 2021.

[38] Bowen Pan, Rameswar Panda, Camilo Fosco, Chung-Ching Lin, Alex Andonian, Yue Meng, Kate Saenko, Aude Oliva, and Rogerio Feris. Va-red^2: Video adaptive redundancy reduction. *arXiv preprint arXiv:2102.07887*, 2021.

[39] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7008–7024, 2017.

[40] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[41] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[42] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

[43] Suraj Srinivas and François Fleuret. Full-gradient representation for neural network visualization. *arXiv preprint arXiv:1905.00780*, 2019.

[44] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

[45] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.

[46] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[47] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

[48] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. *arXiv preprint arXiv:2103.17239*, 2021.

[49] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, 2021.

[50] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 409–424, 2018.

[51] Yue Wang, Jianghao Shen, Ting-Kuei Hu, Pengfei Xu, Tan Nguyen, Richard Baraniuk, Zhangyang Wang, and Yingyan Lin. Dual dynamic inference: Enabling more efficient, adaptive, and controllable deep inference. *IEEE Journal of Selected Topics in Signal Processing*, 14(4):623–633, 2020.

[52] Yulin Wang, Kangchen Lv, Rui Huang, Shiji Song, Le Yang, and Gao Huang. Glance and focus: a dynamic approach to reducing spatial redundancy in image classification. *arXiv preprint arXiv:2010.05300*, 2020.

[53] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021.

[54] Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S Davis, Kristen Grauman, and Rogerio Feris. Blockdrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8817–8826, 2018.

[55] Haotian Yan, Zhe Li, Weijian Li, Changhu Wang, Ming Wu, and Chuang Zhang. Contnet: Why not use convolution and transformer at the same time? *arXiv preprint arXiv:2104.13497*, 2021.

[56] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution adaptive networks for efficient inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2369–2378, 2020.

[57] Kun Yuan, Shaopeng Guo, Ziwei Liu, Aojun Zhou, Fengwei Yu, and Wei Wu. Incorporating convolution designs into visual transformers. *arXiv preprint arXiv:2103.11816*, 2021.

[58] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021.

[59] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. *arXiv preprint arXiv:2012.09164*, 2020.

[60] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. *arXiv preprint arXiv:2012.15840*, 2020.

[61] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] The main contributions are listed in the abstract
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Sec. 5
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] Yes, we have carefully read the guidelines and our paper conforms to them
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments (e.g. for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] In Sec. 4
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We have cited all the dataset.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] We use publicly available dataset without any offense to any group.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]