Mean-Shift Feature Transformer

Takumi Kobayashi[†‡]
[†]National Institute of Advanced Industrial Science and Technology, Japan
[‡]University of Tsukuba, Japan
takumi.kobayashi@aist.go.jp

Abstract

Transformer models developed in NLP make a great impact on computer vision fields, producing promising performance on various tasks. While multi-head attention, a characteristic mechanism of the transformer, attracts keen research interest such as for reducing computation cost, we analyze the transformer model from a viewpoint of feature transformation based on a distribution of input feature tokens. The analysis inspires us to derive a novel transformation method from mean-shift update which is an effective gradient ascent to seek a local mode of distinctive representation on the token distribution. We also present an efficient projection approach to reduce parameter size of linear projections constituting the proposed multi-head feature transformation. In the experiments on ImageNet-1K dataset, the proposed methods embedded into various network models exhibit favorable performance improvement in place of the transformer module. Codes are available at https://github.com/tk1980/MSFtransformer.

1. Introduction

After the renaissance of neural networks, deep models have been attracting keen attention in pattern recognition fields. While convolutional neural networks (CNNs) provide promising performance especially on computer vision tasks, transformer [44] revolutionizes the network architecture. The transformer originated from NLP efficiently encodes (causal) dependency in a sequence by means of attention mechanism replacing complicated recurrent networks [19]. It is now distributed across various recognition fields [16] such as for images [14] and videos [3].

Much research effort has been made to improve transformer networks. In computer vision, a vision transformer (ViT) [14] successfully encodes images by utilizing image-patch tokens. Transformer modules on the patch tokens are capable of extracting non-local visual features [47] beyond the local characteristics exploited by convolutions in CNN. On the other hand, convolutions are incorporated to exhibit synergy with the transformers [32, 36, 48, 49] such as for fusing global and local information as well as utilizing convolutional inductive bias that the patch-token approach misses [12]. The original transformer module is directly embedded even to those sophisticated networks due to its versatile applicability.

The transformer module itself is improved mainly in terms of attention which manifests a distinctive mechanism in the module. Attention is an important process for recognition as studied in neuroscience [28] and the transformer utilizes attention weights (map) among input tokens, the size of which is quadratic in the number of tokens; specifically, multiple attention weights are extracted by means of multi-head attention (MHA) in the transformer. Thus, there are plenty of works to provide efficient computation of attention such as by hashing [23], clustering [34], low-rank approximation [2, 8, 22, 46, 51] and IO-aware implementation [11]. The attention maps are also enhanced, e.g., by aggregating multiple maps [18, 62, 63] and exploring effective similarity (kernel) function [43, 58]. A transformer model leverages the attention weights to transforming input feature tokens toward discriminative feature representation.

In this paper, we focus on the process to *transform* feature vectors in the transformer. In contrast to the approaches which focus only on attention weights, we consider the whole transformation process that aggregates feature tokens in MHA to modify the feature representation via a residual connection. Thereby, rather apart from semantic concept of attention, we regard the transformer mechanism as a feature transformation to explore characteristic points on a probabilistic distribution composed of input feature tokens. Through the lens of the probabilistic density function, we can rewrite the transformer model into a gradient-ascent updating to seek the local maxima, *modes*, of the distribution. This analysis inspires us to derive a simple yet effective transformation from *mean-shift* [40] update of efficient mode seeking on the distribution. In the mean-shift framework utilizing a multi-head approach, linear projection from an input feature space into subspace representation is a key process. We also present an efficient grouped projec-

tion to reduce parameter sizes of the projection matrices while retaining performance. It is related to grouped convolution [26, 50, 56], and we further analyze the grouping for improving back-propagation in a multi-head framework. The proposed feature transformation works in place of the transformer module, thereby being applicable to the extensive methods that improve transformer-based networks mentioned in Section 1.1.

Our contributions are summarized as follows.

- We provide a novel viewpoint to analyze transformer mechanism by utilizing a probabilistic distribution of input feature tokens.
- Based on the analysis, we propose a mean-shift based feature transformation, *MSF-transformer*, as an extension of the transformer. An efficient grouped projection is also presented to enhance efficiency of the MSF-transformer.
- In the experiments on ImageNet dataset, the method is thoroughly evaluated from various aspects regarding feature transformation and exhibits favorable performance on various networks in place of a transformer module.

1.1. Related works

Vision Transformer. Transformer [44] first proposed in NLP makes a great impact on computer vision such as through vision transformer (ViT) [14] which is versatilely applied to diverse vision tasks [3, 14, 29, 60]; comprehensive survey is shown in [16]. In contrast to *words*, ingredients of NLP, images are not intrinsically tokenized, which encourages improving tokenization of simple patch partitioning [14] by Swin-Transformer [30] and T2T [52]. The transformer can be combined with CNNs to further improve the efficacy on vision tasks, such as by means of convolution stems [49], light-weight CNN model [32] and convolutional local dependency extraction [48]. ViTs are also thoroughly evaluated in comparison to CNNs under huge computation budget [35]. In [10], positional self-attention in transformers is shown to be connected to convolution, which induces a sophisticated approach to enhance positional encoding in the self-attention model [12]. Since ViTs are regarded as data-hunger models demanding large computation resources [14, 54], effective training approaches are explored in [37, 41, 42] and an optimizer is sophisticated to efficiently train ViTs [6, 15]. While ViTs are advanced from those various aspects, the original transformer module is directly applied in most models. In this paper, we improve the feature transformation mechanism of the transformer module through analysis about a token distribution.

Attention. Transformers are characterized by their attention mechanism which is embedded in multi-head attention (MHA) [44]. There are some works to improve the attention weights which are built on pair-wise token similarities. Multiple attention maps are fused to enhance the weights in the MHA framework [62, 63] and a residual connection [18]. Similarity functions to compute attention are explored by means of MLP [58] and kernel functions [43]. On the other hand, as the attention weights have time and memory complexity quadratic by the number of tokens, the computation issue is addressed in lots of works. To efficiently search similar pairs of tokens (query and key), clustering [34] and hashing [23] are incorporated in the process of attention computation. The attention weight (matrix) is efficiently described by low-rank approximation [2, 8, 22, 46, 51]. In [11], IO-aware approach is proposed to compute attention weights in a computationally efficient manner. While those approaches mainly focus on attention weights and their computation, we analyze the transformer from a viewpoint of *feature transformation*, not limited to the attention; the proposed method could incorporate those efficient attention computation techniques.

Mean Shift. In computer vision, mean-shift algorithm [7, 9] is mostly applied to clustering. Recently, it is leveraged to loss function to measure affinity between feature representation for self-supervised learning [24, 25], and is also applied to key-point detection on an attention map for instance segmentation [27]. In contrast, we focus on the updating form of mean shift and analyze the transformer mechanism through the lens of the mean-shift updating to show resemblance between those approaches.

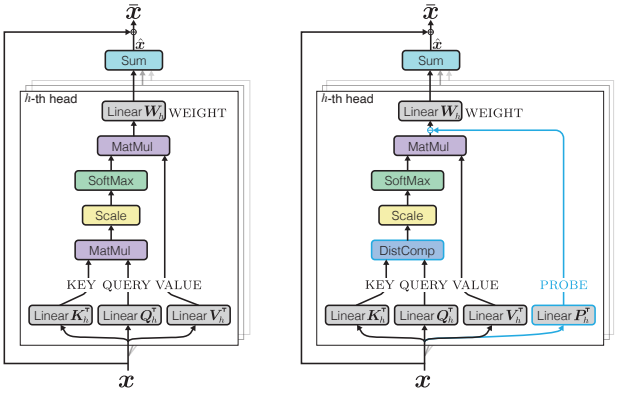
Grouped Convolution. To reduce parameter size of convolution kernels, grouped convolution is applied [26, 50] in CNN. We also employ a grouping approach to enhance parameter efficiency in linear projection of token features. It is similar to interleaved group convolution [56, 57] which stacks couples of group convolution and channel shuffling. We further analyze the effect of interleaved grouping on back-propagation in the multi-head framework.

2. Method

We begin with briefly reviewing a transformer module [14, 44] (Section 2.1), and then formulate the feature transformation framework using a token distribution (Section 2.2) to derive the proposed method, dubbed MSF-transformer (Section 2.2.2). Then, an efficient approach to linearly project features into the distribution is presented in Section 2.3.

2.1. Transformer

Suppose we have m tokens of d-dimensional feature vectors to form a matrix of $\mathcal{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_m] \in \mathbb{R}^{d\times m}$, and transform them by means of multi-head attention (MHA) with H heads [44]. As shown in Figure 1a, the h-th head is equipped with four types of projection matrices $\mathbb{R}^{d\times \hat{d}}$, $\{\boldsymbol{Q}_h, \boldsymbol{K}_h, \boldsymbol{V}_h, \boldsymbol{W}_h\}$. *Transformer* converts a feature vector



(a) Original transformer [44] (b) MSF-transformer

Figure 1. Transformer models.

$\boldsymbol{x}$ into $\bar{\boldsymbol{x}}$ by applying those projection matrices in MHA as

$$\hat{\boldsymbol{x}} = \sum_{h=1}^{H} \boldsymbol{W}_h \left[\boldsymbol{V}_h^\top \mathcal{X}\, \sigma(\mathcal{X}^\top \boldsymbol{K}_h \boldsymbol{Q}_h^\top \boldsymbol{x}) \right], \tag{1}$$

$$\bar{\boldsymbol{x}} = \boldsymbol{x} + \hat{\boldsymbol{x}}, \tag{2}$$

where σ is a softmax function processing m components with a temperature of $\hat{d}^{\frac{1}{2}}$ [44]. MHA (1) contains three ingredients of QUERY $\boldsymbol{Q}_h^\top \boldsymbol{x}$, KEY $\boldsymbol{K}_h^\top \boldsymbol{x}$ and VALUE $\boldsymbol{V}_h^\top \boldsymbol{x}$. Attention weights $\sigma(\mathcal{X}^\top \boldsymbol{K}_h \boldsymbol{Q}_h^\top \boldsymbol{x})$ are constructed by measuring similarity between QUERY and KEY via the softmax function σ in order to effectively aggregate VALUE. The attention weights are distributed over non-local features [47] in contrast to convolution, for improving feature representation. It should be noted that this attention process is performed in $\hat{d}$-dimensional subspaces through projection by $\boldsymbol{Q}_h, \boldsymbol{K}_h$ and $\boldsymbol{V}_h$ from the input d-dimensional features $\mathcal{X}$.

While most works focus on the attention mechanism composed of QUERY, KEY and VALUE, we explicitly incorporate both the subsequent linear projection by WEIGHT $\boldsymbol{W}_h$ in (1) and the residual connection (2) which accompany with MHA in the transformer model [44] (Figure 1a). Namely, we focus on the feature transformation process that an input $\boldsymbol{x}$ is projected into the $\hat{d}$-dimensional subspace via MHA and then back-projected into the d-dimensional input space by $\boldsymbol{W}_h$, followed by residual connection (2) to finally modify the feature representation. So transformed feature $\bar{\boldsymbol{x}}$ is fed into MLP and subsequent transformer layers [14, 44].

2.2. Distribution of input token features

We view the feature tokens from a probabilistic viewpoint. Suppose m samples (tokens) of $\hat{d}$-dimensional features, $\mathcal{K} = [\boldsymbol{k}_1, \cdots, \boldsymbol{k}_m] \in \mathbb{R}^{\hat{d} \times m}$, form a distribution. The distribution can be characterized by their *modes*, local maxima of the probability density function. For detecting the modes, a probe point $\boldsymbol{q} \in \mathbb{R}^{\hat{d}}$ explores the distribution by means of a gradient ascent of the probability $p(\boldsymbol{q})$ as

$$p(\boldsymbol{q}) = \frac{1}{Z} \sum_{i=1}^{m} \mathrm{g}(\boldsymbol{k}_i, \boldsymbol{q}), \tag{3}$$

$$\hat{\boldsymbol{q}} \triangleq \frac{d}{d\boldsymbol{q}} \log p(\boldsymbol{q}) = \frac{\sum_{i=1}^{m} \frac{d}{d\boldsymbol{q}} \mathrm{g}(\boldsymbol{k}_i, \boldsymbol{q})}{\sum_{i=1}^{m} \mathrm{g}(\boldsymbol{k}_i, \boldsymbol{q})}, \ \boldsymbol{q} \leftarrow \boldsymbol{q} + \eta \hat{\boldsymbol{q}}, \tag{4}$$

where g is a kernel function to build a kernel density estimation (KDE) [45] in (3) and η is a step size of gradient ascent.

We relate the probe point $\boldsymbol{q} \in \mathbb{R}^{\hat{d}}$ with a higher-dimensional representation $\boldsymbol{x} \in \mathbb{R}^{d}$ via projection $\boldsymbol{q} = \boldsymbol{Q}^\top \boldsymbol{x}$, thereby formulating the gradient w.r.t $\boldsymbol{x}$ in

$$\hat{\boldsymbol{x}} \triangleq \frac{d\boldsymbol{q}}{d\boldsymbol{x}} \frac{d}{d\boldsymbol{q}} \log p(\boldsymbol{q}) = \boldsymbol{Q}\hat{\boldsymbol{q}}. \tag{5}$$

By incorporating (4) into (5), the probe $\boldsymbol{x}$ is updated to $\bar{\boldsymbol{x}}$ by a gradient ascent of

$$\bar{\boldsymbol{x}} = \boldsymbol{x} + \eta \hat{\boldsymbol{x}} = \boldsymbol{x} + \eta \boldsymbol{Q} \left[\frac{\sum_{i=1}^{m} \frac{d}{d\boldsymbol{q}} \mathrm{g}(\boldsymbol{k}_i, \boldsymbol{q})}{\sum_{i=1}^{m} \mathrm{g}(\boldsymbol{k}_i, \boldsymbol{q})} \right]. \tag{6}$$

2.2.1 Connection to Transformer

A kernel function g can be flexibly designed in the KDE (3). One practical possibility is to set $\mathrm{g}(\boldsymbol{k}, \boldsymbol{q}) = \exp(\hat{d}^{-\frac{1}{2}} \boldsymbol{k}^\top \boldsymbol{q})$, though it produces *ill-defined* probability[1]; the gradient ascent (6) is written as

$$\bar{\boldsymbol{x}} = \boldsymbol{x} + \eta \hat{d}^{-\frac{1}{2}} \boldsymbol{Q} \frac{\sum_i \exp(\hat{d}^{-\frac{1}{2}} \boldsymbol{k}_i^\top \boldsymbol{q}) \boldsymbol{k}_i}{\sum_i \exp(\hat{d}^{-\frac{1}{2}} \boldsymbol{k}_i^\top \boldsymbol{q})} \tag{7}$$

$$= \boldsymbol{x} + \eta \hat{d}^{-\frac{1}{2}} \boldsymbol{Q} \left[\mathcal{K}\, \sigma(\mathcal{K}^\top \boldsymbol{Q}^\top \boldsymbol{x}) \right]. \tag{8}$$

As is the case with $\boldsymbol{q}$, we assume that samples $\mathcal{K}$ are also projections from the higher-dimensional features $\mathcal{X} \in \mathbb{R}^{d \times m}$ via $\mathcal{K} = \boldsymbol{K}^\top \mathcal{X}$, to further rewrite (8) into

$$\bar{\boldsymbol{x}} = \boldsymbol{x} + \eta \hat{d}^{-\frac{1}{2}} \boldsymbol{Q} \left[\boldsymbol{K}^\top \mathcal{X}\, \sigma(\mathcal{X}^\top \boldsymbol{K} \boldsymbol{Q}^\top \boldsymbol{x}) \right], \tag{9}$$

which is a gradient ascent toward the local mode around $\boldsymbol{x}$ in the *pseudo* probability density function of

$$p(\boldsymbol{x}) \propto \sum_{i=1}^{m} \exp(\hat{d}^{-\frac{1}{2}} \boldsymbol{x}_i^\top \boldsymbol{K} \boldsymbol{Q}^\top \boldsymbol{x}). \tag{10}$$

It should be noted that the update form (9) bears resemblance to the transformer (1, 2); only difference lies in the back-projection from $\hat{d}$-dimensional to d-dimensional space where (9) applies $\eta \hat{d}^{-\frac{1}{2}} \boldsymbol{Q}\boldsymbol{K}^\top$ instead of $\boldsymbol{W}\boldsymbol{V}^\top$ in (1). So, it can be said that the transformer (1) flexibly describes the projection matrices by breaking the ties among the matrices

[1] The normalization Z in (3) is not finite.

in (9) for effective feature representation learning. Thus, the transformer is naively interpreted as *feature update* that moves $\boldsymbol{x}$ toward its local mode of a representative and distinctive point on the distribution of input feature tokens.

This analysis also reveals an issue that the transformer (1) might contain a bias toward increasing feature norm $\|\boldsymbol{x}\|_2$ through end-to-end learning since the pseudo probability (10) could be easily increased by enlarging the norm of a probe $\boldsymbol{x}$. Such a bias to increasing feature norm is derived from the ill-posed kernel function $\mathrm{g}(\boldsymbol{k},\boldsymbol{q}) = \exp(\hat{d}^{-\frac{1}{2}}\,\boldsymbol{k}^\top\boldsymbol{q})$, though the bias can be practically mitigated by normalization techniques, such as LayerNorm [1], embedded in the networks [14, 44].

2.2.2 Mean-shift update as Transformer

Following a standard KDE [9], we apply a Gaussian kernel $\mathrm{g}(\boldsymbol{k},\boldsymbol{q}) = \exp(-\frac{1}{2}\hat{d}^{-\frac{1}{2}}\|\boldsymbol{k}-\boldsymbol{q}\|_2^2)$ to formulate the gradient ascent (6) in

$$\bar{\boldsymbol{x}} = \boldsymbol{x} + \eta\hat{d}^{-\frac{1}{2}}\boldsymbol{Q}\Big[\boldsymbol{K}^\top\mathcal{X}\,\sigma\big(\{-\tfrac{1}{2}\|\boldsymbol{K}^\top\boldsymbol{x}_i-\boldsymbol{Q}^\top\boldsymbol{x}\|_2^2\}_{i=1}^m\big) - \boldsymbol{Q}^\top\boldsymbol{x}\Big], \tag{11}$$

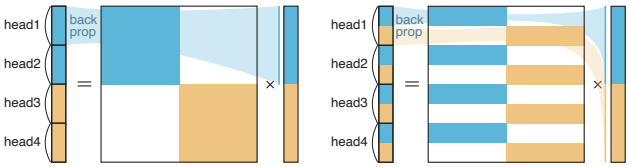
where the softmax function σ produces an m-dimensional vector. This is a mean-shift update [7, 9]. As KDE using the Gaussian kernel provides well defined probability density, the mean-shift update (11) effectively moves $\boldsymbol{x}$ toward its local mode as shown in [9], in contrast to the case of *ill*-posed kernel function $\exp(\hat{d}^{-\frac{1}{2}}\,\boldsymbol{q}^\top\boldsymbol{k})$ in the original transformer (Section 2.2.1); the probability density $p(\boldsymbol{x}) = \sum_{i=1}^m \exp(-\frac{1}{2}\hat{d}^{-\frac{1}{2}}\|\boldsymbol{K}^\top\boldsymbol{x}_i - \boldsymbol{Q}^\top\boldsymbol{x}\|_2^2)$ not necessarily favors a probe point $\boldsymbol{x}$ of larger norm. Other than the kernel function, (11) is different from (9) in that *differential* is computed by subtracting the probe vector $\boldsymbol{Q}^\top\boldsymbol{x}$. This effectively orients the update toward local maxima without bias for feature norm.

Following the relaxation from the gradient ascent (9) to the transformer (1), we break ties among projection matrices in (11) to propose mean-shift feature (MSF) transformer of

$$\bar{\boldsymbol{x}} = \boldsymbol{x} + \sum_{h=1}^{H}\boldsymbol{W}_h\Big[\boldsymbol{V}_h^\top\mathcal{X}\,\sigma\big(\{-\frac{1}{2}\|\boldsymbol{K}_h^\top\boldsymbol{x}_i-\boldsymbol{Q}_h^\top\boldsymbol{x}\|_2^2\}_{i=1}^m\big) - \boldsymbol{P}_h^\top\boldsymbol{x}\Big], \tag{12}$$

where $\boldsymbol{P}_h \in \mathbb{R}^{d\times\hat{d}}$ is additionally introduced to represent PROBE. The architecture of the MSF-transformer is depicted in Figure 1b which modifies the original transformer (Figure 1a) in simple yet effective ways regarding PROBE and Gaussian kernel function[2] based on the above analysis.

[2]The softmax $\sigma(\{-\frac{1}{2}\|\boldsymbol{K}_h^\top\boldsymbol{x}_i-\boldsymbol{Q}_h^\top\boldsymbol{x}\|_2^2\}_{i=1}^m)$ is efficiently computed with a negligible overhead compared to the standard one $\sigma(\mathcal{X}^\top\boldsymbol{K}_h\boldsymbol{Q}_h^\top\boldsymbol{x})$ as shown in supplementary material.



(a) Block-wise grouping (b) Interleaved grouping

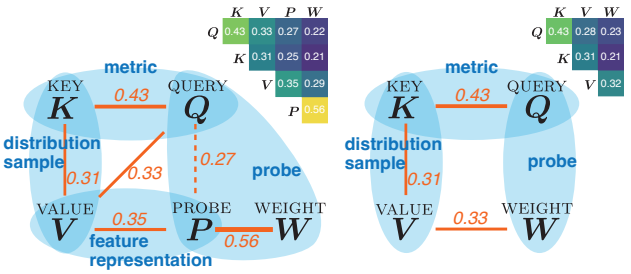
Figure 2. Grouped projection with multiple heads. These are schematic diagrams of 2-grouped projection by matrix-vector multiplication of, e.g., $[\boldsymbol{q}_1^\top,\dots,\boldsymbol{q}_H^\top]^\top = [\boldsymbol{Q}_1,\dots,\boldsymbol{Q}_H]^\top\boldsymbol{x}$ with $H=4$.

2.3. Grouped projection

It is known that KDE and mean-shift update degenerate in a high-dimensional feature space [33]. In transformer models (Figure 1), the issue of high dimensionality is mitigated by projecting high-dimensional feature $\boldsymbol{x}$ into lower $\hat{d}$-dimensional space so that the updating in (1, 12) works effectively. The MHA approach in (1, 12) constructs multiple $\hat{d}$-dimensional subspaces rendering diverse feature distributions to encode various (visual) characteristics at local modes for enhancing feature representation. Those subspace representations are back-projected into the original space via WEIGHT $\boldsymbol{W}_h$ and then merged as shown in Figure 1. While the multi-head projection plays an important role in the transformer models, it consumes considerable amount of memory (parameters); note that, for computation efficiency, projection matrices are concatenated to larger one, e.g., $\boldsymbol{Q} = [\boldsymbol{Q}_1,\cdots,\boldsymbol{Q}_H] \in \mathbb{R}^{d\times H\hat{d}}$, to enjoy efficient matrix multiplication.

Visual characteristics embedded in each projection are considered to be not distributed across whole d feature components in an input $\boldsymbol{x}$ but limited to a subset of features. To effectively exploit such localized features in MHA, we apply grouped linear projection in a similar way to the grouped convolution [26, 50]; as shown in Figure 2, G-grouped projection reduces parameter size of linear projection into $\frac{1}{G}$ and the number of groups is arbitrarily determined, independently of the number of heads H.

We analyze the grouped projection from a viewpoint of back-propagation. The ordinary grouping in Figure 2a produces a block-wise output in which one head is filled with one of groups. Thereby, each head is connected only to small portion of feature components, passing gradient information only to that part of features via back-propagation. This impedes end-to-end learning as each head in MHA contributes only to learning one subset (group) of features without inter-connection among groups in the back-propagation. Therefore, in a similar way to [56, 57], we apply *interleaved* grouped projection to alleviate the above-mentioned drawback of grouping in the back-propagation. As shown in Figure 2b, grouped weights are not block-sparse but interleaved so that each head is filled with the



(a) MSF (Figure 1b) (b) Original (Figure 1a)

Figure 3. Relationships among projection matrices in the transformer models. The upper-triangle matrix shows subspace similarities (13) computed between the projection weights of the transformers embedded in ImageNet pretrained ViT-S. The numbers in diagrams show high similarity scores.

output projections from all groups. It facilitates learning by delivering gradients from one head to whole feature components in back-propagation. This interleaved grouped projection improves multi-head approach, which is thus different from [56, 57] stacking interleaved group convolutions to efficiently approximate dense convolution.

3. Results

We apply the proposed method to transformer networks on ImageNet-1K dataset [13]; the MSF-transformer (Figure 1b) replaces all the transformer modules (Figure 1a) in the networks. The method is first analyzed through ablation studies using ViT-S [4] (Table 5) in Section 3.1, and then evaluated on various network models in Section 3.2.

3.1. Ablation study

Training protocol. ViT-S [4] is optimized by AdamW [31] with 0.05 weight decay and 0.001 initial learning rate which is decayed in a cosine schedule. For data augmentation, we apply random resized cropping [38] with three types of appearance fluctuation [42] as well as mixup [55] and cutmix [53]. We train ViT-S over 100 epochs with 1024 batch size on four GPUs of NVIDIA RTX3090; detailed training settings are shown in supplementary material. For evaluation, top-1 classification accuracies are measured on an ImageNet [13] validation set.

Types of projection. As shown in Figure 1b, the proposed formulation (12) contains $\hat{d}$-dimensional subspace features of PROBE $\boldsymbol{P}_h^\top \boldsymbol{x}$ as well as QUERY, KEY and VALUE, which are back-projected to d-dimensional space by WEIGHT $\boldsymbol{W}_h$; thus there are totally *five* projection matrices in the method.

First, we qualitatively analyze relationships among them from a viewpoint of mean-shift updating (Section 2.2.2), as summarized in Figure 3a. By comparing the mean-shift update (11) with MSF-transformer (12), we can realize that KEY and VALUE are coupled as distribution samples

	Model	QUERY	KEY	VALUE	PROBE	WEIGHT	Acc.	Param (M)
	Simplest							
a	MSF	**Q**	**Q**	**Q**	**Q**	**Q**	61.13	16.60
	2-matrices							
b	MSF	**Q**	**K**	**K**	**Q**	**Q**	77.16	18.37
c	MSF	**Q**	**Q**	**V**	**V**	**V**	69.97	18.37
d	MSF	**Q**	**Q**	**V**	**V**	**Q**	72.54	18.37
e	MSF	**Q**	**Q**	**Q**	**P**	**P**	75.50	18.37
f	MSF	**Q**	**K**	**K**	**K**	**Q**	77.84	18.37
g	MSF	**Q**	**K**	**K**	**Q**	**K**	74.61	18.37
	3-matrices							
h	MSF	**Q**	**K**	**K**	**K**	**W**	78.82	20.14
i	MSF	**Q**	**K**	**K**	**Q**	**W**	78.51	20.14
j	MSF	**Q**	**K**	**K**	**P**	**P**	78.74	20.14
k	MSF	**Q**	**K**	**K**	**P**	**Q**	77.87	20.14
	4-matrices							
l	MSF	**Q**	**Q**	**V**	**P**	**W**	77.23	21.91
m	MSF	**Q**	**K**	**K**	**P**	**W**	79.19	21.91
n	MSF	**Q**	**K**	**V**	**V**	**W**	79.49	21.91
o	MSF	**Q**	**K**	**V**	**P**	**P**	79.52	21.91
p	MSF	**Q**	**K**	**V**	**Q**	**W**	79.19	21.91
q	MSF	**Q**	**K**	**V**	**P**	**Q**	78.97	21.91
	Full							
r	MSF	**Q**	**K**	**V**	**P**	**W**	**79.79**	23.68

Table 1. Various types of configurations for projections in MSF-transformer (12) on 100-epoch trained ViT-S. We report top-1 accuracy (%) on an ImageNet validation set with the number of parameters in each model. Colored letters indicate shared matrices.

	Model	QUERY	KEY	VALUE	WEIGHT	Acc.	Param (M)
	Simplest						
a	Orig.	**Q**	**Q**	**Q**	**Q**	71.34	16.60
	2-matrices						
b	Orig.	**Q**	**K**	**K**	**Q**	76.25	18.37
c	Orig.	**Q**	**Q**	**Q**	**W**	76.04	18.37
d	Orig.	**Q**	**Q**	**V**	**Q**	77.25	18.37
e	Orig.	**Q**	**Q**	**V**	**V**	74.64	18.37
	3-matrices						
f	Orig.	**Q**	**Q**	**V**	**W**	78.04	20.14
g	Orig.	**Q**	**K**	**K**	**W**	78.38	20.14
h	Orig.	**Q**	**K**	**V**	**V**	76.47	20.14
	Full						
i	Orig.	**Q**	**K**	**V**	**W**	**78.98**	21.91

Table 2. Various types of configurations for projections in the original transformer (1) on ViT-S in the same way as Table 1.

$\boldsymbol{k}_i$, while the other threes of QUERY, PROBE and WEIGHT are connected via a probe point $\boldsymbol{q}$. In (12), VALUE is comparable with PROBE to form a differential vector in $\hat{d}$-dimensional feature representation. Besides, the attention weights are computed based on a metric constructed by QUERY and KEY. We can similarly analyze the original transformer (Figure 1a) for clarifying qualitative relation-

ships among QUERY, KEY, VALUE and WEIGHT as shown in Figure 3b.

Next, we further analyze their similarities by using the *learnt projection matrices* of $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}, \boldsymbol{P}$ and $\boldsymbol{W}$ in a quantitative way. Since a subspace is a essential representation to characterize projection, we compute subspace similarity [5] among projection matrices; let singular value decomposition of a matrix $\boldsymbol{X} \in \mathbb{R}^{d \times \hat{d}}$ be $\boldsymbol{X} = \boldsymbol{U}_X \boldsymbol{\Lambda}_X \boldsymbol{V}_X^\top$, and the similarity between subspaces of $\boldsymbol{X}$ and $\boldsymbol{Y}$ is given by

$$\texttt{sim}(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{\hat{d}} \texttt{trace}(\boldsymbol{U}_X \boldsymbol{U}_X^\top \boldsymbol{U}_Y \boldsymbol{U}_Y^\top) = \frac{1}{\hat{d}} \sum_{r=1}^{\hat{d}} cos^2 \theta_r, \tag{13}$$

where θ_r is the r-th canonical angle between two subspaces, $\boldsymbol{X}, \boldsymbol{Y} \in \{\boldsymbol{Q}_h, \boldsymbol{K}_h, \boldsymbol{V}_h, \boldsymbol{P}_h, \boldsymbol{W}_h\}$, and $0 \leq \texttt{sim} \leq 1$. The similarity scores among projection matrices are shown in Figure 3a together with a diagram to describe the relationships. This quantitative measurement is roughly coincident with the qualitative analysis discussed above. A high similarity can be found in the relationship between PROBE and WEIGHT which are used for forward-backward projection in (12). The connections of QUERY-KEY and VALUE-PROBE also exhibit relatively high similarities in accordance with the above qualitative analysis. Then, the same analyses are applied to the original transformer as shown in Figure 3b, demonstrating that the relationships among four projection matrices resemble those of MSF-transformer (Figure 3a). However, as it lacks PROBE, there are less strong relationships like VALUE-PROBE and WEIGHT-PROBE in Figure 3a.

Configuration of projections. While MSF-transformer (12) individually assigns matrices to those projections, there is a possibility to tie some of them by sharing projection matrices according to the discussion in Figure 3a and Section 2.2.2. From that viewpoint, we analyze configurations of projections in Table 1. Since increasing parameters generally improves performance, we compare the configurations under the same budget of parameter size as follows.

The simplest model is given by sharing a single matrix across all the projections (Table 1*a*). While it minimizes memory consumption, such a hard constraint to tie all the projections degrades performance in comparison to the full model (Table 1*n*), and thus we require more parameter budget to attain favorable performance.

By using two projection matrices, the mean-shift update (11) is constructed to interestingly produce favorable performance (Table 1*b*); the smaller model of 18.37M parameters even outperforms ResNet-50 (75.97% in Table 6) of 25.50M-parameterized model. It validates our approach that relates mean-shift update with feature transformation. Table 1*c-f* show the performance results by tying projection matrices in different semantic groups, demonstrating superiority of the mean-shift update (*b*). Particularly, the results of *c-e* fail to validate the connection between QUERY and KEY. Due to Gaussian kernel in MSF-transformer (12), assimilation of QUERY and KEY leading to $\|\boldsymbol{K}_h^\top \boldsymbol{x}_i - \boldsymbol{Q}_h^\top \boldsymbol{x}_i\| = 0 \, \forall i$ inevitably maximizes self-attention weights (diagonal components of attention weights). It may impede metric learning on KEY/QUERY representation, deteriorating performance. On the other hand, the mean-shift update (Table 1*b*) is further improved by slightly touching PROBE to relate with VALUE in Table 1*f*; the remaining connection between QUERY and WEIGHT is qualitatively validated as discussed above. In contrast, by tying unrelated ones, performance is degraded, e.g., in Table 1*g* where a projection matrix is shared among KEY, VALUE and WEIGHT which are not connected semantically nor quantitatively; they exhibit low similarity scores in Figure 3a.

Model	Test Acc.	Training loss	Training Acc.
MSF (12)	79.79	2.50	69.90
Residual (14)	79.28	2.43	71.20
Gauss-Orig.	79.31	2.51	69.34

Table 3. Performance comparison in terms of mean-shift formulation in ViT-S. Gauss-Orig indicates the original transformer equipped with Gaussian kernel, i.e., the first term in (14).

Then, we disentangle the successful configurations of Table 1*bf* to have three projection matrices. As shown in Table 1*h-k*, performance is effectively boosted by increasing parameter sizes.

Under the budge of four matrices, we let a pair of projections share a matrix as shown in Table 1*l-q*. While the assimilation of QUERY and KEY is inferior due to the self-attention issue discussed above, the other configurations enjoy performance improvement by augmenting representations to outperform the original transformer (Table 2*i*). In particular, coupling PROBE to WEIGHT in Table 1*o* produces the best performance; their relationship is strongly suggested in Figure 3a by exhibiting high similarity score. Finally, the full model in Table 1*r* outperforms those configurations.

The similar analysis and discussion are applicable to the original transformer as shown in Table 2. While the simplest form works less effectively, the mean-shift update (9) produces favorable performance (Table 2*b*) under the budget of two projection matrices (Table 2*b-e*). As shown in Table 2*e*, performance is deteriorated by tying unrelated ones as is the case with Table 1*g*. Since the original transformer (1) applies inner-product between QUERY and KEY in computing attention, it is free from the self-attention issue in Table 1 discussed above, as shown in Table 2*cdf*. It should be noted that the models of the original transformer in Table 2 are inferior to those of MSF-transformer (Table 1) under the same budge of parameter size; even the full model in Table 2*i* is defeated by the MSF models equipped with four

Model	QUERY	KEY	VALUE	PROBE	WEIGHT	Group (mode)	Acc.	Param (M)
MSF	$\boldsymbol{Q}$	$\boldsymbol{K}$	$\boldsymbol{V}$	$\boldsymbol{P}$	$\boldsymbol{W}$	1	79.79	23.68
MSF	$\boldsymbol{Q}$	$\boldsymbol{K}$	$\boldsymbol{V}$	$\boldsymbol{P}$	$\boldsymbol{W}$	2 (interleave)	79.55	20.14
MSF	$\boldsymbol{Q}$	$\boldsymbol{K}$	$\boldsymbol{V}$	$\boldsymbol{P}$	$\boldsymbol{W}$	2 (block)	78.87	20.14
MSF	$\boldsymbol{Q}$	$\boldsymbol{K}$	$\boldsymbol{K}$	$\boldsymbol{K}$	$\boldsymbol{W}$	1	78.82	20.14
MSF	$\boldsymbol{Q}$	$\boldsymbol{K}$	$\boldsymbol{K}$	$\boldsymbol{K}$	$\boldsymbol{W}$	2 (interleave)	78.23	18.37
Orig.	$\boldsymbol{Q}$	$\boldsymbol{K}$	$\boldsymbol{V}$	-	$\boldsymbol{W}$	1	78.98	21.91
Orig.	$\boldsymbol{Q}$	$\boldsymbol{K}$	$\boldsymbol{V}$	-	$\boldsymbol{W}$	2 (interleave)	78.57	19.26
Orig.	$\boldsymbol{Q}$	$\boldsymbol{K}$	$\boldsymbol{V}$	-	$\boldsymbol{W}$	2 (block)	78.05	19.26

Table 4. Performance comparison by applying the grouped projection (Section 2.3) to both MSF and original transformers as well as to the parameter-sharing MSF model (Table 1*h*).

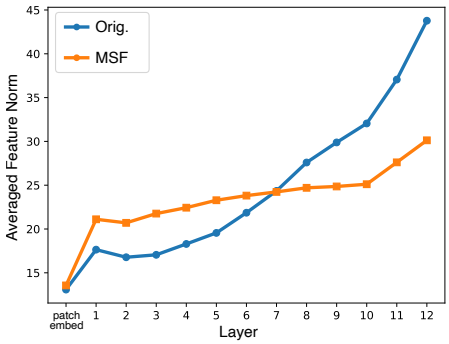


Figure 4. Averaged L_2 norm of features $\boldsymbol{x}$ at the patch-embedding and 12 layers in ViT-S.

projection matrices (Table 1*o*). Thus, these results validate the efficacy of the proposed MSF formulation.

Mean-shift formulation. As discussed in Section 2.2.2, the MSF-transformer derived from mean-shift updating (11) on KDE is distinctive in that it additionally introduces PROBE $\boldsymbol{P}_h^\top \boldsymbol{x}$ to produce a differential against attention-aggregated VALUE features. Focusing on PROBE, the updating vector $\hat{\boldsymbol{x}}$ can be rewritten into

$$\begin{aligned}\hat{\boldsymbol{x}} &= \sum_{h=1}^{H} \boldsymbol{W}_h \Big[\boldsymbol{V}_h^\top \mathcal{X}\, \sigma\big(\{-\tfrac{1}{2}\|\boldsymbol{K}_h^\top \boldsymbol{x}_i - \boldsymbol{Q}_h^\top \boldsymbol{x}\|_2^2\}_{i=1}^m\big) - \boldsymbol{P}_h^\top \boldsymbol{x} \Big], \\ &= \sum_{h=1}^{H} \boldsymbol{W}_h \Big[\boldsymbol{V}_h^\top \mathcal{X}\, \sigma\big(\{-\tfrac{1}{2}\|\boldsymbol{K}_h^\top \boldsymbol{x}_i - \boldsymbol{Q}_h^\top \boldsymbol{x}\|_2^2\}_{i=1}^m\big) \Big] - \tilde{\boldsymbol{P}}\boldsymbol{x},\end{aligned} \tag{14}$$

where $\tilde{\boldsymbol{P}} = \sum_h \boldsymbol{W}_h \boldsymbol{P}_h^\top \in \mathbb{R}^{d\times d}$ is a re-parameterized projection matrix, consuming the same number of parameters as that of $\boldsymbol{P} = [\boldsymbol{P}_1, \cdots, \boldsymbol{P}_H] \in \mathbb{R}^{d\times d}$; usually, $H\hat{d} = d$. The differential vector is also produced in (14) by means of a linear projection using $\tilde{\boldsymbol{P}}$, which may be regarded as a way to add a residual connection [17] to the original transformer model (Figure 1a) equipped with Gaussian kernel, i.e., the first term in (14). The approach (14), however, is less related to the mean-shift model (Section 2.2.2) since WEIGHT $\boldsymbol{W}_h$ does not have any effect on the probe representation of $\tilde{\boldsymbol{P}}\boldsymbol{x}$; in MSF-transformer (12), PROBE is back-projected to d-dimensional space via WEIGHT according to the mean-shift update. We compare the MSF-transformer (12) to the residual-based model (14) in Table 3 which also shows performance of the original transformer using Gaussian kernel for reference. The residual model (14) is inferior to the MSF-transformer while producing better scores on a *training* set due to the lack of mean-shift constraint. We conjecture that degenerating mean-shift model curbs generalization performance by inducing overfitting. These results validate effectiveness of the mean-shift model (Section 2) in feature transformation.

Model	Layer	Width d	Head H	MLP
ViT-Ti [4]	12	192	3	$4d$
ViT-SS [4]	6	384	6	$4d$
ViT-S [4]	12	384	6	$4d$
ViT-B [4]	12	768	12	$4d$
Swin-T [30]	[2,2,6,2]	[96,192,384,768]	[3,6,12,24]	$4d$
Swin-S [30]	[2,2,18,2]	[96,192,384,768]	[3,6,12,24]	$4d$

Table 5. Architectures of transformer-based networks. We apply simple ViT models [4] slightly modified from the original ViT [14] such as by applying global average pooling to aggregate features.

Feature norm. We then analyze feature representations embedded in the transformer models. As discussed in Section 2.2.1, the *ill*-posed kernel of $\exp(\hat{d}^{-\frac{1}{2}}\boldsymbol{k}^\top \boldsymbol{q})$ would induce bias toward enlarging feature magnitude (L_2 norm) in gradient-ascent updating, i.e., the original transformer. To empirically analyze it, Figure 4 evaluates averaged feature norm $\frac{1}{m}\sum_{i=1}^m \|\boldsymbol{x}_i\|_2$ at each layer. While the preprocessing layer of patch embedding produces almost the same feature norms, the two transformer models seem to produce different feature representation in terms of feature norms along layers. The original transformer increases the feature norms at deeper layers, which inevitably demands normalization layers to stabilize training. This implies the above-mentioned bias induced by the *ill*-posed kernel function. On the other hand, the MSF-transformer leverages well-defined mean shift based on a Gaussian kernel to provide features of rather stable norms across layers.

Grouped projection. Next, we analyze the grouped projection (Section 2.3) for reducing parameter size. In the multi-head framework, the grouped projection is applied in two ways of block-wise and interleaved grouping which are different specifically in terms of back-propagation as shown in Figure 2. They are evaluated on the models of MSF-transformer and the original one; the performance results are shown in Table 4 where the grouped projection operates on four projections of QUERY, KEY, VALUE and

	Model	Param (M)	FLOPS (G)	Acc. (%) 100-ep	300-ep
CNN	EfficientNet-B0	5.24	0.77	73.47	76.04
	ResNet-50	25.50	8.24	75.97	78.78
	ResNet-101	44.44	15.71	78.36	80.72
	ResNeXt-50	24.96	8.54	77.67	80.11
	ResNeXt-101	88.59	31.1	81.28	81.71
ViT-Ti	Orig.	5.65	2.50	72.25	75.71
	MSF	6.09	2.67	73.50	76.81
	MSF w/ 2-group	5.20	2.32	72.74	76.20
ViT-SS	Orig.	11.30	4.64	74.65	77.22
	MSF	12.18	4.98	75.50	78.20
	MSF w/ 2-group	10.41	4.29	75.09	77.31
ViT-S	Orig.	21.91	9.16	78.98	80.98
	MSF	23.68	9.85	79.79	81.49
	MSF w/ 2-group	20.14	8.46	79.55	81.41
ViT-B	Orig.	86.29	34.96	81.47	82.67
	MSF	93.37	37.73	81.79	82.86
	MSF w/ 2-group	79.21	32.18	81.70	82.65
Swin-T	Orig. (1, 2)	28.20	8.99	78.92	81.69
	MSF (12)	30.36	9.68	79.61	82.34
	MSF (12) w/ 2-group	26.04	8.29	79.25	81.95
Swin-S	Orig. (1, 2)	49.43	17.49	81.25	83.43
	MSF (12)	53.36	18.88	81.33	83.70
	MSF (12) w/ 2-group	45.51	16.10	81.18	83.52

Table 6. Performance results by embedding two types of MSF-transformer modules into ViT and Swin networks (Table 5) trained from scratch on ImageNet. Top-1 accuracies are reported.

PROBE. The interleaved approach effectively works to reduce parameter size while keeping performance on both the models; especially, MSF-transformer with 2-group interleaved projection outperforms the original transformer while consuming smaller number of parameters (79.55% with 20.14M param vs 78.98% with 21.91M param). On the other hand, the block-wise method interferes with performance as it impedes back-propagation in end-to-end learning. The interleaved grouping also contributes to reducing parameter size of projection-sharing model (Table 1*h*).

3.2. Performance comparison

We embed the MSF-transformer into various networks based on ViT [4, 14] and Swin-Transformer [30] (Table 5) with comparison to representative CNN models [17, 39, 50] of various parameter sizes. For fair comparison, all the networks are trained from scratch on an ImageNet [13] training set by the same training protocol as in Section 3.1 using four GPUs; only the intensity of mixing augmentation and batch size are tuned according to the model size, the details of which are shown in supplementary material. It is noteworthy that both the network models of transformers and CNNs are effectively trained to produce competitive performance with the reported ones, such as ViT-S (76.5%) [4] and ResNeXt-50 (77.8%) [50]. We evaluate the networks trained in 100 and 300 epochs by reporting top-1 classification accuracies on an ImageNet validation set in Table 6.

We apply two types of MSF-transformer (Figure 1b) equipped with full projection and the grouped projection (Figure 2b) using interleaved 2 groups. The MSF models improve performance on various models of diverse parameter sizes. The grouped projection effectively reduces parameter size, especially working on large-scaled models while retaining classification performance. Even on a small-/middle-scaled models, the MSF-transformer produces favorable performance improvement; e.g., ViT-Ti with MSF outperforms EfficientNet-B0 [39] and ViT-S with MSF is competitive to ResNet-family [17, 50] under the same or smaller budget of parameter size. These experimental results show versatile applicability of the proposed method in place of the original transformer module.

Backbone model	iNat2018 (Acc)	iNat2019 (Acc)	ADE (mIoU)
ViT-S Orig.	66.84	73.45	41.81
MSF	68.19	74.60	44.01
MSF w/ 2-group	67.94	74.74	43.50

Table 7. Performance results on fine-grained iNaturalist classification (acc, %) and ADE semantic segmentation (mIoU).

3.3. Transfer learning to downstream tasks

To show transferability, we apply the proposed method to fine-grained visual classification on iNaturalist datasets [20, 21] and semantic segmentation on ADE dataset [61] in a framework of SETR [59]. In those two types of downstream tasks, we leverage a backbone of ImageNet-pretrained ViT-S (Table 6). The performance results in Table 7 demonstrate effectiveness of the proposed method. Particularly, on semantic segmentation, the SETR [59] exploits multiple feature maps, each of which can be improved by our MSF model to produce favorable improvement.

4. Conclusion

We have proposed a feature transformation method to extend the transformer. By analyzing the transformer model from a viewpoint of feature updating on a token distribution, the method is formulated based on the mean-shift update which is a gradient ascent of KDE equipped with Gaussian kernel. We also present an efficient approach to reduce parameter size of feature projections which are essential ingredients in our multi-head transformation method. In the experiments on an ImageNet classification task, the methods are thoroughly evaluated from various aspects and improve performance in place of the original transformer module.

References

[1] Jimmy Le Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv,* 1607.06450, 2016. 4

[2] Irwan Bello. Lambdanetworks: Modeling long-range interactions without attention. In *ICLR*, 2021. 1, 2

[3] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, pages 813–824, 2021. 1, 2

[4] Lucas Beyer, Xiaohua Zhai, and Alexander Kolesnikov. Better plain vit baselines for imagenet-1k. *arXiv,* 2205.01580, 2022. 5, 7, 8

[5] Åke Björck and Gene H. Golub. Numerical methods for computing angles between linear subspaces. *Mathematics of Computation*, 27(123):579–594, 1973. 6

[6] Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When vision transformers outperform resnets without pre-training or strong data augmentations. In *ICLR*, 2022. 2

[7] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE TPAMI*, 17(8):790–799, 1995. 2, 4

[8] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers. In *ICLR*, 2021. 1, 2

[9] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE TPAMI*, 24(5): 603–619, 2002. 2, 4

[10] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. In *ICLR*, 2020. 2

[11] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *arXiv,* 2205.14135, 2022. 1, 2

[12] Stéphane d'Ascoli, Hugo Touvron, Matthew Leavitt, Ari Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. In *ICML*, 2021. 1, 2

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 5, 8

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Trans- formers for image recognition at scale. In *ICLR*, 2021. 1, 2, 3, 4, 7, 8

[15] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *ICLR*, 2021. 2

[16] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, Zhaohui Yang, Yiman Zhang, and Dacheng Tao. A survey on vision transformer. *IEEE TPAMI*, 45(1): 87–110, 2023. 1, 2

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 7, 8

[18] Ruining He, Anirudh Ravula, Bhargav Kanagal, and Joshua Ainslie. Realformer: Transformer likes residual attention. *arXiv,* 2012.11747, 2021. 1, 2

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 1

[20] iNatrualist. The inaturalist competition dataset. https://github.com/visipedia/inat_comp/tree/master/2018, 2018. 8

[21] iNatrualist. The inaturalist 2019 competition dataset. https://www.kaggle.com/c/inaturalist-2019-fgvc6, 2019. 8

[22] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*, pages 5156–5165, 2020. 1, 2

[23] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2021. 1, 2

[24] Takumi Kobayashi. Mutual conditional probability for self-supervised learning. In *BMVC*, 2022. 2

[25] Soroush Abbasi Koohpayegani, Ajinkya Tejankar, and Hamed Pirsiavash. Mean shift for self-supervised learning. In *ICCV*, pages 10326–10335, 2021. 2

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 2, 4

[27] Mingxiang Liao, Zonghao Guo, Yuze Wang, Peng Yuan, and Bailan Feng. Attentionshift: Iteratively estimated part-based attention map for pointly supervised instance segmentation. In *CVPR*, pages 19519–19528, 2023. 2

[28] Grace W. Lindsay. Attention in psychology, neuroscience, and machine learning. *Frontiers in Computational Neuroscience*, 14(29), 2020. 1

[29] Yaoyao Liu, Bernt Schiele, Andrea Vedaldi, and Christian Rupprecht. Continual detection transformer for incremental object detection. In *CVPR*, pages 23799–23808, 2023. 2

[30] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021. 2, 7, 8

[31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 5

[32] Sachin Mehta and Mohammad Rastegari. Mobilevit: Lightweight, general-purpose, and mobile-friendly vision transformer. In *ICLR*, 2022. 1, 2

[33] Miguel Á. Carreira-Perpi nán. A review of mean-shift algorithms for clustering. *arXiv,* 1503.00687, 2015. 4

[34] Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *arXiv,* 2003.05997, 2020. 1, 2

[35] Samuel L Smith, Andrew Brock, Leonard Berrada, and Soham De. Convnets match vision transformers at scale. *arXiv,* 2310.16764, 2023. 2

[36] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck

transformers for visual recognition. In *CVPR*, pages 16519–16529, 2021. 1

[37] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv,* 2106.10270, 2021. 2

[38] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 5

[39] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 8

[40] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1195–1204, 2017. 1

[41] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 2

[42] Hugo Touvron, Matthieu Cord, and Hervé Jégou. Deit iii: Revenge of the vit. In *ECCV*, 2022. 2, 5

[43] Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Transformer dissection: A unified understanding of transformer's attention via the lens of kernel. In *EMNLP*, pages 4344–4353, 2019. 1, 2

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017. 1, 2, 3, 4

[45] M.P. Wand and M.C. Jones. *Kernel Smoothing*. Chapman and Hall, 1995. 3

[46] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv,* 2006.04768, 2020. 1, 2

[47] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, pages 7794–7803, 2018. 1, 3

[48] Zhanghao Wu, Zhijian Liu, Ji Lin, Yujun Lin, and Song Han. Lite transformer with long-short range attention. In *ICLR*, 2020. 1, 2

[49] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross Girshick. Early convolutions help transformers see better. In *NeurIPS*, 2021. 1, 2

[50] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, pages 5987–5995, 2017. 2, 4, 8

[51] Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *AAAI*, pages 14138–14148, 2021. 1, 2

[52] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis E.H. Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *ICCV*, pages 558–567, 2019. 2

[53] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, pages 6023–6032, 2019. 5

[54] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *CVPR*, pages 12104–12113, 2022. 2

[55] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 5

[56] Ting Zhang, Guo-Jun Qi, Bin Xiao, and Jingdong Wang. Interleaved group convolutions. In *ICCV*, pages 4383–4392, 2017. 2, 4, 5

[57] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*, pages 6848–6856, 2018. 2, 4, 5

[58] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *CVPR*, pages 10076–10085, 2020. 1, 2

[59] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, 2021. 8

[60] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, pages 6881–6890, 2021. 2

[61] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017. 8

[62] Daquan Zhou, Bingyi Kang, Xiaojie Jin, Linjie Yang, Xiaochen Lian, Zihang Jiang, Qibin Hou, and Jiashi Feng. Deepvit: Towards deeper vision transformer. *arXiv,* 2103.11886, 2021. 1, 2

[63] Daquan Zhou, Yujun Shi, Bingyi Kang, Weihao Yu, Zihang Jiang, Yuan Li, Xiaojie Jin, Qibin Hou, and Jiashi Feng. Refiner: Refining self-attention for vision transformers. *arXiv,* 2106.03714, 2021. 1, 2