Token Boosting for Robust Self-Supervised Visual Transformer Pre-training

Tianjiao Li[1†]   Lin Geng Foo[1†]   Ping Hu[2]   Xindi Shang[3]   Hossein Rahmani[4]

Zehuan Yuan[3]   Jun Liu[1‡]

[1]Singapore University of Technology and Design

[2]Boston University   [3]ByteDance   [4]Lancaster University

{tianjiao_li,lingeng_foo}@mymail.sutd.edu.sg, pinghu@bu.edu, shangxindi@bytedance.com

h.rahmani@lancaster.ac.uk, yuanzehuan@bytedance.com, jun_liu@sutd.edu.sg

Abstract

Learning with large-scale unlabeled data has become a powerful tool for pre-training Visual Transformers (VTs). However, prior works tend to overlook that, in real-world scenarios, the input data may be corrupted and unreliable. Pre-training VTs on such corrupted data can be challenging, especially when we pre-train via the masked autoencoding approach, where both the inputs and masked "ground truth" targets can potentially be unreliable in this case. To address this limitation, we introduce the Token Boosting Module (TBM) as a plug-and-play component for VTs that effectively allows the VT to learn to extract clean and robust features during masked autoencoding pre-training. We provide theoretical analysis to show how TBM improves model pre-training with more robust and generalizable representations, thus benefiting downstream tasks. We conduct extensive experiments to analyze TBM's effectiveness, and results on four corrupted datasets demonstrate that TBM consistently improves performance on downstream tasks.

1. Introduction

Having rapidly risen in popularity in recent years, Vision Transformer (ViT) [16] and its variants [19] have shown impressive performance across various computer vision tasks, such as image classification, video recognition and 3D action analysis [2, 20, 31, 51, 60]. Amongst this wave of research on Visual Transformers (VTs), there has emerged a popular paradigm – self-supervised VT pre-training [2, 6, 20, 60] – which has attracted a lot of attention in the research community. These works [2, 6, 20, 60] generally pre-train VTs on a large dataset in a self-supervised manner, allowing them to extract semantically meaningful and generalizable features without the need for annotated

data. These pre-trained VTs are practical and effective, showing good downstream performance with only slight fine-tuning, and have quickly become an important research direction.

Among the various self-supervised pre-training strategies, masked autoencoding [2, 16, 20] is a prominent approach that has been widely explored. Masked autoencoding [2, 11, 16, 20, 33, 60, 61, 71] works by randomly masking a portion of input tokens or patches, and letting the VT reconstruct them. Thus, in order to successfully reconstruct the masked tokens or patches, the VT is driven to learn the underlying structure and semantic information of the input data. Due to its natural compatibility with the token-wise representation in VTs, masked autoencoding has been explored for pre-training VTs on data across many fields, such as RGB images [2, 20, 61], pose data [11, 33] and 3D data [71].

However, in many real-world scenarios, the input data can be of low quality and can be unreliable, which is often overlooked. For example, in adverse weather conditions [22], the quality of captured images can be quite bad, with many corrupted pixels that degrade the image. Depth images, which are commonly used for 3D analysis, also often contain errors and noticeable perturbations due to measurement errors introduced by the depth camera sensors [26]. Skeleton data can also have distortions in the joint coordinates [14, 37, 59] due to noise introduced by the sensors, e.g., the pose tracking function of Kinect v2 often introduces average offsets of 50-100 mm per joint [59]. Examples of such corrupted data are depicted in Fig. 1.

In many cases, we have access to large-scale unlabeled data containing such corruptions and noise, and would like to exploit them to pre-train a VT that can be effectively adapted to a wide range of downstream tasks [2, 11, 20, 61]. Yet, this can be rather challenging since the data corruptions can interfere with the learning of representations during self-supervised pre-training, leading to unreliable features and predictions in downstream tasks. This is the case

---

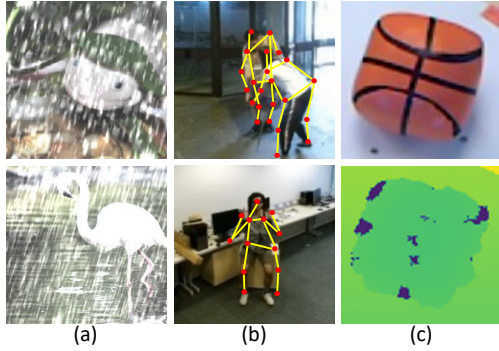† equal contribution

‡ corresponding author

Figure 1. Samples of corrupted data. (a) Images in adverse weather conditions. (b) Skeletons with corresponding RGB images. Skeleton data collected with a pose tracking algorithm [59] can often contain misplaced joints. (c) Depth image (Bottom) with its corresponding RGB image (Top). The collected depth image can be unreliable, e.g., having noticeable perturbations. Images are taken from ImageNet-C [21], NTU RGB+D 120 [36] and Washington RGB-D [26] datasets. Best viewed in color.

in particular for masked autoencoding approaches, where both the inputs and the "ground truth" targets can be corrupted at the same time during pre-training. Hence, we aim to tackle the key question: How can we improve the VT's ability to extract robust and reliable features during self-supervised pre-training on corrupted data? Once this is achieved, the pre-trained capability can then be transferred to downstream tasks, where the VTs can reliably extract generalizable features and perform well even when the testing data is unreliable.

To this end, we propose a novel Token Boosting Module (TBM) to improve the robustness of VTs against unreliable and corrupted data during the pre-training process. TBM is a plug-and-play module that can improve the performance of VTs by boosting the reliability of tokens, making them cleaner such that we can feed robust features to the subsequent layers. We also provide analysis to theoretically demonstrate that our TBM can indeed encourage the learning of cleaner features and tokens during masked autoencoding pre-training.

In summary, our contributions are:

- We design a novel Token Boosting Module (TBM) to learn to extract reliable features from corrupted and unreliable data during the self-supervised pre-training process of VTs. The TBM can be easily incorporated into multiple layers of VTs, and trained in an end-to-end manner to improve robustness and performance on corrupted data.

- We provide theoretical analysis to show that the robustness capabilities of TBM can be learned during the self-supervised masked autoencoding pre-training.

- We conduct experiments on multiple tasks including RGB image classification, 3D skeleton action

recognition, and depth image classification with corrupted data. Experiments consistently show that our TBM provides significant improvements in downstream tasks.

2. Related Work

Self-supervised pre-training. As a practical way to train models without the need for annotated data, self-supervised pre-training has received a lot of attention. Recently, masked autoencoding has become a very popular approach for self-supervised Visual Transformer (VT) pre-training [2,7,16,20,35,45,60,61], together with other alternatives including contrastive learning [8,9,23,47,68,72,74] and self-distillation [6,18,28]. Due to the popularity of pre-training methods and their potential widespread applications in real-world scenarios [2,11,20,33,35,71], it becomes important to improve their robustness to corrupted data, but this is often overlooked by previous methods. For instance, during masked autoencoding pre-training on corrupted data, both inputs and "ground truths" can be unreliable, making it a tricky situation. Therefore, we explore if, under these challenging circumstances, we can allow VTs to learn to extract robust feature representations.

Robustness of VTs. Visual Transformers have recently been a popular and hot research direction, and their robustness has also attracted attention. Some existing works [1, 3, 42, 77] investigate the test-time robustness of VTs, while some others explore the adversarial robustness of VTs [24,41,42]. Different from these works, we aim to improve the VTs' capability to extract clean and robust features during self-supervised pre-training. This capability can benefit downstream tasks and improve their robustness against corrupted input data at test time.

Handling of Corrupted Data. Improving deep models' performance on corrupted data is important for many practical tasks [5, 14, 37, 43, 52, 73]. Previous efforts have been made to handle corrupted input data for supervised settings in tasks like skeleton action recognition [10, 12, 14, 37, 50, 65], depth image classification [5, 73], and face recognition [52]. Alternatively, some other approaches [43, 55, 62, 64] can be performed during data pre-processing to reconstruct a "clean" input sample before using them for corresponding tasks. Many of these methods [43,48,55,62,64] require an additional stage of training, and tend to be specially designed for specific types of data or corruptions only. Here, we investigate a different problem from the above works, and instead explore the handling of corrupted data during *self-supervised VT pre-training*.

3. Method

In many real-world scenarios, the data (e.g., depth images, images in adverse weather conditions) can be col-

lected with substantial corruptions. When such corrupted data is used for self-supervised pre-training, they can lead to unreliable features and predictions, interfering with the learning of generalizable and robust features. In order to improve VTs' performance on corrupted and unreliable data, we introduce a Token Boosting Module (TBM) for VTs that can learn to boost the reliability of features during self-supervised pre-training.

TBM is a plug-and-play module that can be placed between layers in the VT, and can be trained in an end-to-end manner. When placed in the intermediate layers, TBM can make features in VTs cleaner and more reliable. To further improve performance, we incorporate TBM into multiple layers of the VT, such that contextual information at multiple feature levels are used for boosting the quality of the features. This enhanced capability of VTs, achieved during self-supervised pre-training, can thus be transferred to downstream tasks to achieve good performance even when facing corrupted inputs.

In this section, we first give a brief background of masked autoencoding. Then, we explain the feature boosting technique and describe TBM in detail. Lastly, we theoretically show why our TBM can benefit pre-trained VTs with more robust and reliable representations.

3.1. Masked Autoencoding Pre-training

Previous works [2, 16, 20] have explored masked autoencoding to train Transformers, which we follow. In masked autoencoding pre-training [2, 16, 20], a part of the input is often masked, and the Transformer (with a decoder head that is only used during pre-training) is tasked to reconstruct those masked inputs. Driven by the reconstruction losses, the Transformer will learn to capture semantically meaningful information within the input, producing features that tend to generalize well to other downstream tasks. However, this masked autoencoding approach can be challenging when performed with corrupted data, since both the input and masked target could be corrupted, impairing downstream tasks with biased/unreliable feature representations.

3.2. Feature Boosting Technique

In this subsection, inspired by previous works [43, 63], we describe a simple technique to extract robust and reliable features from data corrupted by noise. Specifically, for an unreliable feature F that is going to be fed into a VT layer, this technique allows us to obtain a reliable boosted feature $\hat{R}$. The robust feature $\hat{R}$ (instead of the unreliable feature F) can then be fed into the VT layer for better performance.

Let R represent the ground truth reliable features. We are unable to directly observe the reliable feature R, but are able to observe the unreliable feature F (which are extracted from a real input sample). Importantly, this means that we are unable to simply train a module with direct su-

pervision to produce R from F. We assume that the difference between R and F can be modelled by a feature perturbation P which is randomly drawn from a distribution $\mathcal{P}$, i.e., we have $F = R + P$. Below, we show how, with only access to the unreliable feature F, we can produce $\hat{R}$ (which statistically speaking is an unbiased conditional expectation [54, 56]) that estimates the reliable feature R.

Given F, we first create an intermediate representation, $I = F + Q$ where Q is a synthetic corruption drawn from $\mathcal{P}$ as well, as shown in Fig. 2. Importantly, as P and Q are i.i.d., the two sources of corruptions in I will be indistinguishable to an observer, i.e., we will not be able to tell how much of the corruption comes from P or Q. We then train an autoencoder g (parameterized by θ) to predict F given I as its input, and we denote the output of g as $\hat{F}$ (an estimate of F), as shown in Fig. 2. $\hat{F}$ will later be used to construct the boosted features $\hat{R}$.

Specifically, the autoencoder g is trained with an L2 loss with F as the target, and will be optimized to reconstruct good estimates of F given I as input, i.e., $\hat{F} = \mathbb{E}[F|I]$. Notably, in order to reproduce F perfectly, the autoencoder needs to distinguish between P and Q, but since it never observes P and Q independently, it cannot reproduce F fully, and instead can only produce its best estimate $\hat{F}$. Expanding $\hat{F}$, we observe that $\hat{F} = \mathbb{E}[F|I] = \mathbb{E}[R + P|I] = \mathbb{E}[R|I] + \mathbb{E}[P|I]$.

Next, as Q and P are assumed to be independently drawn from the same distribution, we get $\mathbb{E}[Q|I] = \mathbb{E}[P|I]$. Intuitively, this can be understood as P and Q being indistinguishable from one another, and a simple proof has been provided in the Supplementary. Multiplying the above equation by 2 and performing substitution, we get $2\mathbb{E}[F|I] = \mathbb{E}[R|I] + (\mathbb{E}[R|I] + \mathbb{E}[P|I] + \mathbb{E}[Q|I]) = \mathbb{E}[R|I] + \mathbb{E}[R + P + Q|I] = \mathbb{E}[R|I] + \mathbb{E}[F + Q|I] = \mathbb{E}[R|I] + I$.

By rearranging, we get $\mathbb{E}[R|I] = 2\mathbb{E}[F|I] - I$. We can thus obtain $\hat{R}$ using an estimate $\hat{F}$ as:

$$\hat{R} = 2\hat{F} - I. \qquad (1)$$

In summary, by using this technique, we can still estimate the reliable feature $\hat{R}$ even if we do not have ground truth feature R. This is important, because we do not have access to any ground truth clean data or features (R).

3.3. Token Boosting Module

Next, we present the Token Boosting Module (TBM) to improve reliability of VTs by incorporating the technique described above. One practical issue we face, is how to sample synthetic corruption Q, which requires us to make assumptions on the distribution $\mathcal{P}$. Previous methods [27, 32, 52, 70] handling corrupted data show that modeling perturbations in the deep feature space as Gaussian distributed can be very effective for learning to approximate complicated and intractable distributions underlying
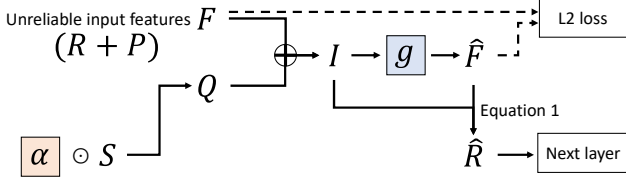
Figure 2. Illustration of the TBM placed at a VT layer. Firstly, TBM takes in the unreliable input feature F, i.e., all tokens. Synthetic corruption Q is produced by scaling Gaussian noise S with learned parameters α. An intermediate representation $I = F + Q$ is produced, and fed into the autoencoder g to reconstruct $\hat{F}$, an estimate of F. Lastly, we apply Eq. 1 with $\hat{F}$ and I as input, producing $\hat{R}$, which is our boosted output feature. $\hat{R}$ is then fed as input to the next VT layer.

the data. Following them, we apply the Gaussian distribution to model the feature perturbation $P \sim \mathcal{P}$, with a mean of 0 and unknown standard deviations. Notably, the Gaussian assumption [27, 32, 52, 70] allows us to "parameterize" corruptions in feature space, and we can then optimize the distribution of the added synthetic corruption Q to be similar to the underlying corruption distribution $\mathcal{P}$, that fulfills the conditions in Sec. 3.2. We empirically observe this works effectively for handling various types of corruptions and tasks.

Let the input feature of TBM be $F \in \mathbb{R}^K$. As shown in Fig. 2, our TBM consists of two sets of learnable parameters. Firstly, we have the scaling parameters $\alpha \in \mathbb{R}^K_{\geq 0}$ that scale the added corruptions. This is necessary to learn the *optimal standard deviation parameters of the synthetic corruptions*, where a different standard deviation is learned for each of K feature dimensions. Importantly, our training loss will optimize α towards making the distribution similar to the natural corruption's distribution $\mathcal{P}$, therefore *the synthetic corruptions and the natural corruptions can be seen as i.i.d.*[1] Secondly, we have the autoencoder g containing learnable parameters θ. This autoencoder g takes in F with added corruptions Q, and reconstructs $\hat{F}$, an estimate of F.

We show an illustration of the TBM in Fig. 2. First, we sample a Gaussian noise vector $S \in \mathbb{R}^K$ with a mean of 0 and standard deviation of 1. The sampled Gaussian noise S is scaled using the learned parameters α in an element-wise manner (denoted by $\odot$). Note that $\alpha \odot S$ corresponds to sampling K Gaussians, with the standard deviation of the Gaussian at each index k to be α_k. We use $\alpha \odot S$ as synthetic corruption Q, and add it to F to form I as follows:

$$I = F + (\alpha \odot S). \tag{2}$$

Next, the autoencoder g is used to reconstruct the estimate of F from I:

$$\hat{F} = g(I). \tag{3}$$

---

[1]We provide detailed analysis that distribution of Q can be trained to approximate distribution of P in Sec. 5 of Supplementary.

Then, we utilize Eq. 1 to obtain a boosted feature $\hat{R}$, which is used as the input to the next layer of the VT.

TBM for features. We insert TBM between self-attention layers in VTs, such that the tokens at the feature layers can be boosted before generating the next set of key, query and value embeddings. In order to further improve the robustness of VTs, we incorporate TBM into multiple layers of the VT. This allows us to enhance the intermediate features of VTs in the presence of different levels of contextual information, which leads to better robustness capabilities.

L2 Reconstruction Loss. To train the autoencoder parameters θ, we introduce an L2 reconstruction loss L_{recon}:

$$L_{recon}(F, \hat{F}) = \lambda \sum_{k=1}^{K} [F_k - \hat{F}_k]^2 \tag{4}$$

where λ is a hyperparameter that controls the weightage of the loss with respect to the end-to-end pre-training objective. This loss L_{recon} trains the autoencoder g to produce good estimates $\hat{F}$, and can be combined with the end-to-end objective in masked autoencoding pre-training to drive TBM parameters θ and α towards improved robustness capabilities, as described in the next sub-section.

3.4. TBM in Masked Autoencoding Pre-training

Our main goal in this paper is to improve the capability of VTs to extract robust features on corrupted and unreliable data during masked autoencoding pre-training, which is a popular and effective paradigm for pre-training VTs. However, one key question remains: *can the TBM be meaningfully trained under the masked autoencoding pre-training objective?* This is not immediately straightforward, especially since both the inputs and the reconstruction targets are unreliable. In this section, we answer the above question and theoretically show how α and θ can be meaningfully trained with the masked autoencoding objective.

We first show our theoretical motivations in more detail using a linear example for the sake of simplicity. Specifically, we explain how a simple linear regression model, tasked to regress some masked inputs $V \in \mathbb{R}^{N_V}$ from the unmasked inputs $U \in \mathbb{R}^{N_U}$, will potentially be able to *achieve a lower loss from the use of our TBM*. This shows that *gradients from the loss will lead to meaningful training of the TBM parameters*. In this example, we assume that the best-fit linear relationship between U and V can be modelled as $V = \beta U + c + \epsilon$, where $\beta \in \mathbb{R}^{N_V \times N_U}, c \in \mathbb{R}^{N_V}, \epsilon \in \mathbb{R}^{N_V}$ respectively refer to the coefficient, intercept and random error coming from a Gaussian with standard deviation of γ. We additionally assume that at least some elements in β are non-zero, which means that there is at least some correlation between U and V.

When trained with adequate data, the fitted linear model (comprising of parameter estimates $\hat{\beta}$ and $\hat{c}$) will converge to β and c, with the MSE error as follows:

$$\mathbb{E}\Big[\frac{1}{N_V}\sum_{j=1}^{N_V}[V_j - (\beta U + c)_j]^2\Big] = \frac{1}{N_V}\sum_{j=1}^{N_V}\mathbb{E}[\epsilon_j^2] = \gamma^2, \quad (5)$$

where the detailed steps are shown in Eq. 23-28 of Supplementary. This (γ^2) represents the best possible result that the linear model can achieve. However, when "natural" corruptions are added towards all inputs in U and V to get $\tilde{U}$ and $\tilde{V}$ (where $\tilde{U} = U + S_U$ and $\tilde{V} = V + S_V$, with S_V and S_U containing elements drawn from a Gaussian with standard deviation σ_N), the MSE will be increased to:

$$\mathbb{E}\Big[\frac{1}{N_V}\sum_{j=1}^{N_V}[\tilde{V}_j - (\beta\tilde{U} + c)_j]^2\Big] \quad (6)$$

$$= \frac{1}{N_V}\sum_{j=1}^{N_V}\mathbb{E}[\epsilon_j^2] + \mathbb{E}[(S_V)_j^2] + \mathbb{E}[(\beta S_U)_j^2] \quad (7)$$

$$= \gamma^2 + \sigma_N^2 + \frac{1}{N_V}\sum_{j=1}^{N_V}\sum_{k=1}^{N_U}\beta_{jk}^2\sigma_N^2. \quad (8)$$

Refer to Eq. 29-40 of Supplementary for more details of the above derivations. The increase in MSE from Eq. 5 to Eq. 8 represents the extra difficulty in conducting inference, when there are corruptions in the input data U (contributing $\frac{1}{N_V}\sum_{j=1}^{N_V}\sum_{k=1}^{N_U}\beta_{jk}^2\sigma_N^2$), as well as corruptions in the "ground truth" inputs V (contributing σ_N^2). We can thus potentially reduce the MSE to $\gamma^2 + \sigma_N^2$ by boosting $\tilde{U}$ (which represents the corrupted inputs) and feeding to the model $\hat{U} \approx U$, which our TBM, if placed before our linear regression model, *will be optimized to do*. In other words, if we can boost $\tilde{U}$ into a clean version $\hat{U} \approx U$, our expected loss (with the full proof in Eq. 41-50 of Supplementary) will be:

$$\mathbb{E}\Big[\frac{1}{N_V}\sum_{j=1}^{N_V}[\tilde{V}_j - (\beta U + c)_j]^2\Big] \quad (9)$$

$$= \frac{1}{N_V}\sum_{j=1}^{N_V}\mathbb{E}[\epsilon_j^2] + \mathbb{E}[(S_V)_j^2] \quad (10)$$

$$= \gamma^2 + \sigma_N^2, \quad (11)$$

which is lower than the MSE if $\tilde{U}$ is directly used (shown in Eq. 8). Thus, during the optimization process, when our TBM is updated via the loss gradients that aim to minimize the pre-training loss, our TBM will tend to learn feature boosting capabilities in order to achieve a smaller loss. This enables the feature boosting capabilities (i.e. learned parameters α and θ) of our TBM to be trained, such that it can function as described in Sec. 3.2.

Although the above is just a linear example, we remark that the concepts involved can generalize to non-linear models such as VTs. In general, *even for non-linear cases*, unreliable inputs will lead to difficulties in masked patch prediction, which will lead to a higher pre-training loss than reliable inputs. Thus, in order to minimize this loss, the gradients will optimize TBM parameters α and θ towards learning how to boost the features, i.e., produce good estimates $\hat{R} = R$, which the TBM's design as described in Sec. 3.2 explicitly allows it to accomplish. Our experimental results also demonstrate the efficacy of such a design.

Note that when trained with supervision labels, TBM can also be trained to boost the reliability of features as well, as input corruptions are likely to negatively affect performance of the supervised task (e.g. 3D action recognition [10, 17, 39, 51], depth image classification [40, 46, 73]) such that the gradients for α and θ will tend to encourage the learning of feature boosting capabilities.

4. Experiments

We first conduct extensive experiments on image classification with added synthetic corruptions, which allows us to systematically vary the type of added corruptions for a more detailed analysis. Then, we test our method on two real-world settings, namely 3D action recognition and depth image classification, where we attain good performance. More experiments can be found in Supplementary.

4.1. Training Setups

In order to evaluate our method more comprehensively, we conduct experiments with two different training setups: the **Self-supervised Training** setup and the **Supervised Training** setup, which we explain below.

Self-supervised Training. Following previous works [2, 16, 20], we add a decoder on top of our VT encoder (inserted with some TBM modules), then we mask input patches and train our VT encoder+decoder by reconstructing these masked patches using MSE loss. After the end-to-end self-supervised training, we follow the *linear probe* setting [20] to train a classifier for testing purposes. Specifically, the decoder is replaced by a classification head, and we train the classification head while keeping the parameters in our VT encoder (including our TBMs) frozen. Then, we report the results using the predictions from the VT encoder+classification head.

Supervised Training. We also aim to evaluate the capability of TBM when trained in a supervised setting. Therefore, we train our VT encoder (including the TBM modules), in a normal supervised manner, where a classification head is attached to the end of the VT encoder and the full

model is trained in an end-to-end manner. The classification head is reused at test time for evaluation.

4.2. Image Classification

Dataset. We conduct both self-supervised and supervised training on ImageNet [15] with the synthetic corruptions from ImageNet-C [21]. ImageNet-C is a corrupted version of the ImageNet test set, and is a commonly used benchmark for robustness evaluation, employing 15 types of synthetically generated corruptions, and some examples can be seen in Fig. 1. ImageNet-C allows us to comprehensively and systematically evaluate the effectiveness of TBM on various types of common corruptions. Specifically, during both *supervised* and *self-supervised training*, we follow the procedure for ImageNet-C [21] and add synthetic corruptions for 50% of input images, where the corruption types are randomly selected (among the 15 types). During testing, we evaluate our method on the corrupted ImageNet-C [21] benchmark.

Network Architecture. To comprehensively evaluate the efficacy of the proposed TBM, we perform experiments using three different sizes of ViT [16]: ViT-Huge, ViT-Large and ViT-Base, as well as DeiT [58] and Swin Transformer (Swin) [38]. We use the Base versions of DeiT and Swin. Following previous works [2, 20], the pre-training decoders are composed of stacked transformer blocks. The autoencoder g in our proposed TBM is comprised of 3 fully-connected+ReLU layers with hidden layers having dimensionality $input\_dim/2$, where $input\_dim$ depends on the architecture of the VT. Scaling values α have the same shape as the input feature map F to the TBM, which depends on the architecture. Our TBM is inserted at 3 different feature levels, i.e., bottom layer, middle layer, and top layer in our ViT, DeiT and Swin backbones, in order to enhance feature robustness with different levels of contextual information.

Training Details. In the *self-supervised training* phase, we follow existing works [20] and use the AdamW optimizer to train the VTs (for all ViTs, DeiT and Swin) for 400 epochs, where 75% of input patches are randomly masked. The initial learning rate is set to 0.01 and gradually decay to 0 following the cosine rule. In the *supervised training* phase, we follow [20] to use LARS optimizer for ViT and we use AdamW optimizer for DeiT and Swin. All experiments are conducted on Nvidia A100 GPUs, and λ is set to 1.

Results. Our results are shown in Tab. 1. Overall, our TBM provides significant performance gains for all five baseline VTs in both self-supervised and fully-supervised settings. This shows that TBM is capable of improving the learning on corrupted data.

Next, we conduct more in-depth analysis using ViT-Huge as the backbone model. We evaluate the effects of TBM on various types of corruptions in Tab. 2. We ob-

Table 1. Performance comparison (%) over multiple baseline VTs on Image Classification using ImageNet-C.

Methods	Supervised	Self-supervised
ViT-Huge [16]	50.1	35.2
ViT-Huge+TBM	**53.2**	**38.6**
ViT-Large [16]	48.7	33.4
ViT-Large+TBM	**51.5**	**36.1**
ViT-Base [16]	41.6	25.9
ViT-Base+TBM	**45.2**	**29.3**
DeiT [58]	44.2	28.7
DeiT+TBM	**47.7**	**32.1**
Swin [38]	45.1	29.6
Swin+TBM	**47.9**	**33.6**

serve that TBM improves performance consistently across *all corruption types* (noise, blue, weather, digital) used in ImageNet-C. We remark that this includes a 3.1% increase on images corrupted with snow, and a 3.6% increase on images corrupted with motion blur, which are examples of practical yet complicated corruptions to tackle. These results suggest that TBM has the capability to generalize to various types of corruptions. We also conduct experiments on another setting **ViT-Huge+TBM (Individual)**, where a single model is trained to deal with each individual type of corruption. This setting leads to a slight improvement as expected, since this setting allows our TBM to fully specialize in tackling a single type of corruption at a time. Nevertheless, our significant improvements over the baseline show that our method (which boosts the features at multiple levels using TBM) is capable of handling these various synthetic corruptions simultaneously, which is a complicated task.

Evaluation with data pre-processing methods. We also want to explore if data pre-processing techniques can be applied to effectively improve the quality of the data before performing self-supervised pre-training. We evaluate the performance of some common smoothing methods, namely the **Median Filter** and the **Gaussian Filter**, as well as three image-based pre-processing methods. As shown in Tab. 3, we find that performance improvements are small. This can be because there are many different types of corruptions, and it can be difficult for a single pre-processing technique to tackle them effectively. It can be more effective to perform the boosting and "cleaning" at the feature level, like our TBM does. Furthermore, we also explore using these five data pre-processing techniques in conjunction with our method (**[Pre-processing] + ViT-Huge + TBM**), where we apply these methods to the input image before feeding them into our ViT-Huge+TBM model. The results when we use these pre-processing techniques in conjunction with our method are comparable to directly using our TBM, suggesting that it is sufficient to directly use our TBM, which indicates the efficacy of our method.

Layers to add TBM. In our method, we apply TBM across

Table 2. Performance comparison (%) of TBM on different types of corruptions on ImageNet-C.

Methods	Noise			Blur				Weather				Digital				Mean
	Gauss.	Shot	Imp.	Defoc.	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	
ViT-Huge [16]	36.7	35.6	35.0	34.3	28.5	39.2	34.4	34.6	26.7	23.8	45.8	28.6	35.9	42.8	45.6	35.2
ViT-Huge+TBM	39.5	39.2	38.7	37.6	30.4	42.8	37.2	37.7	29.5	26.9	49.8	32.2	39.6	47.1	49.2	38.6
ViT-Huge+TBM (individual)	39.8	39.4	39.1	37.9	30.7	42.9	37.3	38.0	29.9	27.2	50.1	32.6	40.0	47.4	49.9	38.8

Table 3. Performance comparison (%) with data pre-processing methods on ImageNet-C.

Methods	Supervised	Self-supervised
ViT-Huge [16]	50.1	35.2
Median Filter	51.3	36.3
Gaussian Filter	51.0	36.1
Shi et al. [52]	51.8	36.4
Moran et al. [43]	52.0	36.6
Xie et al. [62]	51.9	36.5
Median Filter + ViT-Huge + TBM	52.9	38.5
Gaussian Filter + ViT-Huge + TBM	52.7	38.2
Shi et al. [52] + ViT-Huge + TBM	53.0	38.5
Moran et al. [43] + ViT-Huge + TBM	53.1	38.7
Xie et al. [62] + ViT-Huge + TBM	53.0	38.6
ViT-Huge+TBM	53.2	38.6

Table 5. Evaluation of different values of λ for TBM on ImageNet-C.

λ	0	0.1	0.5	1	2	5
Acc. (%)	35.7	37.4	38.1	38.6	38.5	38.5

ImageNet, while the baseline ViT-Huge achieves an accuracy of 85.1%. As accuracy remains about the same as the original ViT-Huge, we conclude that the benefits of TBM mainly lie in improved robustness against corrupted data.

4.3. 3D Skeleton Action Recognition

In 3D skeleton action recognition, models are tasked to classify the performed action using 3D skeleton sequences as input. However, there are often errors in the 3D skeleton data [37], which will negatively affect performance. Thus, this is a real-world setting where we can evaluate the effectiveness of our TBM.

Dataset. NTU RGB+D 60 (NTU60) [48] is a large dataset that has been widely used for 3D action recognition, containing approximately 56k RGB-D sequences from 60 activity classes collected with Kinect v2 sensors. NTU RGB+D 120 (NTU120) [36] is an extension of NTU60, and is currently the largest RGB+D dataset for 3D action analysis, containing more than 114k skeletal sequences across 120 activity classes. We follow the standard evaluation protocols of [36, 48] for 3D action recognition, where the Cross-Subject (xsub) and Cross-View (xview) protocols are tested for NTU60, while the Cross-Subject (xsub) and Cross-Setup (xset) protocols are tested for NTU120.

Network Architecture. We perform experiments using DSTA-Net [51], which is specifically designed for processing skeleton data, and currently achieves state-of-the-art performance among VTs in this field. Following previous works [13, 69], we design the decoder (used during pre-training) as a stack of GCN blocks. Our TBM settings here are similar to the image classification experiments, where autoencoder g consists of 3 fully-connected+ReLU layers, and TBM is inserted at 3 different feature levels, from low to high.

Pre-training Settings. For masked autoencoding pre-training for 3D sequences, we randomly mask "spatial" patches as well as "temporal" patches, and task DSTA-Net to reconstruct them. Thus, DSTA-Net will be trained to encode information about both the *spatial aspect of the human body* and the *temporal aspect of motions*, which will lead to

several layers spanning from the bottom to the top of the VT. Here, we ablate this decision by evaluating several alternative designs on ImageNet-C, with results shown in Tab. 4. **Top layer**, **Mid layer** and **Bot layer** are settings where the TBM is only applied to the top layer, the middle layer, or the bottom layer. Compared to using it in any single layer, our incorporation into multiple layers performs better, demonstrating the efficacy of exploiting context information at different feature levels.

Table 4. Evaluation of TBM applied to different layers. We also conduct testing with TBM incorporated into more layers, and do not find further improvement.

Settings	Baseline [16]	Top layer	Mid layer	Bot layer	Ours (All 3 layers)
Acc. (%)	35.2	37.2	37.1	37.3	38.6

Weightage of losses. We conduct ablation studies on the impact of λ in Tab. 5. $\lambda = 1$ performs the best. When λ is too low, the TBM is not trained well, and when λ is too high, the VT may lose focus on the main objective. Specifically, when we set $\lambda = 0$, we do not train with the reconstruction loss, which effectively disables our boosting mechanism, leading to sub-optimal performance. Notably, when $\lambda = 0$, the model also contains the same number of model parameters as our main ViT-Huge+TBM setting reported in Tab. 1, but performs significantly worse, which indicates that the benefits of our TBM does not merely come from the slightly ($\sim 5\%$) increased number of parameters.

Evaluation on clean ImageNet. We also evaluate the accuracy of ViT-Huge+TBM to be 85.3% on the clean test set of

its features capturing generalizable information that benefits downstream tasks. More details are in Supplementary.

Training Details. In the self-supervised training phase, we train our model for 200 epochs with batch size of 256 using the AdamW optimizer. We set the initial learning rate to 0.01, and set it to gradually decay to 0 following the cosine rule. λ is set to 1. In the *supervised training* phase, we use SGD optimizer following [51].

Results on Self-Supervised Training. We compare with the current state-of-the-art self-supervised models in Tab. 6. It can be observed that adding our TBM to DSTA-Net and then conducting pre-training improves the performance of DSTA-Net consistently across all settings. This also includes a significant 6.0% and 4.0% improvement over DSTA-Net baselines on NTU60 xsub and xview respectively. We also highlight that DSTA-Net+TBM achieves good results on all reported settings.

Table 6. Performance comparison (%) on 3D Skeleton Action Recognition. We follow the evaluation setting of [36, 48]. Models are pre-trained in a self-supervised manner, and tested with a linear probe. We attain good performance on all settings.

Methods	NTU60		NTU120	
	xsub	xview	xsub	xset
LongT GAN [76]	39.1	52.1	-	-
MS^2L [34]	52.6	-	-	-
P&C FS-AEC [57]	50.6	76.3	-	-
P&C Fw-AEC [57]	50.7	76.1	-	-
SeBiReNet [44]	-	79.7	-	69.3
EnGAN-PoseRNN [25]	68.6	77.8	-	-
T-Colorization [67]	71.6	79.9	-	-
TS-Colorization [67]	74.6	82.6	-	-
TSP-Colorization [67]	75.2	83.1	-	-
SkeletonCLR [29]	68.3	76.4	56.8	55.9
3s-SkeletonCLR [29]	75.0	79.8	60.7	62.6
CroSCLR [29]	68.3	76.4	56.8	55.9
3s-CroSCLR [29]	77.8	83.4	-	-
DSTA-Net [51]	73.1	81.5	68.6	70.1
DSTA-Net+TBM	**79.1**	**85.5**	**69.7**	**71.1**

Results on Supervised Training. In Tab. 7, we show the results of training DSTA-Net in a supervised manner for 3D skeleton action recognition. Applying TBM to DSTA-Net provides improvement gains over the baseline DSTA-Net over all reported settings, including a 1.6% gain on NTU60 xsub and 2.5% gain on NTU120 xsub. This allows our DSTA-Net+TBM model to achieve good results for all reported settings.

4.4. Depth Image Classification

In depth image classification, models are tasked to classify images using only the depth image as input (without RGB data). This is challenging as the depth cameras often induce corruptions in the data, which leads to noticeable perturbations and noise. Note that here, we only use the depth image for classification.

Table 7. Performance comparison (%) on 3D Skeleton Action Recognition with models trained in a fully-supervised manner. We follow the setting of [36, 48].

Methods	NTU60		NTU120	
	xsub	xview	xsub	xset
ST-LSTM+TG [37]	69.2	77.7	-	-
ST-GCN [65]	81.5	88.3	70.7	73.2
AS-GCN [30]	86.8	94.2	78.3	79.8
2s-AGCN [50]	88.5	95.1	82.2	84.1
DGNN [49]	89.9	96.1	-	-
SGN [75]	89.0	94.5	79.2	81.5
Shift-GCN [12]	90.7	96.5	85.9	87.6
MS-G3D [39]	91.5	96.2	86.9	88.4
FGCN [66]	90.2	96.3	85.4	87.4
CTR-GCN [10]	92.4	96.8	88.9	90.6
DSTA-Net [51]	91.5	96.4	86.6	89.0
DSTA-Net+TBM	**93.1**	**97.0**	**89.1**	**91.1**

Table 8. Performance comparison (%) on Depth Image Classification using WRGBD dataset based on depth data only.

Methods	Supervised	Self-supervised
CNN-RNN [53]	78.9	-
DECO [5]	84.0	-
VGG_f-RNN [4]	84.0	-
RCFusion [40]	85.9	-
HP-CNN [73]	85.0	-
MMFLAN [46]	84.0	-
ViT-Huge [16]	82.3	69.7
ViT-Huge+TBM	86.1	71.5

Dataset. The Washington RGB-D Object (WRGBD) dataset [26] contains 300 common household objects from 51 different categories. These objects are filmed using RGB-D cameras from multiple angles.

Implementation Details. We perform experiments using ViT-Huge [16], and follow the architecture and training details from our image classification experiments.

Results. We evaluate our method on depth image classification in Tab. 8. TBM provides a 3.8% improvement on the supervised setting and a 1.8% improvement on the self-supervised setting. This further verifies the efficacy of TBM at tackling tasks with corrupted data.

5. Conclusion

In this work, we propose a novel TBM for robust self-supervised VT pre-training on corrupted data. We also theoretically motivate how TBM can learn its feature boosting capabilities using the masked autoencoding pre-training task. When incorporated into VTs and trained in a self-supervised or fully-supervised manner, our TBM leads to performance improvements on corrupted data.

References

[1] Yutong Bai, Jieru Mei, Alan L Yuille, and Cihang Xie. Are transformers more robust than cnns? *Advances in Neural Information Processing Systems*, 34, 2021. 2

[2] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT pre-training of image transformers. In *International Conference on Learning Representations*, 2022. 1, 2, 3, 5, 6

[3] Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Andreas Veit. Understanding robustness of transformers for image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10231–10241, October 2021. 2

[4] Ali Caglayan and Ahmet Burak Can. Exploiting multi-layer features using a cnn-rnn approach for rgb-d object recognition. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018. 8

[5] Fabio Maria Carlucci, Paolo Russo, and Barbara Caputo. De$\hat{2}$ co: Deep depth colorization. *IEEE Robotics and Automation Letters*, 3(3):2386–2393, 2018. 2, 8

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9650–9660, October 2021. 1, 2

[7] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pre-training from pixels. In *International Conference on Machine Learning*, pages 1691–1703. PMLR, 2020. 2

[8] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021. 2

[9] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9640–9649, October 2021. 2

[10] Yuxin Chen, Ziqi Zhang, Chunfeng Yuan, Bing Li, Ying Deng, and Weiming Hu. Channel-wise topology refinement graph convolution for skeleton-based action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13359–13368, 2021. 2, 5, 8

[11] Yuxiao Chen, Long Zhao, Jianbo Yuan, Yu Tian, Zhaoyang Xia, Shijie Geng, Ligong Han, and Dimitris N. Metaxas. Hierarchically self-supervised transformer for human skeleton representation learning, 2022. 1, 2

[12] Ke Cheng, Yifan Zhang, Xiangyu He, Weihan Chen, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with shift graph convolutional network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 183–192, 2020. 2, 8

[13] Bruno Degardin, João Neves, Vasco Lopes, João Brito, Ehsan Yaghoubi, and Hugo Proença. Generative adversarial graph convolutional networks for human action synthesis. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1150–1159, 2022. 7

[14] Girum G Demisse, Konstantinos Papadopoulos, Djamila Aouada, and Bjorn Ottersten. Pose encoding for robust skeleton-based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 188–194, 2018. 1, 2

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6

[16] 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: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1, 2, 3, 5, 6, 7, 8

[17] Lin Geng Foo, Tianjiao Li, Hossein Rahmani, Qiuhong Ke, and Jun Liu. Unified pose sequence modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023. 5

[18] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020. 2

[19] 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 Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2022. 1

[20] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021. 1, 2, 3, 5, 6

[21] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019. 2, 6

[22] Mazin Hnewa and Hayder Radha. Object detection under rainy conditions for autonomous vehicles: a review of state-of-the-art and emerging techniques. *IEEE Signal Processing Magazine*, 38(1):53–67, 2020. 1

[23] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021. 2

[24] Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang. Robust pre-training by adversarial contrastive learn-

ing. *Advances in Neural Information Processing Systems*, 33:16199–16210, 2020. 2

[25] Jogendra Nath Kundu, Maharshi Gor, Phani Krishna Uppala, and Venkatesh Babu Radhakrishnan. Unsupervised feature learning of human actions as trajectories in pose embedding manifold. In *2019 IEEE winter conference on applications of computer vision (WACV)*, pages 1459–1467. IEEE, 2019. 8

[26] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *2011 IEEE international conference on robotics and automation*, pages 1817–1824. IEEE, 2011. 1, 2, 8

[27] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018. 3, 4

[28] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34:9694–9705, 2021. 2

[29] Linguo Li, Minsi Wang, Bingbing Ni, Hang Wang, Jiancheng Yang, and Wenjun Zhang. 3d human action representation learning via cross-view consistency pursuit. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4741–4750, 2021. 8

[30] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3595–3603, 2019. 8

[31] Tianjiao Li, Lin Geng Foo, Qiuhong Ke, Hossein Rahmani, Anran Wang, Jinghua Wang, and Jun Liu. Dynamic spatio-temporal specialization learning for fine-grained action recognition. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IV*, pages 386–403. Springer, 2022. 1

[32] Xuelong Li, Zhenghang Yuan, and Qi Wang. Unsupervised deep noise modeling for hyperspectral image change detection. *Remote Sensing*, 11(3):258, 2019. 3, 4

[33] Yaqian Liang, Shanshan Zhao, Baosheng Yu, Jing Zhang, and Fazhi He. Meshmae: Masked autoencoders for 3d mesh data analysis, 2022. 1, 2

[34] Lilang Lin, Sijie Song, Wenhan Yang, and Jiaying Liu. Ms2l: Multi-task self-supervised learning for skeleton based action recognition. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2490–2498, 2020. 8

[35] Haotian Liu, Mu Cai, and Yong Jae Lee. Masked discrimination for self-supervised learning on point clouds. *arXiv preprint arXiv:2203.11183*, 2022. 2

[36] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C Kot. Ntu rgb+ d 120: A large-scale benchmark for 3d human activity understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2684–2701, 2019. 2, 7, 8

[37] Jun Liu, Amir Shahroudy, Dong Xu, and Gang Wang. Spatio-temporal lstm with trust gates for 3d human action

recognition. In *European conference on computer vision*, pages 816–833. Springer, 2016. 1, 2, 7, 8

[38] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, October 2021. 6

[39] Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. Disentangling and unifying graph convolutions for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 143–152, 2020. 5, 8

[40] Mohammad Reza Loghmani, Mirco Planamente, Barbara Caputo, and Markus Vincze. Recurrent convolutional fusion for rgb-d object recognition. *IEEE Robotics and Automation Letters*, 4(3):2878–2885, 2019. 5, 8

[41] Kaleel Mahmood, Rigel Mahmood, and Marten Van Dijk. On the robustness of vision transformers to adversarial examples. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7838–7847, 2021. 2

[42] Xiaofeng Mao, Gege Qi, Yuefeng Chen, Xiaodan Li, Ranjie Duan, Shaokai Ye, Yuan He, and Hui Xue. Towards robust vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12042–12051, 2022. 2

[43] Nick Moran, Dan Schmidt, Yu Zhong, and Patrick Coady. Noisier2noise: Learning to denoise from unpaired noisy data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12064–12072, 2020. 2, 3, 7

[44] Qiang Nie and Yunhui Liu. View transfer on human skeleton pose: Automatically disentangle the view-variant and view-invariant information for pose representation learning. *International Journal of Computer Vision*, 129(1):1–22, 2021. 8

[45] Zhiliang Peng, Li Dong, Hangbo Bao, Qixiang Ye, and Furu Wei. Beit v2: Masked image modeling with vector-quantized visual tokenizers. *arXiv preprint arXiv:2208.06366*, 2022. 2

[46] Lingfeng Qiao, Zhongliang Jing, Han Pan, Henry Leung, and Wuji Liu. Private and common feature learning with adversarial network for rgbd object classification. *Neurocomputing*, 423:190–199, 2021. 5, 8

[47] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 2

[48] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1010–1019, 2016. 2, 7, 8

[49] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with directed graph neural networks. In *Proceedings of the IEEE/CVF Conference*

on Computer Vision and Pattern Recognition, pages 7912–7921, 2019. 8

[50] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12026–12035, 2019. 2, 8

[51] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Decoupled spatial-temporal attention network for skeleton-based action-gesture recognition. In *Proceedings of the Asian Conference on Computer Vision*, 2020. 1, 5, 7, 8

[52] Yichun Shi and Anil K Jain. Probabilistic face embeddings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6902–6911, 2019. 2, 3, 4, 7

[53] Richard Socher, Brody Huval, Bharath Bhat, Christopher D. Manning, and Andrew Y. Ng. Convolutional-recursive deep learning for 3d object classification. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, page 656–664, Red Hook, NY, USA, 2012. Curran Associates Inc. 8

[54] Jason L Speyer and Walter H Chung. *Stochastic processes, estimation, and control*. SIAM, 2008. 3

[55] Vladimiros Sterzentsenko, Leonidas Saroglou, Anargyros Chatzitofis, Spyridon Thermos, Nikolaos Zioulis, Alexandros Doumanoglou, Dimitrios Zarpalas, and Petros Daras. Self-supervised deep depth denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2

[56] Rolf Steyer and Werner Nagel. *Probability and conditional expectation: Fundamentals for the empirical sciences*, volume 5. John Wiley & Sons, 2017. 3

[57] Kun Su, Xiulong Liu, and Eli Shlizerman. Predict & cluster: Unsupervised skeleton based action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9631–9640, 2020. 8

[58] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10347–10357. PMLR, 18–24 Jul 2021. 6

[59] Qifei Wang, Gregorij Kurillo, Ferda Ofli, and Ruzena Bajcsy. Evaluation of pose tracking accuracy in the first and second generations of microsoft kinect. In *2015 international conference on healthcare informatics*, pages 380–389. IEEE, 2015. 1, 2

[60] Rui Wang, Dongdong Chen, Zuxuan Wu, Yinpeng Chen, Xiyang Dai, Mengchen Liu, Yu-Gang Jiang, Luowei Zhou, and Lu Yuan. Bevt: Bert pretraining of video transformers. In *CVPR*, 2022. 1, 2

[61] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022. 1, 2

[62] Yaochen Xie, Zhengyang Wang, and Shuiwang Ji. Noise2same: Optimizing a self-supervised bound for image denoising. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 20320–20330. Curran Associates, Inc., 2020. 2, 7

[63] Jun Xu, Yuan Huang, Ming-Ming Cheng, Li Liu, Fan Zhu, Zhou Xu, and Ling Shao. Noisy-as-clean: Learning self-supervised denoising from corrupted image. *IEEE Transactions on Image Processing*, 29:9316–9329, 2020. 3

[64] Shi Yan, Chenglei Wu, Lizhen Wang, Feng Xu, Liang An, Kaiwen Guo, and Yebin Liu. Ddrnet: Depth map denoising and refinement for consumer depth cameras using cascaded cnns. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 2

[65] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-second AAAI conference on artificial intelligence*, 2018. 2, 8

[66] Hao Yang, Dan Yan, Li Zhang, Yunda Sun, Dong Li, and Stephen J Maybank. Feedback graph convolutional network for skeleton-based action recognition. *IEEE Transactions on Image Processing*, 31:164–175, 2021. 8

[67] Siyuan Yang, Jun Liu, Shijian Lu, Meng Hwa Er, and Alex C. Kot. Skeleton cloud colorization for unsupervised 3d action representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13423–13433, October 2021. 8

[68] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022. 2

[69] Ping Yu, Yang Zhao, Chunyuan Li, Junsong Yuan, and Changyou Chen. Structure-aware human-action generation. In *European Conference on Computer Vision*, pages 18–34. Springer, 2020. 7

[70] Tianyuan Yu, Da Li, Yongxin Yang, Timothy M Hospedales, and Tao Xiang. Robust person re-identification by modelling feature uncertainty. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 552–561, 2019. 3, 4

[71] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19313–19322, 2022. 1, 2

[72] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021. 2

[73] Hasan FM Zaki, Faisal Shafait, and Ajmal Mian. Viewpoint invariant semantic object and scene categorization with rgb-d sensors. *Autonomous Robots*, 43(4):1005–1022, 2019. 2, 5, 8

[74] David Junhao Zhang, Kunchang Li, Yali Wang, Yunpeng Chen, Shashwat Chandra, Yu Qiao, Luoqi Liu, and

Mike Zheng Shou. Morphmlp: An efficient mlp-like backbone for spatial-temporal representation learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXV*, pages 230–248. Springer, 2022. 2

[75] Pengfei Zhang, Cuiling Lan, Wenjun Zeng, Junliang Xing, Jianru Xue, and Nanning Zheng. Semantics-guided neural networks for efficient skeleton-based human action recognition. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1112–1121, 2020. 8

[76] Nenggan Zheng, Jun Wen, Risheng Liu, Liangqu Long, Jianhua Dai, and Zhefeng Gong. Unsupervised representation learning with long-term dynamics for skeleton based action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. 8

[77] Daquan Zhou, Zhiding Yu, Enze Xie, Chaowei Xiao, Animashree Anandkumar, Jiashi Feng, and Jose M Alvarez. Understanding the robustness in vision transformers. In *International Conference on Machine Learning*, pages 27378–27394. PMLR, 2022. 2