Learning Canonical View Representation for 3D Shape Recognition with Arbitrary Views

Xin Wei[1*], Yifei Gong[2*], Fudong Wang[2], Xing Sun[2✉], Jian Sun[1✉]
[1]Xi'an Jiaotong University, [2] Tencent Youtu Lab
wxmath@stu.xjtu.edu.cn, {yifeigong,winfredsun}@tencent.com
fudong-wang@whu.edu.cn, jiansun@xjtu.edu.cn

Abstract

In this paper, we focus on recognizing 3D shapes from arbitrary views, i.e., arbitrary numbers and positions of viewpoints. It is a challenging and realistic setting for view-based 3D shape recognition. We propose a canonical view representation to tackle this challenge. We first transform the original features of arbitrary views to a fixed number of view features, dubbed canonical view representation, by aligning the arbitrary view features to a set of learnable reference view features using optimal transport. In this way, each 3D shape with arbitrary views is represented by a fixed number of canonical view features, which are further aggregated to generate a rich and robust 3D shape representation for shape recognition. We also propose a canonical view feature separation constraint to enforce that the view features in canonical view representation can be embedded into scattered points in a Euclidean space. Experiments on the ModelNet40, ScanObjectNN, and RGBD datasets show that our method achieves competitive results under the fixed viewpoint settings, and significantly outperforms the applicable methods under the arbitrary view setting.

1. Introduction

Understanding the 3D world is a fundamental problem in computer vision. One of its central challenges is how to represent and recognize objects in the 3D space. Recently, many view-based methods [7, 13, 14, 15, 20, 22, 23, 33, 34, 38, 40, 42, 43] were proposed to recognize 3D shape with multi-view 2D images based on the aggregation of features learned by deep neural networks. Leveraging advances in 2D image descriptors (*e.g.* [18]) and massive image databases [10], they are among the state-of-the-art methods for 3D shape recognition.

However, most of these methods [7, 13, 14, 15, 20, 23, 33, 34, 38, 40, 42, 43] focus on settings with a pre-defined camera setup where the same set of viewpoints are used for every object, *e.g.*, Fig. 1(a). In practical applications, 3D objects are often observed from arbitrary views without knowing their precise camera positions. In this work, we aim to tackle 3D shape recognition with arbitrary views. The setting can be defined as follows. (i) Views are taken from arbitrary viewpoints for each object. (ii) Objects have varying numbers of observation views, *e.g.*, Fig. 1(b).

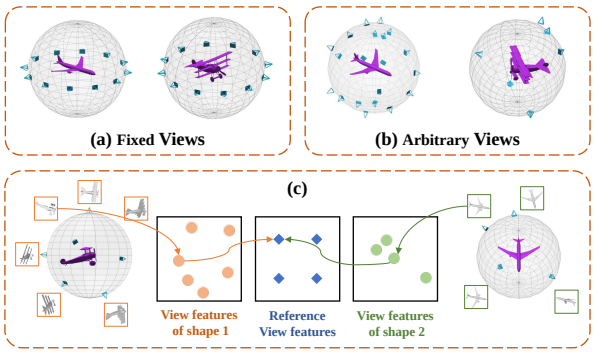


Figure 1. This paper addresses 3D shape recognition with arbitrary views as shown in (b), which is more challenging and realistic than the fixed-viewpoint setting in (a). As shown in (c), given an arbitrary number of unaligned view images, our method learns canonical view features of a 3D shape aligned to a fixed number of learnable reference view features using optimal transport.

Compared with the fixed-viewpoint setup, 3D shape recognition faces new challenges brought by the unaligned inputs from arbitrary views. It is difficult to robustly aggregate features of structurally unaligned views. Moreover, representations learned from a typical neural network are also mutually-unaligned in the feature space, where feature aggregation could result in a loss of discriminability.

To tackle these challenges, an intuitive motivation is to recover the inherent alignment for the arbitrary views. Specifically, if we find a link between the unaligned features from arbitrary views and a set of virtual reference views for observing an object, we can transform the features into

*Equal contribution.

aligned representations for the subsequent aggregation.

Driven by this motivation, we design a novel canonical view representation for 3D shape recognition with arbitrary views. Specifically, the input arbitrary views of each 3D shape are first processed by an image-level feature encoder consisting of a CNN and a Transformer encoder [36]. Then these features of arbitrary views are transformed into canonical view features aligned to a fixed number of learned reference view features. The transformation mapping is derived by the optimal transport [9, 16, 37]. To ensure that the canonical view features are distinct, we require that the canonical view features can be embedded into a Euclidean space (*e.g.*, $\mathbb{R}^3$) with mutually distant coordinates. In this way, each 3D shape is represented by a fixed number of features over the reference views in the feature space, resulting in the canonical representation of each 3D shape. The aligned canonical view features added with spatial embeddings are further encoded and aggregated to generate a discriminative global representation of the 3D shape.

Our main contributions can be summarized as follows. We tackle the challenge of 3D object recognition with arbitrary views by introducing a novel canonical view representation, which recovers the inherent mutual-alignment features among arbitrary views and produces a rich representation of the 3D shape. We further propose a canonical view feature separation loss to ensure feature separability, which improves the discriminability and robustness of the final representation. We conduct experiments on CAD, scanned model and real-world image datasets including ModelNet40 [41], ScanObjectNN [35] and RGBD [25] dataset. The results show that our approach significantly outperforms the state-of-the-art methods under the challenging setting of 3D shape recognition with arbitrary views.

2. Related Work

2.1. 3D shape recognition with multi-view images

View-based methods in 3D shape recognition have proved to be effective while only requiring 2D input images observed from different viewpoints. The key challenge of the view-based methods is how to effectively aggregate the features of multiples views to generate the shape descriptor.

MVCNN [33] is a framework that aggregates multi-view features with max-pooling, achieving superior performance against methods directly working on 3D inputs. Multi-view feature aggregation is further explored in GVCNN [15] where the view features are grouped to obtain more informative representation. Similarly, view-GCN [40] uses Graph Convolutional Neural Network to model the relations among different viewpoints to hierarchically aggregate features of multiple views. RotationNet [23] attempts to tackle the challenge of perturbed objects by predicting the object pose to represent the 3D shape in its aligned form. EMV[13] also tries to solve this problem with group convolution over discrete rotation groups. While achieving impressive performance, these approaches assume having a pre-defined set of viewpoints for each object. This makes them unfitted for the more practical setting where viewpoint positions are arbitrary and different for every object.

To the best of our knowledge, there are few works that go beyond the fixed viewpoints setup. DeepCCFV [22] tries to simulate a constraint-free camera setup in the testing phase and improve the generalization performance. However, it still assumes a pre-defined camera setup for the training data and the retrieval gallery, and the queries are sampled from the pre-defined viewpoints. OVCNet [26] attempts to tackle the task of shape recognition from any view, but mainly targets at the single-view scenario and relies on a challenging task of 3D reconstruction from a single image, while our method focuses on effectively aggregating multi-view images from arbitrary viewpoints.

Compared with the above-mentioned methods, our proposed method is also view-based, but we flexibly relax the fixed viewpoints setup to the arbitrary viewpoints setup. Our method achieves the state-of-the-art results in this challenging setting by employing a canonical view representation that aligns the image-level features of arbitrary views to a set of reference view features.

2.2. Transformer networks

Transformers [4, 11, 36] are initially introduced as an encoder-decoder architecture for machine translation, where the self-attention mechanism is incorporated to model the relationship among a set of inputs. To model the positional information of the sequential inputs, positional encodings are added to the input embedding. They are widely adopted in NLP for their scalability and good generalization performance.

Transformer networks are also proved to be effective for computer vision tasks [5, 6, 12, 17, 27, 39, 44]. DETR [5] is an object detection method based on Transformer, which encodes the image features and decodes the objects in parallel. VIT [12] demonstrated the feasibility of using Transformer as the backbone for image classification, and outperforms the popular CNNs.

In this work, we first utilize the Transformer encoder [36] as an effective way to explore the relationship among features of arbitrary views. After we transform the features into the canonical view representation, we use another Transformer encoder [36] to process the aligned canonical view features added with spatial embeddings, resulting in the final representation of the 3D shape.

3. Canonical View Representation

We first introduce our proposed canonical view representation for 3D shape recognition with arbitrary views, taken

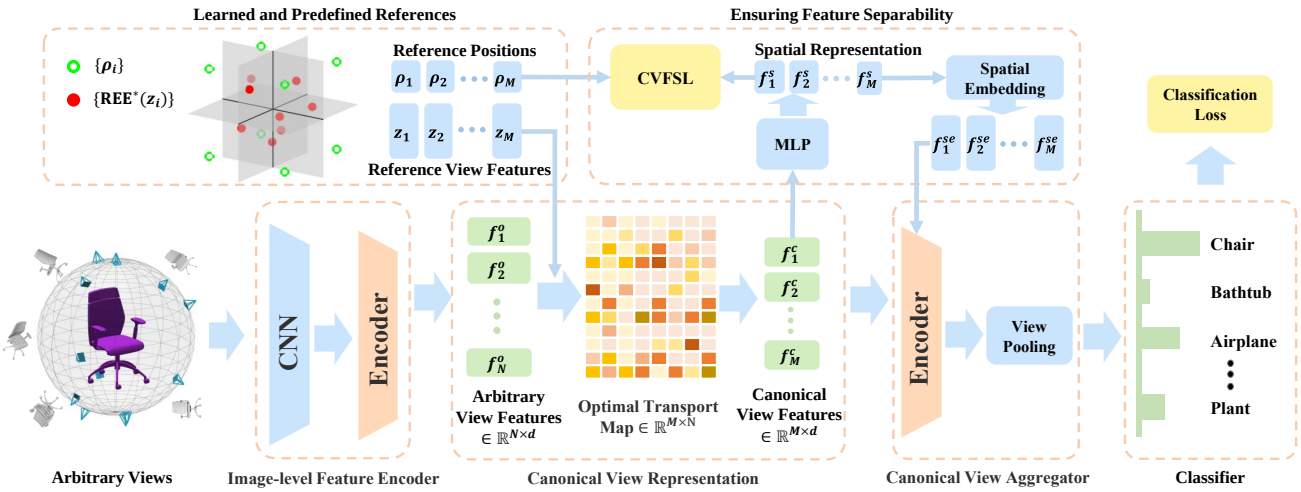


Figure 2. Overview of our approach. The network consists of three components, i.e., Image-level Feature Encoder (ILFE), Canonical View Representation (CVR), and Canonical View Aggregator (CVA). Images from N arbitrary views are first encoded by the ILFE, then the original unaligned features $F^o = \{f_i^o\}_{i=1}^N$ are transformed into a fixed number M of canonical view features $F^c = \{f_i^c\}_{i=1}^M$ aligned to the learned reference view features $Z = \{z_i\}_{i=1}^M$. Optimal transport is performed between F^o and Z to obtain the canonical view features F^c, while a novel Canonical View Feature Separation Loss (CVFSL) ensures canonical view features F^c to be distinct and separable. The CVA with spatial embeddings further explores the inter-viewpoint relationship and aggregates the canonical view features. *Robust Euclidean Embedding (REE) is used to visualize Z in an example 3D space.

as the basis of our 3D shape recognition network presented in Sect. 4. The major objective of this representation is to transform a set of arbitrary view features of a 3D shape to be a fixed number of view features, by learning and aligning to the same number of reference view features in the feature space. The optimally transformed features are dubbed ***canonical view representation*** of a 3D shape.

Suppose that we have extracted features from each view of 3D shape by the Image-level Feature Encoder (in Sect. 4.1). We next present how to model a set of reference view features in the feature space and transform the arbitrary view features to a canonical view representation based on optimal transport, as shown in the Fig. 3. In order to increase the discriminative ability of the canonical view representation, we also propose a constraint to ensure that the canonical view representations are separable in the feature space. Since the involved computations are differentiable, the computations for the canonical view representation will be taken as network modules in our 3D shape recognition network introduced in Sect. 4, and the reference view features and sub-nets in canonical view representation can be learned by network training.

3.1. Formulation

Given varying N arbitrary views of a 3D shape, we first extract their original features $F^o \triangleq \{f_i^o\}_{i=1}^N \in \mathbb{R}^{N\times d}$ by an image-level feature encoder (in Sect. 4.1). Then, to obtain a fixed number (M) of view features from the features F^o of N arbitrary views, we propose to find a feature transform $\mathcal{T} : \mathbb{R}^{N\times d} \rightarrow \mathbb{R}^{M\times d}$ such that $\mathcal{T}(F^o) \in \mathbb{R}^{M\times d}$. We assume that $\mathcal{T}$ is linear which is reasonable in high-dimensional feature space. Now we have

$$\mathcal{T}(F^o) \triangleq \mathbf{T}F^o, \quad f_j^t \triangleq \sum_i \mathbf{T}_{ji} f_i^o, \forall j = 1, ..., M, \quad (1)$$

where $\mathbf{T} \in \mathbb{R}^{M\times N}$ is a linear transform map implementing $\mathcal{T}$ and $F^t \triangleq \{f_j^t\}_{j=1}^M \in \mathbb{R}^{M\times d}$ are the transformed features taken as the candidate canonical view representations. We hope to find an optimal transform map $\mathbf{T}^*$ to construct F^t, which is detailed in the followings.

Reference view representation. We further specify the transform $\mathbf{T}$ as the mapping from N arbitrary view features to a fixed number (M) learnable reference view features $Z \triangleq \{z_j\}_{j=1}^M$, which can be seen as virtual reference views shared by all different 3D shapes. We define a similarity function $S(f_i^t, z_j)$ to measure similarity between f_i^t and z_j for $i \in [1, N], j \in [1, M]$, and solve the following optimization problem to find an optimal transform map $\mathbf{T}^*$:

$$\mathbf{T}^* \triangleq \underset{\mathbf{T}}{\text{argmax}} \sum_j S(f_j^t, z_j) = \sum_j S(\sum_i \mathbf{T}_{ji} f_i^o, z_j). \quad (2)$$

In this paper, a simple yet efficient definition of $S(\cdot, \cdot)$ is adopted as the linear inner product $S(f_i^t, z_j) \triangleq f_i^t \cdot z_j$.

Optimal transport solver. Due to the linearity of S, the optimization problem in Eq. (2) can be rewritten as

$$\mathbf{T}^* \triangleq \underset{\mathbf{T}}{\text{argmin}} \sum_{ij} -\mathbf{T}_{ji} S(f_i^o, z_j), \quad (3)$$

which can be solved by many linear programming algorithms [21, 30]. However, to guarantee the regularization for $\mathbf{T}$ and differentiability for the training procedure, we regularize $\mathbf{T}$ to be a doubly-stochastic matrix [28, 32] and add an entropy-based regularization term to Eq. (3):

$$\mathbf{T}^* \triangleq \underset{T}{\operatorname{argmin}} \sum_{ij} -\mathbf{T}_{ji} S(f_i^o, z_j) + \epsilon \sum_{ij} \mathbf{T}_{ji} \ln(\mathbf{T}_{ji}), \quad (4)$$

where $\epsilon \geq 0$ is a balance weight. Moreover, Eq. (4) is a well-known regularized optimal transport problem [3, 16, 29], that can be solved differentiably with the Sinkhorn algorithm [9].

Canonical view representation. Once the optimal $\mathbf{T}^*$ is solved, we get the canonical view features as $F^c \triangleq \{f_j^c\}_{j=1}^M$, where $f_j^c = \sum_i \mathbf{T}_{ji}^* f_i^o$. Thus the canonical view representation of a 3D shape with arbitrary views are the optimally transformed features under the alignment constraint w.r.t. the reference view features.

3.2. Canonical View Feature Separability

The canonical view representation obtained above for each 3D shape is length-and-order fixed benefiting from the reference view representation, but the resulting features might suffer from homogenization without proper constraint. Thus we propose the **C**anonical **V**iew **F**eature **S**eparation **L**oss (CVFSL) to instill separability among these features. More precisely, we require that the canonical view representation F^c of a 3D shape can be embedded into a spatial representation $F^s \in \mathbb{R}^{M \times k}$ such that F^s are scattered in the k-dimensional Euclidean space.

To achieve this goal, we utilize a two-layer MLP network $\Phi(\cdot)$ with a hidden dimension of 64 to extract the spatial representation $F^s \in \mathbb{R}^{M \times k}$ from $F^c \in \mathbb{R}^{M \times d}$, such that $F^s = \Phi(F^c)$. To make the spatial representation F^s scatter uniformly in the $\mathbb{R}^k$ space, we enforce the constraint that

$$\mathrm{L}_{sep} \triangleq \sum_{j=1}^{M} ||f_j^{s'} - \frac{\rho_j}{||\rho_j||}||_2^2, , \quad (5)$$

where $f^{s'}$ is the l_2-normalization of f^s, and reference positions $\mathrm{P} \triangleq \{\rho_j\}_{j=1}^M \in \{1, -1\}^k, M = 2^k$. When training our network (in Sect. 4) using this loss as one term, it enforces that the canonical view representation of each 3D shape are separable and discriminative. The effectiveness of this design is validated in Sect. 5.5.

4. Network Architecture

As shown in Fig. 2, our network for 3D shape recognition consists of three modules: the Image-level Feature Encoder (ILFE), the Canonical View Representation (CVR), and the Canonical View Aggregator (CVA). The Image-level Feature Encoder is composed of the CNN backbone and a Transformer encoder [36]. Given N arbitrary views $\{I_i\}_{i=1}^N$, the CNN backbone processes each view individually, and the Transformer encoder further processes the whole set of views to output a richer feature for each view, denoted by $F^o \triangleq \{f_i^o\}_{i=1}^N$. The Canonical View Representation module aims to align the features in F^o of arbitrary views to the reference view features $Z \triangleq \{z_j\}_{j=1}^M$ that are also learned during training. We compute a linear transformation map $\mathbf{T}^* \in \mathbb{R}^{M \times N}$ using optimal transport. It can optimally transform F^o to a fixed sized canonical view representation $F^c \triangleq \{f_i^c\}_{i=1}^M$ based on reference view features Z. F^c are then processed by the Canonical View Aggregator to derive a global feature for the 3D shape. In the CVA, a transformer encoder explores the relationship among the view features of the canonical view representation with spatial embedding, followed by a Global Average Pooling (GAP) layer to obtain the global feature for the 3D shape. We next introduce these network modules.

4.1. Image-level Feature Encoder

As shown in Fig. 2, the image-level feature encoder consists of the CNN backbone and the transformer encoder. Given N views $\{I_i\}_{i=1}^N$, the CNN backbone processes the images individually and produces the view features $F^v \triangleq \{f_i^v\}_{i=1}^N \in \mathbb{R}^{N \times d}$. The features F^v are then processed by a transformer encoder [36], where multi-head self-attention and the Feed-Forward Network (FFN) are utilized to extract the information among arbitrary views. In the self-attention layer, the queries, keys and values are obtained by linearly projecting the view features. Namely, the query Q, key K and value V are denoted as

$$Q \triangleq F^v W^Q, \quad K \triangleq F^v W^K, \quad V \triangleq F^v W^V, \quad (6)$$

where $W^Q \in \mathbb{R}^{d \times d}$, $W^K \in \mathbb{R}^{d \times d}$, and $W^V \in \mathbb{R}^{d \times d}$ are learnable linear weights. We utilize the Scaled Dot-Product Attention [36] defined as

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \quad (7)$$

Then the Multi-Head Attention (MHA) is calculated as

$$\begin{aligned} \text{MHA}(Q, K, V) &= \text{Concat}(head_1, ..., head_h)W^O \\ \text{where } head_h &= \text{Attention}(Q, K, V) \end{aligned} \quad (8)$$

Here $W^o \in \mathbb{R}^{hd \times d}$ reduces the dimension of the concatenated attention heads. The relationship among the arbitrary views are explored. The results are fed into an FFN [36], from which we obtain the image-level features denoted as $F^o \triangleq \{f_i^o\}_{i=1}^N$ of the input arbitrary views. FFN$(\cdot)$ [36] is a simple neural network using a two-layer MLP following the standard Transformer architecture. There are also residual connections and layer normalization [2] after every block.

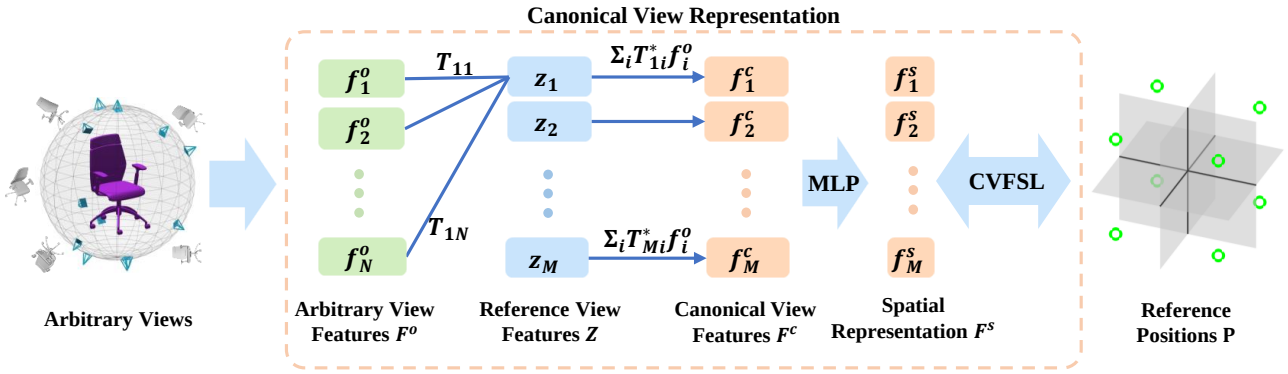


Figure 3. Illustration of the canonical view representation for 3D shape with arbitrary views. Given the unaligned features $F^o \triangleq \{f^o_i\}^N_{i=1}$ and reference view features $Z \triangleq \{z_j\}^M_{j=1}$, the transformation map $\mathbf{T}^* \triangleq \{T^*_{ji}\}$ is calculated with optimal transport in Eq. (4). The transformed feature $F^c \triangleq \{f^c_i\}^M_{i=1}$ are the aligned canonical view features. The Canonical View Feature Separation Loss (CVFSL) ensures that F^c are aware of the reference positions $\{\mathrm{P} = \rho_i\}^M_{i=1}$.

4.2. Canonical View Representation

As demonstrated in Sect. 3, the Canonical View Representation (CVR) module consists of three main operations, including (i) learning the reference view features Z, (ii) transforming the image-level features F^o into the canonical view features F^c with optimal transport, (iii) ensuring separability of the canonical view representation with CVFSL. The illustration of the process is shown in Fig. 3.

Update of Z. We first randomly initialize it as $Z^0 \in \mathbb{R}^{M\times d}$. Then, with the forwarded features $F^o \in \mathbb{R}^{N\times d}$, we construct the objective function in Eq. (4) that is solved differentiably with the Sinkhorn algorithm. In this way, the gradient of both Z and F^o can be calculated so as to update Z during training.

Transformation of F^o. Given F^o, we calculate the canonical view feature as the linearly-transformed features of F^o with the optimal transport map:

$$F^c = \mathbf{T}^* F^o, \tag{9}$$

where $\mathbf{T}^*$ is the solution of Eq. (4).

Separability constraint on F^c. With the canonical view features F^c, a two-layer MLP with hidden dimension of 64 is used to extract the spatial representation $F^s \in \mathbb{R}^{M\times k}$, by $F^s = \mathrm{MLP}(F^c)$. Then, we construct the canonical view feature separation constraint in Eq. (5) to enforce that F^s, inferred from the canonical view representation F^c, scatters uniformly in the $\mathbb{R}^{M\times k}$ space.

4.3. Canonical View Aggregator

The Canonical View Aggregator (CVA) further processes the canonical view features F^c along with the spatial representation F^s, and produces a global representation of the 3D shape. Given the canonical view representation $F^c \in \mathbb{R}^{M\times d}$, we explore the relationship between view features in the canonical view representation and aggregate them into a global feature F^g for the 3D shape.

Transformer encoder with spatial embedding. Given the spatial representation $F^s \in \mathbb{R}^{M\times k}$ calculated in the CVR, we obtain the spatial embedding $F^{se} \in \mathbb{R}^{M\times d}$ by $F^{se} = \Psi(F^s)$, where $\Psi(\cdot)$ is designed as a two-layer MLP network with 64 hidden units and a LeakyReLU layer. Thus we calculate the query Q, key K, and value V by

$$\begin{aligned} Q &\triangleq (F^c + F^{se})W^Q, \\ K &\triangleq (F^c + F^{se})W^K, \\ V &\triangleq F^c W^V. \end{aligned} \tag{10}$$

We then calculate the multi-head attention as in Eq. (8), after which the outputs are fed into a Feed Forward Network, resulting in the same number of features $F^{ce} \in \mathbb{R}^{M\times d}$.

Global representation of the 3D shape. We obtain the global representation $f^g \in \mathbb{R}^{1\times d}$ of the 3D shape by performing Global Average Pooling (GAP) on the outputs of the Transformer encoder F^{ce}, by $f^g = \mathrm{GAP}(F^{ce})$.

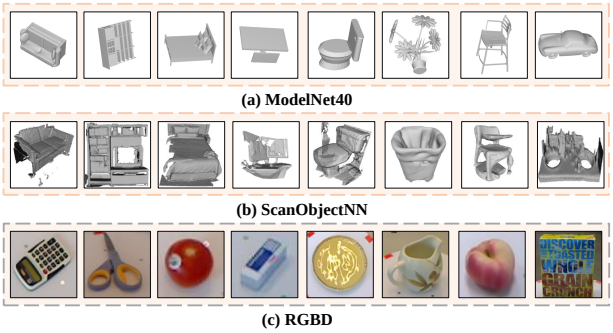
4.4. Classifier

We construct the classification module by a two-layer MLP network with hidden dimension of $\frac{d}{2}$. The output of the MLP is then fed into a softmax layer and the resulting logits represent the probability of each class.

4.5. Network Training

Training Loss. The training loss of our network consists of the classification loss L_{cls} and the Canonical View Feature Seperation Loss L_{sep}. The overall loss is defined as

$$\begin{aligned} \mathrm{L} &\triangleq \mathrm{L}_{cls} + \mathrm{L}_{sep} \\ &= -\sum_{c=1}^{C} y_c \log p_c + \lambda(\sum_{j=1}^{M} ||f^{s'}_j - \frac{\rho_j}{||\rho_j||}||^2_2), \end{aligned} \tag{11}$$



(a) ModelNet40

(b) ScanObjectNN

(c) RGBD

Figure 4. Examples of data in three different datasets.

where y_c and p_c are the true probability and predicted probability of class c, while $f_j^{s'}$ and ρ_j are defined in Eq. (5).

Hyper-parameters and backbone. We adopt the ResNet-18 [18] pretrained on ImageNet[10] as our CNN backbone network in Image-level Feature Encoder and set the feature dimension to $d = 512$. $M = 2^k$ is the number of canonical views, which affects how the Canonical View Representation module processes inputs. We use $M = 8, k = 3$ for experiments on ModelNet40 [41] and ScanobjectNN [35], while $M = 4, k = 2$ for RGBD [25]. Further discussion on the effect of M is in Sect. 5.5. λ is the weighting factor for the canonical view feature separation loss as in Eq. (11), which we set $\lambda = 0.1$ for the experiments. ϵ is the balance weight defined in (4), empirically set to 0.05.

Training Details. For all the experiments, we train our network for 60 epochs, with a batch-size of 20 on a NVIDIA V100 GPU. For the aligned and rotated setting, each batch contains 20 shapes with 400 multi-view images. For the arbitrary-view setting, the number of views for each shape varies. Variable-length view features from the CNN backbone network are zero-padded to the max number of views (20) and batched together. We use SGD with momentum as the optimizer. The initial learning rate, weight decay, momentum are 10^{-3}, 10^{-3}, 0.9 respectively. The learning rate follows the warm-up strategy [19] in the first epoch, and it linearly increases from 0 to 10^{-3}. Then it is reduced to 10^{-5} following a cosine quarter-cycle. Our code will be available on http://github.com/weixmath/CVR.

5. Experiments

We evaluate the performance of our method on multiple datasets including ModelNet40 [41], ScanObjectNN [35] and RGBD [35], examples of data in these datasets are shown in Fig. 4. For each dataset, we conduct experiments under the arbitrary view setting and the fixed viewpoint setting, where the 3D objects are either aligned or rotated. To keep the comparisons fair, we re-implement MVCNN [33] and GVCNN [15] denoted as MVCNN-M and GVCNN-M, and they utilize the exact same backbone network and training settings as ours.

5.1. Data Preparation

ModelNet40 and ScanObjectNN. For the arbitrary view setting on ModelNet40 [41] and ScanobjectNN [35], we generate the projected views with the following steps: (i) Randomly choose 6 to 20 points from a spherical surface as camera locations. (ii) Project the object from the chosen viewpoints to obtain the 2D views (cameras are assumed to point to the centroid of the object). For the fixed viewpoint setting with object rotation, we obtain the 2D views by first rotating the object around X-axis by a random angle between 0 and 180 degrees and then projecting the object from 20 fixed viewpoints that constitute a dodecahedron similar to [23, 40]. As for the aligned setting, we follow the setup used in other work like [23, 40]. We compare our performance with the state-of-the-art methods applicable to the specific setting. Note that since ScanObjectNN provides 3D models in the form of point clouds, we first reconstruct them into meshes with Poisson Surface Reconstruction [24].

RGBD dataset. The RGBD dataset [25] contains real-world pictures of objects from a large number of viewpoints, without providing 3D scans of these objects. Thus we simulate the arbitrary view settings by randomly sampling 4 to 12 images from each object instance. We also perform 10-fold cross validation on this dataset. In each round, we randomly leave one instance from each class out for testing while the rest are used in training.

5.2. Experiments on ModelNet40

This dataset consists of 12,311 3D shapes from 40 categories, with 9,483 training models and 2,468 test models for shape classification. It is the most widely adopted benchmark for 3D shape classification. Various methods reported results on this dataset using different shape representations including voxels, point clouds and multi-view images.

The experimental results on ModelNet40 [41] are shown in Tab. 1. Among the previous methods, view-GCN [40] and RotationNet [23] are two powerful methods and produce state-of-the-art results when the objects are aligned. However, their classification accuracies drop dramatically by more than 9.3% under the rotated object setting, in which the projected 2D images are not well aligned. Our method outperforms them by 3.97% per class and 5.22% per instance accuracy, showing that our method can obtain more robust representations from perturbed objects.

For the arbitrary view setting, we can see that our proposed method achieves notably better accuracy than the compared methods with the margin of 3.34% per instance and 3.03% per class accuracy. Note that RotationNet [23] and view-GCN [40] are not applicable to the arbitrary view setting because RotationNet [23] assumes pre-defined viewpoints while view-GCN [40] requires given fixed viewpoint positions to construct view-graph in training and testing.

Table 1. Shape classification accuracy (in %) on ModelNet40. 'NA' / '-': method is not applicable or result was not reported.

Method	Aligned		Rotated		Arbitrary Views	
	Per Class Acc.	Per Ins. Acc.	Per Class Acc.	Per Ins. Acc.	Per Class Acc.	Per Ins. Acc.
MVCNN-M	94.30%	96.35%	87.95%	88.17%	78.86%	83.20%
GVCNN-M	94.46%	96.07%	89.69%	88.10%	80.98%	83.57%
RotationNet [23]	-	97.37%	84.74%	85.29%	NA	NA
View-GCN [40]	**96.50%**	**97.60%**	85.90%	88.25%	NA	NA
Ours	95.77%	97.16%	**91.12%**	**92.22%**	**84.01%**	**86.91%**

Table 2. Shape classification accuracy (in %) on ScanObjectNN. 'NA' represents that the method is not applicable to this setting.

Method	Aligned		Rotated		Arbitrary Views	
	Per Class Acc.	Per Ins. Acc.	Per Class Acc.	Per Ins. Acc.	Per Class Acc.	Per Ins. Acc.
MVCNN-M	85.71%	87.82%	78.21%	80.62%	58.58%	63.29%
GVCNN-M	86.64%	88.68%	82.86%	83.70%	58.84%	65.35%
RotationNet [23]	84.88%	86.90%	74.68%	76.16%	NA	NA
View-GCN [40]	**88.67%**	90.39%	81.99%	83.50%	NA	NA
Ours	88.39%	**90.74%**	**84.70%**	**85.59%**	**68.07%**	**71.36%**

Table 3. Shape classification accuracy (in%) on RGBD.

Method	Setting	#View	Per Ins. Acc.
MDSICNN [1]	Fixed	≥ 120	89.6 %
CFK[8]		≥ 120	86.8 %
MMDCNN [31]		≥ 120	86.8 %
RotationNet [23]		12	89.3%
View-GCN [40]		12	**94.3**%
MVCNN-M	Arbitrary	4-12	89.0%
GVCNN-M			89.8 %
Ours			**91.8**%

5.3. Experiments on ScanObjectNN

ScanObjectNN [35] is a recently proposed real-world 3D object classification dataset with scanned indoor scene data. It contains around 15000 objects that are categorized into 15 categories with 2902 unique object instances. ScanObjectNN offers more practical challenges including background occurrence, object partiality, and different deformation variants. The results on ScanObjectNN are shown in Tab. 2. Under the arbitrary view setting, our approach significantly outperforms MVCNN-M and GVCNN-M by more than 6.01% and 9.23% on per-instance and per-class accuracy respectively. As for the fixed viewpoint setting, while our approach performs similarly with the current state-of-the-art on aligned objects, it achieves better results on rotated objects, improving per-instance and per-class accuracy by 2.09% and 2.71%.

5.4. Experiments on RGBD Dataset

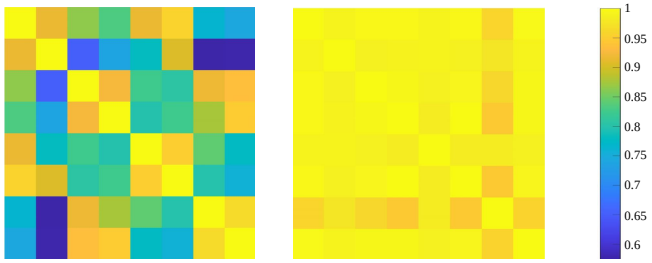
To further evaluate our method for recognizing real captured multi-view images, we conduct experiments on multi-view shape recognition with images from the RGBD dataset [25]. We randomly select images captured by camera with different elevation angles and the view number is varying from 4 to 12. As shown in Tab. 3, for the arbitrary view setting, our method outperforms MVCNN-M and GVCNN-M by 2%, which shows that our method is capable of dealing with real multi-view images captured from arbitrary views. Our method in arbitrary view setting (4 to 12 views) also exceeds the results of RotationNet in the setting of fixed 12 views.

5.5. Ablation Study

In this section, we take a closer look at the effects of key components of our network. Experiments are conducted on ModelNet40 under the arbitrary view setting.

Effects of image-level feature encoder. We examine the effects of the Image-level Feature Encoder (ILFE) defined in Sect. 4.1. We compare it with a baseline network that extracts features with a CNN and aggregates them with Max-pooling, the same structure as MVCNN [33]. To evaluate the effects of the ILFE, We use the ILFE instead of the CNN to extract the view features. As shown in Tab. 5, the ILFE brings 1.92% and 2.92% improvement on per-instance and per-class accuracies over the baseline. This proves the effectiveness of the Transformer that explores the relationship among arbitrary views.

Empirical analysis of canonical view representation. We evaluate the effects of the Canonical View Representation (CVR) module defined in Sect. 4.2, and our choice of using optimal transport to obtain canonical view features. As shown in Tab. 4, if we remove the CVR module completely, the per-instance and per-class accuracies drop by 1.87% and 2.00% respectively. Aside from optimal transport, Transformer decoder [36] is a popular network structure that can align an arbitrary number of input features into fixed sized features. While structurally identical to a Transformer encoder, a Transformer decoder takes learnable reference view features Z as query and image-level features



(a) (b)

Figure 5. Visualization of canonical view features F^c with (a) and without (b) the canonical view feature separation constraint. Each element of the matrix is the cosine similarity of paired canonical view features.

F^0 as key and value. We substitute the CVR module with a Transformer decoder while the rest of the network is kept unchanged. Surprisingly, the results further drop by 2.04% and 1.75%. This shows that optimal transport is a superior way to align features and brings a significant performance boost to our network.

Effects of canonical view feature separation constraint. We now study empirically how the Canonical View Feature Separation Loss (CVFSL) affects the performance. As shown in Tab. 5, with CVFSL, our method achieves notably better results with 1.59% and 1.73% improvements on per-instance and per-class accuracies. Moreover, we can observe in Fig. 5 (b) that without CVFSL, the resulting features have a flat similarity matrix. This means that canonical view features are not properly distinguishable in the feature space, resulting in a non-informative representation of the 3D shape. With CVFSL, the features are much more diverse as shown in (a), which can explain the larger performance gain with the CVFSL enabled. We also compare it with a cosine similarity loss that simply forces canonical view features to be different. The results drop by 1.06% and 1.21% in two accuracies. Therefore, we conclude that the canonical view feature separation constraint is vital in obtaining an informative canonical view representation and a robust final feature representation of the 3D shape.

Selection of the number of reference view features. In this paper, we have introduced the reference view features to which the features from arbitrary views are aligned. Here we evaluate the effect of the number (M) of reference view features. As shown in Tab. 6, different choices of M results in notable differences in performance on ModelNet40 with arbitrary views. Specifically, $M = 8$ results in the best performance, followed by $M = 16$, and $M = 4$ at last. We can infer that on the arbitrary-view setting of ModelNet40, $M = 8$ is preferable. Note that this result may vary on different datasets and settings with different distributions of viewpoints.

Effects of the canonical view aggregator. Now we examine the effects of the Canonical View Aggregator (CVA) module defined in Sect. 4.3. We remove the CVA from our network, instead we directly perform view pooling on the output features from CVR. Comparing the results shown in Tab. 5, we find that removing the CVA leads to a performance drop of 1.16% and 1.97% on per-instance and per-class accuracy. Therefore, it is shown that the CVA module, which makes use of the aligned spatial encoding and further models the relationship among features in canonical representation, is crucial to the performance of our network.

Table 4. Comparison of optimal transport with Transformer decoder in CVR.

	Per Class Acc.	Per Ins. Acc.
w/o CVR	82.01%	85.04%
Transformer decoder	79.97%	83.29%
Optimal transport (ours)	**84.01%**	**86.91%**

Table 5. Ablation study on each module of our network.

ILFE	CVR	CVA	CVFSL	Per Class Acc.	Per Ins. Acc.
				78.86%	83.20%
✓				81.78%	85.12%
✓	✓		✓	82.04%	85.75%
✓	✓	✓		82.28%	85.32%
✓	✓	✓	✓	**84.01%**	**86.91%**

Table 6. Results by choosing different number of reference view features.

	Per Class Acc.	Per Ins. Acc.
M = 4	82.41%	85.69%
M = 8	**84.01%**	**86.91%**
M = 16	82.71%	86.22%

6. Conclusion and discussion

In this work, we propose a novel canonical view representation to tackle the challenge of 3D shape recognition with arbitrary views. We incorporate optimal transport with the canonical view feature separation constraint to transform the features of arbitrary views into an aligned canonical view representation, enabling us to aggregate and derive a rich and robust feature representation for the 3D shape. The experimental results prove the effectiveness of our method. As discussed in Sect. 3 and Sect. 4, the learned reference view features in Z set up a common reference for aligning arbitrary views to a fixed number of learnable reference views. This approach can potentially be applied to other multi-view vision tasks, such as view synthesis, view-based 3D reconstruction or generation, which we may investigate in our future work.

Acknowledgment This work was supported by NSFC with grant numbers of 11971373, U20B2075, 11690011, U1811461, 12026605, 12090021, 61721002, and National Key R&D Program 2018AAA0102201.

References

[1] Umar Asif, Mohammed Bennamoun, and Ferdous A Sohel. A multi-modal, discriminative and spatially invariant cnn for rgb-d object labeling. *IEEE TPAMI*, 40(9):2051–2065, 2017. 7

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 4

[3] Jean-David Benamou, Guillaume Carlier, Marco Cuturi, Luca Nenna, and Gabriel Peyré. Iterative bregman projections for regularized transportation problems. *SIAM Journal on Scientific Computing*, 37(2):A1111–A1138, 2015. 4

[4] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, volume 33, pages 1877–1901, 2020. 2

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020. 2

[6] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. *arXiv preprint arXiv:2012.00364*, 2020. 2

[7] Jiaxin Chen, Jie Qin, Yuming Shen, Li Liu, Fan Zhu, and Ling Shao. Learning attentive and hierarchical representations for 3d shape recognition. In *ECCV*, 2020. 1

[8] Yanhua Cheng, Rui Cai, Xin Zhao, and Kaiqi Huang. Convolutional fisher kernels for rgb-d object recognition. In *3DV*, pages 135–143. IEEE, 2015. 7

[9] Marco Cuturi. Sinkhorn distances: lightspeed computation of optimal transport. In *NIPS*, volume 2, page 4, 2013. 2, 4

[10] 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. IEEE, 2009. 1, 6

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2018. 2

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[13] Carlos Esteves, Yinshuang Xu, Christine Allen-Blanchette, and Kostas Daniilidis. Equivariant multi-view networks. In *ICCV*, pages 1568–1577, 2019. 1, 2

[14] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *AAAI*, volume 33, pages 3558–3565, 2019. 1

[15] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *CVPR*, pages 264–272, 2018. 1, 2, 6

[16] Sira Ferradans, Nicolas Papadakis, Gabriel Peyré, and Jean-François Aujol. Regularized discrete optimal transport. *SIAM Journal on Imaging Sciences*, 7(3):1853–1882, 2014. 2, 4

[17] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *arXiv preprint arXiv:2012.09688*, 2020. 2

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1, 6

[19] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *CVPR*, pages 558–567, 2019. 6

[20] Xinwei He, Tengteng Huang, Song Bai, and Xiang Bai. View n-gram network for 3d object retrieval. In *ICCV*, 2019. 1

[21] Frederick S. Hillier. *Linear Programming*. Springer US, Boston, MA, 2013. 4

[22] Zhengyue Huang, Zhehui Zhao, Hengguang Zhou, Xibin Zhao, and Yue Gao. Deepccfv: Camera constraint-free multi-view convolutional neural network for 3d object retrieval. In *AAAI*, volume 33, pages 8505–8512, 2019. 1, 2

[23] Asako Kanezaki, Yasuyuki Matsushita, and Yoshifumi Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *CVPR*, pages 5010–5019, 2018. 1, 2, 6, 7

[24] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *SGP*, volume 7, 2006. 6

[25] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *ICRA*, pages 1817–1824. IEEE, 2011. 2, 6, 7

[26] Sainan Liu, Vincent Nguyen, Isaac Rehg, and Zhuowen Tu. Recognizing objects from any view with object and viewer-centered representations. In *CVPR*, pages 11784–11793, 2020. 2

[27] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021. 2

[28] M. MARCUS and R. REE. Diagonals of doubly stochastic matrices. *The Quarterly Journal of Mathematics*, 10(1):296–302, 1959. 4

[29] Grégoire Mialon, Dexiong Chen, Alexandre d'Aspremont, and Julien Mairal. A trainable optimal transport embedding for feature aggregation and its relationship to attention. In *ICLR*, 2021. 4

[30] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, 1994. 4

[31] Mohammad Muntasir Rahman, Yanhao Tan, Jian Xue, and Ke Lu. Rgb-d object recognition with multimodal deep convolutional neural networks. In *ICME*, pages 991–996. IEEE, 2017. 7

[32] Richard Sinkhorn. Diagonal equivalence to matrices with prescribed row and column sums. *The American Mathematical Monthly*, 74(4):402–405, 1967. 4

[33] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks

for 3d shape recognition. In *ICCV*, pages 945–953, 2015. 1, 2, 6, 7

[34] Jong-Chyi Su, Matheus Gadelha, Rui Wang, and Subhransu Maji. A deeper look at 3d shape classifiers. In *Second Workshop on 3D Reconstruction Meets Semantics, ECCV*, 2018. 1

[35] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *CVPR*, pages 1588–1597, 2019. 2, 6, 7

[36] 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*, volume 30, 2017. 2, 4, 7

[37] Cédric Villani. *Optimal transport–Old and new*. 2008. 2

[38] Chu Wang, Marcello Pelillo, and Kaleem Siddiqi. Dominant set clustering and pooling for multi-view 3d object recognition. In *BMVC*, 2017. 1

[39] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, 2021. 2

[40] Xin Wei, Ruixuan Yu, and Jian Sun. View-gcn: View-based graph convolutional network for 3d shape analysis. In *CVPR*, pages 1850–1859, 2020. 1, 2, 6, 7

[41] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, 2015. 2, 6

[42] Ze Yang and Liwei Wang. Learning relationships for multi-view 3d object recognition. In *ICCV*, pages 7505–7514, 2019. 1

[43] Tan Yu, Jingjing Meng, and Junsong Yuan. Multi-view harmonized bilinear network for 3d object recognition. In *CVPR*, pages 186–194, 2018. 1

[44] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. *arXiv preprint arXiv:2012.09164*, 2020. 2