Self-supervised Spatial Reasoning on Multi-View Line Drawings

Siyuan Xiang* ‡ Anbang Yang* ‡ Yanfei Xue‡ Yaoqing Yang§ Chen Feng† ‡
‡New York University Tandon School of Engineering §University of California, Berkeley
https://ai4ce.github.io/Self-Supervised-SPARE3D/

Abstract

Spatial reasoning on multi-view line drawings by state-of-the-art supervised deep networks is recently shown with puzzling low performances on the SPARE3D dataset [14]. Based on the fact that self-supervised learning is helpful when a large number of data are available, we propose two self-supervised learning approaches to improve the baseline performance for view consistency reasoning and camera pose reasoning tasks on the SPARE3D dataset. For the first task, we use a self-supervised binary classification network to contrast the line drawing differences between various views of any two similar 3D objects, enabling the trained networks to effectively learn detail-sensitive yet view-invariant line drawing representations of 3D objects. For the second type of task, we propose a self-supervised multi-class classification framework to train a model to select the correct corresponding view from which a line drawing is rendered. Our method is even helpful for the downstream tasks with unseen camera poses. Experiments show that our method could significantly increase the baseline performance in SPARE3D, while some popular self-supervised learning methods cannot.

1. Introduction

Human visual reasoning, especially spatial reasoning, has been widely studied from psychological and educational perspectives [17, 19]. Researches show that trained humans can achieve good performance on spatial reasoning tasks [32] because they can solve these tasks using spatial memory, logic, and imagination. However, the spatial reasoning of deep networks is yet to be explored and improved. In other visual learning tasks such as image classification, object detection, and segmentation, state-of-the-art deep networks have shown their superior performance to humans by memorizing indicative visual patterns from enormous image instances for prediction. However, it seems difficult for deep networks to reason using the same mechanism about the spatial information such as the view consistency and camera poses from 2D images [14].

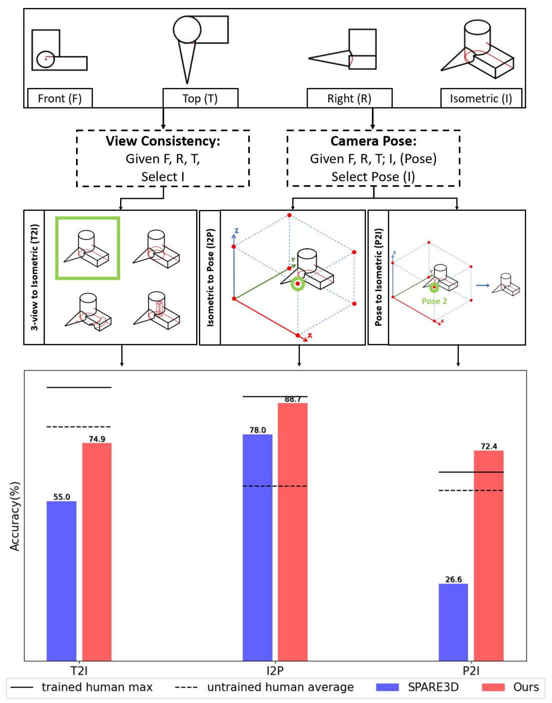

Figure 1. We significantly improve SPARE3D baselines using self-supervised learning approaches. Note that in this figure, all the networks are trained using the same amount of data (data generated from 5,000 CAD models) as used on the SPARE3D dataset.

To our best knowledge, we are the first to investigate spatial reasoning tasks in the SPARE3D[1] dataset [14], that provides several challenging spatial reasoning tasks in line drawings (Figure 1). The dataset is unique in that: (1) it uses line drawings as inputs, (2) it is a non-categorical dataset, meaning there is no class label information for each object, (3) it defines spatial reasoning instead of recognition tasks. Specifically, we focus on the view-consistency reasoning task (the Three-view to Isometric or *T2I*) and the two camera-pose reasoning tasks (the Isometric to Pose or *I2P*, and the Pose to Isometric or *P2I*). The task examples are illustrated in Figure 1 and for more details of the task

*Equal contribution.
†The corresponding author is Chen Feng cfeng@nyu.edu.

[1]Note that we use the latest dataset and benchmark results updated by the SPARE3D authors after CVPR 2020.

settings, we refer the readers to the SPARE3D paper [14].

To improve the DNN's performance, our first effort is to explore the supervised learning network's ability in three aspects: (1) the quantity of data used for learning, (2) the network's capacity (width and depth), and (3) the network's structure. However, we find changing these factors could not bring significant improvement (all these results are shown in the supplementary), while it might cost more time and computational resources to generate data for supervised learning.

Compared with supervised learning, self-supervised learning is helpful when a large number of unlabeled data are available. Moreover, in many visual learning tasks, self-supervised learning as a pre-training task can learn better visual representations that can be further fine-tuned via supervision for downstream tasks, achieving similar or even better results than supervised learning [6]. Based on these findings, we design two self-supervised learning methods for both view consistency reasoning task and camera pose reasoning task on the SPARE3D dataset.

For the view consistency reasoning task, we design a contrastive spatial reasoning method to tackle the challenges in SPARE3D tasks (Figure 2). This is necessary because most of the existing contrastive learning methods do not explicitly consider the relationship between different views, nor do they force deep networks to focus on detailed differences between images. We demonstrate both qualitative and quantitative results that show *our method helps deep networks capture these small details while being view invariant*, which is crucial for the view consistency task.

For the camera pose reasoning task, we propose a self-supervised learning network for improving the deep network's ability to find the correlation between the pose and the image rendered from the pose (see Figure 3). Our experiment results show that our self-supervised learning network is not only helpful for the downstream tasks with seen views but also helpful for the tasks with unseen views. This shows a potential to even improve pose estimation by transferring the learned representations from a camera pose reasoning pretext task to related downstream tasks.

In sum, our major contributions are:

- A novel contrastive learning method by self-supervised binary classifications, which enables deep networks to effectively learn detail-sensitive yet view-invariant multi-view line drawing representations for view consistency reasoning task;
- A self-supervised multi-class classification method to learn pose-aware representations from multi-view line drawing images;
- Significantly improved spatial reasoning performance of deep networks on line drawings based on the above, some of which surpass human performance.

2. Related Work

Self-supervised learning has achieved great success due to the performance improvement on many visual learning tasks [11, 12, 18, 21, 23, 26, 28, 36, 39, 43]. Successful self-supervised learning frameworks use different tricks to create artificial labels based solely on the inputs. One way is to use spatial information or spatial relationships between image patches in a single image. For example, Gidaris et al. [10] designed the pretext task by asking the network to predict image rotations. Another way is to ask the network to recover the positions of shuffled image patches [20, 29, 37], or predict the relative position [8]. In addition, the color signal contained in an RGB image could also be used. By recovering the color for grayscale images generated from RGB images, networks can learn the semantic information of each pixel [24, 46]. Despite the success of the spatial reasoning tasks in the SPARE3D dataset, these methods are ineffective since they only use visual information from a single image. Our method aims to consider the common information between different images and, thus, is more suitable to solve those reasoning problems.

One way of grouping various self-supervised learning methods is to divide them into generative vs. contrastive ones [27]. On the one hand, generative ones learn visual representations via pixel-wise loss computation and thus forcing a network to focus on pixel-based details; on the other hand, contrastive ones learn visual representations by contrasting the positive and negative pairs [1, 38, 40, 41]. Many researchers explored contrastive learning by comparing shared information between multiple positive or negative image pairs. These methods often attempt to minimize the distance of the features extracted from the same data source and maximize the distance between features from different data sources in the feature space. Hjelm et al. [16] propose Deep InfoMax based on the idea that the global feature extracted from an image should be similar to the same image's local feature and should be different from a different image's local feature. Based on this method, Bachman et al. [3] further use image features extracted from different layers to compose more negative or positive image pairs. SimCLR [5] differs from the previous two methods in that it only considers the global features of the augmented image pairs to compose positive and negative pairs. SimCLRv2 [6] make further improvements on Imagenet [7] by conducting contrastive learning with a large network, fine-tuning using labeled data, and finally distilling the network to a smaller network. Compared to SimCLR, MoCo [15] stores all generated samples to a dictionary and uses them as negative pairs instead of generating negative pairs in each step. These tactics could help reduce the batch size requirement while still achieving good performance. Differently, SwaV [4] trains two networks that can interact and learn from each other, with one network's input being the

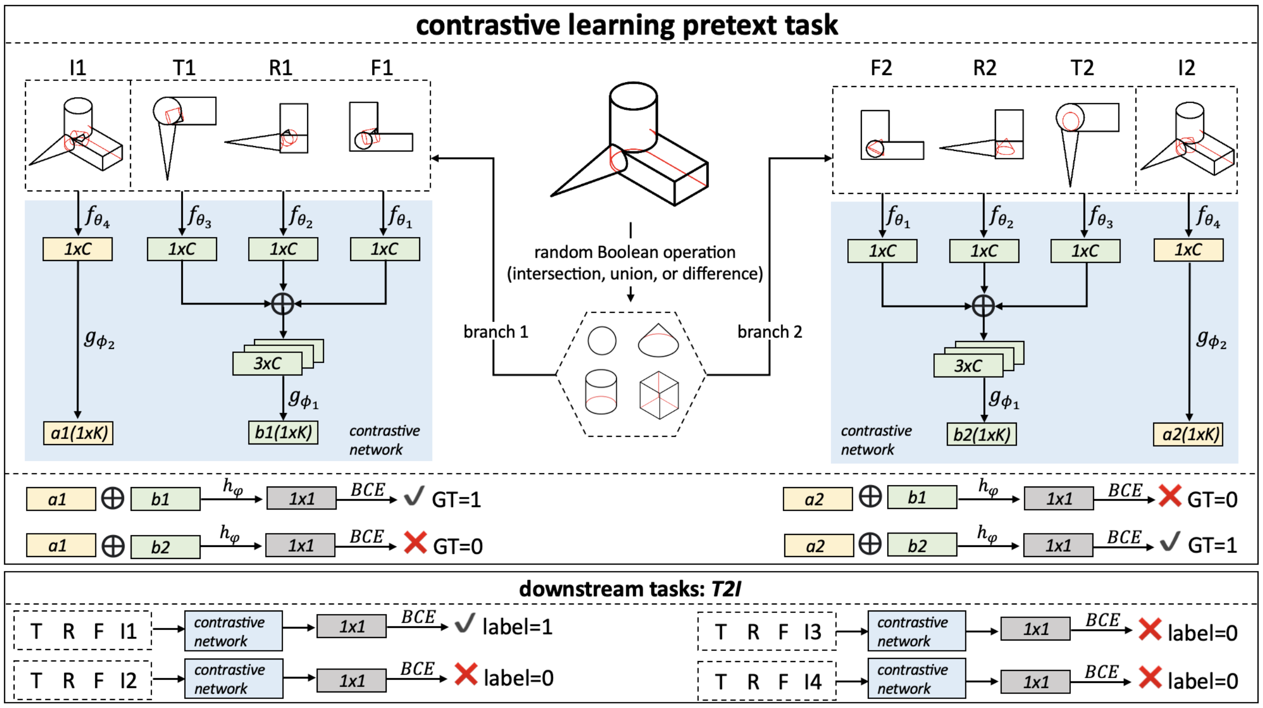

Figure 2. **Contrastive spatial reasoning architecture for task *T2I*.** We train a contrastive learning network, then use the learned representations to the downstream task *T2I*. Front (F), Right (R), Top (T) represent line drawings and an isometric (I) line drawing. a_1, b_1, a_2, b_2 are the encoded feature vectors; C, K are the dimension of the latent vectors. The f_{θ_1}, f_{θ_2}, f_{θ_3}, and f_{θ_4} are CNN networks; g_{ϕ_1}, g_{ϕ_2}, and h_ψ are MLP networks. $\bigoplus$ is a concatenation operation. BCE represents binary cross-entropy loss.

augmented pair of another network's input. Besides, researchers also theoretically analyze why these contrastive learning methods work well [2, 25, 34, 35].

However, *it is difficult to apply the aforementioned contrastive learning methods directly to tasks which requires consideration of the relationships between multi-view images*. Contrastive multiview coding [33] has a misleading name in our context because that "multiview" in fact, means different input representations instead of views from different camera poses. Therefore it is not suitable for SPARE3D tasks. Kim et al. [22] propose a method to solve a few-shot visual reasoning problem on RAVEN dataset [44], which is perhaps more relevant to us due to the use of contrastive learning in visual reasoning. Yet because it is designed for analogical instead of spatial reasoning, it is not directly applicable to SPARE3D tasks either.

3. Self-supervised Spatial Reasoning

In this section, we explore a self-supervised learning method to learn the representations for all three tasks in SPARE3D: a contrastive spatial reasoning network for view consistency reasoning task *T2I*, a self-supervised learning network for camera pose reasoning task *I2P* and *P2I*. For the camera pose reasoning task, we extend the downstream task settings to show our method is helpful when the camera poses in the downstream task are not seen in the pretext task.

3.1. Contrastive learning network for task *T2I*

According to the discussion in the Introduction 1, we believe contrastive learning can help the network learn better line drawing representations. Because it is not difficult to obtain a large number of unlabeled CAD models, we design our specialized contrastive spatial reasoning method as illustrated in Figure 2. Our method can be divided into three steps: 3D model augmentation, line-drawing feature extraction, contrastive loss computation.

Step 1: 3D data augmentation. Our data augmentation happens in 3D instead of 2D because the visual differences between different views are caused by 3D boolean operation. We generate two sets of images: a branch 1 set and a branch 2 set for each augmented CAD model. Each image set contains the three-view drawings and the isometric view drawing. We denote the branch 1 set by $\{F_1, R_1, T_1, I_1\}$, and the branch 2 set by $\{F_2, R_2, T_2, I_2\}$, where F, R, T, and I stand for front, right, top views and the isometric view separately. The branch 1 and branch 2 image sets are generated from two different modifications to the original CAD model. Each modification is a random Boolean operation (intersection, union, or difference) on the original CAD model with a random primitive (sphere, cube, cone, or cylinder.) Figure 2 gives an example of generating the two image sets.

Step 2: line-drawing feature extraction. In this paper, f, g, h represent neural networks; θ, ϕ, ψ are the network weights in the three networks respectively. F_i, R_i, T_i, and I_i ($i \in \{1,2\}$) are fed into four convolutional neural networks (CNN) individually. The four networks are denoted by $f_{\theta_1}, f_{\theta_2}, f_{\theta_3}$, and f_{θ_4}, where $f_{\theta_j} : \mathbb{R}^{3\times H\times W} \to \mathbb{R}^C, j \in \{1,2,3,4\}$; H and W are the height and width of the images. Note that the four networks share the *same architecture* but with *different parameters*. Then, the outputs from f_{θ_1}, f_{θ_2}, and f_{θ_3} are concatenated and fed into a one-layer MLP $g_{\phi_1} : \mathbb{R}^{3C} \to \mathbb{R}^K$, and the output from f_{θ_4} is fed into another one-layer MLP $g_{\phi_2} : \mathbb{R}^C \to \mathbb{R}^K$. g_{ϕ_1}, g_{ϕ_2} also share the *same architecture* but with *different parameters*. The outputs from g_{ϕ_1} and g_{ϕ_2} are noted as a a_i and b_i ($i \in \{1,2\}$) , which encode the information from the 3-view images and the isometric image respectively.

Step 3: Contrastive loss computation. After having the four latent codes (a_1, a_2, b_1, and b_2), we concatenate each a and each b, which gives four combinations $a_1 \bigoplus b_1$, $a_1 \bigoplus b_2$, $a_2 \bigoplus b_1$, and $a_2 \bigoplus b_2$ (see Figure 2). Then we send the four concatenated latent codes to a binary classifier $h_\psi : \mathbb{R}^{2K} \to (0,1)$, where the h_ψ is a two-layer MLP. The outputs from h_ψ are $\hat{p}_1, \hat{p}_2, \hat{p}_3$, and $\hat{p}_4$ respectively, and are used to compute the binary cross entropy (BCE) loss with the ground truth. We define the ground truth to be 1 if the original two latent codes used to concatenate are from the same image pairs, and the ground truth to be 0 from different image pairs. Therefore, $p_1 = p_4 = 1, p_2 = p_3 = 0$. The final loss is $\frac{1}{4}\sum_{k=1}^{4} BCE(\hat{p}_k, p_k)$, ($k \in \{1,2,3,4\}$).

Difference with SimCLR. Unlike existing contrastive learning methods such as SimCLR, the representation from our method is designed to be both multi-view-consistent and detail-sensitive. Technically, our method differs from them in that: 1) our data augmentation operates on 3D CAD models with Boolean operations (intersection, union, or difference), instead of single-image operations like random cropping, color distortion, and Gaussian blur; 2) our positive pairs are two sets of multi-view line drawings (three-view line drawings and isometric line drawing) that are rendered in the same data branch, and our negative pairs are image sets from different data branches, unlike being sampled from other data instances; 3) we use binary cross-entropy loss to optimize the network, unlike the NT-Xent loss [13, 45] which has the temperature parameter to tune. Our experimental results also show the advantage of using BCE loss over the NT-Xent loss.

Representation adaption for task *T2I*. We notice that all the learned parameters θ, ϕ, ψ can be loaded to the neural network for further supervised fine-tuning because the contrastive spatial reasoning method is just a pre-training step, and it uses exactly the same network architecture as the network for the supervised learning (see Figure 2 downstream tasks: *T2I*).

3.2. Self-supervised learning network for task *I2P* and *P2I*

We propose a self-supervised learning pretext task, as can be seen in Figure 3. As mentioned in Introduction 1, the learned representations from the pretext task can be helpful to both *I2P* and *P2I* tasks, even when the camera poses in the downstream tasks are not seen in the pretext task. The network design can be divided into two steps: line-drawing feature extraction; loss computation.

Step 1: line-drawing feature extraction. For the three view images, similar as the network design for contrastive spatial reasoning for task *T2I*, $\{F, R, T\}$ are sent to the share-weight neural network (CNN) $f_{\eta_1}, f_{\eta_2}, f_{\eta_3}$ separately to obtain three features, where $f_{\eta_j} : \mathbb{R}^{3\times H\times W} \to \mathbb{R}^C, j \in \{1,2,3\}$. Then the three vectors are concatenated and fed into a one-layer MLP $g_{\omega_1} : \mathbb{R}^{3C} \to \mathbb{R}^K$, generating c_1 which encodes the information from the 3-view images.

For the eight isometric view images, each of the images goes through the same network. For the sake of simplicity, we use I_1 (isometric view image rendered from camera pose 1) as an example. I_1 is sent to a neural network (CNN) f_{η_4} followed by a one-layer MLP $g_{\omega_2} : \mathbb{R}^C \to \mathbb{R}^K$, and the output latent vector is noted as d_1.

After having c_1 and d_1 for three view feature and isometric view feature separately, we concatenate them and send the concatenated vector to a two-layer MLP $g_{\omega_3} : \mathbb{R}^K \to \mathbb{R}^8$. The obtained 8-dimensional vector e_1 can represent the camera-pose probability logits of the eight candidate isometric views. Therefore, for all the eight isometric view images $I_1, I_2, \ldots, I_8$, we will have eight 1×8 vectors, $e_1, e_2, \ldots, e_8$ separately. We concatenate the eight vectors to obtain a 8×8 matrix, forming the output of the self-supervised learning pretext task.

Step 2: loss computation. This matrix is used to calculate the BCE loss with the ground truth matrix. The ground truth matrix is a 8×8 identity matrix, and each row is the logits for the corresponding isometric view. For example, the logit value of the first value in the first row is 1, while other values are 0, since the logits in the first row represents the isometric view pose 1. The BCE loss is: $\frac{1}{64}\sum_{i=1}^{8}\sum_{j=1}^{8} BCE(\hat{p}_{ij}, p_{ij})$, $(i, j \in 1, 2, \cdots, 8)$, the ground truth $p_{ij} = 1$ when $i = j$, otherwise $p_{ij} = 0$.

Representation adaption for task *I2P* and *P2I*. In task *I2P*, three view images and one isometric view image (rendered from one of the poses in view $1, 2, 5, 6$, pose defined in SPARE3D dataset) are given, and the network is asked to select the correct viewpoint from view $1, 2, 5, 6$. Therefore, for each question, we send the three-view image F, R,

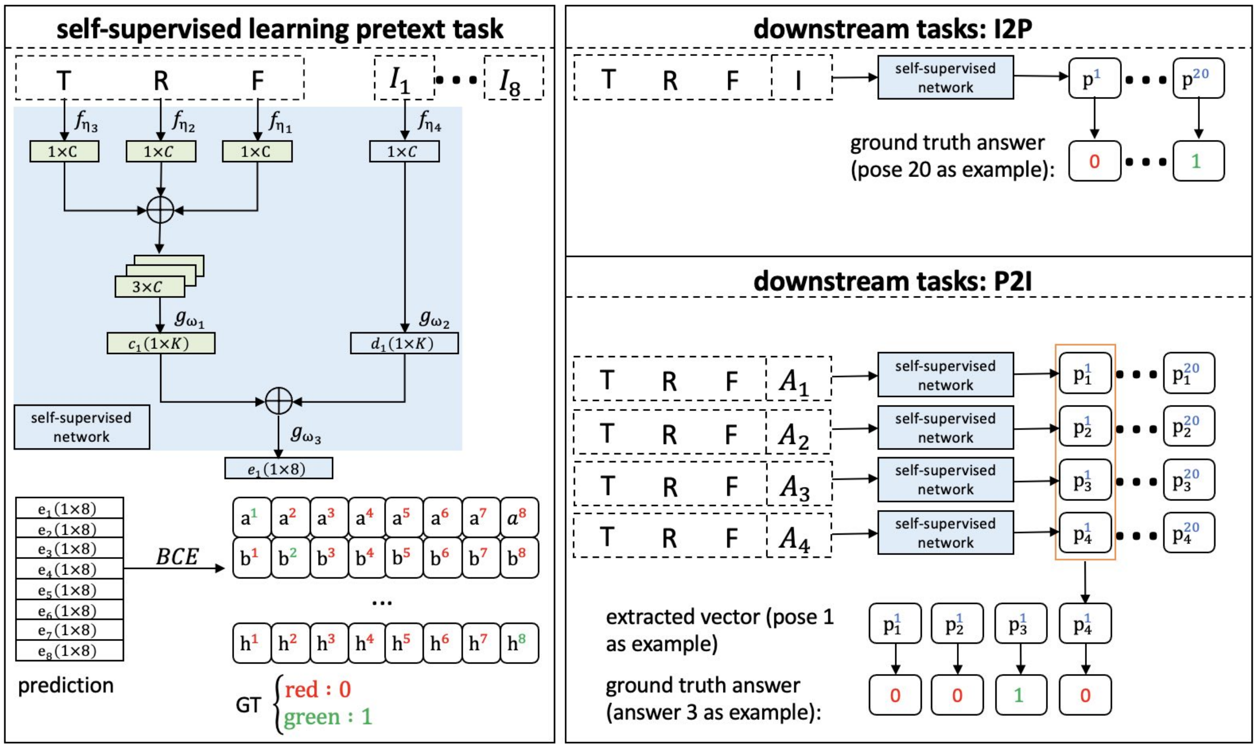
self-supervised learning pretext task
T R F
I_1 ••• I_8
f_{η_3} f_{η_2} f_{η_1} f_{η_4}
1×C 1×C 1×C 1×C
3×C
g_{ω_1} g_{ω_2}
$c_1(1\times K)$ $d_1(1\times K)$
g_{ω_3}
$e_1(1\times 8)$
self-supervised network
$e_1(1\times 8)$ … $e_8(1\times 8)$
BCE
prediction
GT red : 0 green : 1
downstream tasks: I2P
T R F I
self-supervised network
p^1 ••• p^{20}
ground truth answer (pose 20 as example):
0 ••• 1
downstream tasks: P2I
T R F A_1 A_2 A_3 A_4
extracted vector (pose 1 as example)
ground truth answer (answer 3 as example):
0 0 1 0

Figure 3. **Self-supervised learning network architecture for task *I2P* and *P2I*.** We train a self-supervised learning network (left subfigure), then use the learned representations to the downstream tasks *I2P* and *P2I* (right subfigure). The f_{η_1}, f_{η_2}, f_{η_3}, and f_{η_4} are CNN networks; g_{ω_1}, g_{ω_2}, and g_{ω_3} are MLP networks; c_1, d_1, e_1 are the encoded feature vectors.

T, and the isometric image I to the trained self-supervised network and obtain a eight-digit vector. Then we select the first, second, fifth, and sixth values in the vector as the prediction pose of the network, which can represent the potential view that is used in this task. Then, the max value in the prediction pose vector is considered as the predicted pose, and we give value 1 to the corresponding pose and 0 to other poses to form a one hot encoder. Finally, the BCE loss is computed using the predicted one hot encoder and the ground truth one hot encoder.

In task *P2I*, the three-view images and a selected pose from 8 potential poses are sent to the network. The network then selects the isometric view image corresponding to the pose from four potential answers (four rendered isometric view images). For representation adaption, we send the three-view image F, R, T and one potential answer image A_i, $i \in \{1, 2, 3, 4\}$ to the trained self-supervised network and obtain four 8 dimensional vectors. For each vector, the value represents the probability of the pose that the answer image is rendered from. Finally, we extract the value from the column that corresponds to the given pose for each vector, forming a four-dimensional vector. For example, the four vectors in the first column will be extracted if the given pose is 1. Then we use the extracted vector to compute the BCE loss with the ground truth one hot encoder.

Extension experiment for camera pose reasoning task. Besides testing the self-supervised network's learned representations on *I2P* and *P2I* tasks in the SPARE3D dataset, we are also curious on whether this self-supervised learning could help camera pose reasoning that contains poses unseen during training? Therefore, we design two extension downstream task of *I2P* and *P2I* by increasing the potential camera pose number from 4, 8 to 20, 20 respectively. The 20 camera poses are viewed in the supplementary.

4. Experiments and Discussions

In this section, we introduce the experiment details for training two of our self-supervised learning networks and the baseline methods. Then, we compare the classification accuracy of our method with baseline methods on all three tasks, including the extension *I2P* and *P2I* tasks. Additionally, for *T2I* task, we also visualize the attention maps obtained from our methods and supervised baseline methods, demonstrating that our method can better localize the differences between the candidate answers. All experiments are implemented with PyTorch [30], using NVIDIA GeForce GTX 1080 Ti GPU.

Data for training and testing. For all the tasks, we

Table 1. **Comparison of performance on *T2I* for SL method vs. SSL method.** SL and SSL represent supervised learning and self-supervised learning, respectively. 5K and 14K are the training data amount. Fine-tuning means we further use the 5K training data in SPARE3D to fine-tune the parameters. For supervised learning, we evaluate the network performance for: (1) using *early fusion* or *late fusion* structure (details in the supplementary), (2) whether or not using ImageNet pre-trained parameters.

SL	early-fusion(5K)	early-fusion(pretrained, 5K)	late-fusion(pretrained , 5K)	early-fusion(14K)	early-fusion(pretrained, 14K)	late-fusion(pretrained, 14K)
	55.0	30.6	25.2	63.6	51.4	27.4
SSL	Jigsaw puzzle[29]	Colorization[46]	SimCLR[5]	RotNet[10]	Ours (NT-Xent loss)	Ours (BCE loss)
	27.4	23.4	31.0	30.6	48.4	**74.9**

use the same test data in SPARE3D dataset for testing (except the *extension I2P* and *extension P2I* tasks). To prevent the network from "memorize" the data, we avoid using the same models for self-supervised training, supervised fine-tuning, and testing for all the tasks. Next, we will discuss how we generate training data for different tasks.

In our contrastive learning approach for *T2I* task, we first generate image sets for two branches. In practice, we download $14,051$ models from the ABC dataset for training and 737 models for testing. Then we use PythonOCC [31] to apply different random Boolean operations on each model, generating four images for branch 1 and branch 2, respectively. Therefore, we have $14,051 \times 8$ rendered images.

All $14,051 \times 8$ images are used for three self-supervised baselines that we compare with, namely Jigsaw Puzzle, Colorization, SimCLR, and RotNet. For supervised learning with 5K dataset, we use the original data from SPARE3D paper, with $5,000$ questions for training. For supervised learning with 14K dataset, we use the $14,051$ CAD models to generate new questions for training.

For the self-supervised network for *I2P* and *P2I*, we generate different training sets with CAD model amount ranging from $5,000$ to $40,000$, with $5,000$ as a step. For each CAD model, it generates three-view images(F, R, T) and eight isometric view images($I_1, I_2, \cdots, I_8$). For a fair comparison, we also generate different scale datasets for supervised learning, ranging from $5,000$ to $40,000$, and all the implementation settings are the same as in SPARE3D.

For both the *extension I2P* and *extension P2I* tasks, we need to: 1) create supervised training dataset with different size, ranging from $5,000$ to $40,000$, and 2) create a test set with $1,000$ questions, which is the same number as in the original *I2P* and *P2I* settings. We follow the implementation settings as in SPARE3D, and the only change is we increase the potential view number to 20.

Hyperparameter settings. We tune the learning rate and batch size for each self-supervised learning method and supervised learning method for all tasks.

For *T2I*, our contrastive learning network, with either NT-Xent loss or BCE loss, uses a learning rate of 0.00005, batch size 4. Jigsaw puzzle, Colorization, SimCLR, and RotNet all use learning rate 0.00001 and batch size $10, 30, 70, 16$, respectively. For supervised learning, we follow the hyperparameter settings as in SPARE3D.

For the *I2P* and *P2I* tasks, the learning rate and batch size for our self-supervised network are 0.00005 and 4. For the *extension I2P* and *extension P2I* tasks, the learning rate is 0.00001, and the batch size are 70 and 20 respectively. For supervised learning, we follow the hyperparameter settings as in SPARE3D.

Details of the two proposed self-supervised networks. For both our contrastive learning method and self-supervised method, we use the VGG-16 network as the backbone for image feature extraction, and parameters keep the same. Note that for the contrastive learning network using NT-Xent loss, we use a_1, b_1, a_2, b_2 as the latent vectors. a_1, b_1 and a_2, b_2 are considered as positive pairs, while other remaining pairs, including the pairs within the batch, are negative. The value of C is set to be $18,432$ (by flattening the $6 \times 6 \times 512$ feature map). The value of K is set to be 4096, the same as in the SPARE3D paper. The size of input line drawings is $200 \times 200 \times 3$. The output of the last convolutional layer is a $6 \times 6 \times 512$ feature map.

Self-supervised learning baseline network adaptation for *T2I*. For task *T2I*, we use Jigsaw puzzle [29], Colorization [46], SimCLR [5], and RotNet [10] as four self-supervised learning baseline methods to compare with our method, see Table 1. For the network structures of the Jigsaw puzzle, Colorization, and RotNet, we follow these papers' original design, only replacing the backbone networks with VGG-16 to ensure that the learned parameters can be loaded to the networks of the downstream tasks. For SimCLR, we define the positive "image pairs" in our case as the $\{F, R, T\}$ and I images, which are generated from the same CAD model. We then use the same *contrastive network* as in Figure 2 to extract the features for $\{F, R, T\}$ and I images separately, and the learned parameters can be loaded to the downstream tasks.

***Extension I2P* and *extension P2I* task implementation.** For these two tasks that use 20 views as potential camera pose, we compare our self-supervised learning pre-trained networks with the supervised learning baseline methods. For supervised learning baseline methods of *extension I2P* and *extension P2I*, we follow the network design as *I2P* and *P2I* respectively. The only difference is that, for each task, the output of the last MLP layer g_{ω_3} changes to a 20 dimensional latent vector. Therefore, when loading the learned parameters from the self-supervised network, we discard the last layer MLP of the trained model and train the parameters using the extension supervised learning training data for 20 views.

4.1. *T2I* task result analysis

Classification accuracy for task *T2I*. As can be seen in Table 1, our methods (both with or without fine-tuning) outperform other methods, including self-supervised baseline methods and supervised methods. Our fine-tuned result can achieve 74.9% accuracy on *T2I* task, approaching the average untrained human performance of 80.5%. Here direct evaluation means we use the learned parameters from the trained contrastive learning network, and fine-tuning means we further use the $5,000$ training data for supervised learning to fine-tune the learned parameters.

Although we use more data in the contrastive pre-training, the higher accuracy of our method is not only due to increased data volume. As aforementioned, we use $14,051$ CAD models to generate image sets for contrastive learning. We also use these models to generate $14,051$ questions for purely supervised learning. This ensures the number of CAD models used for our method is the same as for purely supervised learning. With the same number of CAD models for training (14K dataset), we find the best performance that supervised learning can achieve is 63.6%. Although increasing the data volume can help improve the baseline performance for supervised learning, from 55.0% to 63.6%, the result is still significantly lower than our method, which is 74.9%.

We believe the good performance of our method is because contrastive learning helps the network learn the *detail-sensitive* yet view-invariant visual representations in the line drawings. This reason could also explain an interesting phenomenon that we observe, which is that the direct evaluation (without fine-tuning) using the learned parameters from contrastive spatial reasoning can achieve 71.4% accuracy (see Table 1). Thus, a good visual representation should be able to transfer to the downstream tasks with little further training.

Qualitatively, we show that our contrastive learning method can help the network learn the *detail-sensitive* yet view-invariant visual representations. we visualize the attention map for our contrastive learning method and supervised learning method using the schemes in [42]. For supervised learning, we generate attention maps on both *early fusion* and *late fusion* method with no pre-trained parameters. For *early fusion*, the input line drawings, including front, right, top, and one isometric line drawing, are concatenated before being sent to the CNN. Therefore, each composite

Table 2. **Comparison of performance on *I2P* and *P2I* tasks for SL method vs. SSL method.**

Data amount (K)	5	10	15	20	25	30	35	40
I2P(SL)	83.6	86.4	87.7	88.5	88.7	90.4	90.6	91.1
I2P(SSL)	88.7	93.2	95.1	96.4	96.7	97.7	97.5	**98.0**
P2I(SL)	65.4	67.1	68.5	67.8	69.8	69.6	68.5	70.4
P2I(SSL)	72.4	80.8	81.9	82.1	82.8	83.1	83	**83.4**

Table 3. **Comparison of performance on *extension I2P* and *extension P2I* tasks for SL method vs. SSL method.** Note that for SSL methods, we fine-tune the network for the downstream tasks.

Data amount(K)	5	10	15	20	25	30	35	40
I2P(SL)	15.1	14.5	15.9	14.8	17.7	17.8	16.3	17.5
I2P(SSL)	42.9	45.5	48.4	50.1	51.3	51.7	54.8	**56.5**
P2I(SL)	51.4	47.2	33.7	52.9	44.3	43.3	52.7	44.8
P2I(SSL)	67.4	73.5	66.2	76.2	75.0	68.1	75.7	**77.9**

image will have one attention map. We put the attention map with the corresponding candidate isometric line drawing, leaving the attention map for the front, right, and top as empty. For *late fusion* and our method, after having the attention map for each input image, we put it together with the corresponding input line drawing. All the results are shown in Figure 4 (more results in the supplementary). The comparison between the three rows of attention maps generated from the three methods shows that our method can help the CNN better capture the tiny detailed differences between the candidate answer drawings, which is the key to selecting the correct answer from four similar options.

4.2. *I2P* and *P2I* task result analysis

Classification accuracy for task *I2P* and *P2I*. For a fair comparison, we change the supervised learning network's structure to match the structure of our self-supervised learning method. More details will be discussed in the supplementary. The results can be seen in Table 2. With the increase of data amount for training, both supervised learning, and our self-supervised learning-based method achieve higher accuracy. For task *I2P*, the best accuracy achieves 98.0%, and for task *P2I*, the best accuracy is 83.4%. The best performance happens when using the $40,000$ scale dataset with our self-supervised learning method.

Therefore, we claim that the increased data amount can help both supervised and self-supervised learning methods. However, our self-supervised learning method has the advantages that: 1) more efficient to improve the accuracy, compared with using the same amount of data as supervised learning methods, and 2) more helpful when the supervised learning method does not perform well. For the second advantage, since the number of potential cameras poses in task *I2P* and *P2I* are 4 and 8 respectively, it is natural to expect that it will be more difficult for the neural network to solve the *P2I* task. In *P2I* task, we find that even after increas-

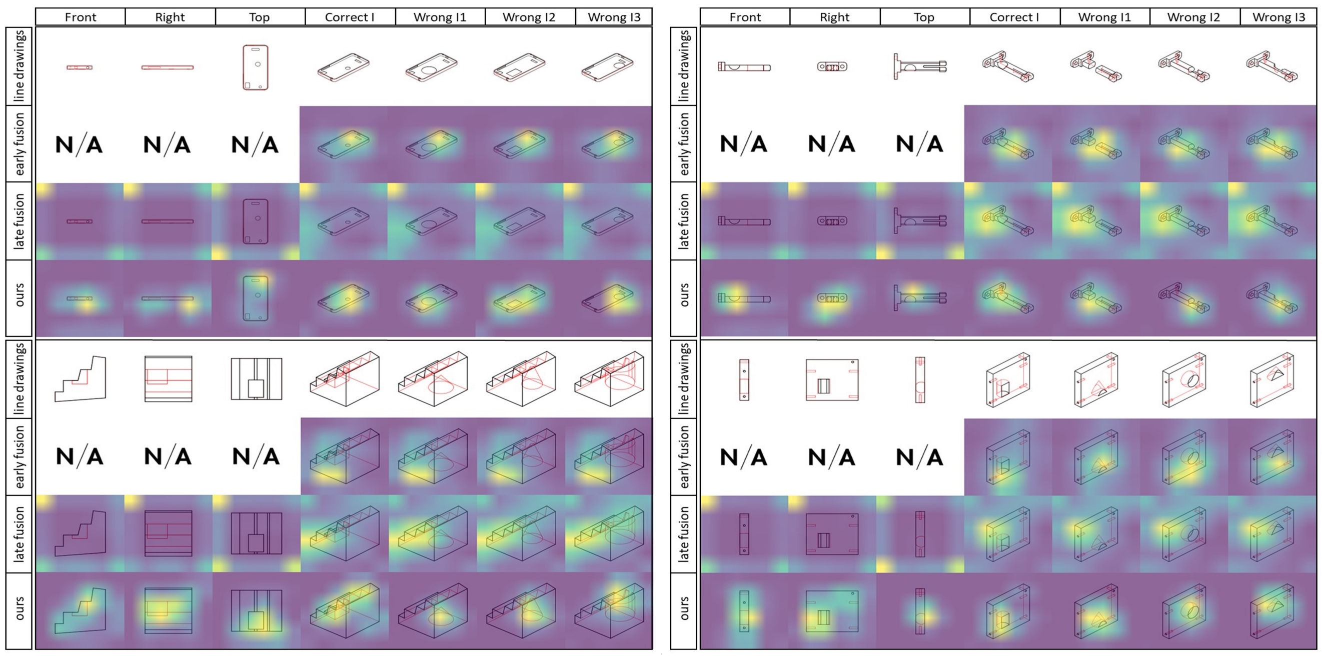

Figure 4. **Attention maps for SL vs. SSL method in *T2I* task.** For each CAD model, the first row are the line drawings. The second and third row are the attention maps generated from supervised learning using *early fusion* and *late fusion*, respectively. The fourth row are the attention maps generated from our method. N/A indicates no attention map for the corresponding view. Best viewed in color.

ing the training data amount for supervised learning, the performance improvement is limited. However, our self-supervised learning method can achieve around 10% higher accuracy than the supervised learning method, revealing its advantage over the supervised learning method.

Classification accuracy for task *extension I2P* and *extension P2I*. We also show that our self-supervised learning methods can help *extension I2P* and *extension P2I*. Table 3 shows our self-supervised learning-based method outperforms the supervised learning method for different data amounts on both tasks. We believe it is because the "rough" pose reasoning in the pretext task can help the "fine" pose reasoning in the downstream task. "rough" is because the network only needs to reason about 8 poses; "fine" means in the downstream tasks, the network is required to reason about more views, and these views are located in between the 8 poses. Once the network has the ability to determine the "rough" pose and use it as prior information, it will be easier for the network to reason about the "fine" pose.

4.3. Limitations and discussion

One limitation is that our methods are designed for the tasks in the SPARE3D dataset. Therefore, they are beneficial for *object level* spatial reasoning, yet not directly for *scene level* spatial reasoning. One main difference between these two settings is whether the views are "outside-in" for 3D shapes or "inside-out" for scenes [9]. To make spatial reasoning helpful in robot manipulation, navigation or other scenarios, *scene level* spatial reasoning is necessary.

Besides, our methods are tested on a specific dataset and solve some basic spatial reasoning tasks. More effort is required to explore how our methods can be applied to more general spatial reasoning tasks, which can help the community better utilize the power of spatial reasoning to solve real-world problems.

5. Conclusion

In this paper, we focus on enhancing the deep networks' performance on multi-view line drawing spatial reasoning tasks on the SPARE3D dataset. Specifically, we focus on two types of tasks: 1) view consistency task, which contains task *T2I*, 2) camera pose reasoning task, which contains task *I2P* and *P2I*. We quantitatively and qualitatively show the advantage of using our self-supervised learning methods for all three tasks.

In the future, We plan to explore how our network design for reasoning about 2D and 3D information could benefit the more general vision-related tasks like AI-assisted design, localization, and navigation.

Acknowledgment

The research is supported by NSF FM Program under EEC-2036870. Siyuan Xiang gratefully thanks the IDC Foundation for its scholarship. The authors gratefully acknowledge the constructive comments and suggestions from the anonymous reviewers and area chairs.

References

[1] Ankesh Anand. Contrastive self-supervised learning, 2020. https://ankeshanand.com/blog/2020/01/26/contrative-self-supervised-learning.html.

[2] Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.

[3] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, pages 15535–15545, 2019.

[4] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33, 2020.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[6] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in Neural Information Processing Systems*, 33, 2020.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. Ieee, 2009.

[8] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015.

[9] Carlos Esteves, Yinshuang Xu, Christine Allen-Blanchette, and Kostas Daniilidis. Equivariant multi-view networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1568–1577, 2019.

[10] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[11] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 3828–3838, 2019.

[12] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6391–6400, 2019.

[13] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.

[14] Wenyu Han, Siyuan Xiang, Chenhui Liu, Ruoyu Wang, and Chen Feng. Spare3d: A dataset for spatial reasoning on three-view line drawings. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 14690–14699, 2020.

[15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[16] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[17] Sherry Hsi, Marcia C Linn, and John E Bell. The role of spatial reasoning in engineering and the design of spatial instruction. *Journal of engineering education*, 86(2):151–158, 1997.

[18] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[19] Harrison J Kell, David Lubinski, Camilla P Benbow, and James H Steiger. Creativity and technical innovation: Spatial ability's unique role. *Psychological science*, 24(9):1831–1836, 2013.

[20] Dahun Kim, Donghyeon Cho, Donggeun Yoo, and In So Kweon. Learning image representations by completing damaged jigsaw puzzles. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 793–802. IEEE, 2018.

[21] Ildoo Kim, Woonhyuk Baek, and Sungwoong Kim. Spatially attentive output layer for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9533–9542, 2020.

[22] Youngsung Kim, Jinwoo Shin, Eunho Yang, and Sung Ju Hwang. Few-shot visual reasoning with meta-analogical contrastive learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[23] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1920–1929, 2019.

[24] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European Conference on Computer Vision*, pages 577–593. Springer, 2016.

[25] Jason D Lee, Qi Lei, Nikunj Saunshi, and Jiacheng Zhuo. Predicting what you already know helps: Provable self-supervised learning. *arXiv preprint arXiv:2008.01064*, 2020.

[26] Fenglin Liu, Chenyu You, Xian Wu, Shen Ge, Xu Sun, et al. Auto-encoding knowledge graph for unsupervised medical report generation. *Advances in Neural Information Processing Systems*, 34, 2021.

[27] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *arXiv preprint arXiv:2006.08218*, 2020.

[28] Ishan Misra and Laurens van der Maaten. Self-supervised

learning of pretext-invariant representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6707–6717, 2020.

[29] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016.

[30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

[31] PythonOCC. 3D CAD/CAE/PLM development framework for the Python programming language, 2018. http://www.pythonocc.org.

[32] Ajay Ramful, Thomas Lowrie, and Tracy Logan. Measurement of spatial ability: Construction and validation of the spatial reasoning instrument for middle school students. *Journal of Psychoeducational Assessment*, 35(7):709–727, 2017.

[33] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.

[34] Yuandong Tian, Lantao Yu, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning with dual deep networks. *arXiv preprint arXiv:2010.00578*, 2020.

[35] Christopher Tosh, Akshay Krishnamurthy, and Daniel Hsu. Contrastive learning, multi-view redundancy, and linear models. *arXiv preprint arXiv:2008.10150*, 2020.

[36] Yude Wang, Jie Zhang, Meina Kan, Shiguang Shan, and Xilin Chen. Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12275–12284, 2020.

[37] Chen Wei, Lingxi Xie, Xutong Ren, Yingda Xia, Chi Su, Jiaying Liu, Qi Tian, and Alan L Yuille. Iterative reorganization with weak spatial constraints: Solving arbitrary jigsaw puzzles for unsupervised representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1910–1919, 2019.

[38] Chenyu You, Nuo Chen, and Yuexian Zou. Self-supervised contrastive cross-modality representation learning for spoken question answering. *arXiv preprint arXiv:2109.03381*, 2021.

[39] Chenyu You, Ruihan Zhao, Fenglin Liu, Sandeep Chinchali, Ufuk Topcu, Lawrence Staib, and James S Duncan. Class-aware generative adversarial transformers for medical image segmentation. *arXiv preprint arXiv:2201.10737*, 2022.

[40] Chenyu You, Ruihan Zhao, Lawrence Staib, and James S Duncan. Momentum contrastive voxel-wise representation learning for semi-supervised volumetric medical image segmentation. *arXiv preprint arXiv:2105.07059*, 2021.

[41] Chenyu You, Yuan Zhou, Ruihan Zhao, Lawrence Staib, and James S Duncan. Simcvd: Simple contrastive voxel-wise representation distillation for semi-supervised medical image segmentation. *arXiv preprint arXiv:2108.06227*, 2021.

[42] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

[43] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1476–1485, 2019.

[44] Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. Raven: A dataset for relational and analogical visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5317–5327, 2019.

[45] Chi Zhang, Baoxiong Jia, Feng Gao, Yixin Zhu, Hongjing Lu, and Song-Chun Zhu. Learning perceptual inference by contrasting. *Advances in Neural Information Processing Systems*, 32, 2019.

[46] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, pages 649–666. Springer, 2016.