

# TriCoLo: Trimodal Contrastive Loss for Text to Shape Retrieval

## Supplemental Materials

Yue Ruan<sup>1\*</sup>   Han-Hung Lee<sup>1\*</sup>   Yiming Zhang<sup>1</sup>   Ke Zhang<sup>1</sup>   Angel X. Chang<sup>1,2</sup>  
<sup>1</sup>Simon Fraser University   <sup>2</sup>Alberta Machine Intelligence Institute (Amii)  
{yuer, hla300, yza440, ke\_zhang\_4, angelx}@sfu.ca  
<https://3dlg-hcvc.github.io/tricolo/>

In this supplement to the main paper, we provide details about the statistics of the ‘chairs and tables’ dataset (Sec. 1), our rendering process (Sec. 2) and our model implementation (Sec. 3), as well as additional evaluation experiments (Sec. 4). In Sec. 5, we provide visualizations of our trimodal embedding space (Sec. 5.1), additional qualitative results (Sec. 5.3), and brief discussion of shape similarity metrics (Sec. 5.4).

### 1. Data statistics

Category	Text			Shape		
	Train	Val	Test	Train	Val	Test
Chair	26257	3313	3206	5221	659	641
Table	33520	4122	4246	6700	827	851
Total	59777	7435	7452	11921	1486	1492

Table 1. ‘Chairs and tables’ statistics [2].

### 2. Render settings

For the rendering setup, we use Pyrender<sup>1</sup>. The object is placed at the center (0, 0, 0). The camera is placed at (0, 1, 0.6) with the focal length set to 35mm and the sensor width to 32mm while being pointed towards the center (0, 0, 0). We render 12 images by rotating the camera 30 degrees per render with render resolution set to 224.

### 3. Model details

#### 3.1. Voxel encoder details

For the voxel encoder, we use a 5-layer sparse 3D CNN architecture with input resolution 64<sup>3</sup>. Tab. 2 shows the architectural details. Here BN stands for Batch Normalization and LR stands for Leaky ReLU, each layer of convolution is followed by normalization then activation.

Layer	Kernel	Stride	Channels	BN	LR
conv1	3	1	32	Y	Y
max_pool1	2	2	-	-	-
conv2	3	1	64	Y	Y
max_pool2	2	2	-	-	-
conv3	3	1	128	Y	Y
max_pool3	2	2	-	-	-
conv4	3	1	256	Y	Y
max_pool4	2	2	-	-	-
conv5	3	1	512	Y	Y
max_pool5	2	2	-	-	-
fc6	-	-	512	N	N

Table 2. Voxel encoder for resolution 64<sup>3</sup>

#### 3.2. Incorporating CLIP

To use CLIP [8] for our retrieval task, we feed 6 multi-view images of an object into the image encoder for CLIP separately then average the vectors to get the image embedding. Specifically, we use the ViT-L/14 pretrained model from CLIP. For retrieval, we encode the text using the pretrained transformer-based CLIP text encoder and then retrieve relevant shapes by taking the dot product of the text and shape embeddings. We compared the zero-shot performance of CLIP embedding to the embedding from CLIP with additional MLPs (which is equivalent to taking the cosine similarity) and found that the CLIP+MLP embeddings work better. For incorporating CLIP into our model, we take the CLIP embeddings of each image, and project the embeddings using a two-layer MLP. The projected embeddings of the 6 multi-view images are averaged, and the weights of the MLP are trained using a Cross-Entropy loss. We train using the Adam [6] optimizer until the validation performance drops, with a learning rate of 0.00035 on A40. For fast training, we preprocess the data and store frozen clip embeddings in a cache.

\*indicates equal contribution.

<sup>1</sup><https://github.com/mmatl/pyrender>

Model	#Params	Resolution	BS	Memory
Bi(V)	6.6M	32 <sup>3</sup>	128	2.6 GB
			32	3.3 GB
		64 <sup>3</sup>	64	5.1 GB
			128	10.1 GB
			256	18.3 GB
Bi(I)	13.3 M	64 <sup>2</sup>	128	2.7 GB
			128 <sup>2</sup>	9.9 GB
		224 <sup>2</sup>	128	30.4 GB
			32	2.6 GB
			64	5.2 GB
128 <sup>2</sup>	128	9.9 GB		
	256	20.8 GB		
	Tri(I+V)	20.4 M	v64 <sup>3</sup> i128 <sup>2</sup>	128

Table 3. Memory usage and number of parameters for the Bi(I), Bi(V) and Tri(I+V) models.

### 3.3. Triplet loss

Given an anchor text embedding  $\mu_{t_j}$  and its positive shape embedding  $\mu_{s_j}$ , we sample the semi-hard examples  $\mu_{s_k}$  such that  $\langle \mu_{t_j}, \mu_{s_j} \rangle < \langle \mu_{t_j}, \mu_{s_k} \rangle < \langle \mu_{t_j}, \mu_{s_j} \rangle + \alpha$  is satisfied, where  $\alpha$  is the margin. The semi-hard example is sampled online by selecting from the mini-batch as in Schroff et al. [9]. Semi-hard sampling has been shown to work better than naive sampling or hard negative sampling for triplet loss and is also used in Tang et al. [11]. We then apply the triplet loss using the triplets  $(\mu_{t_j}, \mu_{s_j}, \mu_{s_k})$ , here  $\mu_s$  can be the voxel or image embeddings:

$$l_j = \sum_{\text{all satisfied } k} \max(\langle \mu_{t_j}, \mu_{s_j} \rangle - \langle \mu_{t_j}, \mu_{s_k} \rangle + \alpha, 0) \quad (1)$$

### 3.4. Model size

We show the number of parameters and memory for our models with different hyperparameters in Tab. 3. Here BS stands for batch size. Bi(I) is trained with 6 multi-view images and a ResNet-18 backbone. Tri(I+V) is trained with voxel resolution 64<sup>3</sup> and image resolution 128<sup>2</sup>. Here, the memory usage shown is calculated using `torch.cuda.max_memory_reserved` during training.

## 4. Additional experiments

We conduct additional experiments to investigate the impact of image and voxel resolution (Sec. 4.1) and image backbone architecture (Sec. 4.2) on text-to-shape retrieval. We present the performance of our model on an extended set of 13 object categories from ShapeNet [1] (Sec. 4.3).

	resolution	RR@1(↑)	RR@5(↑)	NDCG@5(↑)	MRR(↑)
Bi(I)	64	9.98 ± 0.28	28.60 ± 0.19	19.56 ± 0.20	19.88 ± 0.24
	128	<b>11.61 ± 0.20</b>	30.65 ± 0.19	21.36 ± 0.23	21.46 ± 0.25
	224	11.45 ± 0.18	<b>32.06 ± 0.22</b>	<b>22.01 ± 0.26</b>	<b>21.86 ± 0.27</b>
Bi(V)	32	8.45 ± 0.32	26.00 ± 0.35	17.32 ± 0.20	17.72 ± 0.19
	64	<b>9.59 ± 0.27</b>	<b>27.14 ± 0.48</b>	<b>18.54 ± 0.13</b>	<b>19.03 ± 0.08</b>

Table 4. Comparison of resolution settings on shape retrieval for Bi(I) and Bi(V) on the validation set. We find that increasing the resolution increases the performance.

	Model	RR@1(↑)	RR@5(↑)	NDCG@5(↑)	#Params
Bi(I)	EfficientNet-B0	<b>12.52 ± 0.22</b>	<b>32.53 ± 0.28</b>	<b>22.79 ± 0.31</b>	7.2M
	ResNet-18	11.61 ± 0.20	30.65 ± 0.19	21.36 ± 0.23	13.3M
	ResNet-34	11.41 ± 0.21	30.68 ± 0.17	21.21 ± 0.25	23.4M

Table 5. Comparison of different architectures on shape retrieval for Bi(I) on the validation set. We find that increasing the model parameters actually decreases the performance from ResNet-18 to ResNet-34. Using a more powerful model with fewer parameters (i.e. EfficientNet), we see that the performance improves.

### 4.1. Resolution experiments

We conduct experiments for different resolutions of images (64<sup>2</sup>, 128<sup>2</sup> and 224<sup>2</sup>) and voxels (32<sup>3</sup> and 64<sup>3</sup>). In Tab. 4 we see that the performance increases with higher resolutions.

### 4.2. Image backbone experiments

We conduct experiments with different image backbones for Bi(I) to see how different model sizes will impact the retrieval performance. The results are shown in Tab. 5. When we increase model parameter size from ResNet-18 [5] to ResNet-34 it can be seen that the performance drops, this is in conflict with prior work [3, 8] on contrastive learning that sees performance gains when using larger encoder models. This could support our hypothesis that our dataset is relatively small, and using bigger models will result in overfitting. To verify this we conduct another experiment using a model that uses fewer parameters but has comparable performance to ResNet-152, namely EfficientNet-B0 [10]. We see that EfficientNet-B0 performs better than ResNet-18 and ResNet-34. For our use case, it may be more desirable to use models that are more lightweight, but still offer good performance. Note that we don’t use EfficientNet-B0 in our main model because it actually uses more than 2x memory in training compared to ResNet-18 due to the operations in EfficientNet requiring more memory for backpropagation.

### 4.3. Retrieval on ShapeNet 13 categories

To show the effectiveness of our method for retrieval beyond ‘chairs and tables’, we also collected a set of descriptions for 11 additional categories of objects from ShapeNet [1]. Using a similar setting as Chen et al. [2], we asked Amazon Mechanical Turk workers to provide de-

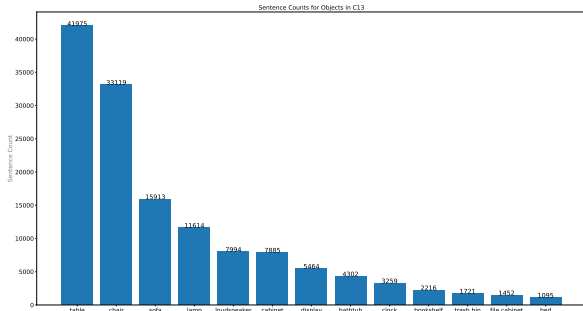


Figure 1. Histogram of number of sentences for the 13 different object categories ShapeNetCore. We collected additional sentences for shapes other than ‘chairs and table’ to investigate the performance of our models on a more diverse object dataset.

	Text	Image	Voxels	RR@1	RR@5	NDCG@5
ZS*	CLIP	CLIP	-	6.81	16.97	11.97
Bi(I)	CLIP	CLIP	-	5.78	20.17	13.03
Bi(I)	GRU	MVCNN	-	8.44	25.64	17.15
Bi(V)	GRU	-	3D-CNN	8.87	27.91	18.52
Tri(I+V)	GRU	MVCNN	3D-CNN	<b>10.63</b>	<b>31.01</b>	<b>21.03</b>

Table 6. Shape retrieval results on ShapeNet c13 val set. We observe a similar trend as on the ‘chairs and tables’ dataset.

descriptions for a random set of 5 objects. We exclusively engaged high-quality workers (with acceptance rate of  $> 95\%$  on more than 200 HITs) from countries that have English as a native language (US, Canada, Great Britain and Australia). We collected up to 5 descriptions per object. This together with the original ‘chairs and tables’ dataset, results in a dataset with over 138K descriptions for 27, 510 objects across 13 object categories (see Figure 1 for distribution of sentences for each object category).

We use the same standard settings as in our Text2Shape experiments with with 6 multi-view images, resolution  $128^2$  and  $64^3$  for images and voxels respectively and a batch size of 128. Experiments on this ShapeNet C13 dataset show a similar trend as for the ‘chairs and table’ dataset, with Tri(I+V) outperforming Bi(I) and Bi(V) (see Table 6). Comparison with zero-shot CLIP (ZS\*) shows that the pre-trained CLIP model is able to retrieve some relevant shapes in a zero-shot setting on this broader set of shapes, but has lower performance. Our Bi(I) with GRU and MVCNN trained from scratch is able to outperform the bimodal models with CLIP text and image encoder trained on the data, but still underperforms Tri(I+V), showing the value of trimodal embedding. With more categories and less training data for ShapeNet C13 compared to the ‘chairs and tables’ dataset, the pretrained CLIP embeddings is more effective on ShapeNet C13.

## 5. Qualitative results and discussion

We provide visualizations of the embedding space (Sec. 5.1), types of different errors (Sec. 5.2), additional qualitative examples of text-to-shape retrieval results (Sec. 5.3) and a discussion of the shape similarity metrics (Sec. 5.4).

### 5.1. Visualization for the embedding space

We visualize the joint embedding spaces for different models by projecting our embeddings to 2D using t-distributed stochastic neighbor embedding (t-SNE) [12]. The embedding spaces for the bimodal and trimodal models are shown in Fig. 2. We also take a closer look at the Tri(I+V) embedding space in Fig. 3 and show that the embeddings of similar shapes cluster together.

### 5.2. Examples of error types

Fig. 4 shows examples of the types of errors we analyzed in the main paper. These failure cases demonstrate potential directions for future work.

### 5.3. Additional visualizations of retrieval results

Fig. 5 shows the best matching shapes each model retrieves for novel queries. These results show our network can be used for easy and rapid search through large 3D collections. We show qualitative comparisons of the models in Fig. 6. From examples 1,4 and 5, we see that Bi(I) is unable to match high-level words such as *stretched*, *tennis* and *picnic* as well as Bi(V) and Tri(I+V). Examples 2 and 3 show that sometimes Bi(V) also proposes poor retrieval results. Since Tri(I+V) considers both images and voxels, it is more robust and retrieves fewer incorrect results than Bi(I) and Bi(V) (examples 2,4,5). It is also challenging for the models to match small details (*notch* in example 3) and the correct number of parts (*single drawer* in example 6).

### 5.4. Shape similarity metrics

In addition to measuring  $F1^\tau$  for  $\tau = 0.1$  we also follow prior shape retrieval work [7] and measure  $F1^\tau$  for  $\tau = 0.3, 0.5$ , as well as the Chamfer Distance (CD), and (Abstract) Normal Consistency (NC). We note that these are all point-wise metrics and we sample 10K points uniformly on the mesh surface of GT and retrieved shapes for computing these metrics. Note that both CD and  $F1^\tau$  depend on the absolute scale of meshes. To compute them, we follow Fan et al. [4] who define 1 unit as 1/10 of the largest length of the ground truth’s bounding box and rescale the ground truth and retrieved meshes individually. See Tab. 7 (which expands on Tab. 3 from the main paper). While our results show that the average of these metrics increases overall for our best model, Tri(I+V), we find that these shape similarity metrics are not very informative in measuring fine shape

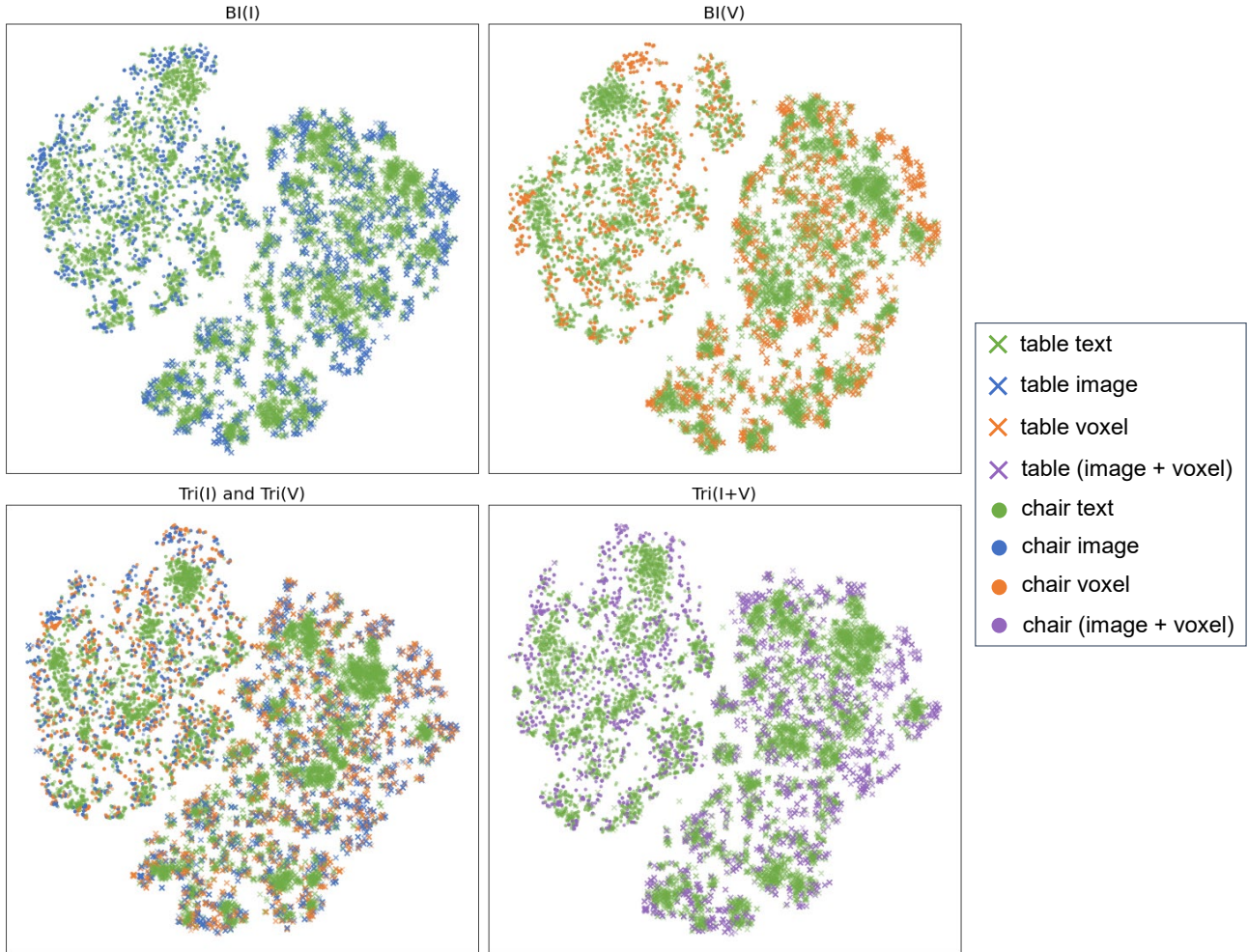


Figure 2. Joint embedding spaces for different models with t-SNE. The modalities are indicated by color (green for text, blue for image, orange for voxel, and purple for image + voxel). Tables are indicated by  $\times$  and chairs by  $\bullet$ . By training with the NT-XEnt contrastive loss, we are able to push the different modalities together and separate embeddings for tables (lower right) and chairs (upper left).

	RR@1( $\uparrow$ )	RR@5( $\uparrow$ )	NDCG@5( $\uparrow$ )	MRR( $\uparrow$ )	CD( $\downarrow$ )	NC( $\uparrow$ )	$F1^{0.1}$ ( $\uparrow$ )	$F1^{0.3}$ ( $\uparrow$ )	$F1^{0.5}$ ( $\uparrow$ )
Bi(I)	11.61 $\pm$ 0.20	30.65 $\pm$ 0.19	21.36 $\pm$ 0.23	21.46 $\pm$ 0.25	2.01 $\pm$ 0.02	0.62 $\pm$ 0.002	11.97 $\pm$ 0.20	34.37 $\pm$ 0.31	48.89 $\pm$ 0.36
Tri(I)	12.19 $\pm$ 0.45	32.33 $\pm$ 0.60	22.54 $\pm$ 0.54	22.62 $\pm$ 0.49	1.91 $\pm$ 0.02	0.63 $\pm$ 0.002	12.49 $\pm$ 0.21	35.56 $\pm$ 0.28	50.28 $\pm$ 0.29
Bi(V)	9.59 $\pm$ 0.27	27.14 $\pm$ 0.48	18.54 $\pm$ 0.13	19.03 $\pm$ 0.08	1.96 $\pm$ 0.02	0.62 $\pm$ 0.002	12.21 $\pm$ 0.09	35.01 $\pm$ 0.18	49.60 $\pm$ 0.24
Tri(V)	9.83 $\pm$ 0.21	27.75 $\pm$ 0.35	18.97 $\pm$ 0.21	19.32 $\pm$ 0.20	1.89 $\pm$ 0.03	0.63 $\pm$ 0.002	12.64 $\pm$ 0.13	35.75 $\pm$ 0.30	50.44 $\pm$ 0.37
Tri(I+V)	<b>12.52 <math>\pm</math> 0.28</b>	<b>32.67 <math>\pm</math> 0.61</b>	<b>22.87 <math>\pm</math> 0.46</b>	<b>22.68 <math>\pm</math> 0.32</b>	<b>1.88 <math>\pm</math> 0.02</b>	<b>0.63 <math>\pm</math> 0.001</b>	<b>12.85 <math>\pm</math> 0.17</b>	<b>36.02 <math>\pm</math> 0.32</b>	<b>50.70 <math>\pm</math> 0.35</b>

Table 7. Comparison of bimodal and trimodal models for text-to-shape retrieval on the validation set. Models are trained with a batch size of 128, solid color voxels at a resolution of  $64^3$ , 6 multi-view images at a resolution of  $128^2$  each. Having a trimodal embedding (Tri(I),Tri(V)) gives better performance than the bimodal embeddings (Bi(I),Bi(V)). By summing the image and voxel representations from the trimodal embeddings (Tri(I+V)), we can further improve retrieval performance.

details and do not always capture whether two shapes are semantically similar. Nevertheless, these metrics are popular in 3D shape generation and retrieval literature and we report them here for completeness.



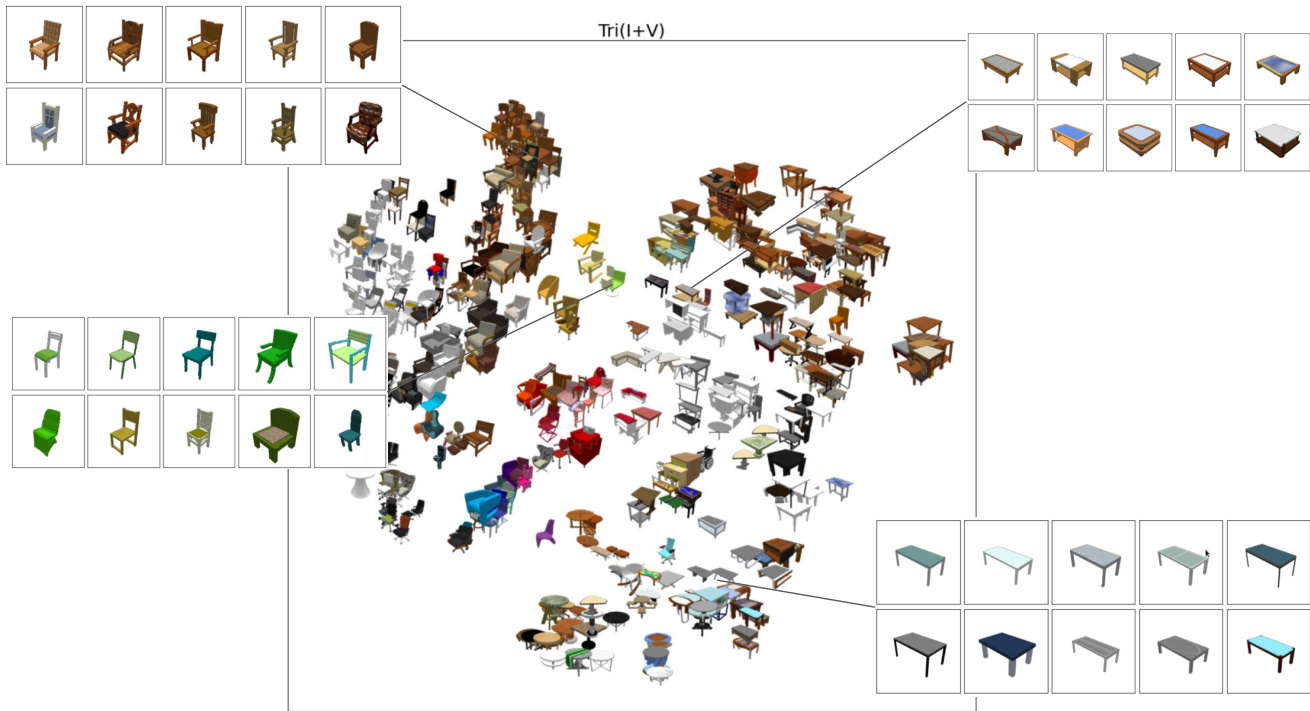


Figure 3. Detailed joint embedding space for Tri(I+V) with t-SNE. We also zoom in on four regions of the joint embedding space. It is clear that similar shapes are close to each other in the embedding space.






a black desk chair with two legs made of metal piping which join under the back of the chair along with arm rests and a split back rest  GT  	color mismatch  	big shape error  
	small shape error  	missing part  

Figure 4. Examples of the four types of error we analyzed in our manual analysis.













	Bi(I)	Bi(V)	Tri(I+V)
a dark brown colored wooden table			
circular table with glass			
chair with arm			
chair without arm			

Figure 5. Retrieved shapes from test set using Bi(I), Bi(V), and Tri(I+V) for custom sentences. Note all models retrieve shapes that match the color (*dark brown*), materials (*wooden, glass*), shape (*circular*), and the presence and absence of chair arms.

---

1 A wooden designer chair, which is good for a stretched sitting.



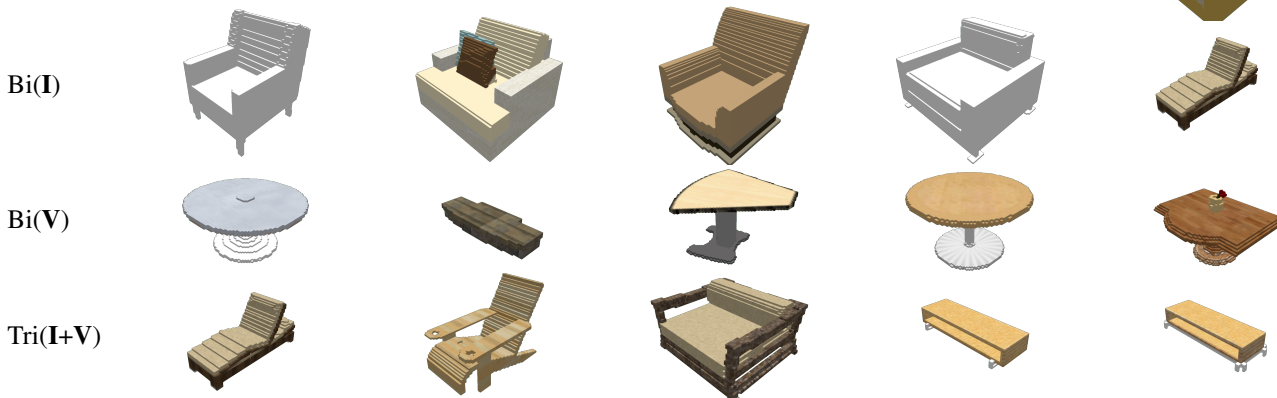
---

2 This short bar stool has a curved metal back in gray. The round cushion is blue and appears to be vinyl.



---

3 a low seat with olive exterior and grey-ish interior. There is a notch where one's neck might rest. It looks to be raised on a very low pedestal, maybe brown in color



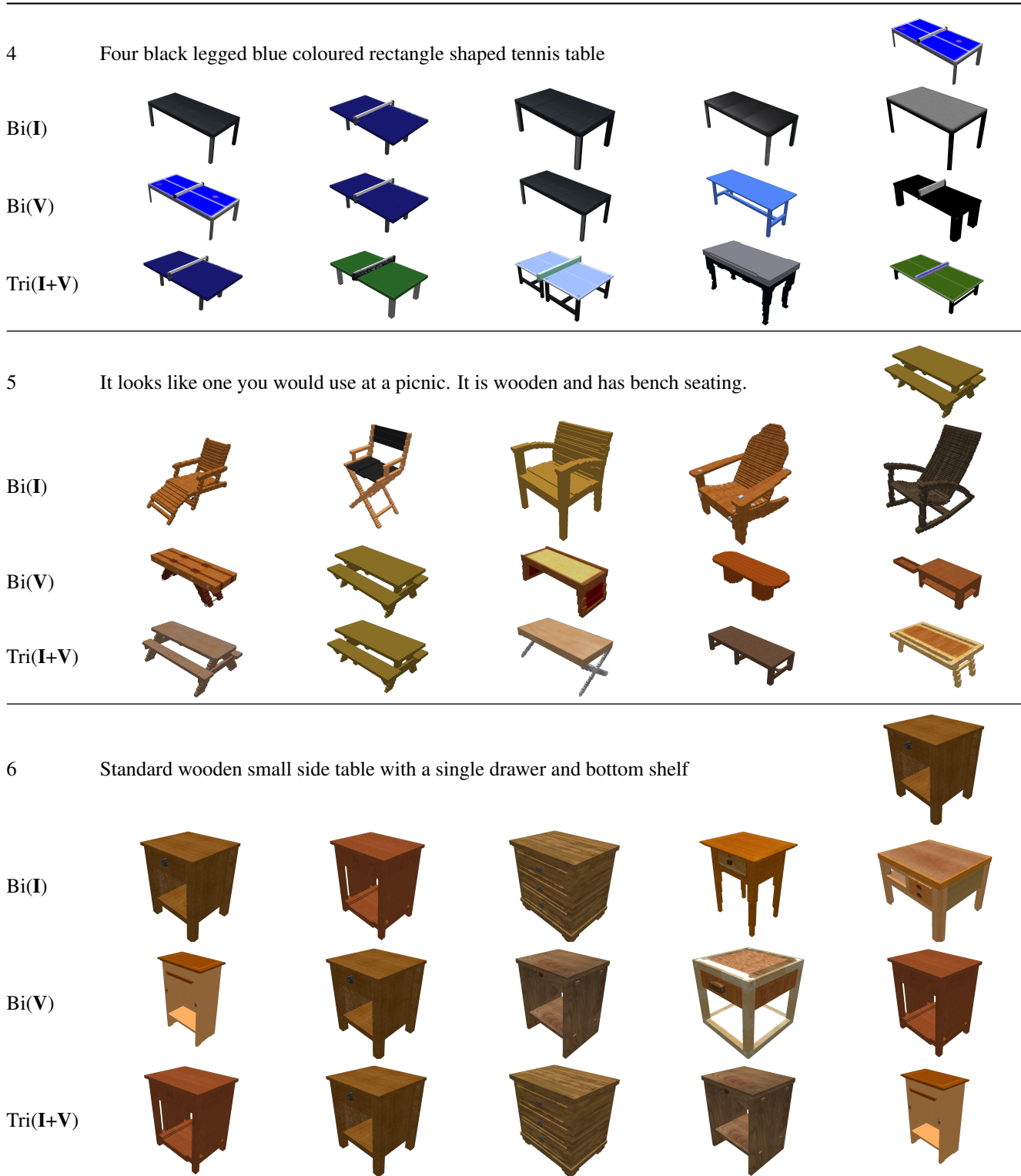


Figure 6. Examples of top 5 retrieved shapes from the validation set using Bi(I), Bi(V), and Tri(I+V). We can see that Bi(I) understands abstract concepts such as *stretched*, *tennis* and *picnic* poorly (examples 1,4,5). It is also challenging to pick up on small details (*notch* in example 3). Although the retrieval results of Tri(I+V) are not always the best among the three models (Bi(V) results are better for example 6), Tri(I+V) is the most stable overall and retrieves more results that are consistent with the description (examples 2,4,5).

## References

- [1] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015. [2](#)
- [2] Kevin Chen, Christopher B Choy, Manolis Savva, Angel X Chang, Thomas Funkhouser, and Silvio Savarese. Text2shape: Generating shapes from natural language by learning joint embeddings. In *Proc. of Asian Conference on Computer Vision (ACCV)*, 2018. [1](#), [2](#)
- [3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020. [2](#)
- [4] Haoqiang Fan, Hao Su, and Leonidas Guibas. A point set generation network for 3D object reconstruction from a single image. In *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. [3](#)
- [5] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. [2](#)
- [6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. [1](#)
- [7] Weicheng Kuo, Anelia Angelova, Tsung-Yi Lin, and Angela Dai. Mask2CAD: 3D shape prediction by learning to segment and retrieve. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 260–277. Springer, 2020. [3](#)
- [8] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021. [1](#), [2](#)
- [9] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015. [2](#)
- [10] Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2019. [2](#)
- [11] Chuan Tang, Xi Yang, Bojian Wu, Zhizhong Han, and Yi Chang. Parts2words: Learning joint embedding of point clouds and texts by bidirectional matching between parts and words. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6884–6893, 2023. [2](#)
- [12] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9 (11), 2008. [3](#)