

Template is all you need: 2D to 3D reconstruction with template learned by contrastive learning

Ming-Yang Ho
Graduate Institute of Biomedical
Electronics and Bioinformatics
r08945027@csie.ntu.edu.tw

Yu-Shan Huang
Graduate Institute of
Communication Engineering
r09942089@ntu.edu.tw

Chih-Jung Chang
Department of
Mathematics
b06201018@ntu.edu.tw

Abstract

3D reconstruction from 2D images has been investigated for several years, and many approaches have been proposed. However, most of them generated 3D objects with sparse representation, which might compromise the dense properties of 3D shape and render it inaccurate. In this study, we leverage the concept of a template that can preserve the basic shape of one category of an object to enhance the performance significantly via a 2-phase training framework. Embedding for each instance with warping LSTM and implicit template networks are jointly optimized first, and an encoder enabling mapping 2D images of different views to the corresponding embedding is further trained. To expedite the process, we propose a contrastive learning solution and compare and discuss the results generated from these two approaches. As a result, both of our two approaches can surpass all the other previous works and achieve better evaluation results with Chamfer distance, demonstrating the effectiveness of the implicit template for 2D to 3D reconstruction.

Code is available at: <https://github.com/Kaminyou/Template-is-all-you-need>

1. Introduction

Dense 3D shape representation from 2D images has recently become an important issue. However, since the training data comes primarily from synthetic datasets, most existing methods can perform reconstruction well only on synthetic images, and some are even restricted to training data. As a result, we proposed a method that can be generalized to a testing dataset and to unseen real image data.

The common problem for 2D-to-3D reconstruction is the lack of geometry information. An existing template of each category can help represent the common characteristic in

each category. Furthermore, every instance can be viewed as a transformation from the template. That is, we can reconstruct a new instance with only its transformation and the template shape. In our method, the template is optimized during training and used for reconstruction afterward.

Besides, feature extraction from images is also a paramount issue that identical embedding must be obtained with images of different views corresponding to one object. While an L1 constraint with a 2-phase training process can successfully address this problem, we further find that a contrastive learning strategy can help retrieve better latent features and perform some experiments in an end-to-end training manner.

The contributions of this paper are highlighted as below:

- We propose two frameworks, two-phase training, and end-to-end training, for 2D-to-3D reconstruction, which can be applied to unseen data.
- With the concept of a template that is optimized during training, better reconstruction results can be achieved.

2. Related work

2.1. Signed Distance Function (SDF)

3D representations for shape learning have long been an issue worth exploring. From early voxel, point cloud, mesh to recently proposed deep implicit function.[9, 1] In this task, we use [9] as our 3D shape representation, which learns a generative model to produce a continuous field. In this work, the signed distance function is designed as a Multi-layer perceptron neural network. Given a 3D spatial point as input, the function outputs the scalar, representing the nearest distance from the input point to the surface. The sign of the scalar shows that whether the spatial point is inside (negative) or outside (positive) the object.

$$SDF(x) = s \quad (1)$$

where: $x \in \mathbb{R}^3$, $s \in \mathbb{R}$. The surface is represented by the function where $SDF(x) = 0$.

2.2. 3D reconstruction from 2D RGB images

After [9] was proposed, several works using DeepSDF representation to solve the 2D-to-3D reconstruction problem come along. Instead of learning the embedding of instances jointly while training the decoder, the information from images is introduced to improve the reconstruction. [14] extracts both global and local features from 2D images to get detail-rich results. Besides, rendering algorithms [10, 7] designed for DeepSDF make unsupervised reconstruction using only 2D images as guides achievable. Our task focuses on the supervised setting.

2.3. Template Learning for Signed distance function

Previous papers related to 2D-to-3D reconstruction often introduce the concept of "Template" [8]. However, a deep implicit function is able to deal with individual shape while not the relationship between instances. [15] proposes a new way to interpret the decoder of deep Signed distance function and thus overcomes this limitation. In this work, the decoder is divided into a warping function and an implicit template, basing on the insight that every instance in the same category can be viewed as a transformation from the only template of this category. As like [9], this work needs to additionally optimize the embedding while testing, which lacks efficiency.

As a result, we would like to improve this work by proposing a methodology that can be directly applied to both the testing set and other unseen data.

3. Methodology

Our framework is designed on the basis of DeepSDF [9] and Deep Implicit Template [15]. Given an image x where $x \in \mathbb{R}^{244 \times 244 \times 3}$, an informative embedding c ($c \in \mathbb{R}^m$, where m denotes the size of this latent vector) should be extracted by an encoder \mathcal{E} ; that's:

$$c = \mathcal{E}(x) \quad (2)$$

Given this c and an 3D point p ($c \in \mathbb{R}^3$), an new 3D coordinates \tilde{p} would be obtained after a warping function \mathcal{W} , that's:

$$\tilde{p} = \mathcal{W}(p, c) \quad (3)$$

An template function \mathcal{T} can further map the \tilde{p} to corresponding signed distance value s , that's:

$$s = \mathcal{T}(\tilde{p}) \quad (4)$$

The whole framework comprises two phases. The first phase, the embedding training phase, attempts to generate

a unique embedding c for every instance X , where X have corresponding several 2D images x of different views, and the encoder training phase aims to construct an encoder that is able to map an image x , where $x \in X$, to its embedding c .

3.1. Embedding training phase

Two networks \mathcal{W} and \mathcal{T} with trainable embedding for each instance are trained in the embedding training phase. While the warping function \mathcal{W} comprises an LSTM cell with iteration times of 8, the \mathcal{T} function is comprised of FCN with a hidden size of 256. During training, every trainable embedding is randomly generated with Gaussian distribution and updated with the whole network simultaneously. A progressive reconstruction loss is utilized to optimize the network, which is defined as:

$$\mathcal{L}_{rec}^{(t)} = \sum_{k=1}^K \sum_{i=1}^N L_{\epsilon_t, \lambda_t}(v, v_{k,i}) \quad (5)$$

$$\mathcal{L}_{rec} = \sum_{t \in \{2,4,6,8\}} \mathcal{L}_{rec}^{(t)} \quad (6)$$

where $v_{k,i}$ denotes the ground truth of signed distance and $L_{\epsilon_s, \lambda_s}$ is a smoothed L1 loss; please refer to [4] for the detail. t denotes the step during the wrapping process. K denotes the number of instances, and N denotes the number of images with different views.

Two other regularizations on p to limited the difference and distortion during the transformation of p to \tilde{p} are defined as the following equations that are called point-wise regularization and point pair regularization, respectively:

$$\mathcal{L}_{pw} = \sum_{k=1}^K \sum_{i=1}^N h(\|\mathcal{W}(p_i, c_k) - p_i\|_2) \quad (7)$$

$$\mathcal{L}_{pp} = \sum_{k=1}^K \sum_{i \neq j} \max\left(\frac{\|\Delta p_i - \Delta p_j\|_2}{\|p_i - p_j\|_2} - \epsilon, 0\right) \quad (8)$$

where h is the Huber kernel and $\Delta p = \mathcal{W}(p, c) - p$.

The total loss \mathcal{L} is the weighted sum of the above losses, that's:

$$\mathcal{L} = \mathcal{L}_{rec} + 0.005 \times \mathcal{L}_{pp} + 0.00002 \times \mathcal{L}_{pw}. \quad (9)$$

We chose Adam to optimize the whole network with this objective function.

3.2. Encoder training phase

After obtaining a unique embedding for each instance, we attempted to train encoder \mathcal{E} to map image x of different

views to corresponding embedding c . We accomplished this goal by training with L1 loss, that's:

$$\mathcal{L}_1 = \sum_{k=1}^K \sum_{i=1}^N \|\mathcal{E}(x_{k,i}) - c_{k,i}\|_1 \quad (10)$$

ResNet-18 [6] backbone was chose as the encoder and Adam algorithm was utilized to optimize this encoder.

3.3. Evaluation metric

Although subjectively compare the generated 3D objects with the ground truth is one way to evaluate our approach, one metric, Chamfer Distance (CD), which is commonly utilized to gauge 3D reconstruction results, is introduced for objective evaluation. The CD is defined as:

$$d_{CD}(S_1, S_2) = \frac{1}{S_1} \sum_{a \in S_1} \min_{y \in S_2} \|a - y\|_2 + \frac{1}{S_2} \sum_{b \in S_2} \min_{a \in S_1} \|b - a\|_2 \quad (11)$$

where S_1 and S_2 denote two 3D point clouds, and a and b denote single 3D coordinates in S_1 and S_2 , respectively.

4. Experiments

4.1. Dataset and Preprocessing

Three categories, chair, sofa, and plane, subsets of ShapeNet core v2, a dataset that comprises multifarious synthetic 3D objects were selected as the training and testing data for our experiments. 3D objects in the OBJ format are provided initially. To generate 2D RGB images from various views, Blender[3] with a script that can simulate camera from random angles were utilized to capture 50 images of "easy views" and 50 images of "hard views," which indicate the larger difference of change of angles between two sequential generated images.

Afterward, the 3D objects in SDF format, which contains 3D coordinates (x, y, z) and corresponding SDF value, were converted from OBJ format via pipeline leveraging CLH11, Pangolin, nanoflann Eigen3 packages. 3D objects in PLY format were also required to evaluate the reconstruction results, which were generated via the above packages.

We split the training and testing sets as the allocation in the DeepSDF [9] that enabled fair comparison of results. Those images were randomly selected from the searching results with the Google search engine for the reconstruction of real-world images.

4.2. Result

Figure 1, 2, and 3 show the results of chair, sofa, and plane categories, respectively. For the chair category, it can be observed that details were well-constructed for training data, and some tiny lines would not be constructed in the testing data. Although a 3D chair can be inferred with an

image of a real-world chair and some basic shape was remained, the chair arms were erroneously generated.

Because the shape of the sofa is intrinsically simple, both the 3D shapes in training and testing sets can be accurately generated. However, most real-world sofa images contain multiple sofas, which do not exist in the training data. Thus, although the model could generate the U-shape contour, it would awkwardly connect them.

When it comes to the plane, the exquisite propellers or other details cannot be well-generated in both training and testing sets. Besides, the wings were becoming somehow larger when the real-world image was inferred.

The results of the objective Chamfer distance evaluation are provided in Table 1. Since images captured from different views would produce different results, we considered two scenarios: (1.) inference with a fixed view (2) inference with random views. The results show that there is no too much discrepancy between these two scenarios, which promises the robustness of our framework. Furthermore, our results achieved the best Chamfer distances among all the categories compared with all previous works, for instance, 3D CNN [14], DISN [14], and Pixel2mesh [12], which demonstrates the benefit of leveraging 3D template for 2D to 3D reconstruction.

	chairs		sofas		planes	
	mean	median	mean	median	mean	median
3D CNN	11.15	NA	9.76	NA	10.47	NA
DISN	7.54	NA	8.71	NA	9.01	NA
Pixel2mesh	11.13	NA	6.54	NA	6.10	NA
Ours (fix)	1.61	1.02	1.50	0.72	4.19	0.16
Ours (rnd)	1.82	1.16	1.49	0.63	4.52	0.18

Table 1. Comparison of Chamfer distance ($\times 0.001$) between our framework and other previous methods for the testing set, where "fix" indicates that images of a fixed view were utilized for inference. In contrast "rnd" indicates images of different views were randomly selected during inference.

5. End-to-end Training

The approach introduced in Section 3 in part relies on the intermediary trainable embeddings c trained in the first phase, thus having two potential issues: 1. The final results of 2D-to-3D reconstruction will hugely depend on the quality of such embeddings. 2. The number of trainable parameters scales linearly with the number of training data. To address this problem, we discuss the methodology and training performance in an end-to-end manner in this section.

5.1. Naive Methodology

Different from the 2-phase training in Section 3, in end-to-end training the encoder and the SDF are jointly optimized. Concretely, given an image x and a 3D point


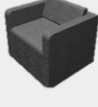




	Train		Test		Real world
rgb					
gt					NA
3d					

Figure 1. Reconstruction results of chairs with 4 samples from training and testing set and 1 from real-world image.

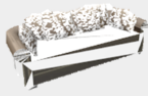

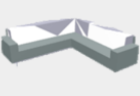
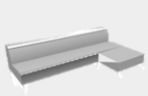


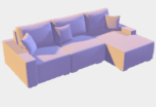
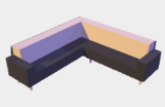
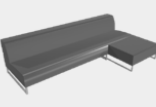
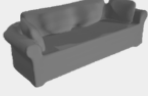
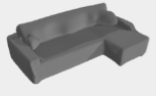
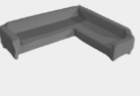
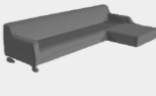
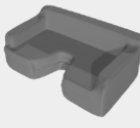
	Train		Test		Real world
rgb					
gt					NA
3d					

Figure 2. Reconstruction results of sofas with 4 samples from training and testing set and 1 from real-world image.

$p \in \mathbb{R}^3$, the signed distance s can be computed as follows:

$$s = \mathcal{T}(W(p, \mathcal{E}(x))) \quad (12)$$

where \mathcal{E} , \mathcal{T} , and \mathcal{W} are the encoder, SDF, and warping function, respectively. The training loss for this end-to-end training is the same as the one introduced in Section 3.1. Note that we do not need the intermediary embeddings c here.

As mentioned in Section 3.2, to successfully perform 3D reconstruction, the encoder must learn to encode different 2D views of the same instance to the same embedding in latent space. However, without the L_1 loss induced in the second phase of 2-phase training (Equation 10), the encoder can only learn such information implicitly from the objective function. Therefore, the constraint on the learned latent















	Train		Test		Real world
rgb					
gt					NA
3d					

Figure 3. Reconstruction results of planes with 4 samples from training and testing set and 1 from real-world image.

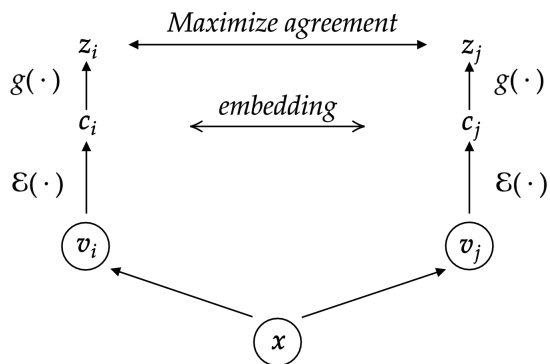


Figure 4. The framework for contrastive learning. Two views are generated based on the source image x . A base encoder $\mathcal{E}(\cdot)$ and a projection head $g(\cdot)$ are trained to maximize agreement using a contrastive loss.

space is not as strong as the one in Section 3.2. Therefore, to ensure a decent latent space, we induce contrastive learning into our end-to-end training.

5.2. Contrastive Learning

Contrastive learning has recently become the mainstream method for self-supervised representation learning. Most approaches [5, 2, 11, 13] leverage the task of instance discrimination, where augmented views from the same im-

age are enforced to have similar embeddings. We adopt SimCLR[2] to obtain such discrimination. As illustrated in Figure 4, the framework comprises of the following four components:

- A data augmentation module that generates two different views of an image, denoted v_i and v_j .
- A base encoder \mathcal{E} that extracts embeddings from views to obtain $c_i = \mathcal{E}(v_i)$.
- A projection head g that maps embeddings to the space where contrastive loss is applied. We use a MLP to obtain $z_i = g(c_i)$
- A contrastive loss function defined for instance discrimination task.

Specifically, the contrastive loss for a positive pair of example (i, j) is defined as:

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_k \exp(\text{sim}(z_i, z_k)/\tau)} \quad (13)$$

where τ is a temperature parameter, and $\text{sim}(u, v) = u^T v / \|u\| \|v\|$ denotes the dot product between ℓ_2 normalized u and v . The encoder pretrained by this method is then used as the initial weight for the later end-to-end training.

Figure 5 shows the advantage of pretraining the encoder with contrastive learning. Compared to the classic ImageNet-pretrained encoder, the encoder pretrained by contrastive-learning method leads to better convergence.

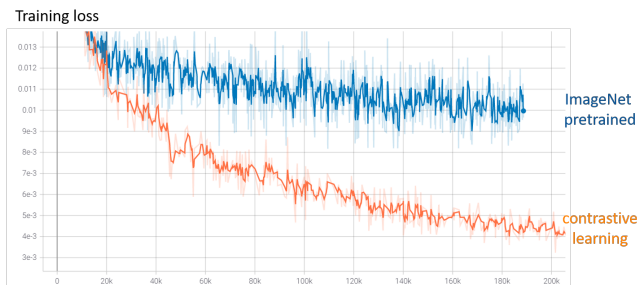


Figure 5. End-to-end training loss of the plane category using two different weight initialization.

The visualization of the learned latent space in Figure 6 also proves the importance of pretraining with contrastive learning. Instances are not distinguishable in latent space otherwise.

6. Comparison

Figure 7, 8 and Table 2 compare the results obtained from 2-phase training in Section 3 and end-to-end training in Section 5. As being shown both qualitatively and quantitatively, when contrastive learning is not applied beforehand, the end-to-end method usually produces worse results (e.g., the chairs category and the sofas category in the last row of Figure 7). However, if the model is pretrained with contrastive learning (i.e., the planes category), it can actually achieve comparable reconstruction performance compared to 2-phase training.

	chairs		sofas		planes	
	mean	median	mean	median	mean	median
2-phase (fix)	1.61	1.02	1.50	0.72	4.19	0.16
2-phase (rnd)	1.82	1.16	1.49	0.63	4.52	0.18
end-to-end* (fix)	1.53	0.93	2.01	1.40	1.76	0.29
end-to-end* (rnd)	1.59	1.00	2.54	1.26	1.75	0.28

Table 2. Chamfer distance ($\times 0.001$) of the results obtained from the 2-phase training and the end-to-end training. Note that contrastive learning is not applied for the chairs category and the sofas category in the end-to-end training.

7. Conclusion

We have presented two frameworks for the 2D-to-3D reconstruction task utilizing the concept of a template. One performs 2-phase training, which has the best result, and the other performs end-to-end training to achieve efficiency, which can be further improved with contrastive learning. The experiment demonstrates that we successfully learn the template containing common geometry characteristics and

the dense 3D representation of each instance, even those from unseen data.

8. Code availability

<https://github.com/Kaminyou/Template-is-all-you-need>.

9. Author contributions

Y.S. Huang proposed the initial idea. M.Y. HO contributed all data preprocessing, including converting OBJ to 100 PNG images, SDF, and PLY. Y.S. Huang implemented warping and SDF submodules. M.Y. HO proposed solution 1 to enhance the performance, and C.J. Chang proposed solution 2 to allow end-to-end training. Y.S. Huang implemented solution one and C.J. Chang implemented solution 2. M.Y. HO implemented and conducted embedding analysis and result evaluation. C.J. Chang contributed result generation function. M.Y. HO, Y.S. Huang, C.J. Chang took responsibility for the training and testing for "sofa", "chair", and "plane" classes, respectively. All authors contributed to the final manuscript.

10. What you have learned

10.1. M.Y. HO

The struggling data preprocessing made me familiar with manifold 3D object representations. For rendering the 2D images, this was also my first time writing scripts for Blender. Although several infrastructures and OS-related issues occurred during this process, this was a good opportunity for me to glance at some essential components in the Unix system. The topic of "3D reconstruction from 2D image" was relatively intriguing. To figure out what was not well-explored or how I could improve or leverage the previous work was time-consuming but enabled me to be familiar with this topic. During the training process, we met many problems, such as the non-decreasing loss value. Coming up with solutions and finally addressing the problems benefited me a lot and would be helpful in the future.

10.2. Y.S. Huang

The 2D-to-3D reconstruction task is always an appealing topic for me. I wish to go further on this. It's nice to have a chance to learn to implement a deep signed distance function, which is nowadays a famous representation for 2D-to-3D reconstruction task, in advance. Our codes are heavily dependent on the reference paper. It took a long time to understand the architecture and designs of that work. This experience will be pretty useful for my future research. I was kind of at a loss when the result turned out not satisfying. Thanks to every amazing idea from my group, the problems were solved, and we eventually get a good result for every

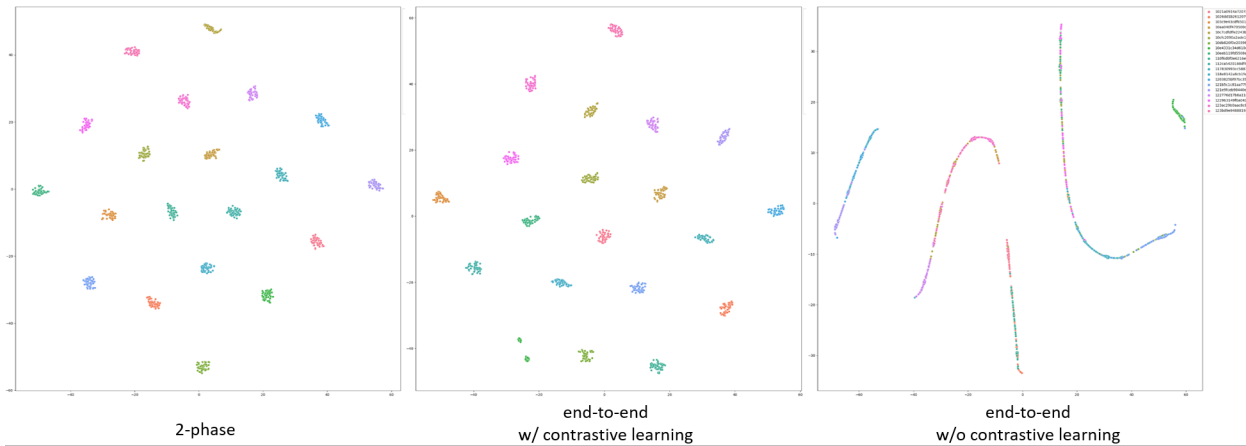


Figure 6. T-SNE visualization of the learned embeddings for the first 20 instances of planes using different training methods. Left: 2-phase training; Middle: end-to-end training with contrastive learning; Right: end-to-end training without contrastive learning. Colors represent different instances.



Figure 7. Comparison of our representation obtained from 2-phase training and end-to-end training. Note that contrastive learning is not applied for the chairs category and the sofas category in the end-to-end training.

proposed hypothesis. The process during this project is very inspiring.

10.3. C.J. Chang

This is my very first time being introduced to the topic of 2D-to-3D reconstruction. It's been quite a journey for us, starting from a straightforward idea to a constructive method with some pretty decent results. There was a time

the reconstruction always came out underwhelming. We spent a lot of effort analyzing every submodule of our framework and, in the end, figured out what kind of constraints should be added to the structure to enforce better information in the latent space for 2D-to-3D reconstruction. It is also amazing how contrastive learning boosts up the performance of a quite naive methodology and produces comparable results. I am truly thankful to have these two



Figure 8. Comparison of the learned templates obtained from 2-phase training and end-to-end training. Note that contrastive learning is not applied for the chairs category and the sofas category in the end-to-end training.

supporting and brilliant members to work with on this interesting topic. The experience gained through this project is definitely helpful for any future study.

11. Conflict of interest

None declared.

References

- [1] 1
- [2] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. 5
- [3] B. O. Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 3
- [4] Y. Duan, H. Zhu, H. Wang, L. Yi, R. Nevatia, and L. J. Guibas. Curriculum deepsdf. In *European Conference on Computer Vision*, pages 51–67. Springer, 2020. 2
- [5] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019. 5
- [6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3
- [7] Y. Jiang, D. Ji, Z. Han, and M. Zwicker. Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2
- [8] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018. 2
- [9] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2, 3
- [10] S. P. B. S. M. P. Z. C. Shaohui Liu, Yinda Zhang. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2
- [11] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding, 2019. 5
- [12] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018. 3
- [13] Z. Wu, Y. Xiong, S. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance-level discrimination, 2018. 5
- [14] Q. Xu, W. Wang, D. Ceylan, R. Mech, and U. Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *arXiv preprint arXiv:1905.10711*, 2019. 2, 3
- [15] Z. Zheng, T. Yu, Q. Dai, and Y. Liu. Deep implicit templates for 3d shape representation, 2020. 2