Supplemental material for "SeqXY2SeqZ: Structure Learning for 3D Shapes by Sequentially Predicting 1D Occupancy Segments From 2D Coordinates"

Zhizhong Han[1,2], Guanhui Qiao[1], Yu-Shen Liu[1⋆], and Matthias Zwicker[2]

[1] School of Software, BNRist, Tsinghua University, Beijing, P. R. China
[2] Department of Computer Science, University of Maryland, College Park, USA
h312h@umd.edu, qiaogh18@mails.tsinghua.edu.cn, liuyushen@tsinghua.edu.cn, zwicker@cs.umd.edu

1 The encoding process

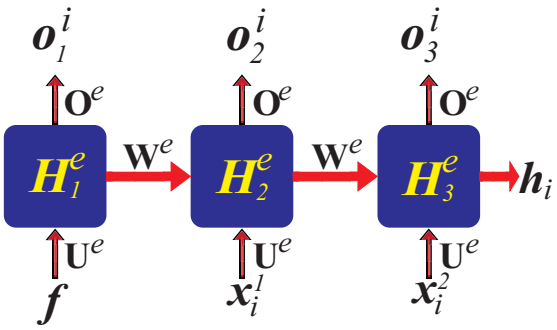

Fig. 1. The encoding process.

In the encoder RNN shown in Fig. 1, we use the following equations to calculate the hidden states at each step,

$$\boldsymbol{H}_a^e = ReLU(\mathbf{U}^e \boldsymbol{f}), a = 1, \tag{1}$$

$$\boldsymbol{H}_a^e = ReLU(\mathbf{U}^e \boldsymbol{x}_i^{a-1} + \mathbf{W}^e \boldsymbol{H}_{a-1}^e), a > 1, \tag{2}$$

where $\mathbf{U}^e$ and $\mathbf{W}^e$ are learnable parameters. $\boldsymbol{f}$ is the shape condition which can be either learnable parameter in autoencoding experiment or feature vector

⋆ Corresponding Author. This work was supported by National Key R&D Program of China (2018YFB0505400) and NSF (award 1813583).

extracted from images in single image reconstruction experiment, and $\{\boldsymbol{x}_i^{a-1}\}$ are location embeddings which are also learnable parameter, and $a \in [1, 2, 3]$.

Then, we can calculate the output $\boldsymbol{o}_a^i$ at each step as defined below,

$$\boldsymbol{o}_a^i = \mathbf{O}^e \boldsymbol{H}_a^e + \boldsymbol{b}^e, \tag{3}$$

where $\mathbf{O}^e$ and $\boldsymbol{b}^e$ are learnable parameters.

2 The decoding process

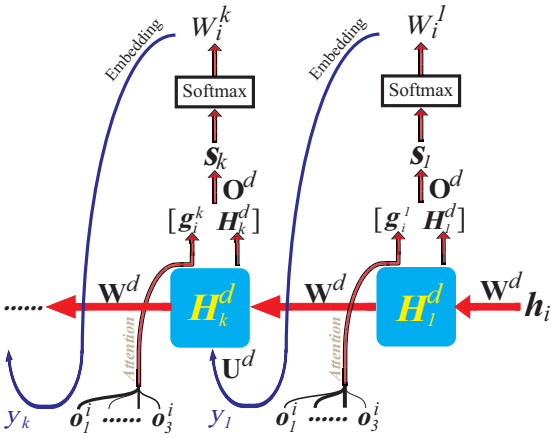

Fig. 2. The decoding process.

In the decoder shown in Fig. 2, we use attention weights d_a^k learned by attention mechanism [1] at each step k to aggregate the output $\boldsymbol{o}_a^i$ at all steps of encoder below,

$$\boldsymbol{g}_i^k = \sum_{a=1}^{3} d_a^k \times \boldsymbol{o}_a^i. \tag{4}$$

The hidden state at each step is calculated as follows,

$$\boldsymbol{H}_k^d = ReLU(\mathbf{W}^d \boldsymbol{h}_i), k = 1, \tag{5}$$

$$\boldsymbol{H}_k^d = ReLU(\mathbf{U}^d \boldsymbol{y}_{k-1} + \mathbf{W}^d \boldsymbol{H}_{k-1}^d), k > 1, \tag{6}$$

where $\mathbf{W}^d$ and $\mathbf{U}^d$ are learnable parameters, $\{\boldsymbol{y}_{k-1}\}$ are location embeddings which are learnable parameters.

Then, we combine the hidden state and the aggregated output from encoder to learn a feature for start and end location prediction as follows,

$$\boldsymbol{s}_k = \mathbf{O}^d[\boldsymbol{g}_i^k \ \boldsymbol{H}_k^d] + \boldsymbol{b}^d, \tag{7}$$

where $\mathbf{O}^d$ and $\boldsymbol{b}^d$ are learnable parameters.

Finally, we predict the start or end location w_i^k at each step by a softmax layer defined below,

$$\boldsymbol{y}_k = \mathbf{W}\boldsymbol{s}_k + \boldsymbol{b}, \tag{8}$$

$$w_i^k = \arg\max \ softmax(\boldsymbol{y}_k), \tag{9}$$

where $\mathbf{W}$ and $\boldsymbol{b}$ are learnable parameters.

3 More analysis

Occupancy segment visualization. With the auto-encoded shapes, we justify the efficiency of our voxel tubelization by visualizing the number of predicted occupancy segments at each 2D coordinate in Fig. 3. For a simple car in Fig. 3 (a), it is enough to use only one occupancy segment to represent the geometry at each 2D coordinate. Although the table in Fig. 3 (b) is more complex, we can still leverage no more than three occupancy segments to represent the geometry at almost all 2D coordinates. In Fig. 3 (c) and (d), we visualize the reconstructed chairs with different numbers of occupancy segments. For complex structures in chairs, it is still enough to reconstruct almost a whole shape using only two occupancy segments.

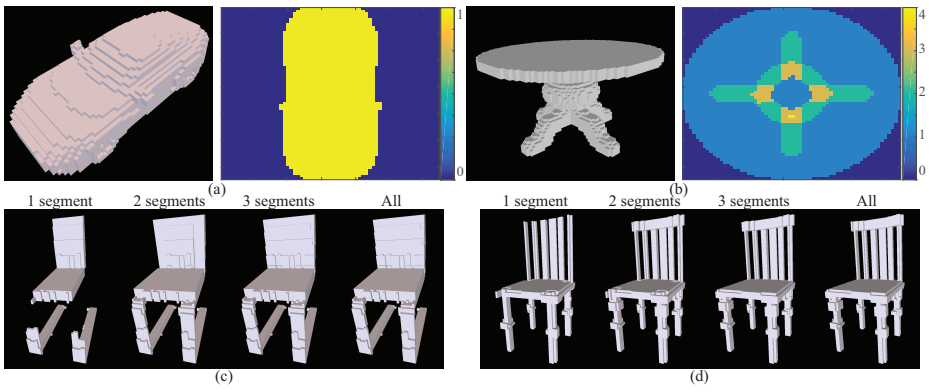

Fig. 3. The efficiency demonstration of occupancy segments at 2D coordinates. We show the numbers of occupancy segments at all 2D coordinates as an image for a car (a) and table (b). We also visualize the shapes reconstructed by different numbers of occupancy segments at all 2D coordinates in (c) and (d). In all cases, only few occupancy segments are needed to represent a complex shape.

Interpolation. We visualize the shape condition space learned in auto-encoding experiment by visualizing the interpolation between two shapes. We interpolate the features between two learned shape conditions, which are further leveraged to reconstruct shapes shown in Fig. 4. The transition shows how one shape is gradually transformed to another one by growing or shrinking occupancy segments in each tube at all 2D locations.

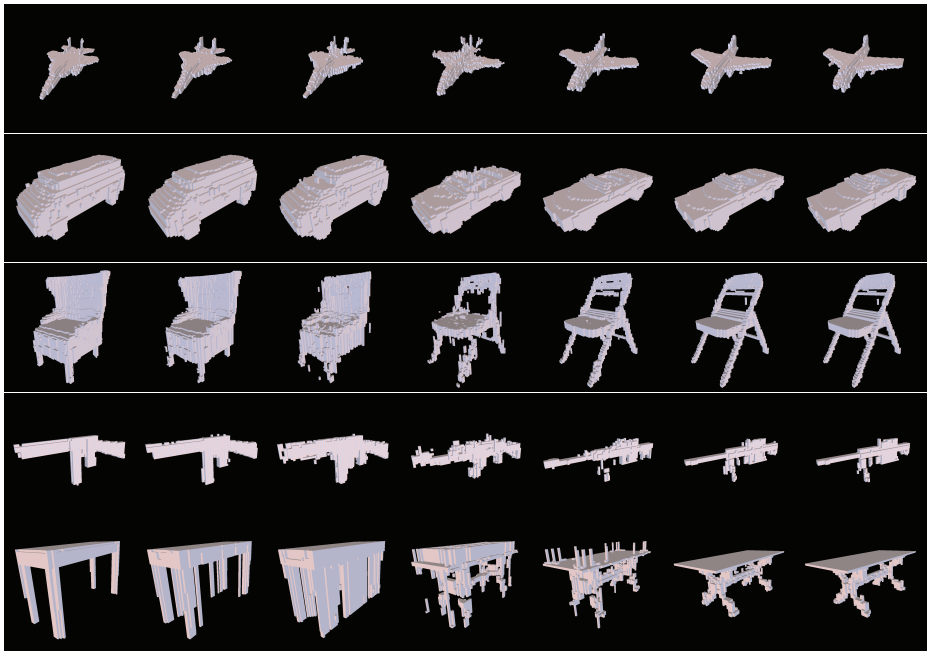

Fig. 4. Interpolated shapes in the learned 3D feature space. We use the learned shape conditions of two shapes to establish a line in the feature space, and then uniformly sample five features along this line by interpolating the two shape conditions. Finally, we generate a interpolated shape using each one of the five interpolated features as shape condition.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. CoRR **abs/1409.0473** (2014)