HyperCube: Implicit Field Representations of Voxelized 3D Models (Student Abstract)

Magdalena Proszewska[1], Marcin Mazur[2], Tomasz Trzciński[3,4,5], Przemysław Spurek[2]

[1]University of Edinburgh
[2]Jagiellonian University in Kraków
[3]Warsaw University of Technology
[4]IDEAS NCBR
[5]Tooploox

m.proszewska@ed.ac.uk, marcin.mazur@uj.edu.pl, tomasz.trzcinski@pw.edu.pl, przemyslaw.spurek@uj.edu.pl

Abstract

Implicit field representations offer an effective way of generating 3D object shapes. They leverage an implicit decoder (IM-NET) trained to take a 3D point coordinate concatenated with a shape encoding and to output a value indicating whether the point is outside the shape. This approach enables the efficient rendering of visually plausible objects but also has some significant limitations, resulting in a cumbersome training procedure and empty spaces within the rendered mesh. In this paper, we introduce a new HyperCube architecture based on interval arithmetic that enables direct processing of 3D voxels, trained using a hypernetwork paradigm to enforce model convergence. The code is available at https://github.com/mproszewska/hypercube.

Introduction The recently introduced in (Chen and Zhang 2019) implicit decoder (IM-NET) architecture has several advantages over a standard convolutional model. First of all, it can produce outputs of various resolutions, including those not observed in the training. Furthermore, IM-NET learns shape boundaries instead of voxel distributions over the volume, which results in surfaces of a higher quality. On the other hand, IM-NET has some limitations. First of all, the point coordinates are concatenated with the shape embedding and to reconstruct an object the model needs to possess knowledge about all objects present in the entire dataset. Therefore IM-NET architecture is hard to train on many different classes. Moreover, the implicit decoder processes only points sampled from within voxels, instead of the entire voxels. This yields problems at the classification boundaries at object edges and gives severe rendering artifacts.

In this paper, we address the above limitations by introducing a novel approach (HyperCube) to the implicit representation of 3D objects. It leverages a hypernetwork architecture to produce weights of a target implicit decoder, based on the input feature vector defining a voxel. This target decoder assigns an *inside* or *outside* of a shape label to each processed voxel. Such architecture is more compact than IM-NET and therefore much faster to train, while it does not need to know the distribution of all objects in the training dataset to obtain object reconstructions. Furthermore, its design allows a flexible adjustment of the target network processing feature vectors. This enables us to input the entire voxels into the model leveraging interval arithmetic and the IntervalNet architecture (Morawiecki et al. 2019), and leads to the inception of our HyperCube-Interval model. The HyperCube-Interval architecture takes as an input relatively small 3D cubes (hence the name), instead of 3D point samples within the voxels. Therefore, it does not produce empty space in the reconstructed mesh representation, as visualized in Fig. 4.

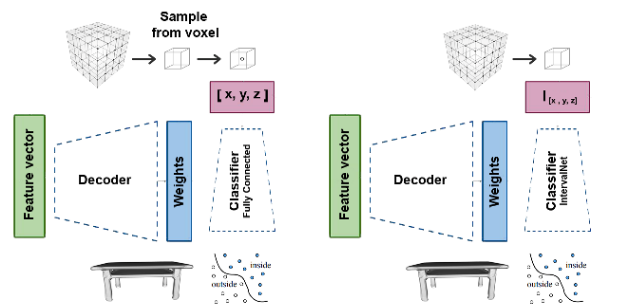

Figure 1: Comparison of HyperCube (left) and HyperCube-Interval (right) architectures.

Our solution is an extension of the data reprehension technique from IM-NET with the hypernetwork paradigm used in the HyperCloud model (Spurek et al. 2020a,b). As a consequence, we take the best of both methods and obtain a reconstruction quality of the IM-NET, while reducing the training and inference time as in the case of the HyperCloud.

HyperCube and HyperCube-Interval Our HyperCube model takes a voxel representation $X \subset \mathbb{R}^3$ and passes it to a hypernetwork $H_\phi \colon X \to \theta$ to output weights of a (small) target network $T_\theta \colon X \to \{0, 1\}$. Next, X (i.e., any point uniformly sampled from X) is compared with the output from the target network T_θ (we take a voxel grid and predict inside/outside labels). To train our model we use the mean squared error loss function.

The above architecture gives competitive qualitative and quantitative results to IM-NET, yet it offers a significant processing speedup. However, the remaining shortcoming of IM-NET, namely the reconstruction artifacts close to clas-

		Plane	Car	Chair	Rifle	Table
MSE	IM-NET	**2.14**	4.99	11.43	**1.91**	10.67
	HyperCube	2.44	**4.37**	**9.07**	**1.91**	**9.37**
IoU	IM-NET	**78.77**	89.26	65.65	**72.88**	71.44
	HyperCube	65.35	**90.05**	**72.61**	63.97	**73.78**
CD	IM-NET	**4.23**	5.44	9.05	**3.77**	11.54
	HyperCube	4.74	**3.36**	**8.35**	4.20	**8.82**

Table 1: *Reconstruction capabilities*. The mean is calculated for reconstructions of 100 first elements from the test set in each category. MSE is multiplied by 10^3, IoU by 10^2, and CD by 10^4.

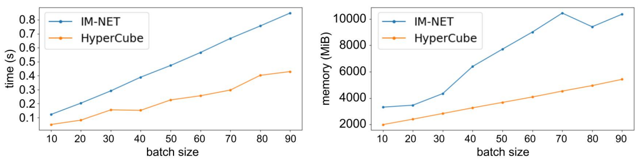

Figure 2: Comparison of training times and GPU memory used by IM-NET and HyperCube.

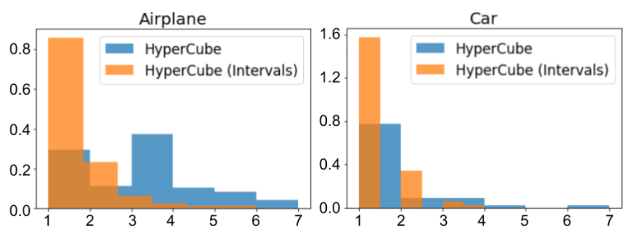

Figure 3: The number of connected components produced by meshes obtained via architectures with and without intervals.

sification boundaries resulting from sampling strategy, remains. To address this limitation and process entire 3D cubes instead of sampled points, we leverage interval arithmetic (Dahlquist and Björck 2008) and a neural architecture that implements it, i.e., the IntervalNet (Morawiecki et al. 2019). This leads to the HyperCube-Interval model that is a copy of HyperCube using IntervalNet instead of MLP as a target network. We use worst-case accuracy instead of cross-entropy to ensure that the whole voxel is correctly classified.

Training time and memory footprint Fig. 2 displays a comparison between our HyperCube method and the competing IM-NET. For a fair comparison, we evaluated the architectures proposed in (Chen and Zhang 2019). The models were trained on the ShapeNet dataset. Our HyperCube approach leads to a significant reduction in both training time and memory footprint due to a more compact architecture.

Reconstruction capabilities For the quantitative comparison of our method with the current state-of-the-art solutions in the reconstruction task, we follow the approach introduced in (Chen and Zhang 2019). Metrics for encoding and reconstruction are based on point-wise distances, e.g.,

		Plane	Car	Chair	Rifle	Table
MSE	IM-NET	**2.98**	10.98	17.11	**2.41**	13.38
	IM-NET %	0.45	**0.70**	0.68	0.56	0.64
	HyperCube	2.99	**7.47**	**16.46**	2.61	**13.23**
	HyperCube %	**0.57**	**0.70**	**0.72**	**0.68**	**0.69**
IoU	IM-NET	56.05	77.36	50.46	51.53	54.08
	IM-NET %	0.61	**0.71**	0.72	0.69	0.75
	HyperCube	**61.68**	**86.34**	**53.52**	**59.80**	**61.23**
	HyperCube %	**0.67**	**0.71**	**0.76**	**0.73**	**0.77**
CD	IM-NET	7.38	5.72	13.99	8.06	17.36
	IM-NET %	0.58	0.71	**0.78**	0.72	**0.82**
	HyperCube	**5.02**	**4.28**	**12.92**	**4.96**	**12.49**
	HyperCube %	**0.66**	**0.74**	**0.78**	**0.74**	0.81

Table 2: *Generative capabilities*. The mean of minimum MSE/CD, maximum IoU between the test set and the collection of generated objects with 5 times greater resolution, % of the test set objects matched as closest ones.

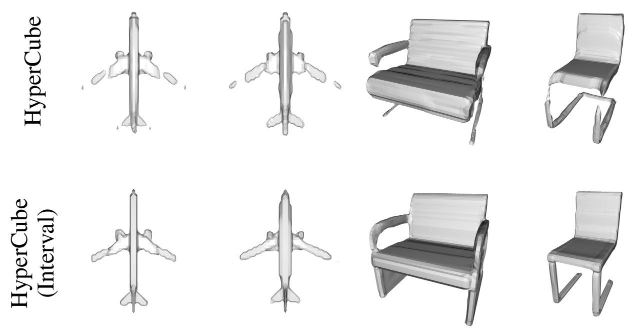

Figure 4: Competition between architectures working on points (HyperCube) and intervals (HyperCube-Interval).

Chamfer Distance (CD), Mean Squared Error (MSE), and Intersection over Union (IoU) on voxels. Results presented in Table 1 show that HyperCube achieve comparable results to the reference method.

Generative capabilities We examine the generative capabilities of the provided HyperCube model compared to IM-NET. Table 2 demonstrates that both models perform similarly along all metrics.

Intervals vs points The classification boundary can be regularized using IntervalNet. In Fig. 4 we present such examples. As we can see, HyperCube-Interval produces single objects without empty space. To verify it, we calculate the number of connected components produced by mesh and visualize them on histograms, see Fig. 3. HyperCube-Interval produces better models for Airplane and Car classes.

Conclusions In this work, we introduce a new implicit field representation of 3D models. It is more lightweight and faster to train than existing solutions while offering competitive or superior results. Finally, our method allows incorporating interval arithmetic which enables processing entire 3D voxels, instead of their sampled version, hence yielding more plausible and higher quality 3D reconstructions.

Acknowledgements

This research was funded by National Science Centre, Poland, grants no 2020/39/B/ST6/01511, 2021/43/B/ST6/01456, and 2022/45/B/ST6/02817. Some of the work was supported by funds from the flagship project entitled "Artificial Intelligence Computing Center Core Facility" from the DigiWorld Priority Research Area within the Excellence Initiative – Research University program at Jagiellonian University in Kraków.

References

Chen, Z.; and Zhang, H. 2019. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5939–5948.

Dahlquist, G.; and Björck, Å. 2008. *Numerical methods in scientific computing, volume I*. SIAM.

Morawiecki, P.; Spurek, P.; Śmieja, M.; and Tabor, J. 2019. Fast and stable interval bounds propagation for training verifiably robust models. *arXiv preprint arXiv:1906.00628*.

Spurek, P.; Winczowski, S.; Tabor, J.; Zamorski, M.; Zieba, M.; and Trzciński, T. 2020a. Hypernetwork approach to generating point clouds. *Proceedings of the 37th International Conference on Machine Learning (ICML)*.

Spurek, P.; Zieba, M.; Tabor, J.; and Trzciński, T. 2020b. Hyperflow: Representing 3d objects as surfaces. *arXiv preprint arXiv:2006.08710*.