Deep Learning on Meshes and Point Clouds

Ruben Wiersma

ETH Zurich, Switzerland

Abstract

Data-driven algorithms have proven valuable for many tasks, with high-profile success in image understanding and synthesis and language modelling. In this course, we look at how such algorithms can be used for data on curved surfaces. Our aim is to give researchers the required background to use such algorithms in an informed way in their own research. In the first part, we consider the types of data and tasks that are relevant for mesh- and point-cloud surfaces, the requirements on our algorithms (e.g., scaling and generalization over the representation), and review the state-of-the-art for how these requirements can be met. In the second part, we will give a hands-on tutorial on setting up a neural network for learning a simple task on 3D meshes.

CCS Concepts

• ***Computing methodologies*** $\rightarrow$ ***Shape analysis; Neural networks;***

1. Introduction

Deep learning has achieved remarkable results on 2D image data and text, often surpassing classical methods on tasks such as classification and segmentation [LBH15, VSP*17]. This motivates work on generalizing deep learning to 3D geometric data [GWH*20, BBL*17, LSL*19, BBCV21]. Many real-world tasks require reasoning about 3D geometry rather than 2D projections. Relevant applications include part segmentation [CFG*15], shape correspondence for texture transfer and registration [LRR*17, AO23], medical shape analysis (e.g., morphological phenotyping [TSR*23]) and the acceleration of physically-based simulations, for example, predicting arterial wall stress from geometry [SDHL*24].

Alongside the development of geometric deep learning techniques, 3D capture hardware has become more affordable and widely available and many 3D datasets are now available, such as ShapeNet [CFG*15], ModelNet [WSK*15], ScanNet [DCS*17], S3DIS [ASZ*16], Thingi10k [ZJ16], ABC [KMJ*19] and Objaverse-XL [DLW*23], which provide (annotated) point clouds and meshes for training and evaluation.

This tutorial provides a high-level map of deep learning on 3D shapes. Rather than covering every technique exhaustively, we aim to introduce readers to the key concepts and landmarks so they can explore the literature independently and apply these methods to their own tasks. The material assumes some familiarity with optimization or machine learning and foundational concepts are reviewed from a geometric perspective. This document loosely follows the contents of the lecture, annotated with references. A recording of the lecture, lecture slides and tutorial code are available at rubenwiersma.nl/deeplearning.

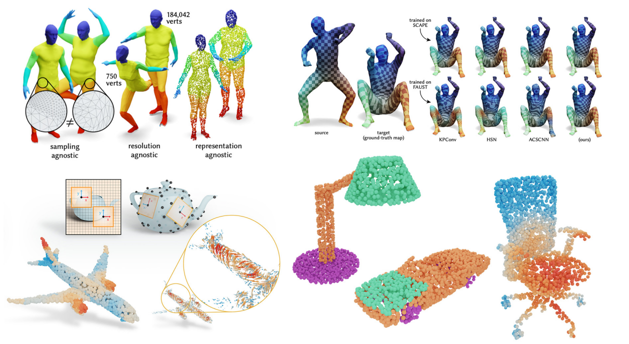

Figure 1: *A collage of results and methods in deep learning on meshes and point clouds. Top: Segmentation and correspondence predictions from DiffusionNet [SACO22], demonstrating its robustness to different discretizations. Middle left: illustrating the differences between image- and 3D data. Bottom left: visualizations of features learned on point clouds using DeltaConv [WNEH22]. Bottom right: segmentation outputs and activation maps for DeltaConv.*

2. Machine Learning Fundamentals

In classical algorithm design, the programmer specifies the full procedure mapping an input x to an output y. In machine learning, that procedure is implemented by a model $f_\theta : X \rightarrow Y$ whose parameters θ are trained on a dataset of input-output pairs (X,Y). The parameters are optimized ("trained" in machine learning), typically to minimize the average squared error between predictions $f_\theta(x)$ and ground-truth labels y.

DOI: 10.2312/egt.20261000

What distinguishes deep learning from other machine learning methods is, in our view, the combination of three factors: 1) the dataset (X,Y) is large (in the order of millions of samples), 2) the model f_θ is nonlinear and layered and 3) the parameters θ are optimized using stochastic gradient descent with backpropagation (due to the large dataset size and parameter count). The combination of these factors is feasible due to the ability to parallelize the evaluation on consumer-grade GPU hardware. Note that alternative methods could be preferred when one of these components is absent. For example, if a dataset is limited in size, optimizing complex nonlinear models with gradient descent may not be the right choice. Simpler models and better-suited optimizers can often be faster and more accurate. See Solomon's "Numerical Algorithms" [Sol15] for a helpful refresher on these techniques.

There are three common implementations for the model f_θ: multi-layer perceptrons, convolutional neural networks and transformers.

Multi-Layer Perceptrons (MLPs). An MLP consists of a sequence of layers, where a layer is a linear transformation followed by a nonlinear activation function $\sigma(\mathbf{Wx})$. An example activation function is the rectified linear unit $\sigma_{\text{relu}}(x) = \max(0,x)$. Each layer can be interpreted from a geometric perspective as partitioning space with a hyperplane. Multiple layers partition the input domain into smaller chunks. Another perspective is to see MLPs as universal function approximators [HSW89].

Convolutional Neural Networks (CNNs). Applying an MLP directly to a flattened image is inefficient and fails to exploit spatial structure. Convolutional Neural Networks [LBBH02] solve this by applying a shared learnable kernel at every location, yielding both weight sharing and translation invariance (up to a certain degree [SKVG20]). Stacking convolution layers with pooling produces features of increasing abstraction.

Transformers. A helpful analogy for understanding transformers in relation to CNNs is the bilateral filter [PKTD09]. A bilateral filter weights contributions from neighboring pixels by both spatial proximity and signal similarity. When computing a convolution in CNNs, only the spatial proximity (location) of a neighboring pixel is used to determine its weight. Transformers [VSP*17, KDW*21] increase the flexibility of the kernel by including signal similarity, similar to a bilateral filter, in a process called self-attention. This added expressivity comes at the cost of more computation.

3. From 2D to 3D: Challenges and Representations

Transferring these architectures to 3D data introduces several fundamental challenges: 3D surfaces discretized as point clouds and meshes are not defined on a regular grid, so the regular structure that makes convolution efficient on images is absent. The domain is curved, rather than flat. And unlike images, where directions are globally consistent, 3D shapes lack a canonical coordinate system. Operations must work with local coordinate frames (charts) or first compute a global parametrization (not considered in this tutorial, but an example can be found in [MGA*17]).

The primary surface representations considered here are point clouds (unordered sets of 3D positions) and triangular meshes (which include edge and face connectivity). Voxel grids are a natural extension of 2D pixel grids to 3D and are effective (e.g., [LRS*18, CGS19]), but they are not the focus in this tutorial. Implicit representations such as signed distance functions are typically sampled to a mesh or point cloud before analysis, so they are also not treated directly.

An important practical consideration is dataset size. Dominant 3D benchmarks such as ShapeNet [CFG*15] and ModelNet40 [WSK*15] contain tens of thousands of shapes. This is very small compared to state-of-the-art image datasets (e.g., LAION-5B contains 5 billion images [SBV*22]). Even Objaverse-XL [DLW*23], containing roughly 10 million shapes, does not come close to this scale. Therefore, benchmark saturation on 3D datasets may reflect data limitations as much as algorithmic ones.

4. Deep Learning on Point Clouds and Meshes

In the following, we will discuss some of the most widespread and well-known techniques for deep learning on point clouds and meshes. This treatment is intended to cover the basics for those new to the field. Please refer to dedicated surveys for a more comprehensive view of the field [GWH*20, BBL*17, LSL*19, BBCV21].

4.1. PointNet and PointNet++

The key constraint for deep learning methods on point clouds is permutation invariance: the output must not depend on the order in which points are provided. Otherwise, the method may fail on cases that do not follow the same order. PointNet [QSMG17] achieves this by applying a shared MLP to each point and then aggregating the output features globally using a per-feature maximum over all points.

A limitation of this approach is that taking a global maximum discards all neighborhood information. As a simple example: understanding curvature requires knowledge of a point's surroundings, which is something PointNet cannot, by definition, analyse. Thus, PointNet++ [QYSG17] applies an MLP on local neighborhoods (defined by k-nearest neighbors or a radius graph), then aggregates with a local maximum, and stacks these operations hierarchically with pooling, mirroring the structure of a CNN.

Despite its age, the core ideas in PointNet++ remain competitive. For example, PointNeXt [QLP*22] shows that PointNet++ can benefit from improved training and scaling strategies introduced by follow-up works. This leads to two practical take-aways when applying 3D deep learning methods: first, it could be useful to build on an existing state-of-the-art codebase, as it is difficult to recreate all the right strategies from scratch and, second, it is worth trying a simpler model if it can be switched out easily; many improvements can be explained by changes outside the model.

4.2. Message Passing on Graphs

A unifying framework for operations on 3D surfaces is message passing on graphs [GSR*17]. Consider the input point cloud or mesh as a graph, where the nodes represent points and the edges of the graph connect nearby points (via k-NN, radius graph or mesh edges). In every layer of the network, a *message* is computed at each node and *passed* over the edges of the graph for aggregation.

This paradigm can be used to understand a range of geometric deep learning methods and is used as a framework in libraries, such as PyTorch Geometric [FL19]. In PointNet++, the *message* is computed per point by an MLP. The *passing* operation takes the maximum over neighbors. EdgeConv [WSL*18] is a close variant that computes relative features over edges. Graph Convolutional Networks (GCN) [KW16] compute a message with a linear transformation and take a weighted-sum aggregation, followed by a non-linearity.

The choice of neighborhood matters. For example, mesh edges respect the connectivity of the input shape, where k-NN or radius graphs do not. Another consideration is how well the aggregation operators map to the GPU. For example, the uniform neighborhood sizes in k-NN graphs map well to GPU tensor operations.

4.3. Incorporating Geometry

Purely combinatorial graph operations ignore the geometric information on the surface. This could lead to issues with generalization. For example, irregular sampling densities and non-uniform mesh quality can cause a network to perform well on a specific dataset's connectivity structure. Geometry processing has studied these problems for decades an offers many useful tools in this regard. An example of a method building on these tools is DiffusionNet [SACO22]. DiffusionNet learns features analogous to heat-kernel signatures [SOG09], but with learnable diffusion time parameters. The resulting features are robust across different discretizations and representation types (meshes, point clouds, etc.). Other examples are MeshCNN [HHF*19], which learns directly on the mesh edges using geometric input features and DeltaConv [WNEH22], which employs differential operators to construct anisotropic convolution operators on surfaces.

Accounting for rotations 3D shapes may appear in arbitrary orientations. Consequently, 3D deep learning methods should ideally be invariant to such rotations. Several solutions are proposed in the literature: data augmentation with random rotations; learned canonical alignment (as in [QSMG17]); invariant input features such as heat-kernel signatures (used in [SACO22]) or edge lengths and dihedral angles (as in [HHF*19]); features encoded in local frames and enforcing equivariance or invariance in the network architecture, e.g., with group equivariance [TSK*18, EABMD17, BBCV21].

4.4. Transferring Transformers to 3D

Until now, we have mainly considered analogies and applications of MLPs and CNNs to 3D surfaces. Self-attention, the core building block of transformers, also transfers naturally to graph-structured data [VCC*18, ZJJ*, WJW*24]. One can treat each node as the query and neighboring nodes as keys and values. The main challenge is computational: attention over all pairs does not scale well to large point clouds. Practical approaches for point clouds and images apply attention within local neighborhoods [ZJJ*], serialize the point cloud along a space-filling curve [WJW*24] or combine CNN-based feature extraction with attention at a coarser scale [KDW*21]. Transformers often use a positional encoding for each token. It should be noted that positional encoding in 3D should also respect the same invariance and equivariance requirements discussed above.

5. Conclusion

Deep learning on 3D shapes is a rich and active field. The practical challenges of data scale, geometric invariances and robustness have parallels in classical geometry processing and computer graphics and familiarity with both fields can be highly valuable. Concluding on a practical note: start with simple models, inspect the input and output of your methods carefully and add complexity where justified by the task and dataset. Code and slides accompanying this tutorial are available at rubenwiersma.nl/deeplearning.

References

[AO23] ATTAIKI S., OVSJANIKOV M.: Understanding and Improving Features Learned in Deep Functional Maps. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (Vancouver, BC, Canada, June 2023), IEEE, pp. 1316–1326. URL: https://ieeexplore.ieee.org/document/10204073/, doi:10.1109/CVPR52729.2023.00133. 1

[ASZ*16] ARMENI I., SENER O., ZAMIR A. R., JIANG H., BRILAKIS I., FISCHER M., SAVARESE S.: 3{D} semantic parsing of large-scale indoor spaces. In *CVPR* (2016). 1

[BBCV21] BRONSTEIN M. M., BRUNA J., COHEN T., VELIČKOVIĆ P.: Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, May 2021. arXiv:2104.13478 [cs]. URL: http://arxiv.org/abs/2104.13478, doi:10.48550/arXiv.2104.13478. 1, 2, 3

[BBL*17] BRONSTEIN M. M., BRUNA J., LECUN Y., SZLAM A., VANDERGHEYNST P.: Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine 34*, 4 (2017), 18–42. 1, 2

[CFG*15] CHANG A. X., FUNKHOUSER T., GUIBAS L., HANRAHAN P., HUANG Q., LI Z., SAVARESE S., SAVVA M., SONG S., SU H., XIAO J., YI L., YU F.: *ShapeNet: An Information-Rich 3D Model Repository*. Tech. Rep. arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 1, 2

[CGS19] CHOY C., GWAK J., SAVARESE S.: 4{D} Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *CVPR* (2019). 2

[DCS*17] DAI A., CHANG A. X., SAVVA M., HALBER M., FUNKHOUSER T., NIESSNER M.: ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Honolulu, HI, July 2017), IEEE, pp. 2432–2443. URL: https://ieeexplore.ieee.org/document/8099744/, doi:10.1109/CVPR.2017.261. 1

[DLW*23] DEITKE M., LIU R., WALLINGFORD M., NGO H., MICHEL O., KUSUPATI A., FAN A., LAFORTE C., VOLETI V., GADRE S. Y., VANDERBILT E., KEMBHAVI A., VONDRICK C., GKIOXARI G., EHSANI K., SCHMIDT L., FARHADI A.: Objaverse-xl: A universe of 10m+ 3d objects. *arXiv preprint arXiv:2307.05663* (2023). 1, 2

[EABMD17] ESTEVES C., ALLEN-BLANCHETTE C., MAKADIA A., DANIILIDIS K.: Learning SO(3) Equivariant Representations with Spherical {CNNs}. In *ECCV* (2017). 3

[FL19] Fey M., Lenssen J. E.: Fast Graph Representation Learning with PyTorch Geometric. *ICLR Workshop on Representation Learning on Graphs and Manifolds* (2019). arXiv: 1903.02428. URL: http://arxiv.org/abs/1903.02428. 3

[GSR*17] Gilmer J., Schoenholz S. S., Riley P. F., Vinyals O., Dahl G. E.: Neural Message Passing for Quantum Chemistry. *CoRR abs/1704.01212* (2017). _eprint: 1704.01212. URL: http://arxiv.org/abs/1704.01212. 2

[GWH*20] Guo Y., Wang H., Hu Q., Liu H., Liu L., Bennamoun M.: Deep Learning for 3D Point Clouds: A Survey. *IEEE TPAMI* (2020), 1. 1, 2

[HHF*19] Hanocka R., Hertz A., Fish N., Giryes R., Fleishman S., Cohen-Or D.: Meshcnn: a network with an edge. *ACM Trans. Graph. 38*, 4 (July 2019). URL: https://doi.org/10.1145/3306346.3322959, doi:10.1145/3306346.3322959. 3

[HSW89] Hornik K., Stinchcombe M., White H.: Multilayer feedforward networks are universal approximators. *Neural Networks 2*, 5 (Jan. 1989), 359–366. URL: https://www.sciencedirect.com/science/article/pii/0893608089900208, doi:10.1016/0893-6080(89)90020-8. 2

[KDW*21] Kolesnikov A., Dosovitskiy A., Weissenborn D., Heigold G., Uszkoreit J., Beyer L., Minderer M., Dehghani M., Houlsby N., Gelly S., Unterthiner T., Zhai X.: An image is worth 16x16 words: Transformers for image recognition at scale. 2, 3

[KMJ*19] Koch S., Matveev A., Jiang Z., Williams F., Artemov A., Burnaev E., Alexa M., Zorin D., Panozzo D.: ABC: A Big CAD Model Dataset for Geometric Deep Learning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (Long Beach, CA, USA, June 2019), IEEE, pp. 9593–9603. URL: https://ieeexplore.ieee.org/document/8954378/, doi:10.1109/CVPR.2019.00983. 1

[KW16] Kipf T. N., Welling M.: Semi-Supervised Classification with Graph Convolutional Networks. *CoRR abs/1609.02907* (2016). _eprint: 1609.02907. URL: http://arxiv.org/abs/1609.02907. 3

[LBBH02] LeCun Y., Bottou L., Bengio Y., Haffner P.: Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*, 11 (2002), 2278–2324. 2

[LBH15] LeCun Y., Bengio Y., Hinton G.: Deep learning. *Nature 521*, 7553 (May 2015), 436–444. URL: https://www.nature.com/articles/nature14539, doi:10.1038/nature14539. 1

[LRR*17] Litany O., Remez T., Rodola E., Bronstein A., Bronstein M.: Deep Functional Maps: Structured Prediction for Dense Shape Correspondence. In *2017 IEEE International Conference on Computer Vision (ICCV)* (Venice, Oct. 2017), IEEE, pp. 5660–5668. URL: http://ieeexplore.ieee.org/document/8237865/, doi:10.1109/ICCV.2017.603. 1

[LRS*18] Li Y., Rui B., Sun M., Wu W., Di X., Chen B., Bu R., Sun M., Wu W., Di X., Chen B.: PointCNN: Convolution on x-transformed points. In *NeurIPS* (2018). 2

[LSL*19] Liu W., Sun J., Li W., Hu T., Wang P.: Deep Learning on Point Clouds and Its Application: A Survey. *Sens 19*, 19 (2019). 1, 2

[MGA*17] Maron H., Galun M., Aigerman N., Trope M., Dym N., Yumer E., Kim V. G., Lipman Y.: Convolutional neural networks on surfaces via seamless toric covers. *ACM Trans. Graph 36*, 4 (2017), 71. 2

[PKTD09] Paris S., Kornprobst P., Tumblin J., Durand F.: Bilateral filtering: Theory and applications. *Foundations and trends in computer graphics and vision 4*, 1 (2009), 1–73. 2

[QLP*22] Qian G., Li Y., Peng H., Mai J., Al Kader Hammoud H. A., Elhoseiny M., Ghanem B.: Pointnext: revisiting pointnet++ with improved training and scaling strategies. In *Proceedings of the 36th International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2022), NIPS '22, Curran Associates Inc. 2

[QSMG17] Qi C. R., Su H., Mo K., Guibas L. J.: PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017* (2017), vol. 2017-Janua, pp. 77–85. arXiv: 1612.00593. doi:10.1109/CVPR.2017.16. 2, 3

[QYSG17] Qi C. R., Yi L., Su H., Guibas L. J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems* (2017), pp. 5099–5108. Journal Abbreviation: NeurIPS. 2

[SACO22] Sharp N., Attaiki S., Crane K., Ovsjanikov M.: DiffusionNet: Discretization Agnostic Learning on Surfaces. *ACM Transactions on Graphics 41*, 3 (June 2022), 1–16. URL: https://dl.acm.org/doi/10.1145/3507905, doi:10.1145/3507905. 1, 3

[SBV*22] Schuhmann C., Beaumont R., Vencu R., Gordon C., Wightman R., Cherti M., Coombes T., Katta A., Mullis C., Wortsman M., Schramowski P., Kundurthy S., Crowson K., Schmidt L., Kaczmarczyk R., Jitsev J.: LAION-5B: An open large-scale dataset for training next generation image-text models, Oct. 2022. arXiv:2210.08402 [cs]. URL: http://arxiv.org/abs/2210.08402, doi:10.48550/arXiv.2210.08402. 2

[SDHL*24] Suk J., De Haan P., Lippe P., Brune C., Wolterink J. M.: Mesh neural networks for SE(3)-equivariant hemodynamics estimation on the artery wall. *Computers in Biology and Medicine 173* (May 2024), 108328. URL: https://linkinghub.elsevier.com/retrieve/pii/S0010482524004128, doi:10.1016/j.compbiomed.2024.108328. 1

[SKVG20] Semih Kayhan O., Van Gemert J. C.: On Translation Invariance in CNNs: Convolutional Layers Can Exploit Absolute Spatial Location. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (Seattle, WA, USA, June 2020), IEEE, pp. 14262–14273. URL: https://ieeexplore.ieee.org/document/9156444/, doi:10.1109/CVPR42600.2020.01428. 2

[SOG09] Sun J., Ovsjanikov M., Guibas L.: A Concise and Provably Informative Multi-Scale Signature Based on Heat Diffusion. In *Proceedings of the Symposium on Geometry Processing* (Goslar, DEU, 2009), SGP '09, Eurographics Association, pp. 1383–1392. 3

[Sol15] Solomon J.: *Numerical algorithms: methods for computer vision, machine learning, and graphics*. CRC press, 2015. 2

[TSK*18] Thomas N., Smidt T., Kearnes S., Yang L., Li L., Kohlhoff K., Riley P.: Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds. *CoRR abs/1802.0* (2018). arXiv: 1802.08219. URL: http://arxiv.org/abs/1802.08219. 3

[TSR*23] Thomas O. O., Shen H., Raaum R. L., Harcourt-Smith W. E. H., Polk J. D., Hasegawa-Johnson M.: Automated morphological phenotyping using learned shape descriptors and functional maps: A novel approach to geometric morphometrics. *PLOS Computational Biology 19*, 1 (Jan. 2023), e1009061. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1009061, doi:10.1371/journal.pcbi.1009061. 1

[VCC*18] Veličković P., Cucurull G., Casanova A., Romero A., Liò P., Bengio Y.: Graph Attention Networks. *International Conference on Learning Representations* (2018). 3

[VSP*17] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser Ł., Polosukhin I.: Attention is all you need. *Advances in neural information processing systems 30* (2017). 1, 2

[WJW*24] Wu X., Jiang L., Wang P.-S., Liu Z., Liu X., Qiao Y., Ouyang W., He T., Zhao H.: Point Transformer V3: Simpler, Faster, Stronger. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2024), pp. 4840–4851. URL: https://ieeexplore.ieee.org/document/10658198/, doi:10.1109/CVPR52733.2024.00463. 3

[WNEH22] WIERSMA R., NASIKUN A., EISEMANN E., HILDEBRANDT K.: DeltaConv: anisotropic operators for geometric deep learning on point clouds. *ACM Transactions on Graphics 41*, 4 (July 2022), 105:1–105:10. URL: https://dl.acm.org/doi/10.1145/3528223.3530166, doi:10.1145/3528223.3530166. 1, 3

[WSK*15] WU Z., SONG S., KHOSLA A., YU F., ZHANG L., TANG X., XIAO J.: 3D ShapeNets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2015), vol. 07-12-June, pp. 1912–1920. arXiv: 1406.5670. doi:10.1109/CVPR.2015.7298801. 1, 2

[WSL*18] WANG Y., SUN Y., LIU Z., SARMA S. E., BRONSTEIN M. M., SOLOMON J. M.: Dynamic Graph CNN for Learning on Point Clouds. *CoRR abs/1801.07829* (2018). _eprint: 1801.07829. URL: http://arxiv.org/abs/1801.07829. 3

[ZJ16] ZHOU Q., JACOBSON A.: Thingi10K: A Dataset of 10,000 3D-Printing Models, 2016. _eprint: 1605.04797. URL: https://arxiv.org/abs/1605.04797. 1

[ZJJ*] ZHAO H., JIANG L., JIA J., TORR P. H. S., KOLTUN V.: Point Transformer. 3