



# Geometric Learning

Academy of Mathematics and Systems Sciences, CAS

May 18, 2023

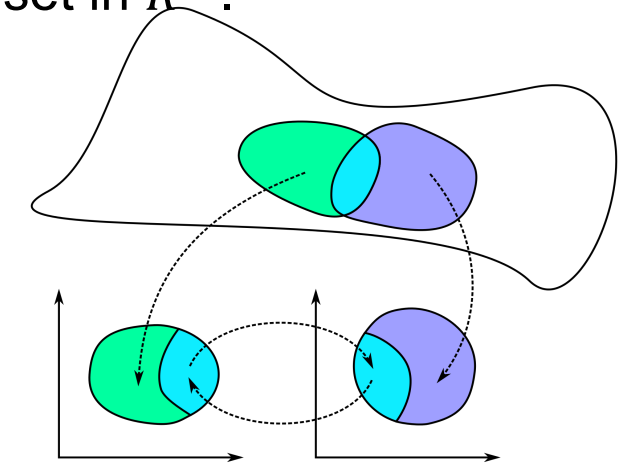
# Contents

# Differentiable Manifold

## Differentiable Manifold

$M$ is a differentiable manifold with dimension $m$ if:

- $M$ is a Hausdorff and second countable topological space.
- $M$ has a differentiable atlas $\{\phi_i\}$. Image of $\phi_i$ is an open set in $R^m$.



Atlas

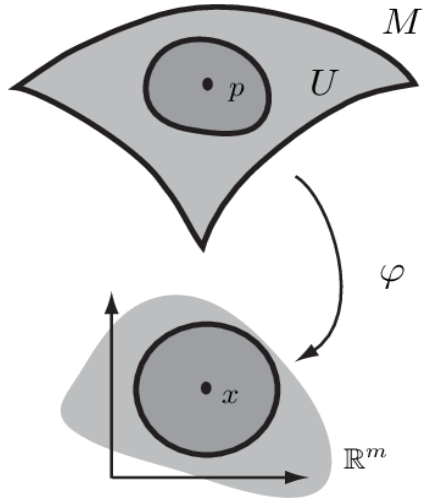
Embedding

$n \leq 2m + 1$

## Submanifold of Euclidian Space

$M$ is a submanifold of $R^n$ with dimension $m$ if:

- $M$ is a subset of $R^n$
- $\forall x \in M, \exists\, x \in B, s.t. B \cap M$ and $U$ and diffeomorphic, where $B$ is a neighbor of $x \in R^n$ and $U$ is an open set in $R^m$.

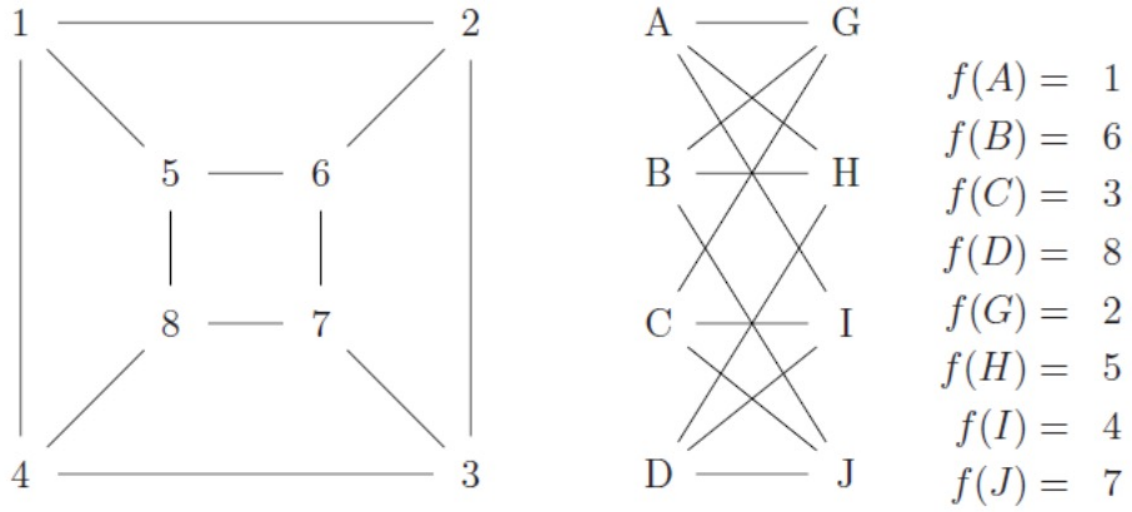



Locally Euclidian

# Potential Structures on Manifold

- Metric space and isomorphism.
  - $(M, d)$ is a set $M$ equipped with distance function $d$.
  - $(M_1, d_1), (M_2, d_2)$ are isomorphic if there exists an invertible map $T: M_1 \to M_2$ satisfy

$$d_1(x, y) = d_2(T(x), T(y))$$




A manifold $M$ can be a metric space.
For example, Riemannian manifold.

# Potential Structures on Manifold

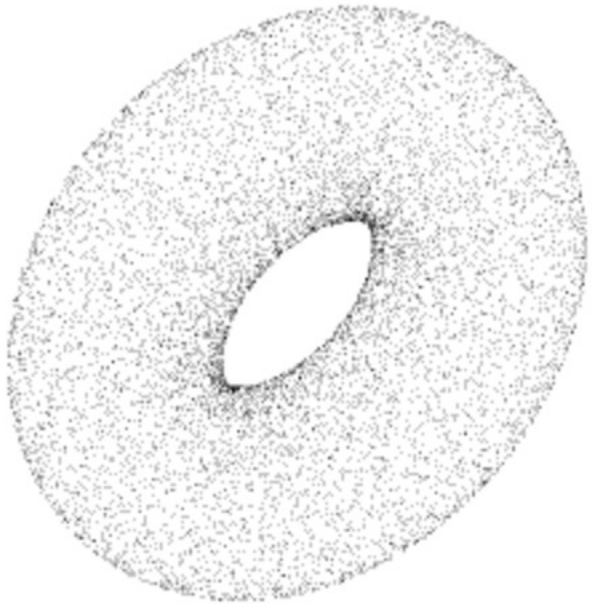
- A measure space is a triple $(X, \mathfrak{F}, \mu)$, where
  - $X$ is a set
  - $\mathfrak{F}$ is a $\sigma$-algebra on $X$
  - $\mu$ is a measure on $(X, \mathfrak{F})$

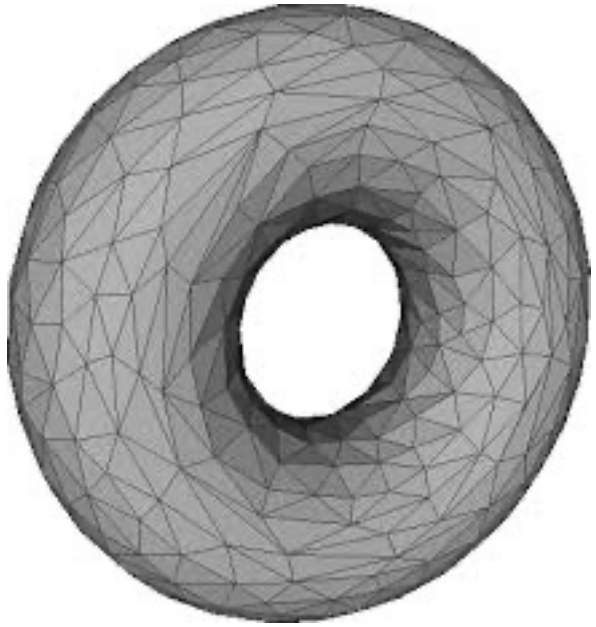
- A metric measure space is a tripe $(X, d_X, \mu_X)$ where
  - $(X, d_X)$ is a compact metric space
  - $\mu_X$ is a Borel probability measure on $X$ and $\mu_X$ has full support:

$$supp[\mu_X] = X$$

A manifold $M$ can be a measure space,
even a metric measure space.

# 3D Geometric Data





Point clouds: points only

Mesh: points + edges + faces

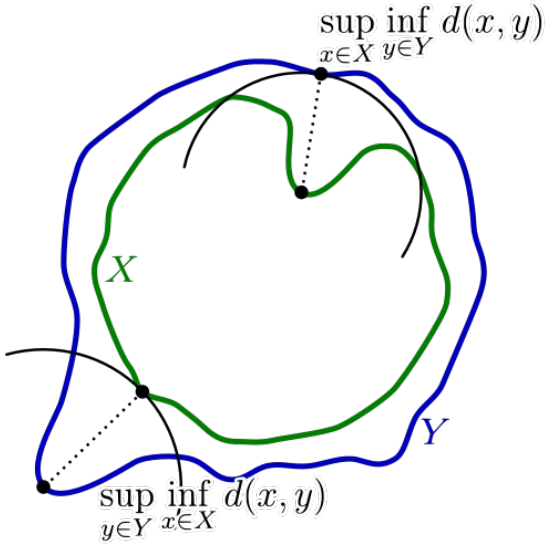
How to recognize an object from geometric data?
How to match the shapes of objects?

# Object Matching

- Comparing objects by Hausdorff distance:
  - Assume $A, B$ are compact subsets of metric space $(Z, d)$. The Hausdorff distance $d_{\mathcal{H}}$ between $A, B$ is defined by

$$d_{\mathcal{H}}^{Z}(A,B) := \max\left(\sup_{a\in A}\inf_{b\in B} d(a,b), \sup_{b\in B}\inf_{a\in A} d(a,b)\right)$$

- Comparing objects by intrinsic distance:
  - Assume $(A = \{a_i\}, d_A), (B = \{b_j\}, d_B)$ are metric space. Then $d_A$ and $d_B$ correspond to a matrix.
  - Measure the difference by measuring the difference of these matrices.



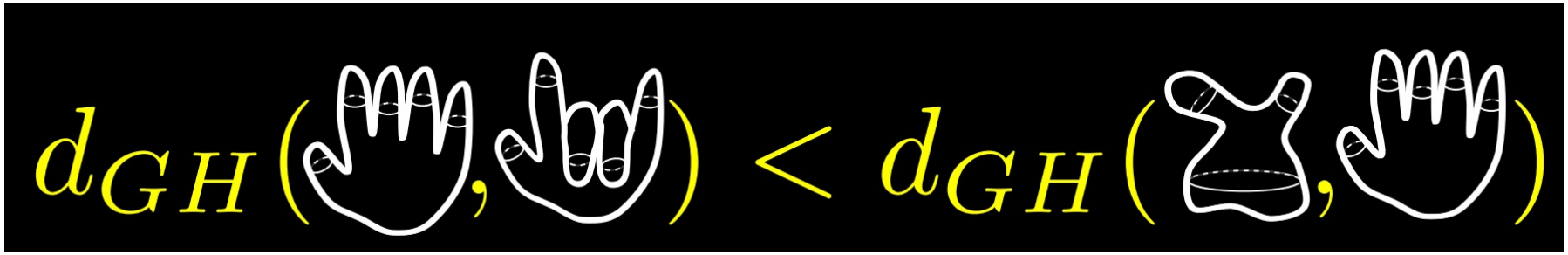

Hausdorff distance

# Object Matching

- The Gromov-Haudsorff distance combines these ideas. Suppose $X,\ Y$ are two compact metric spaces, then the Gromov-Hausdorff distance is defined as:

$$d_{\mathcal{GH}}(X,Y) \coloneqq \inf_{Z,f,g} d_{\mathcal{H}}^{Z}(f(X),g(Y))$$

where $f,\ g$ are isometric embeddings into the metric space Z.




- Surprisingly, $d_{\mathcal{GH}}(X,Y)$ is a metric of all compact metric spaces quotient isometry equivalence.

# Object Matching

- A correspondence *R* of $X$ and $Y$ is a subset of $X \times Y$ s.t.
  - $\forall x \in X, \exists y \in Y \; s.t. (x, y) \in R$
  - $\forall y \in Y \; \exists x \in X \; s.t. (x, y) \in R$

- The following equation gives the way to calculate $d_{\mathcal{GH}}$:

$$d_{\mathcal{GH}}(X, Y) = \frac{1}{2} \inf_{R \in \mathcal{R}(X,Y)} \sup_{\substack{x_1, x_2 \in X \\ y_1, y_2 \in Y \\ s.t.(x_i, y_i) \in R}} \Gamma_{X,Y}(x_1, y_1, x_2, y_2)$$

where $\mathcal{R}(X, Y)$ is the collection of correspondence between $X$ and $Y$, $\Gamma_{X,Y}(x, y, x', y') = |d_X(x, x') - d_Y(y, y')|$.

# Object Matching

- However, finding the best correspondence is NP-hard!

- There are several ways to solve the problem:
  - Solve landmark correspondence
  - MDS+PCA [Bronstein et al. (2006)]
  - Functional maps [Sun et al. (2009)]

# Functional Maps [Ovsjanikov et al. (2012)]

- A mapping $f: M \to N$ induces a linear mapping between function space:

$$f_{\#\#}: \{g: N \to R\} \to \{h: M \to R\}$$

$$g \to g \circ f$$

  - Choose a fifnite-dimensional space of $L_2(M)$ and $L_2(N)$ and a basis in them, the $f_{\#\#}$ corresponds to a matrix C.

- Consider eigen functions of Laplacian over a compact Riemannian manifold $M$:

$$\Delta_M u = \lambda u$$

  If we choose the basis as the first *k* eigen functions of $\Delta_M$ and $\Delta_N$ (denote the $i$-th eigen function as $\phi_i$, $\psi_i$), then we can recover the most likely mapping $f$ from its matrix C.

# Functional Maps [Sun et al. (2009)]

- These functions are fingerprints of this Riemannian manifold. To be specific,
  - $T: M \to N$ is isomorphic between Riemannian manifold $M,\ N$

  $\Leftrightarrow$ {eigen values of $\Delta_M$ }= {eigen values of $\Delta_N$ } and $\phi_i = \psi_i \circ T$
- By heat equation, we have that:

$$\delta_x(\cdot) = \lim_{t \to 0} \sum^{\infty} e^{-\lambda t} \phi_i(x) \phi_i(\cdot)$$

- As a result, $\langle f_{\#\#}(\delta_{(\cdot)}), \delta_{(\cdot)} \rangle \approx |\Psi C \Phi^T|, |\Psi C \Phi^T|$^ gives the probability of image of $f(x)$, where

$$\Phi = [\phi_i(x_j)]_{\mathrm{ji}}, \qquad \Psi = [\psi_i(y_j)]_{\mathrm{ji}}$$

  and "^" means column-wise normalization, $\{x_j\}$ is the data set observed on the manifold.

# High-dimensional Geometric Data

High-dimensional data lie along Low-dimensional manifolds

Dimension: $28 \times 28$

Dimension: 2

# High-dimensional Geometric Data

High-dimensional data lie along Low-dimensional manifolds

Dimension: $178 \times 218 \times 3$

Dimension: 100

How to lower the dimension of data in the meanwhile preserve the structure of manifolds?

# Manifold Learning

- From high-dimensional data to lower-dimensional representation:

$$\{x_i \in R^n\} \rightarrow \{y_i \in R^m\},\ m \ll n$$

where $x_i$ corresponds to $y_i$. In the meanwhile, the structure of the manifold should be preserved.

- MDS: preserve extrinsic metric

$$\min \sum_{i,j} \left\| d_{R^n}(x_i, x_j) - d_{R^m}(y_i, y_j) \right\|_2^2$$

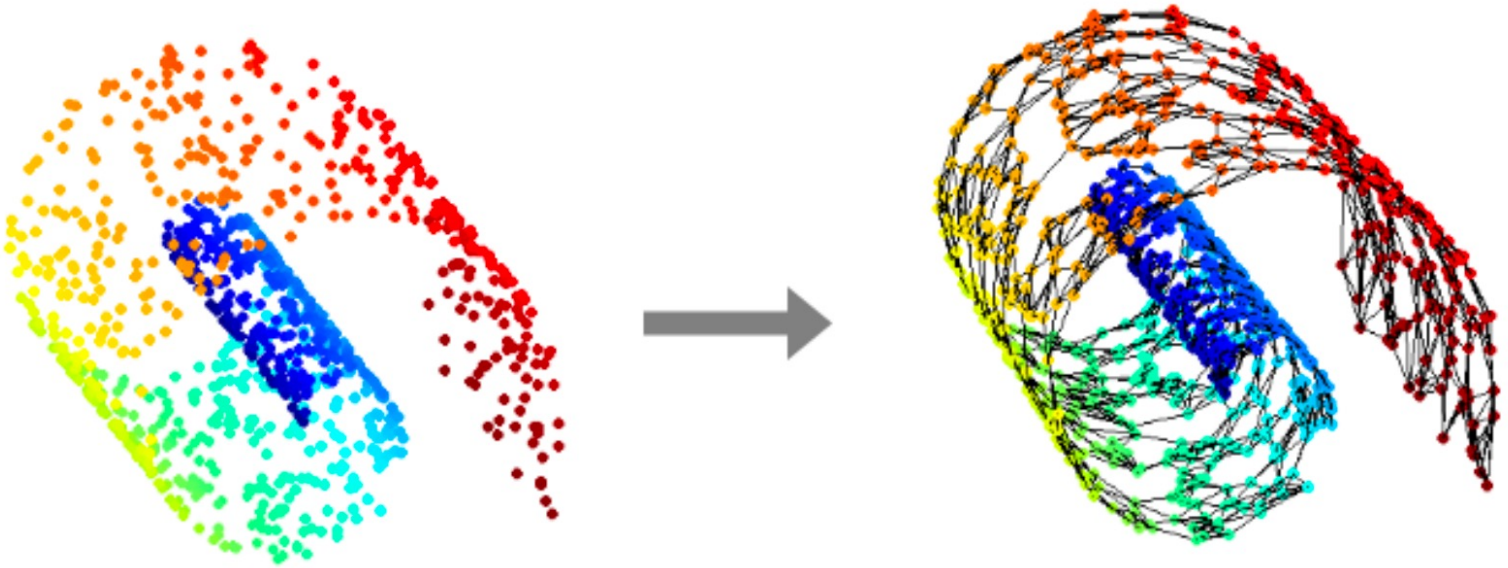
- Isomap: preserve intrinsic metric [Tenenbaum et al. (2000)]

$$\min \sum_{i,j} \left\| d_M(x_i, x_j) - d_{R^m}(y_i, y_j) \right\|_2^2$$

# Manifold Learning

- Isomap: preserve intrinsic metric

$$\min \sum_{i,j} \left\| d_M(x_i, x_j) - d_{R^m}(y_i, y_j) \right\|_2^2$$



Construct graph structure for geodesic length

# Contents

- Deep Learning for 3D Shapes
  - Learning representation of points clouds
    - Recognization
    - Segmentation
  - Learning developable approximation
  - Learning for shape matching

- Deep Learning for High-dimensional Geometry
  - Wasserstein Autoencoders
  - Gromov-Wasserstein Autoencoders

# Learning Representation of Point Clouds

- Distinguishing different objects by traditional methods needs an estimation of metric, which makes it time-consuming.
- Traditional segmentation methods have been developed for certain usages. No general pipeline for segmentation.
- Now that there are many available datasets of point clouds and their classification (such as “chairs”, “cups” and so on) and semantic (or other standards) segmentation (e.g. chair leg, cup handle).
- Deep learning can be used to recognize objects and segmentation.

# Learning Representation of Point Clouds

- Loss function for 3D object classification: the softmax classification loss + a regularizer.
- Let's denote the dataset by $\{(x_i, y_i)\}$, where $x_i$ is a point cloud and $y_i$ is the label.

$$L = \sum_i \log \frac{e^{y_i}}{\sum_j e^{f_\theta(x)_j}}$$

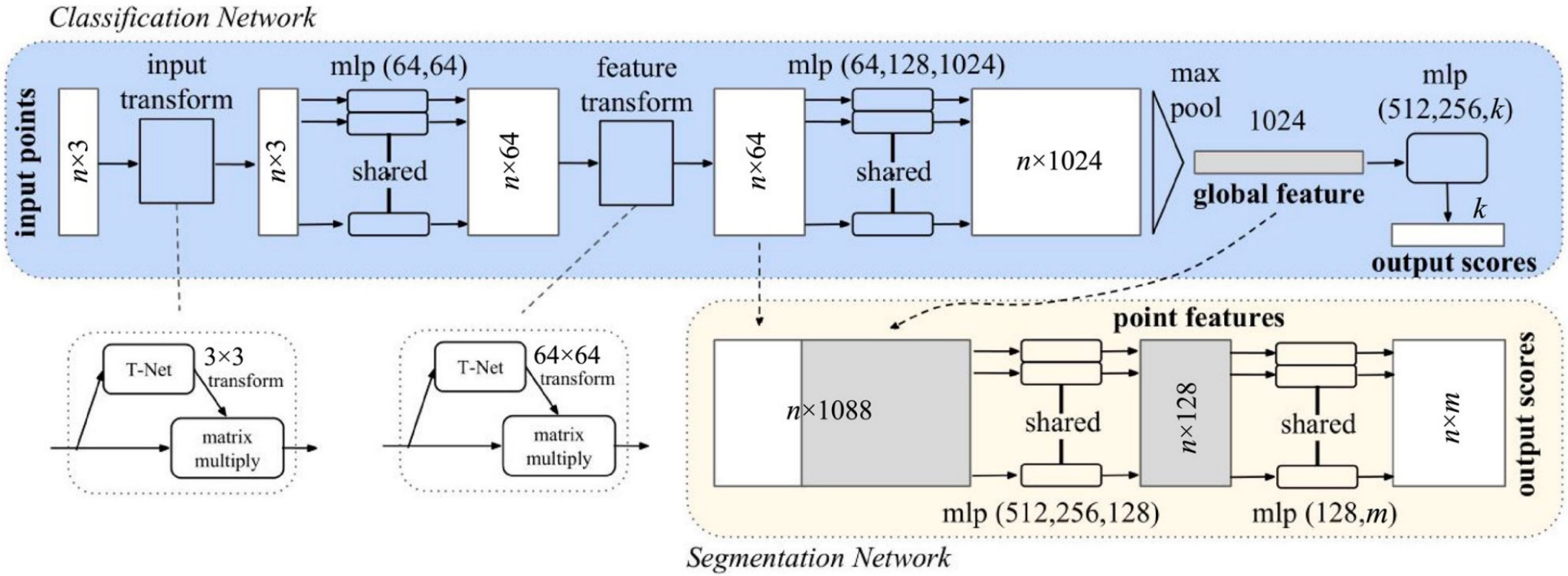
- Loss function for 3D object segmentation: the Mean Intersection over Union (MIoU) loss.
- Let's denote the probability of a point of class i be classified into class j by $p_{ij}(\theta)$. Then the MIoU loss is defined as:

$$\text{MIoU} = \frac{1}{k}\sum_{i=1}^{k} \frac{p_{ii}(\theta)}{\sum_j p_{ij}(\theta) + \sum_j p_{ji}(\theta) + p_{ii}(\theta)}$$

which can be interpreted as the mean of $\frac{\text{True Positive}}{\text{False Negativ+False Positive+True Positive}}$

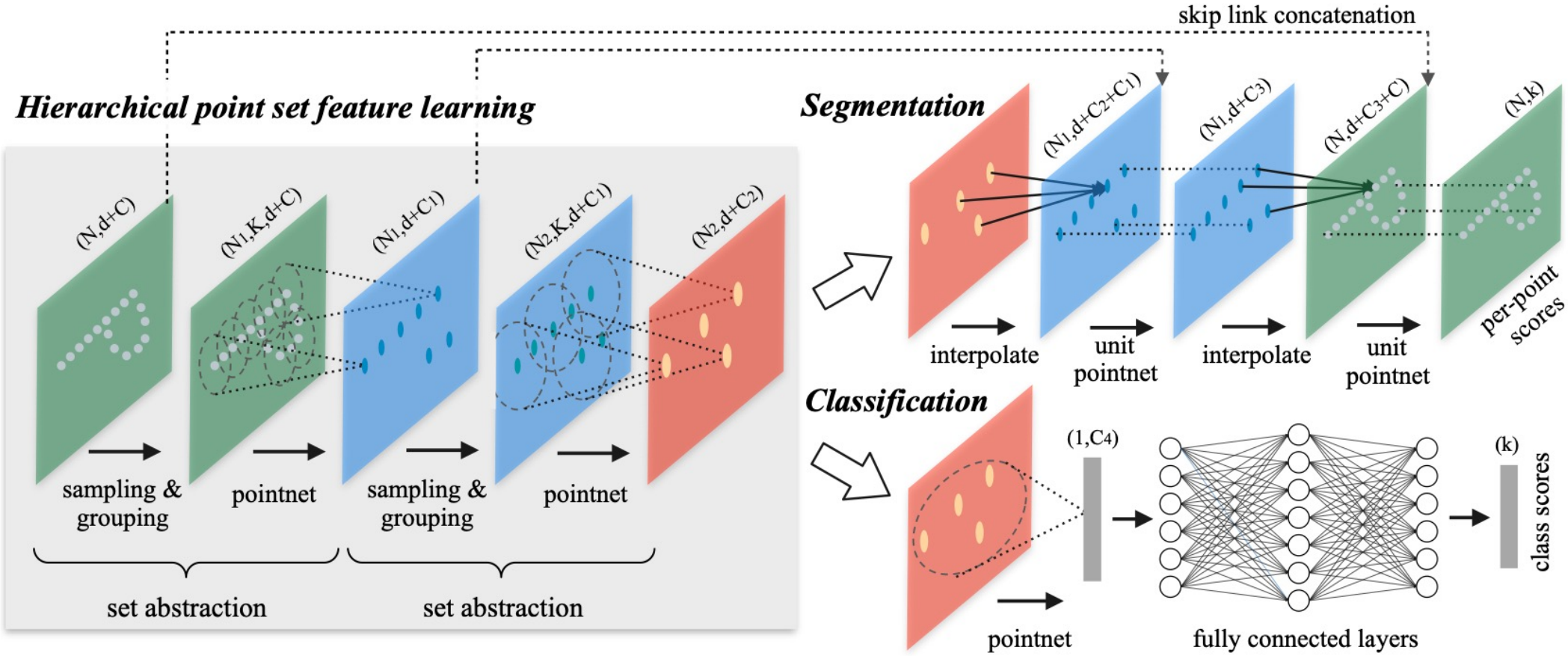
# Learning Representation of Point Clouds: PointNet

- PointNet: learn global features and the point features of the point cloud.
- The global feature can be used to do 3D object classification while the point features can be used to do segmentation [Qi et al. (2017)].



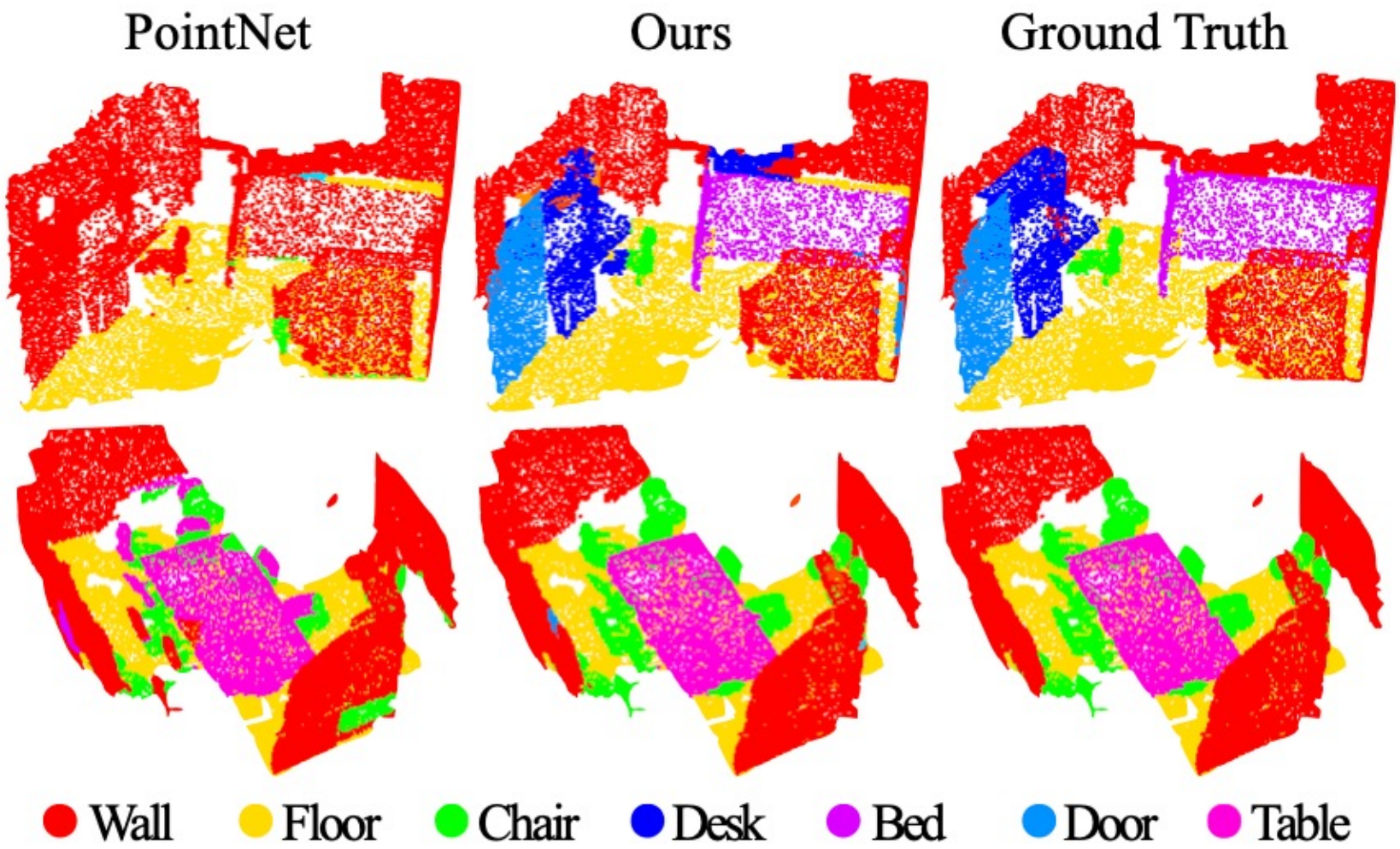

PointNet

# Learning Representation of Point Clouds: PointNet++




PointNet++ adopts a hierarchical structure to further capture local features [Qi et al. (2017)]

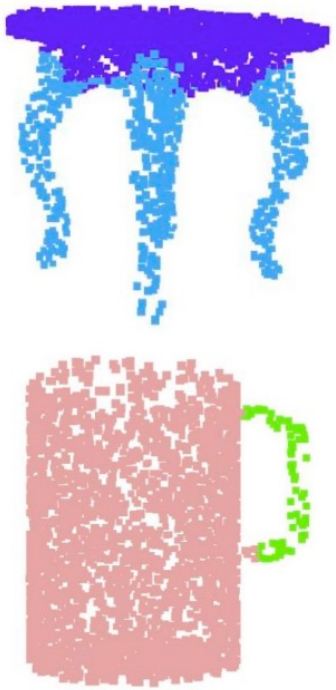
# PointNet vs PointNet++




PointNet vs PointNet++

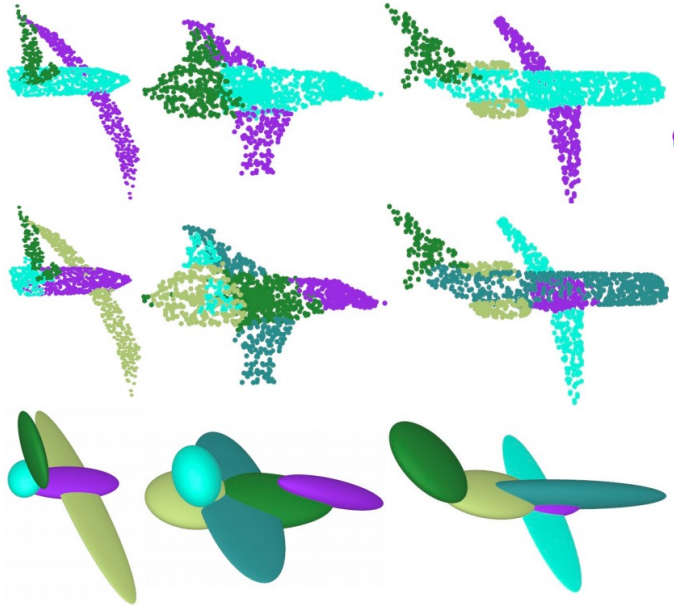
# Other Segmentation

## Fitting by semantics

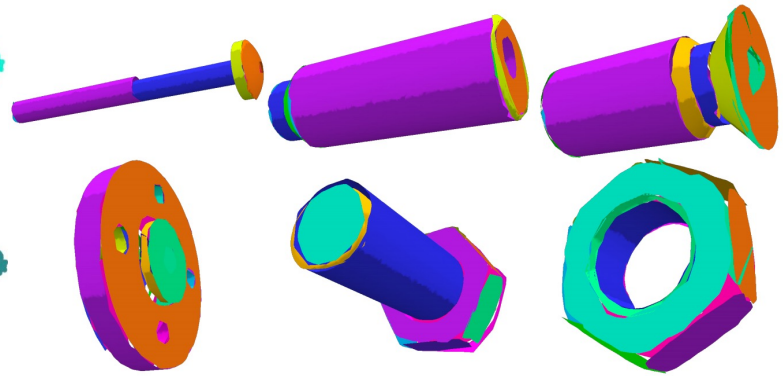


PointNet

## Fitting by primitives



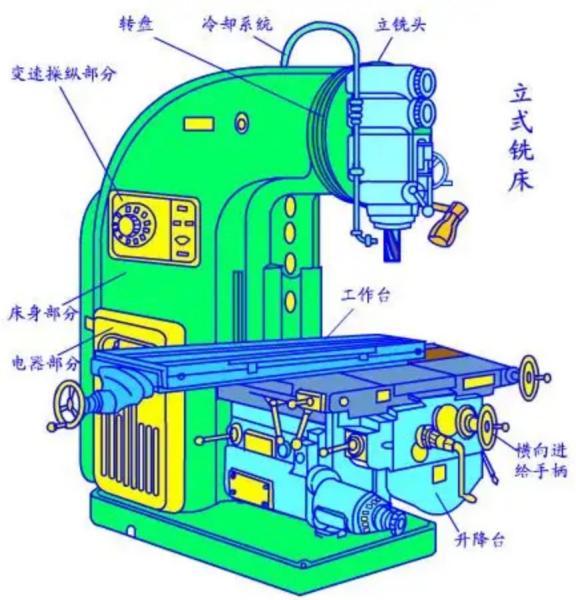
PRIFIT: fit by ellipsoids
Sharma et al. (2022)



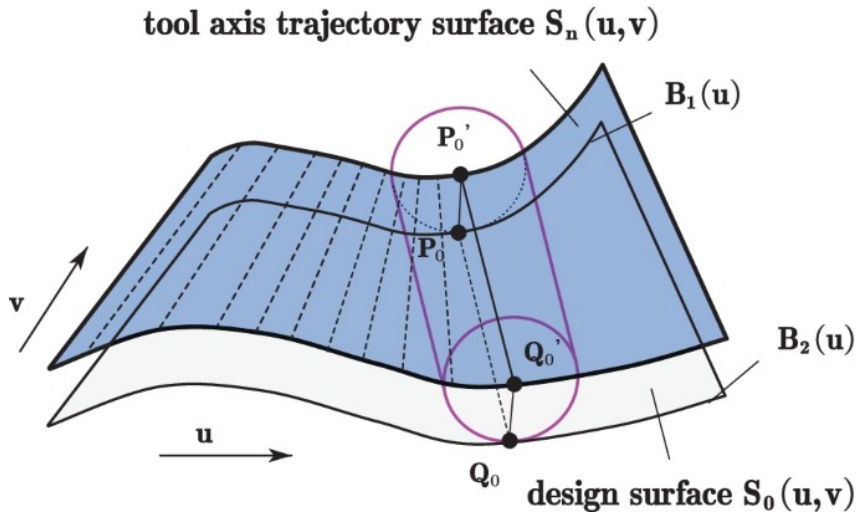
SPFN: fit by planes, cylinders……
Li et al. (2019)

**Different datasets, additional loss terms for specific tasks, and similar structure (based on PointNet++)**

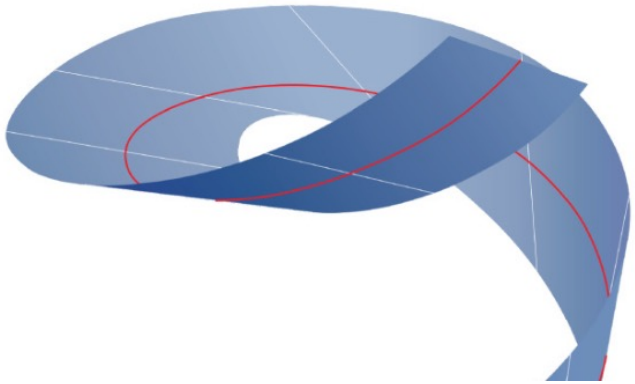
# Learning Developable Approximation




数控铣床




铣刀切割



可展曲面

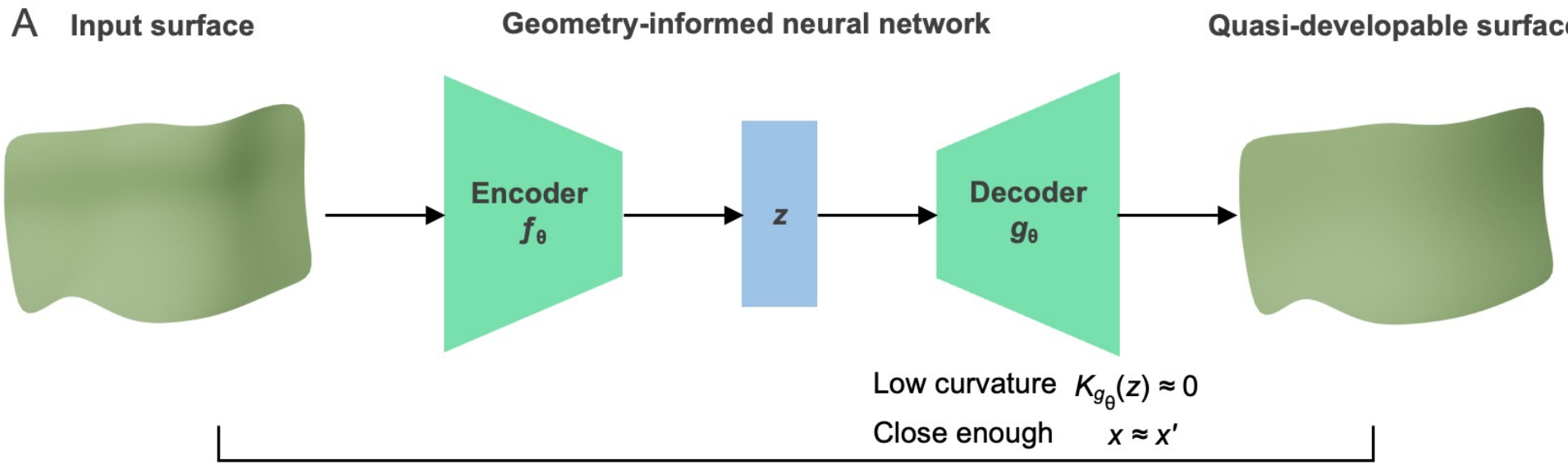
铣刀切割⇔可展曲面

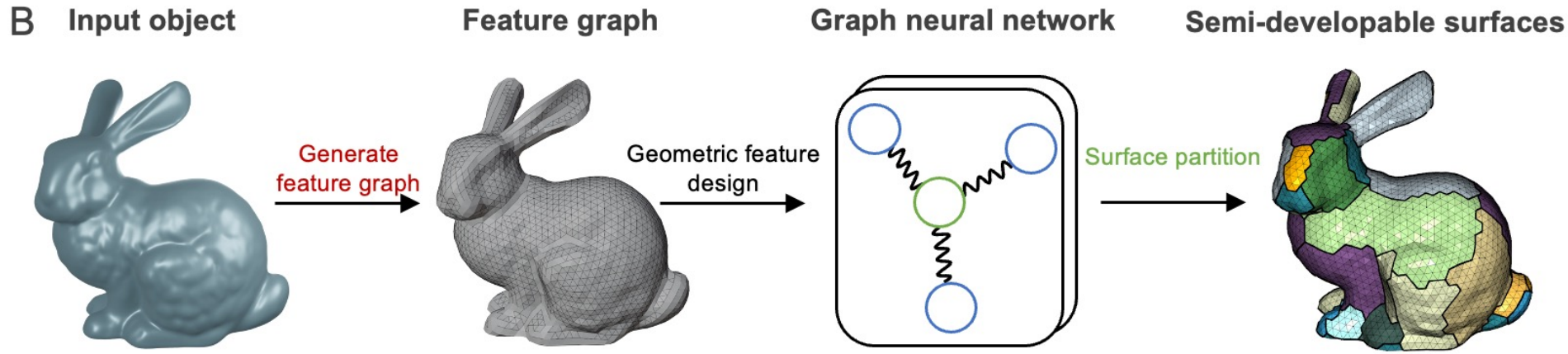
# Developable Approximation

- Developable approximation: transform a complex surface to a set of quasi-developable surfaces.

Mitani & Suzuki  Shatz et. al.  Stein et. al.  Binninger et. al.  Zhao & Fang

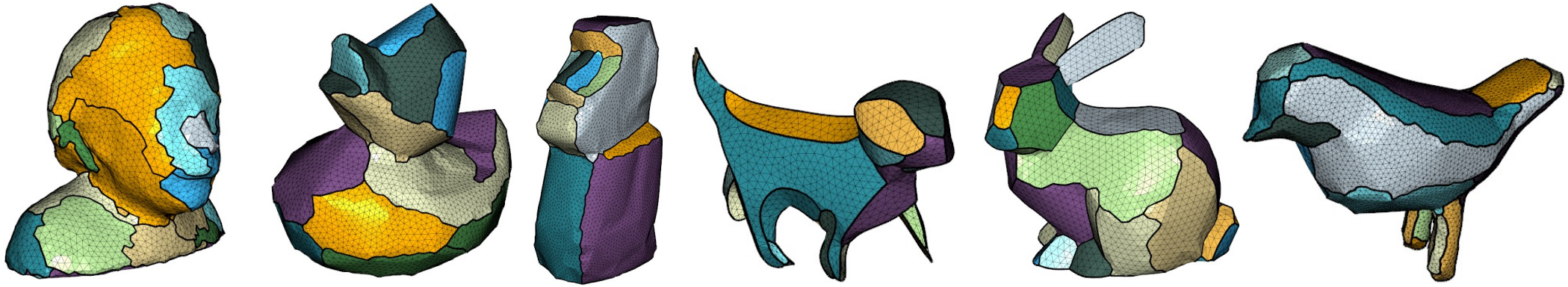
# Learning Developable Approximation



Segmentation

Segmentation + Lower curvature



# Learning for Shape Matching

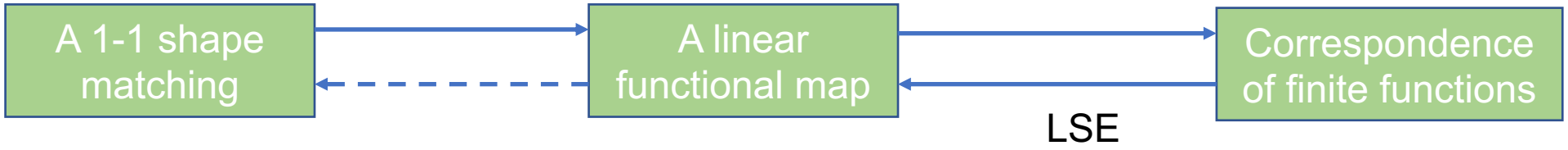
- Recall that $d_{\mathcal{GH}}(X,Y) = \frac{1}{2}\inf_{R\in\mathcal{R}(X,Y)} \sup_{\substack{x_1,x_2\in X\\ y_1,y_2\in Y\\ s.t.(x_i,y_i)\in R}} \Gamma_{X,Y}(x_1,y_1,x_2,y_2)$. To calculate the $d_{\mathcal{GH}}$ is to find the best correspondence, which is NP-hard.

- Note that we have a lot of data of the best correspondence.

- It is natural to use supervised learning to learn the correspondence then accelerate the solving process [Boscaini et al. (2016)].

# Learning for Shape Matching

- Functional map: a linear map $L: \{g: N \to R\} \to \{h: M \to R\}$ can give a way of shape matching, which can be estimated by LSE if we have already known that $L(f_i) = g_i$.




- Consequently, learning a shape matching correspondence is to learn a functional correspondence:

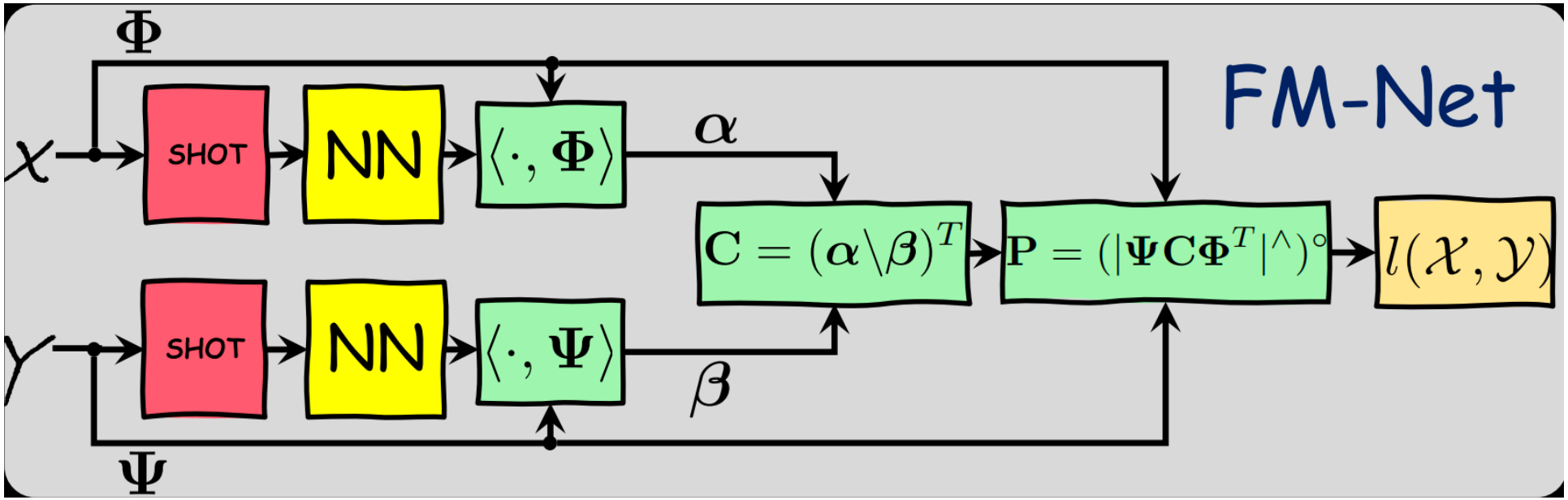
$$L(f_{i,\theta}) = g_{i,\theta}$$

- Luckly, NN has strong ability to represent functions.

# Learning for Shape Matching [Litany et al. (2017)]

- Dataset: $\{(X, Y, \pi^*)\}$ where $X, Y$ are meshes and $\forall x \in V(X), \pi^*(x)$ gives the best correspondent point of $x$.
- Loss function:

$$l_{sup}(X, Y) = \sum_{i \in V(X)} \sum_{j \in V(Y)} p_{ij,\theta} d_y^2(j, \pi^*(i))$$



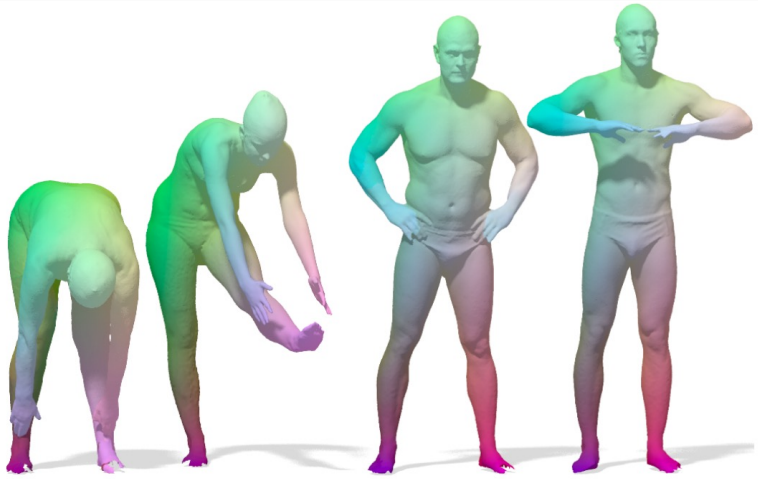
# Learning for Shape Matching [Halimi et al. (2019)]

- Dataset: $\{(X, Y, \pi^*)\}$ where $X, Y$ are meshes and $\forall x \in V(X), \pi^*(x)$ gives the best correspondent point of $x$.
- Loss function:

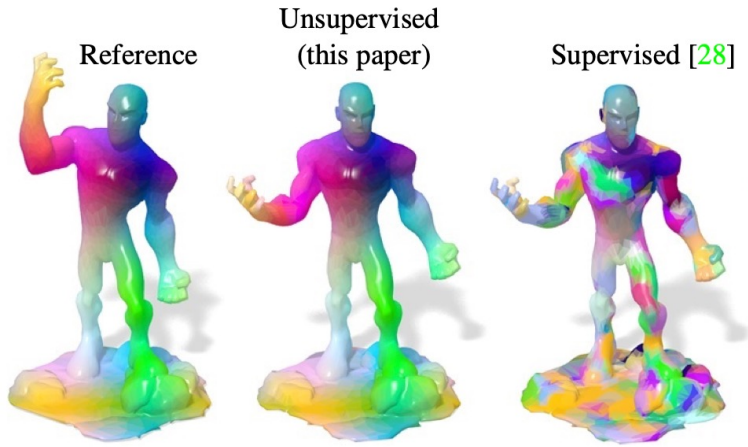
$$l_{uns}(X, Y) = \|D_X - P D_Y P^T\|$$

where $D_X$, $D_Y$ are distance matrices of $X$ and $Y$.

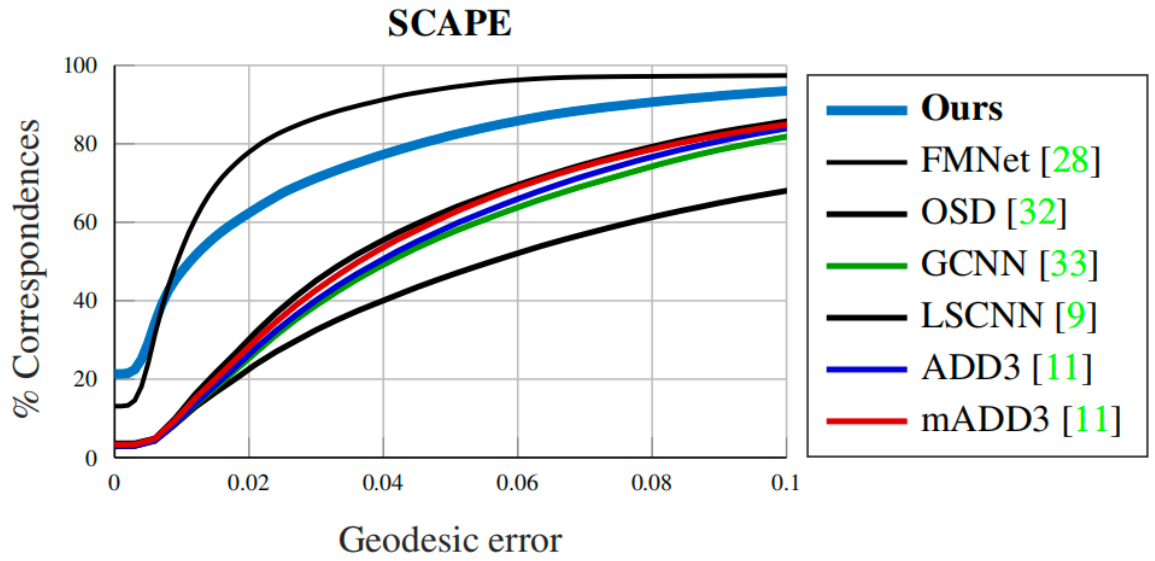
# Learning for Shape Matching



Results of FMNet



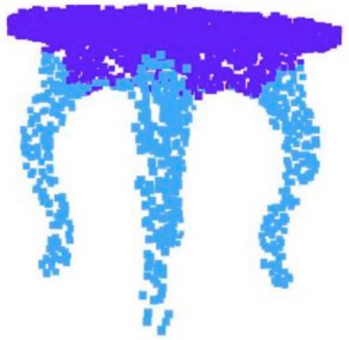

Unsupervised vs Supervised on single



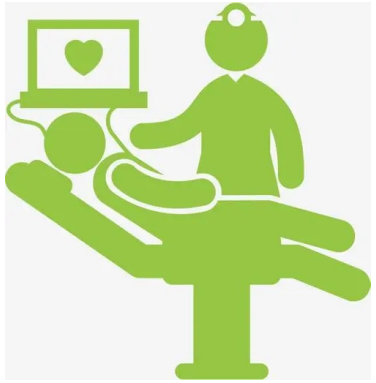

Unsupervised vs Supervised

# Wasserstein Metric and Gromov-Wasserstein metric

## Data of different type



Evenly distributed,
low-dimensional data



Unevenly distributed,
high-dimensional data

## Modeling and metric

- Metric space
  - Hausdorff distance
  - Gromov-Hausdorff metric

- Manifold assumption
- Metric measure space
  - Wasserstein metric
  - Gromov-Wasserstein metric

# Wasserstein Metric

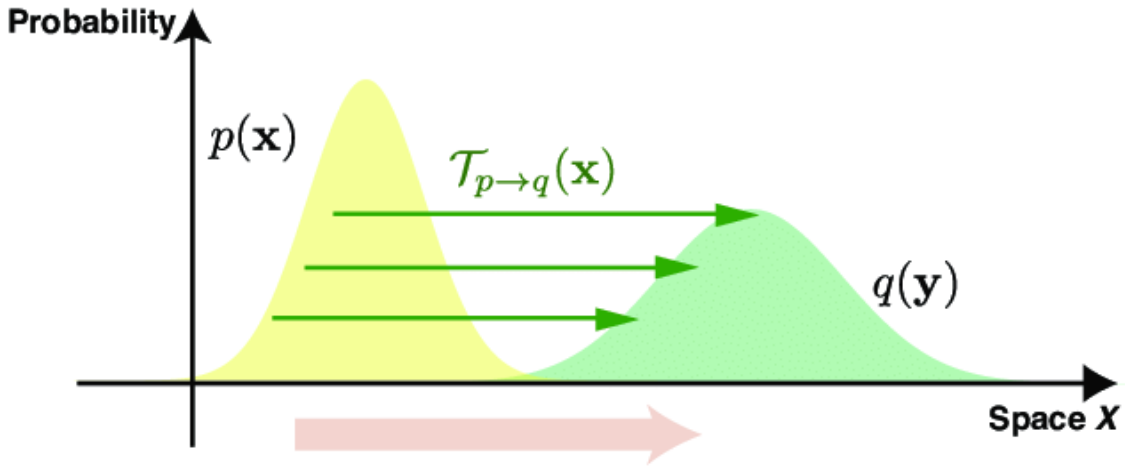
- Given two probability measure $\mu_A, \mu_B$ with compact support $A_0,\ B_0$ in $R^n$, the $L^k$ Wasserstein distance is defined as

$$\mathcal{W}_k(\mu_A, \mu_B)^k := \inf_{\mu \in \mathcal{M}(\mu_A, \mu_B)} \int \|x - y\|^k d\mu(x, y)$$

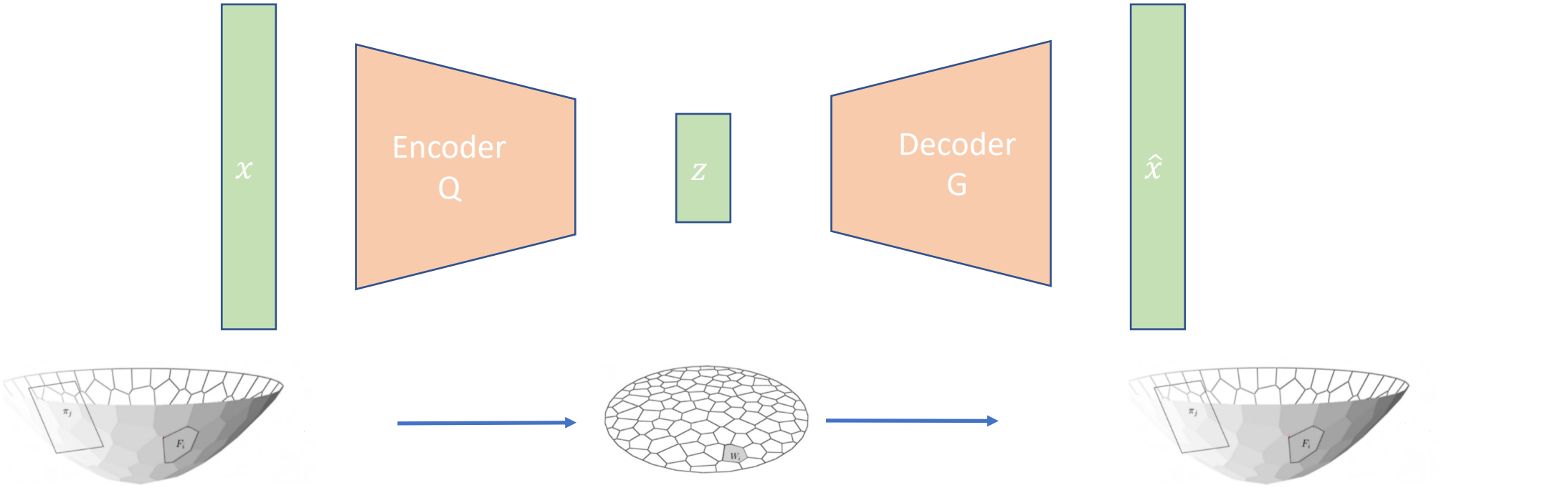
$\mathcal{M}(\mu_A, \mu_B)$ is the set of coupling of $\mu_A$ and $\mu_B$:

$$\mathcal{M}(\mu_A, \mu_B) := \{\text{measure } \mu \text{ on } A_0 \times B_0, s.t. \mu(A_0 \times B) = \mu_B(B), \mu(A \times B_0) = \mu_A(A)\}$$

- Wasserstein metric is also known as mass transportation distance.



# Wasserstein Autoencoder [Tolstikhin et al. (2017)]




Input data distribution

Prior distribution

Reconstructed distribution

$(X, P_X) \longrightarrow (\hat{Z}, Q_Z)$

$Q_Z(z) = \int p_Q(z|x) dP_X(x)$

$(Z, P_Z) \longrightarrow (\hat{X}, P_G)$

$P_G(x) = \int p_G(x|z) dP_Z(z)$

Learn $(X, P_X)$ by minimizing $\mathcal{W}(P_X, P_G)$

# Wasserstein Autoencoder

For $P_G$ as defined above with deterministic $G$ and any function $G: Z \to X$

$$\inf_{\gamma \in \mathcal{M}(P_X, P_G)} E_{\gamma(X,Y)}[c(x,y)] = \inf_{\mathrm{Q}:\mathrm{Q}_Z=\mathrm{P}_Z} E_{P_x} E_{Q(Z|X)}[c(X, G(z))]$$

- Loss function can be written as:

$$L = E_{P_x} E_{Q(Z|X)}[c(X, G(z))] + \lambda D_Z(Q_Z, P_Z)$$

where $D_z(\cdot,\cdot)$ is an arbitrary divergence. For example:

- JS divergence

$$\mathrm{JS}(P,Q) = \frac{1}{2}\mathrm{KL}\left(P, \frac{P+Q}{2}\right) + \frac{1}{2}\mathrm{KL}(Q, \frac{P+Q}{2})$$

- Maximum mean discrepancy (MMD)

$$\mathrm{MMD}[\mathcal{F}, P, Q] \coloneqq \sup_{f \in \mathcal{F}} \left(E_P[f(x)] - E_Q[f(y)]\right)$$

When $\mathcal{F}$ is an RKHS of a positive-definite reproducing kernel $k: Z \times Z \to R$,

$$\mathrm{MMD}_k(P,Q) = \left\| \int_Z k(z,\cdot) dP(z) - k(z,\cdot) dQ(z) \right\|_{\mathcal{H}_k}$$

where $\mathcal{H}_k$ is the RKHS of real-valued functions mapping $Z$ to $R$.

# Wasserstein Autoencoder

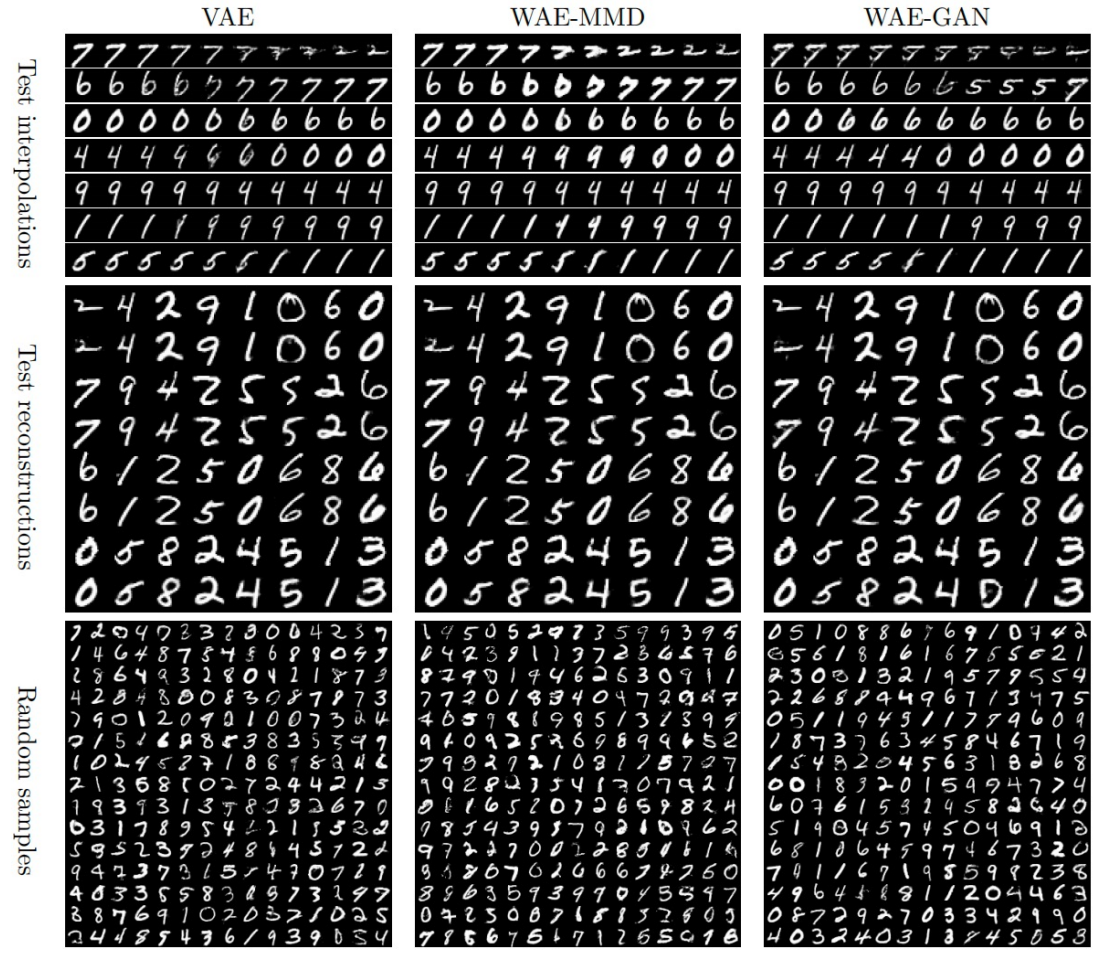



Figure 2: VAE (left column), WAE-MMD (middle column), and WAE-GAN (right column) trained on MNIST dataset. In "test reconstructions" odd rows correspond to the real test points.


Figure 3: VAE (left column), WAE-MMD (middle column), and WAE-GAN (right column) trained on CelebA dataset. In "test reconstructions" odd rows correspond to the real test points.

**Good reconstruction and generation**

# Gromov-Wasserstein metric [Mémoli (2011)]

- Wasserstein metric can be regarded as a probabilistic relaxation of the Hausdorff metric:

$$d_{\mathcal{H}}^{Z}(A,B) \coloneqq \max\left(\sup_{a\in A}\inf_{b\in B} d(a,b), \sup_{b\in B}\inf_{a\in A} d(a,b)\right)$$

$$= \inf_{R\in\mathcal{R}(A,B)} \sup_{(a,b)\in R} \|a-b\|$$

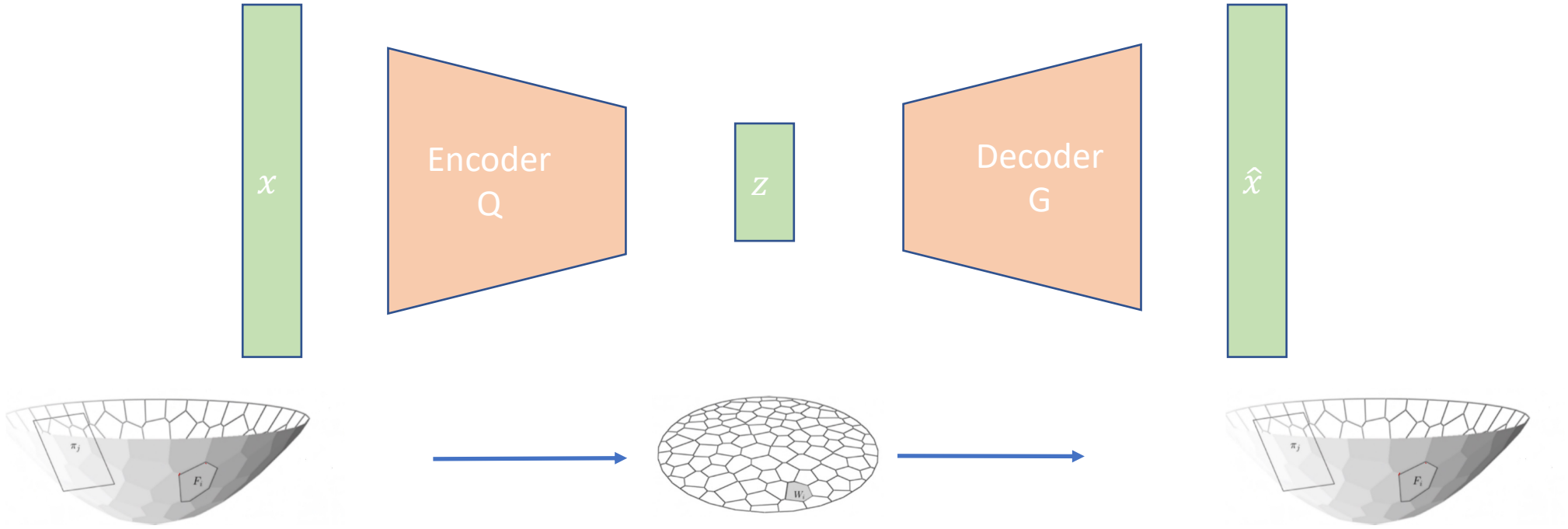
$$\mathcal{W}_{\infty}(P,Q) = \inf_{\mu\in\mathcal{M}(\mu_A,\mu_B)} \sup_{(x,y)\in \mathrm{supp}[\mu]} \|x-y\|$$

- Gromov-Wasserstein metric is a relaxation of the Gromov-Hausdorff metric:

$$d_{\mathcal{GH}}(X,Y) = \frac{1}{2}\inf_{R\in\mathcal{R}(X,Y)} \sup_{\substack{x_1,x_2\in X\\ y_1,y_2\in Y\\ s.t.(x_i,y_i)\in R}} \Gamma_{X,Y}(x_1,y_1,x_2,y_2)$$

$$d_{\mathcal{GW},p} = \frac{1}{2}\inf_{\mu\in\mathcal{M}(\mu_A,\mu_B)} \left(\int \left(\Gamma_{X,Y}(x_1,y_1,x_2,y_2)\right)^p d\mu\otimes d\mu\right)^{1/p}$$

# Gromov-Wasserstein Autoencoder [Gong et al. (2022)]




Input data distribution

Prior distribution

Reconstructed distribution

$$(X, d_X, P_X) \longrightarrow (\hat{Z}, d_Z, Q_Z) \qquad (Z, d_Z, P_Z) \longrightarrow (\hat{X}, d_X, P_G)$$

$$Q_Z(z) = \int p_Q(z|x) dP_X(x) \qquad P_G(x) = \int p_G(x|z) dP_Z(z)$$

Minimize $d_{\mathcal{GW},p}$ between original data $x$ and representation $z$

# Gromov-Wasserstein Autoencoders

- Denote the priori distribution by $\pi_\theta(z)$, decoder by $p_\theta(x|z)$ and joint distribution by $p_\theta(x,z) = \pi_\theta(z)p_\theta(x|z)$.
- The Gromov-Wasserstein loss can be written as

$$L_{\mathcal{GW}} := E_{p_\theta(x,z)}E_{p_\theta(x',z')}[\|d_X(x,x') - Cd_Z(z,z')\|^k]$$

where $C$ is a scaling parameter.

- To make training stable, three regularizers are added:
- Denote the data distribution by $p_{data}(x)$, encoder by $q_\phi(z|x)$ and joint distribution by $q_\phi(x,z) = p_{data}(x)q_\phi(z|x)$. Wasserstein regularizer can be written as

$$L_{\mathcal{W}} = E_{q_\phi(x,z)}E_{p_\theta(x'|z)}[d_X(x,x')]$$

# Gromov-Wasserstein Autoencoders

- To fulfill the coupling condition in the abovementioned metric, a sufficient regularizer is added as:

$$L_D \coloneqq E_{q_\phi(x,z)}\left[f_\psi(x,z)\right] - E_{p_\theta(x,z)}\left[f_\psi(x,z)\right]$$

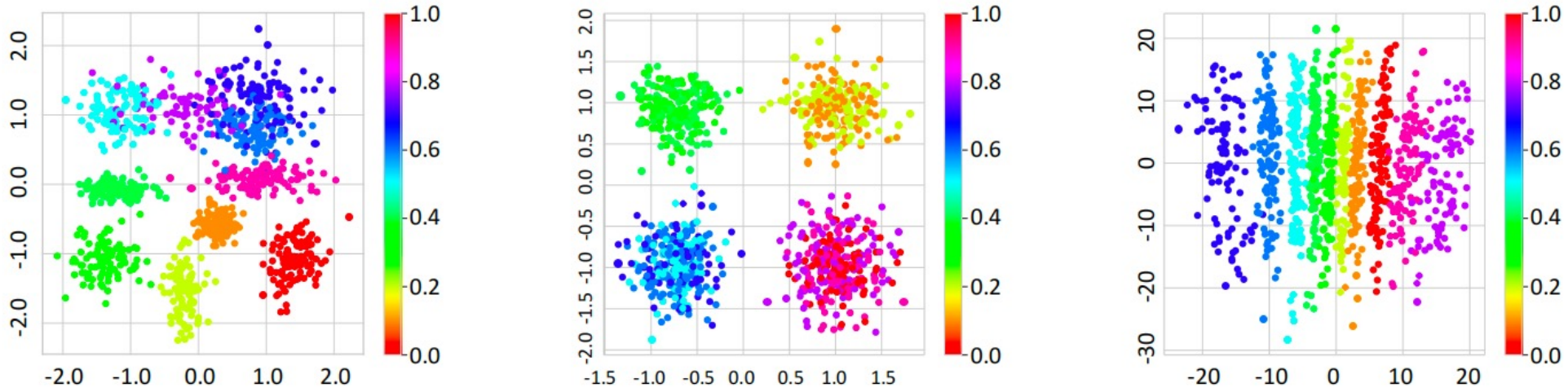
- The entropy regularization is used to avoid degenerate solutions

$$R_H \coloneqq H_q(z|x) = E_{q_\phi(x,z)}[-\log q_\phi(z|x)]$$

- Then the total object can be written as:

$$\min_{\theta,\phi} \max_{\psi} L \coloneqq L_{\mathcal{GW}} + \lambda_{\mathcal{W}} L_{\mathcal{W}} + \lambda_D L_D - \lambda_H R_H$$

# Gromov-Wasserstein Autoencoders



(a) VAE (Kingma & Welling, 2014).

(b) FactorVAE (Kim & Mnih, 2018) ($\gamma$=10).

(c) GWAE (FNP, $\lambda_W$=1, $\lambda_D$=10, $\lambda_{\mathcal{H}}$=0.3).

Figure 2: Comparison of the learned latent spaces in 3D Shapes (Burgess & Kim, 2018) and $L = 16$. The vertical and horizontal axes in the scatter plots respectively represent two of the 16 ($= L$) latent variables with the highest and the second-highest informativeness (Do & Tran, 2020) w.r.t. the *object hue* factor. Note that a single factor value varies along only one axis in a disentangled representation.

**Good representation and clustering**

# Reference


- 1. Boscaini, D., Masci, J., Rodolà, E., & Bronstein, M. (2016). Learning shape correspondence with anisotropic convolutional neural networks. *Advances in neural information processing systems*, *29*.
- 2. Tolstikhin, I., Bousquet, O., Gelly, S., & Schoelkopf, B. (2017). Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*.
- 3. Gong, F., Nie, Y., & Xu, H. (2022, October). Gromov-Wasserstein multi-modal alignment and clustering. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 603-613).
- 4. Mémoli, F. (2011). Gromov–Wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, *11*, 417-487.
- 5. Ovsjanikov, M., Ben-Chen, M., Solomon, J., Butscher, A., & Guibas, L. (2012). Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (ToG)*, *31*(4), 1-11.
- 6. Bronstein, A. M., Bronstein, M. M., & Kimmel, R. (2006). Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences*, *103*(5), 1168-1172.
- 7. Sun, J., Ovsjanikov, M., & Guibas, L. (2009, July). A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum* (Vol. 28, No. 5, pp. 1383-1392). Oxford, UK: Blackwell Publishing Ltd.

# Reference


- 8. Tenenbaum, J. B., Silva, V. D., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *science*, *290*(5500), 2319-2323.
- 9. Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 652-660).
- 10. Qi, C. R., Yi, L., Su, H., & Guibas, L. J. (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, *30*.
- 11. Sharma, G., Dash, B., RoyChowdhury, A., Gadelha, M., Loizou, M., Cao, L., ... & Kalogerakis, E. (2022, August). PriFit: learning to fit primitives improves few shot point cloud segmentation. In *Computer Graphics Forum* (Vol. 41, No. 5, pp. 39-50).
- 12. Li, L., Sung, M., Dubrovina, A., Yi, L., & Guibas, L. J. (2019). Supervised fitting of geometric primitives to 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 2652-2660).
- 13. Litany, O., Remez, T., Rodola, E., Bronstein, A., & Bronstein, M. (2017). Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the IEEE international conference on computer vision* (pp. 5659-5667).
- 14. Halimi, O., Litany, O., Rodola, E., Bronstein, A. M., & Kimmel, R. (2019). Unsupervised learning of dense shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 4370-4379).