Isometry-Invariant and Subdivision-Invariant Representations of Embedded Simplicial Complexes

Taejin Paik

May 24, 2023

Seoul National University
paiktj@snu.ac.kr

3D Objects

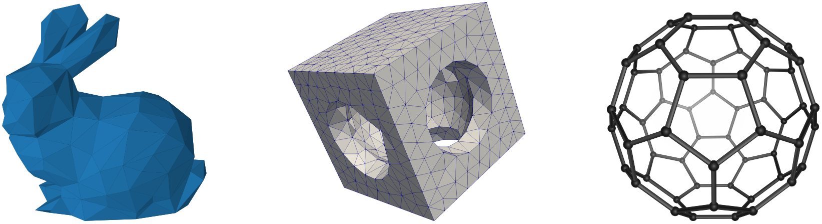


Figure 1: 3D objects

https://www.thingiverse.com/thing:151081
https://github.com/krober10nd/SeismicMesh
https://en.wikipedia.org/wiki/Fullerene

Triangular Mesh

- Triangular mesh can be used to represent complex 3D objects, and it is widely used in computer graphics and computer vision.
- The mesh is created by connecting vertices with edges to form triangular faces.
- We can consider a triangular mesh as a type of simplicial complex.

Triangular Mesh

- Triangular mesh can be used to represent complex 3D objects, and it is widely used in computer graphics and computer vision.
- The mesh is created by connecting vertices with edges to form triangular faces.
- We can consider a triangular mesh as a type of simplicial complex.

Triangular Mesh

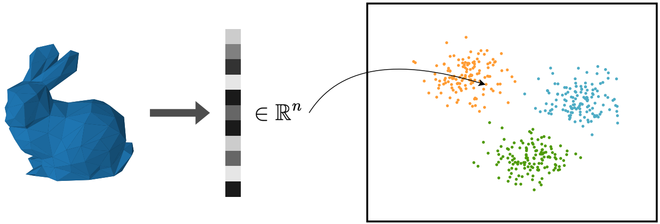
- Triangular mesh can be used to represent complex 3D objects, and it is widely used in computer graphics and computer vision.
- The mesh is created by connecting vertices with edges to form triangular faces.
- We can consider a triangular mesh as a type of simplicial complex.

How to analyze?

Tasks:

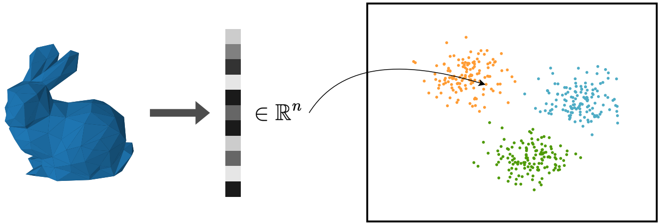
- Classifying
  - Identifying the category or group that a 3D object belongs to based on its features or characteristics.
- Regression
  - Predicting a numerical value for a target variable based on the features or characteristics of a 3D object.
- Clustering
  - Grouping similar 3D objects together based on their features or characteristics, without prior knowledge of their categories or groups.

Vector Representation



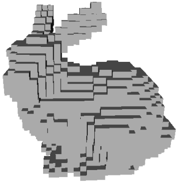
- Our goal is to find a suitable representation vector with constant size based on our task.
- This vector can be used to cluster similar simplicial complexes together, study their properties, and for supervised learning tasks.

Vector Representation



- Our goal is to find a suitable representation vector with constant size based on our task.
- This vector can be used to cluster similar simplicial complexes together, study their properties, and for supervised learning tasks.

Classical Approaches

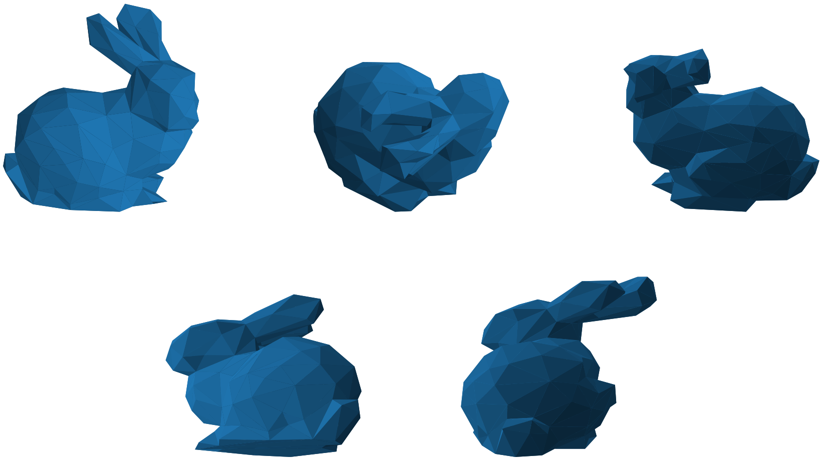


Gupta, A., Watson, S., & Yin, H. (2020, July). 3d point cloud feature explanations using gradient-based methods. In 2020 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

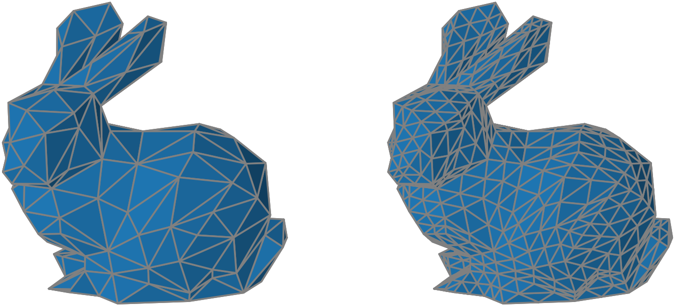
Su, H., Maji, S., Kalogerakis, E., & Learned-Miller, E. (2015). Multi-view convolutional neural networks for 3d shape recognition. In Proceedings of the IEEE international conference on computer vision (pp. 945-953).

https://keras.io/examples/vision/pointnet/

Isometry Invariance



Subdivision Invariance



Our Goal

- Finding a representation vector of each simplicial complex in Euclidean space
- Invariances
  - Isometry Invariance
  - Subdivision Invariance

First, let's briefly look at the definition of a simplicial complex.

Our Goal

- Finding a representation vector of each simplicial complex in Euclidean space
- Invariances
  - Isometry Invariance
  - Subdivision Invariance

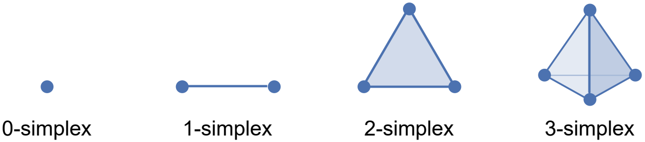
First, let's briefly look at the definition of a simplicial complex.

Preliminaries

Simplex

Definition

A k-**simplex** is a convex hull of $k+1$ affinely independent points in an ambient space $\mathbb{R}^n$.



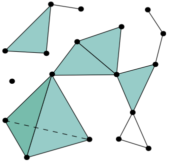
https://umap-learn.readthedocs.io/en/latest/how_umap_works.html

Simplicial Complex

Definition

A **simplicial complex** K is defined as a finite collection of simplices that satisfy:

1. If τ is a face of σ and $\sigma \in K$, then $\tau \in K$.
2. Assume that σ_0 and σ_1 are elements of K. Then, $\sigma_0 \cap \sigma_1$ is a face of σ_0 and σ_1 if it is not the empty set.



https://en.wikipedia.org/wiki/Simplicial_complex

Embedded Simplicial Complex

- We primarily focus on **embedded simplicial complex**, which is defined as a union of simplices in a given simplicial complex K with the subspace topology inherited from the ambient Euclidean space.
- For a simplicial complex K, we will also call the embedded simplicial complex K for ease of notation.

Embedded Simplicial Complex

- We primarily focus on **embedded simplicial complex**, which is defined as a union of simplices in a given simplicial complex K with the subspace topology inherited from the ambient Euclidean space.
- For a simplicial complex K, we will also call the embedded simplicial complex K for ease of notation.

Notations

Definition

For an embedded simplicial complex K in $\mathbb{R}^n$, we write MK to denote $\left\{M^{-1}x \mid x \in K\right\}$ for $M \in \mathrm{GL}_n(\mathbb{R})$.

Also, for a vector v in $\mathbb{R}^n$, we denote the set $\{x + v \mid x \in K\}$ as $K + v$.

Definition

Suppose f is a function from S^{n-1} and R be a matrix $\mathrm{O}(n)$. Then, the function obtained by applying the matrix R to the input of f, that is, the function $x \mapsto f(Rx)$, is denoted as R^*f.

Notations

Definition

For an embedded simplicial complex K in $\mathbb{R}^n$, we write MK to denote $\left\{M^{-1}x \mid x \in K\right\}$ for $M \in \mathrm{GL}_n(\mathbb{R})$.

Also, for a vector v in $\mathbb{R}^n$, we denote the set $\{x + v \mid x \in K\}$ as $K + v$.

Definition

Suppose f is a function from S^{n-1} and R be a matrix $\mathrm{O}(n)$. Then, the function obtained by applying the matrix R to the input of f, that is, the function $x \mapsto f(Rx)$, is denoted as R^*f.

Invariance and Equivariance

- **Invariance** refers to the property of remaining unchanged under certain transformations or operations.

$$\text{ex)}\ f : S^2 \to \mathbb{R}^m,\ \text{and}\ \mathcal{P}(R^* f) = \mathcal{P}(f)\ \text{for}\ R \in O(3).$$

  - Pooling in neural networks (maximum or average)

- **Equivariance** is the property of a function or operation that preserves its behavior under a transformation of its inputs. It means that if we apply a transformation to the input of a function, the output of the function will be transformed in the same way.

$$\text{ex)}\ f : S^2 \to \mathbb{R}^m,\ \text{and}\ \mathcal{P}(R^* f) = R^*(\mathcal{P}(f)) : S^2 \to \mathbb{R}^k$$
$$\text{for}\ R \in O(3).$$

Invariance and Equivariance

- **Invariance** refers to the property of remaining unchanged under certain transformations or operations.

  ex) $f : S^2 \to \mathbb{R}^m,$ and $\mathcal{P}(R^* f) = \mathcal{P}(f)$ for $R \in O(3)$.

  - Pooling in neural networks (maximum or average)

- **Equivariance** is the property of a function or operation that preserves its behavior under a transformation of its inputs. It means that if we apply a transformation to the input of a function, the output of the function will be transformed in the same way.

  ex) $f : S^2 \to \mathbb{R}^m,$ and $\mathcal{P}(R^* f) = R^*(\mathcal{P}(f)) : S^2 \to \mathbb{R}^k$
  for $R \in O(3)$.

Proposed Approach

Overview of Our Approach

We'll introduce the operators here one by one:

- $K \subset \mathbb{R}^3$
- $\mathcal{F}_K : S^2 \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})$
- $\mathcal{DF}_K : S^2 \to \mathbb{R}^a$
- $\mathcal{P}(\mathcal{DF}_K) \in \mathbb{R}^b$

Properties:

- $\mathcal{DF}_{RK+v} = R^*(\mathcal{DF}_K)$
- $\mathcal{P}(\mathcal{R}^*(\mathcal{DF}_K)) \simeq \mathcal{P}(\mathcal{DF}_K)$

for $R \in \mathrm{O}(3)$ and $v \in \mathbb{R}^3$.

First, we introduce the Euler curve transform $\mathcal{F}$.

Overview of Our Approach

We'll introduce the operators here one by one:

- $K \subset \mathbb{R}^3$
- $\mathcal{F}_K : S^2 \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})$
- $\mathcal{DF}_K : S^2 \to \mathbb{R}^a$
- $\mathcal{P}(\mathcal{DF}_K) \in \mathbb{R}^b$

Properties:

- $\mathcal{DF}_{RK+v} = R^*(\mathcal{DF}_K)$
- $\mathcal{P}(\mathcal{R}^*(\mathcal{DF}_K)) \simeq \mathcal{P}(\mathcal{DF}_K)$

for $R \in \mathrm{O}(3)$ and $v \in \mathbb{R}^3$.

First, we introduce the Euler curve transform $\mathcal{F}$.

Euler Characteristic

Let K be a simplicial complex.

$$\chi(K) = \sum_{i=0}^{\infty} (-1)^i c_i$$

where c_i is the number of i-dimensional simplices in K.

$$\chi(K) = \sum_{i=0}^{\infty} (-1)^i \mathrm{rk}\, H_i(K).$$

Therefore, the Euler characteristic is not affected by the subdivision of the simplicial complex.

Euler Characteristic

Let K be a simplicial complex.

$$\chi(K) = \sum_{i=0}^{\infty} (-1)^i c_i$$

where c_i is the number of i-dimensional simplices in K.

$$\chi(K) = \sum_{i=0}^{\infty} (-1)^i \mathrm{rk}\ H_i(K).$$

Therefore, the Euler characteristic is not affected by the subdivision of the simplicial complex.

Semialgebraic Set (1)

Definition

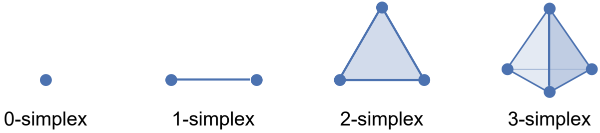
A semialgebraic set is a subset of n-dimensional Euclidean space that can be expressed as a finite union or intersection of sets of two types:

$$\{\bar{x} \in \mathbb{R}^n : f(\bar{x}) > 0\} \text{ and } \{\bar{x} \in \mathbb{R}^n : g(\bar{x}) = 0\},$$

where f and g are polynomials in $\bar{x} = (x_1, \ldots, x_n)$ with real coefficients.

Semialgebraic Set (2)

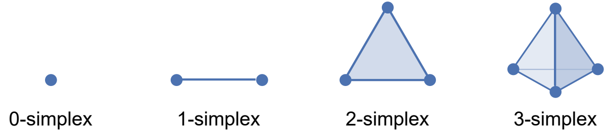
- Each simplex is a semialgebraic set.



- Semialgebraic sets are closed under unions.
- Therefore, an embedded simplicial complex is a semialgebraic set.

Semialgebraic Set (2)

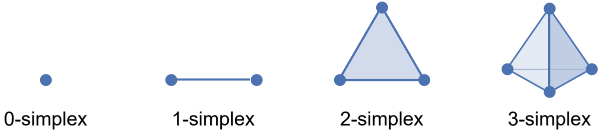
- Each simplex is a semialgebraic set.



- Semialgebraic sets are closed under unions.
- Therefore, an embedded simplicial complex is a semialgebraic set.

Semialgebraic Set (2)

- Each simplex is a semialgebraic set.



- Semialgebraic sets are closed under unions.
- Therefore, an embedded simplicial complex is a semialgebraic set.

Semialgebraic Set (3)

- One of the main properties of semialgebraic sets is that they admit a well-defined notion of Euler characteristics.

Theorem (van den Dries)

Each semialgebraic set $K \subseteq \mathbb{R}^m$ has a finite partition $K = C_1 \cup \cdots \cup C_j$ into cells C_i.

- Euler characteristic for semialgebraic set is (well) defined similarly.

$$\chi(K) = \sum_{i=0}^{\infty} (-1)^i c_i$$

where c_i is the number of i-cells.

van den Dries, L. P. D. (1998). Tame topology and o-minimal structures (Vol. 248). Cambridge university press.

Semialgebraic Set (3)

- One of the main properties of semialgebraic sets is that they admit a well-defined notion of Euler characteristics.

Theorem (van den Dries)

Each semialgebraic set $K \subseteq \mathbb{R}^m$ has a finite partition $K = C_1 \cup \cdots \cup C_j$ into cells C_i.

- Euler characteristic for semialgebraic set is (well) defined similarly.

$$\chi(K) = \sum_{i=0}^{\infty} (-1)^i c_i$$

where c_i is the number of i-cells.

---

van den Dries, L. P. D. (1998). Tame topology and o-minimal structures (Vol. 248). Cambridge university press.

Euler Integration

Additivity property: for semialgebraic sets A and B, we have

$$\chi(A \cup B) = \chi(A) + \chi(B) - \chi(A \cap B).$$

Definition

Let X be a semialgebraic set in $\mathbb{R}^n$. We call an integer-valued function $f : X \to \mathbb{Z}$ **constructible** if $f^{-1}(i)$ is semialgebraic subset for every $i \in \mathbb{Z}$. We denote the set of bounded compactly supported constructible functions on X as $\mathrm{CF}(X)$.

- For every $f \in \mathrm{CF}(X)$, we define

$$\int_X f \, d\chi := \sum_{i \in \mathbb{Z}} i \cdot \chi\left(f^{-1}(i)\right).$$

Euler Integration

Additivity property: for semialgebraic sets A and B, we have

$$\chi(A \cup B) = \chi(A) + \chi(B) - \chi(A \cap B).$$

Definition

Let X be a semialgebraic set in $\mathbb{R}^n$. We call an integer-valued function $f : X \to \mathbb{Z}$ **constructible** if $f^{-1}(i)$ is semialgebraic subset for every $i \in \mathbb{Z}$. We denote the set of bounded compactly supported constructible functions on X as $\mathrm{CF}(X)$.

- For every $f \in \mathrm{CF}(X)$, we define

$$\int_X f \, d\chi := \sum_{i \in \mathbb{Z}} i \cdot \chi\left(f^{-1}(i)\right).$$

Euler Curve Transform

- The Euler curve transform is an operator denoted as $\mathcal{R}$ that maps from $\mathrm{CF}(\mathbb{R}^n)$ to $\mathrm{CF}\left(S^{n-1} \times \mathbb{R}\right)$:

$$\mathcal{R}(f)(v, r) = \int_{\mathbb{R}^n} f(x) \cdot 1_{x \cdot v \leq r}(x) d\chi(x).$$

- For a simplicial complex K, if we put $f = 1_K$,

$$\mathcal{R}(1_K)(v, r) = \int_{\mathbb{R}^n} 1_{\{x \in K \mid x \cdot v \leq r\}} d\chi = \chi(K_{v,r})$$

where $K_{v,r} := \{x \in \mathbb{R}^n \mid x \cdot v \leq r\} \cap K$.

Euler Curve Transform

- The Euler curve transform is an operator denoted as $\mathcal{R}$ that maps from $\mathrm{CF}(\mathbb{R}^n)$ to $\mathrm{CF}\left(S^{n-1} \times \mathbb{R}\right)$:

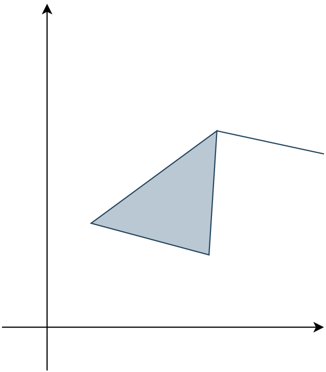
$$\mathcal{R}(f)(v, r) = \int_{\mathbb{R}^n} f(x) \cdot 1_{x \cdot v \leq r}(x) d\chi(x).$$

- For a simplicial complex K, if we put $f = 1_K$,

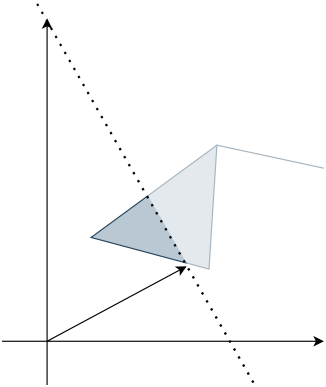
$$\mathcal{R}(1_K)(v, r) = \int_{\mathbb{R}^n} 1_{\{x \in K | x \cdot v \leq r\}} d\chi = \chi(K_{v,r})$$

where $K_{v,r} := \{x \in \mathbb{R}^n \mid x \cdot v \leq r\} \cap K$.

$$K_{v,r} := \{x \in \mathbb{R}^n \mid x \cdot v \leq r\} \cap K$$



$$K_{v,r} := \{x \in \mathbb{R}^n \mid x \cdot v \leq r\} \cap K$$



Injectivity

Theorem (Ghrist et al., 2018)

The Euler curve transform $\mathcal{R} : \mathrm{CF}(\mathbb{R}^n) \to \mathrm{CF}(S^{n-1} \times \mathbb{R})$ *is injective.*

Therefore, instead of the original simplicial complex K, we can deal with $\mathcal{R}(1_K)$.

Ghrist, R., Levanger, R., & Mai, H. (2018). Persistent homology and Euler integral transforms. Journal of Applied and Computational Topology, 2, 55-60.

- To simplify notation, for each embedded simplicial complex K, we define a function $\mathcal{F}_K$:

$$\begin{aligned} \mathcal{F}_K : S^{n-1} &\longrightarrow \mathrm{Map}(\mathbb{R} \to \mathbb{R}) \\ v &\longmapsto \quad \begin{aligned}(\mathcal{F}_K(v) : \quad \mathbb{R} &\longrightarrow \mathbb{R}) \\ r &\longmapsto \chi(K_{v,r})\end{aligned} \end{aligned}$$

- Note that $\mathcal{R}(1_K)(v, r) = \chi(K_{v,r})$

- For each direction $v \in S^{n-1}$, we will call the curve $\mathcal{F}_K(v)$ the **Euler curve**.
- By the injectivity, $K_1 \neq K_2$ implies $\mathcal{F}_{K_1} \neq \mathcal{F}_{K_2}$.

- To simplify notation, for each embedded simplicial complex K, we define a function $\mathcal{F}_K$:

$$\begin{aligned} \mathcal{F}_K : S^{n-1} &\longrightarrow \mathrm{Map}(\mathbb{R} \to \mathbb{R}) \\ v &\longmapsto \quad \begin{aligned}(\mathcal{F}_K(v) : \quad \mathbb{R} &\longrightarrow \mathbb{R}) \\ r &\longmapsto \chi(K_{v,r}) \end{aligned} \end{aligned}$$

- Note that $\mathcal{R}(1_K)(v, r) = \chi(K_{v,r})$

- For each direction $v \in S^{n-1}$, we will call the curve $\mathcal{F}_K(v)$ the **Euler curve**.
- By the injectivity, $K_1 \neq K_2$ implies $\mathcal{F}_{K_1} \neq \mathcal{F}_{K_2}$.

Euler Curves

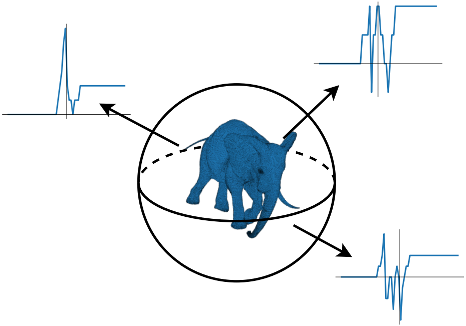


Figure 6: An example of $\mathcal{F}_K : S^2 \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})$

Properties (1)

Proposition

Let K be an embedded simplicial complex in $\mathbb{R}^3$. Then, for $R \in \mathrm{O}(3)$ and $w \in \mathbb{R}^3$,

$$\mathcal{F}_{RK+w}(v)(r) = \mathcal{F}_K(Rv)(r - v \cdot w).$$

That is, the transform is O(3)-equivariant, and if the embedded simplicial complex is translated, then the resulting function is also translated.

Paik, T. (2023). Invariant Representations of Embedded Simplicial Complexes. arXiv preprint arXiv:2302.13565.

Properties (2)

Proposition

Assume that there are translation-invariant functionals $\{\mathcal{D}_i\}_{i=1}^m$ *on the set* $\mathrm{Map}(\mathbb{R} \to \mathbb{R})$*, that is, if there exist* $t \in \mathbb{R}$ *such that* $f(x) = g(x + t)$ *for every* x*, then* $\mathcal{D}_i f = \mathcal{D}_i g \in \mathbb{R}$ *for every* $1 \leq i \leq m$*. Let* $\mathcal{DF}_K$ *be a function*

$$\begin{aligned} \mathcal{DF}_K : S^2 &\longrightarrow \mathbb{R}^m \\ v &\longmapsto \{\mathcal{D}_1 \circ \mathcal{F}_K(v), \ldots, \mathcal{D}_m \circ \mathcal{F}_K(v)\}. \end{aligned}$$

Then, $\mathcal{DF}_*$ *is* O(3)*-equivariant, subdivision-invariant, and translation-invariant on the set of embedded simplicial complexes.*

Paik, T. (2023). Invariant Representations of Embedded Simplicial Complexes. arXiv preprint arXiv:2302.13565.

Translation-Invariant Operator (1)

- One of the simplest translation-invariant functionals for $\mathrm{Map}(\mathbb{R} \to \mathbb{R})$ is the maximum functional, i.e.,

$$\max\{f(x) : x \in \mathbb{R}\}.$$

- However, the maximum functional cannot capture the various features of a function.

Translation-Invariant Operator (1)

- One of the simplest translation-invariant functionals for $\mathrm{Map}(\mathbb{R} \to \mathbb{R})$ is the maximum functional, i.e.,

$$\max\{f(x) : x \in \mathbb{R}\}.$$

- However, the maximum functional cannot capture the various features of a function.

Translation-Invariant Operator (2)

- Instead, we can stack several translation-equivariant operators and apply the max functional at the end to get various invariant features.
  - Let $T_c(f)$ denote the translation operator by c, that is, the function $f(x-c)$ where c is a constant.
  - Let $\{G_i : \mathrm{Map}(\mathbb{R} \to \mathbb{R}) \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})\}_{i=1}^n$ be translation-equivariant operators.
  - Let $\mathcal{H}$ be the maximum functional on $\mathrm{Map}(\mathbb{R} \to \mathbb{R})$.
  - For a function $f \in \mathrm{Map}(\mathbb{R} \to \mathbb{R})$,

$$
\begin{aligned}
\mathcal{H} \circ G_n \circ \cdots \circ G_1 \circ T_c(f) &= \mathcal{H} \circ G_n \circ \cdots \circ T_c \circ G_1(f) \\
&= \cdots \\
&= \mathcal{H} \circ T_c \circ G_n \circ \cdots \circ G_1(f) \\
&= \mathcal{H} \circ G_n \circ \cdots \circ G_1(f).
\end{aligned}
$$

Translation-Invariant Operator (2)

- Instead, we can stack several translation-equivariant operators and apply the max functional at the end to get various invariant features.
  - Let $T_c(f)$ denote the translation operator by c, that is, the function $f(x-c)$ where c is a constant.
  - Let $\{G_i : \mathrm{Map}(\mathbb{R} \to \mathbb{R}) \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})\}_{i=1}^{n}$ be translation-equivariant operators.
  - Let $\mathcal{H}$ be the maximum functional on $\mathrm{Map}(\mathbb{R} \to \mathbb{R})$.
  - For a function $f \in \mathrm{Map}(\mathbb{R} \to \mathbb{R})$,

$$\begin{aligned}
\mathcal{H} \circ G_n \circ \cdots \circ G_1 \circ T_c(f) &= \mathcal{H} \circ G_n \circ \cdots \circ T_c \circ G_1(f) \\
&= \cdots \\
&= \mathcal{H} \circ T_c \circ G_n \circ \cdots \circ G_1(f) \\
&= \mathcal{H} \circ G_n \circ \cdots \circ G_1(f).
\end{aligned}$$

Translation-Invariant Operator (2)

- Instead, we can stack several translation-equivariant operators and apply the max functional at the end to get various invariant features.
  - Let $T_c(f)$ denote the translation operator by c, that is, the function $f(x-c)$ where c is a constant.
  - Let $\{G_i : \mathrm{Map}(\mathbb{R} \to \mathbb{R}) \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})\}_{i=1}^{n}$ be translation-equivariant operators.
  - Let $\mathcal{H}$ be the maximum functional on $\mathrm{Map}(\mathbb{R} \to \mathbb{R})$.
  - For a function $f \in \mathrm{Map}(\mathbb{R} \to \mathbb{R})$,

$$
\begin{aligned}
\mathcal{H} \circ G_n \circ \cdots \circ G_1 \circ T_c(f) &= \mathcal{H} \circ G_n \circ \cdots \circ T_c \circ G_1(f) \\
&= \cdots \\
&= \mathcal{H} \circ T_c \circ G_n \circ \cdots \circ G_1(f) \\
&= \mathcal{H} \circ G_n \circ \cdots \circ G_1(f).
\end{aligned}
$$

Translation-Invariant Operator (2)

- Instead, we can stack several translation-equivariant operators and apply the max functional at the end to get various invariant features.
  - Let $T_c(f)$ denote the translation operator by c, that is, the function $f(x-c)$ where c is a constant.
  - Let $\{G_i : \mathrm{Map}(\mathbb{R} \to \mathbb{R}) \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})\}_{i=1}^n$ be translation-equivariant operators.
  - Let $\mathcal{H}$ be the maximum functional on $\mathrm{Map}(\mathbb{R} \to \mathbb{R})$.
  - For a function $f \in \mathrm{Map}(\mathbb{R} \to \mathbb{R})$,

$$
\begin{aligned}
\mathcal{H} \circ G_n \circ \cdots \circ G_1 \circ T_c(f) &= \mathcal{H} \circ G_n \circ \cdots \circ T_c \circ G_1(f) \\
&= \cdots \\
&= \mathcal{H} \circ T_c \circ G_n \circ \cdots \circ G_1(f) \\
&= \mathcal{H} \circ G_n \circ \cdots \circ G_1(f).
\end{aligned}
$$

Translation-Invariant Operator (2)

- Instead, we can stack several translation-equivariant operators and apply the max functional at the end to get various invariant features.
    - Let $T_c(f)$ denote the translation operator by c, that is, the function $f(x-c)$ where c is a constant.
    - Let $\{G_i : \mathrm{Map}(\mathbb{R} \to \mathbb{R}) \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})\}_{i=1}^n$ be translation-equivariant operators.
    - Let $\mathcal{H}$ be the maximum functional on $\mathrm{Map}(\mathbb{R} \to \mathbb{R})$.
    - For a function $f \in \mathrm{Map}(\mathbb{R} \to \mathbb{R})$,

$$
\begin{aligned}
\mathcal{H} \circ G_n \circ \cdots \circ G_1 \circ T_c(f) &= \mathcal{H} \circ G_n \circ \cdots \circ T_c \circ G_1(f) \\
&= \cdots \\
&= \mathcal{H} \circ T_c \circ G_n \circ \cdots \circ G_1(f) \\
&= \mathcal{H} \circ G_n \circ \cdots \circ G_1(f).
\end{aligned}
$$

Cross-Correlation

Definition

For a bounded measurable function f and a bounded compact supported measurable function g, the cross-correlation is defined as:

$$(f \star g)(\tau) \triangleq \int_{-\infty}^{\infty} f(t+\tau)g(t)\, dt.$$

For $c \in \mathbb{R}$, we have

$$\begin{aligned}
(T_c(f) \star g)(\tau) &= \int_{-\infty}^{\infty} T_c(f)(t+\tau)g(t)\, dt \\
&= \int_{-\infty}^{\infty} f(t+\tau-c)g(t)\, dt \\
&= (f \star g)(\tau - c) = T_c(f \star g)(\tau).
\end{aligned}$$

1D-Convolution

- CNN(Convolutional Neural Network):
  - Discretization of the cross-correlation.
  - CNN can be used to approximate a translation-equivariant operator

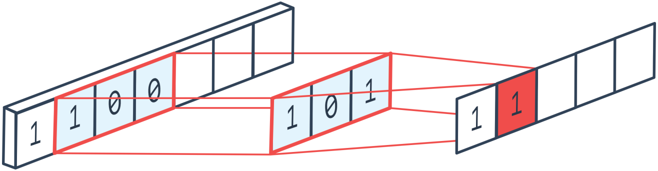


Figure 7: How 1D-CNN works

https://ai.stackexchange.com/questions/28767/what-does-channel-mean-in-the-case-of-an-1d-convolution

- So far, we have defined

$$\mathcal{F}_K : S^2 \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})$$

and

$$\mathcal{DF}_K : S^2 \to \mathbb{R}^a.$$

- Properties:
  - $\mathcal{DF}_{K+v} = \mathcal{DF}_K$ for $v \in \mathbb{R}^3$.
  - $\mathcal{DF}_{RK} = R^*(\mathcal{DF}_K)$ for $R \in \mathrm{O}(3)$.

- But, we cannot process all the data in practice, and need to discretize.

- So far, we have defined

$$\mathcal{F}_K : S^2 \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})$$

  and

$$\mathcal{DF}_K : S^2 \to \mathbb{R}^a.$$

- Properties:
  - $\mathcal{DF}_{K+v} = \mathcal{DF}_K$ for $v \in \mathbb{R}^3$.
  - $\mathcal{DF}_{RK} = R^*(\mathcal{DF}_K)$ for $R \in \mathrm{O}(3)$.

- But, we cannot process all the data in practice, and need to discretize.

- So far, we have defined

$$\mathcal{F}_K : S^2 \to \mathrm{Map}(\mathbb{R} \to \mathbb{R})$$

  and

$$\mathcal{D}\mathcal{F}_K : S^2 \to \mathbb{R}^a.$$

- Properties:
  - $\mathcal{D}\mathcal{F}_{K+v} = \mathcal{D}\mathcal{F}_K$ for $v \in \mathbb{R}^3$.
  - $\mathcal{D}\mathcal{F}_{RK} = R^*(\mathcal{D}\mathcal{F}_K)$ for $R \in \mathrm{O}(3)$.

- But, we cannot process all the data in practice, and need to discretize.

In practice:

1. Draw n points from S^2 as uniformly as possible:

$$\{x_1, x_2, \ldots, x_n\} = X \subset S^2.$$

2. For each point, we obtain a discretized Euler curve

$$\mathcal{F}_K : X \to \mathbb{R}^d.$$

   (where d is a predetermined resolution of the Euler curves.)

3. Using CNNs, activation functions, and a max pooling layer, we obtain

$$\mathcal{DF}_K : X \to \mathbb{R}^a.$$

In practice:

1. Draw n points from S^2 as uniformly as possible:
$$\{x_1, x_2, \ldots, x_n\} = X \subset S^2.$$

2. For each point, we obtain a discretized Euler curve
$$\mathcal{F}_K : X \to \mathbb{R}^d.$$
(where d is a predetermined resolution of the Euler curves.)

3. Using CNNs, activation functions, and a max pooling layer, we obtain
$$\mathcal{DF}_K : X \to \mathbb{R}^a.$$

In practice:

1. Draw n points from S^2 as uniformly as possible:

$$\{x_1, x_2, \ldots, x_n\} = X \subset S^2.$$

2. For each point, we obtain a discretized Euler curve

$$\mathcal{F}_K : X \to \mathbb{R}^d.$$

   (where d is a predetermined resolution of the Euler curves.)

3. Using CNNs, activation functions, and a max pooling layer, we obtain

$$\mathcal{DF}_K : X \to \mathbb{R}^a.$$

O(3)-Equivariant Operator

- The only thing left to do is to build an O(3)-invariant operator.
- As before, to get a O(3)-invariant operator, we can stack several O(3)-equivariant operators first, and then stack an O(3)-invariant operator using a pooling layer.
- Now we need to make an operator

$$\mathcal{T} : \mathrm{Map}(X \to \mathbb{R}^{m_1}) \to \mathrm{Map}(X \to \mathbb{R}^{m_2})$$

that can approximate an O(3)-equivariant operator:

$$\text{``}\mathcal{T}(R^* f) \simeq R^*(\mathcal{T}(f))\text{''}.$$

O(3)-Equivariant Operator

- The only thing left to do is to build an O(3)-invariant operator.
- As before, to get a O(3)-invariant operator, we can stack several O(3)-equivariant operators first, and then stack an O(3)-invariant operator using a pooling layer.
- Now we need to make an operator

$$\mathcal{T} : \mathrm{Map}(X \to \mathbb{R}^{m_1}) \to \mathrm{Map}(X \to \mathbb{R}^{m_2})$$

that can approximate an O(3)-equivariant operator:

$$\text{“}\mathcal{T}(R^* f) \simeq R^*(\mathcal{T}(f))\text{”}.$$

O(3)-Equivariant Operator

- The only thing left to do is to build an O(3)-invariant operator.
- As before, to get a O(3)-invariant operator, we can stack several O(3)-equivariant operators first, and then stack an O(3)-invariant operator using a pooling layer.
- Now we need to make an operator

$$\mathcal{T} : \mathrm{Map}(X \to \mathbb{R}^{m_1}) \to \mathrm{Map}(X \to \mathbb{R}^{m_2})$$

that can approximate an O(3)-equivariant operator:

$$\text{“}\mathcal{T}(R^* f) \simeq R^*(\mathcal{T}(f))\text{”}.$$

Schema

We are going to make $\mathcal{T}$ via a graph neural network:

1. Predetermined $r > 0$ and uniform samples $X \subset S^2$.
2. Construct a graph
   - Consider the points $X = \{x_1, \ldots, x_n\}$ as nodes of a graph.
   - Connect the points where the distance between two points is less than r.
3. Consider the function $\mathcal{DF}_K : X \to \mathbb{R}^a$ as node features.
4. We perform a graph neural network on the graph.

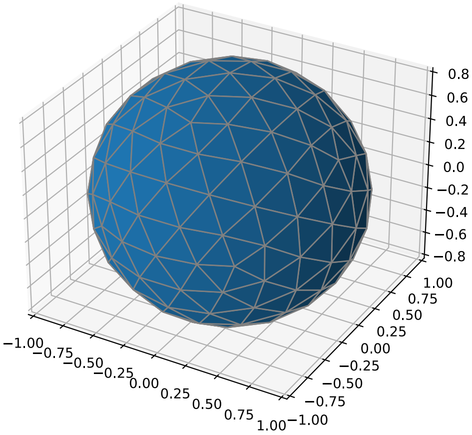
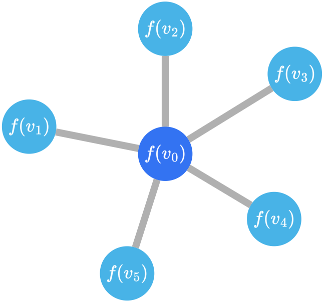


Figure 8: Subdivision of the icosahedron

Graph Convolutional Network (1)

- Graph Convolutional Network (GCN) is a type of neural network that operates on graph structures.
- GCN is an operator that updates feature vectors for each node in a graph by modeling interactions between neighboring nodes and using both the graph structure and node features.

Graph Convolutional Network (2)



Neighborhood of v_0

- Node feature $f : V \to \mathbb{R}^k$.
- New updated node feature $\mathcal{T}(f)$:

$$\mathcal{T}(f)(v) = \mathrm{Mean}\{W_\theta \cdot f(u) : u \in \tilde{\mathcal{N}}(v)\}$$

where $\tilde{\mathcal{N}}(v) = \{v\} \cup \mathcal{N}(v)$.

Extension

- Now, we need to show

$$\text{“}\mathcal{T}(R^* f) \simeq R^*(\mathcal{T}(f))\text{”}$$

  for a function $f \in \text{Map}(X \to \mathbb{R}^{m_1})$, but this does not make sense. ($\because \mathcal{T} : \text{Map}(X \to \mathbb{R}^{m_1}) \to \text{Map}(X \to \mathbb{R}^{m_2})$)
- Thus, we extend $\mathcal{T}$:

$$\mathcal{T}' : \text{Map}(S^2 \to \mathbb{R}^{m_1}) \to \text{Map}(S^2 \to \mathbb{R}^{m_2})$$

  where $\mathcal{T}'(f)(x) = \text{Mean}\{W_\theta \cdot f(u) : \ \|x - u\| < r, u \in X\}$. Obviously, $\mathcal{T}'(f)\big|_X = \mathcal{T}(f\big|_X)$.
- Now, the statements $\mathcal{T}'(R^* f)$ and $R^*(\mathcal{T}'(f))$ make sense.

Extension

- Now, we need to show

$$\text{“}\mathcal{T}(R^*f) \simeq R^*(\mathcal{T}(f))\text{”}$$

for a function $f \in \mathrm{Map}(X \to \mathbb{R}^{m_1})$, but this does not make sense. ($\because \mathcal{T} : \mathrm{Map}(X \to \mathbb{R}^{m_1}) \to \mathrm{Map}(X \to \mathbb{R}^{m_2})$)

- Thus, we extend $\mathcal{T}$:

$$\mathcal{T}' : \mathrm{Map}(S^2 \to \mathbb{R}^{m_1}) \to \mathrm{Map}(S^2 \to \mathbb{R}^{m_2})$$

where $\mathcal{T}'(f)(x) = \mathrm{Mean}\{W_\theta \cdot f(u) : \; \|x-u\| < r, u \in X\}$. Obviously, $\mathcal{T}'(f)\big|_X = \mathcal{T}(f\big|_X)$.

- Now, the statements $\mathcal{T}'(R^*f)$ and $R^*(\mathcal{T}'(f))$ make sense.

Extension

- Now, we need to show

$$\text{``}\mathcal{T}(R^* f) \simeq R^*(\mathcal{T}(f))\text{''}$$

for a function $f \in \mathrm{Map}(X \to \mathbb{R}^{m_1})$, but this does not make sense. ($\because \mathcal{T} : \mathrm{Map}(X \to \mathbb{R}^{m_1}) \to \mathrm{Map}(X \to \mathbb{R}^{m_2})$)

- Thus, we extend $\mathcal{T}$:

$$\mathcal{T}' : \mathrm{Map}(S^2 \to \mathbb{R}^{m_1}) \to \mathrm{Map}(S^2 \to \mathbb{R}^{m_2})$$

where $\mathcal{T}'(f)(x) = \mathrm{Mean}\{W_\theta \cdot f(u) : \ \|x - u\| < r, u \in X\}$. Obviously, $\mathcal{T}'(f)\big|_X = \mathcal{T}(f\big|_X)$.

- Now, the statements $\mathcal{T}'(R^* f)$ and $R^*(\mathcal{T}'(f))$ make sense.

O(3)-Equivariance

Theorem

Let f be a bounded measurable function on S^2. Assume that $x_1, x_2, \ldots, x_n$ are independent identically distributed random variables from the uniform distribution on S^2. Then, for $R \in \mathrm{O}(3)$ and $\epsilon > 0$,

$$\mathbb{P}\left[\|R^*\mathcal{T}'(f)(x) - \mathcal{T}'(R^*f)(x)\|_\infty > \epsilon\right] \longrightarrow 0$$

for every $x \in S^2$ as n goes to infinity.

- Using several GCN layers, activation layers, and a pooling layer at the end, we can approximate an O(3)-invariant operator.

Paik, T. (2023). Invariant Representations of Embedded Simplicial Complexes. arXiv preprint arXiv:2302.13565.

Summary

- First, sample points $X = \{x_1, \ldots, x_n\} \subset S^2$ uniformly.

1 $K \subset \mathbb{R}^3$

2 $\mathcal{F}_K : X \to \mathbb{R}^d$

3 $\mathcal{DF}_K : X \to \mathbb{R}^m$

4 $\mathcal{P}(\mathcal{DF}_K) \in \mathbb{R}^k$

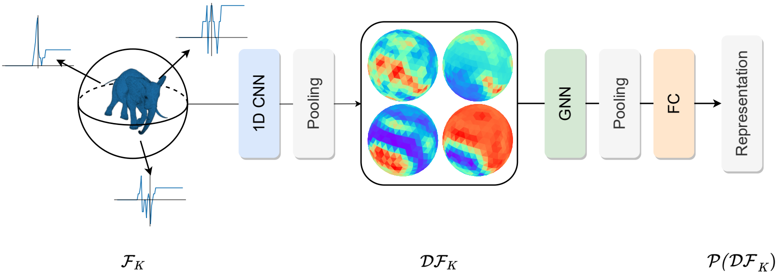


Figure 9: Simplified structure of the proposed architecture

Discretization Error

- Dealing with discretized Euler curves and using CNNs instead of the cross-correlation.
- The theorem for the GCN is about what happens as n goes to infinity, but we don't know how large the error will be for finite n.
- By stacking multiple neural network layers with discretization errors, there is a possibility that the discretization error could be amplified in the overall deep learning model.

Therefore, we conducted a very simple experiment to measure this.

Discretization Error

- Dealing with discretized Euler curves and using CNNs instead of the cross-correlation.
- The theorem for the GCN is about what happens as n goes to infinity, but we don't know how large the error will be for finite n.
- By stacking multiple neural network layers with discretization errors, there is a possibility that the discretization error could be amplified in the overall deep learning model.

Therefore, we conducted a very simple experiment to measure this.

Discretization Error

- Dealing with discretized Euler curves and using CNNs instead of the cross-correlation.
- The theorem for the GCN is about what happens as n goes to infinity, but we don't know how large the error will be for finite n.
- By stacking multiple neural network layers with discretization errors, there is a possibility that the discretization error could be amplified in the overall deep learning model.

Therefore, we conducted a very simple experiment to measure this.

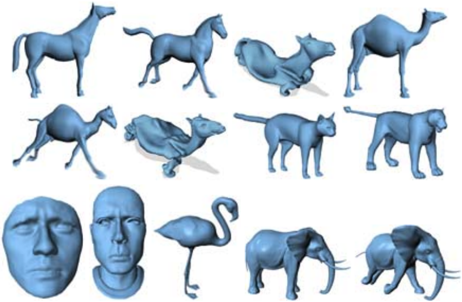
Discretization Error

- Dealing with discretized Euler curves and using CNNs instead of the cross-correlation.
- The theorem for the GCN is about what happens as n goes to infinity, but we don't know how large the error will be for finite n.
- By stacking multiple neural network layers with discretization errors, there is a possibility that the discretization error could be amplified in the overall deep learning model.

Therefore, we conducted a very simple experiment to measure this.

Experiment

ANIM (ANimals in Motion)



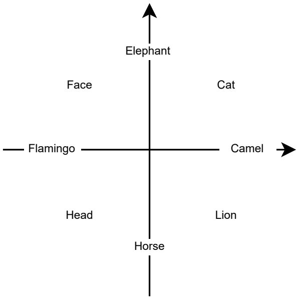
- The dataset I used
  - 229 mesh data
  - 8 classes

---

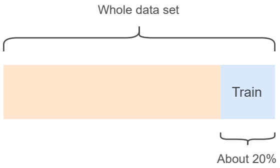
http://people.csail.mit.edu/sumner/research/deftransfer/data.html

Training Details

- Train the model to converge on the vertices of a regular octagon for each class.

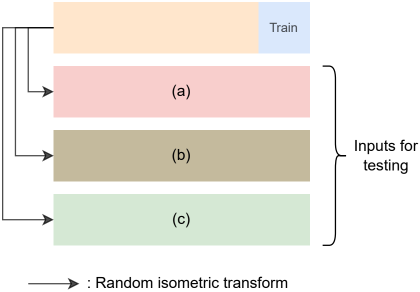


- Randomly picked 5 for each class to train.

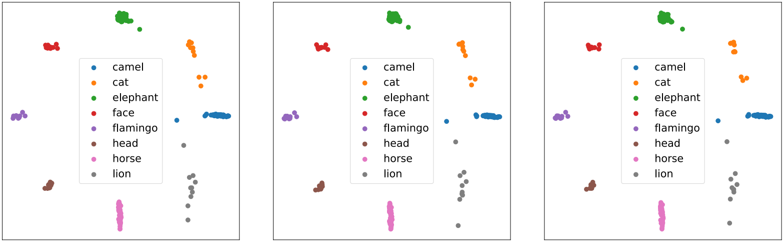


Test Details

- Apply a random isometric transformation to each data three times, to obtain three new data sets, which are then used as inputs to the neural network.



Experimental Results



(a) (b) (c)

Figure 10: The outcomes for 3 datasets.

Discussion

- Theoretical error bound of each layer
- Computational cost
- Usage of persistent homology/cohomology techniques

Discussion

- Theoretical error bound of each layer
- Computational cost
- Usage of persistent homology/cohomology techniques

Discussion

- Theoretical error bound of each layer
- Computational cost
- Usage of persistent homology/cohomology techniques

Discussion

- Theoretical error bound of each layer
- Computational cost
- Usage of persistent homology/cohomology techniques

Thank You