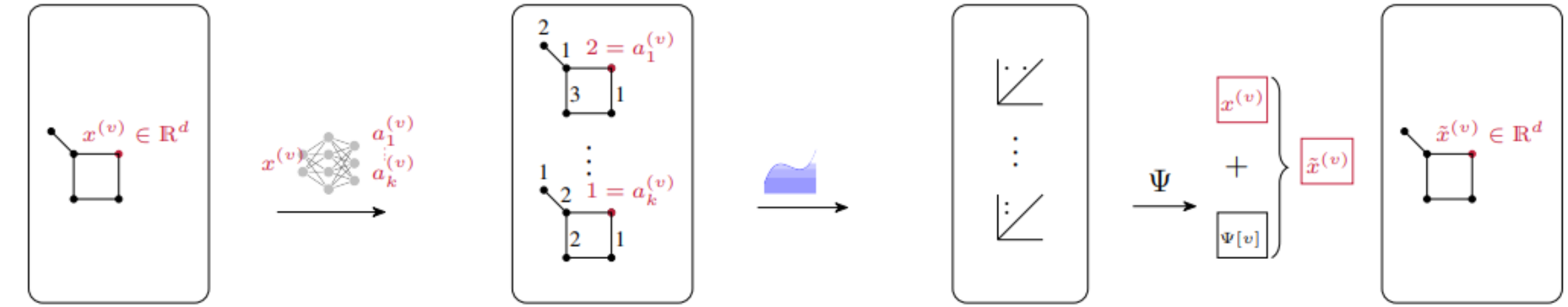
Lee deok joong

TOPOLOGICAL GRAPH NEURAL NETWORKS

ICLR 2022



(a) node attributes $x^{(v)}$ (b) node map Φ (c) k views of graph G (d) filtrations f_i (e) Persistence diagrams (f) aggregation (g) output $\tilde{x}^{(v)}$

목차

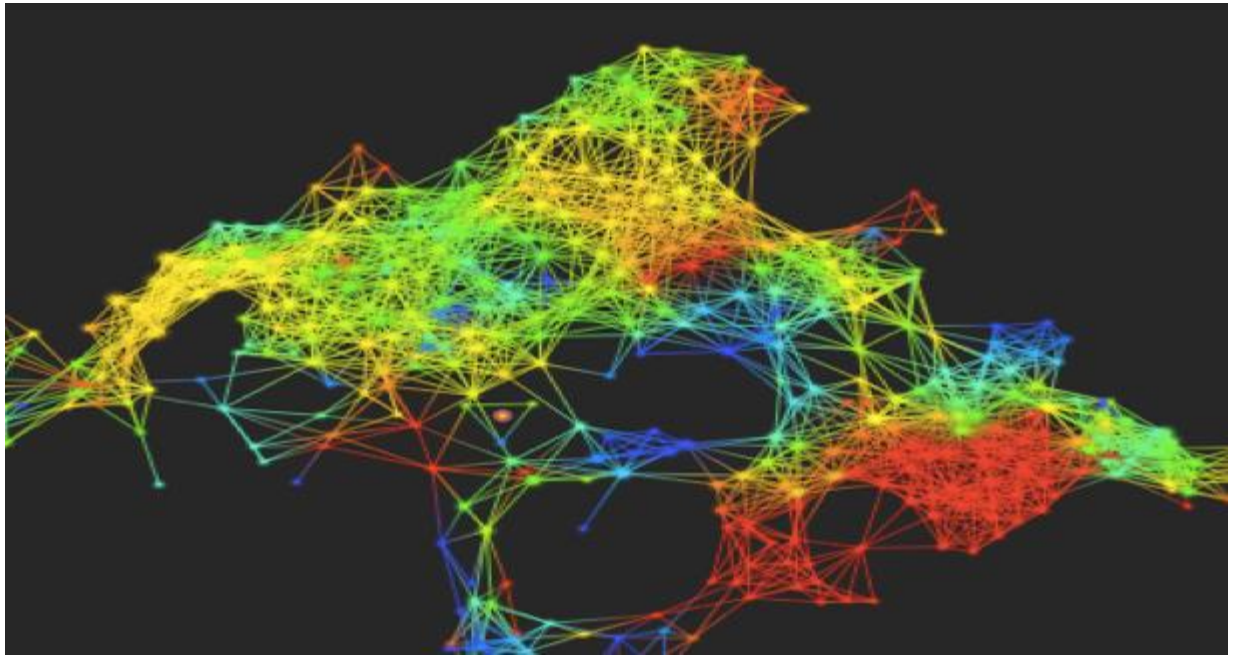
01

Introduction

01 Introduction

- 이 논문은 무슨 내용을 다루는가?

1. 최근 부상하고 있는 topological data analysis 분야에 뿌리를 둔 방법으로 어떤 GNN이든 topology-aware하게 만들어주는 TOpological Graph Layer(TOGL) novel layer를 제안한다.
2. 정확히는 위상수학 기반의 'persistent homology'를 사용하여 global topological information을 통합하는 novel layer라고 할 수 있다.
3. 간단히 얘기하자면, 위상수학을 기반으로하여 topological information을 잘 활용할 수 있는 learnable한 함수를 만들어 이를 node feature에 residual scheme로 입혀주게 된다.



https://towardsdatascience.com/from-tda-to-dl-d06f234f51d

- TDA [topological data anaylsis]

Data에서 위상학적 특성을 추출해 통계적으로 분석을 하는 분야.

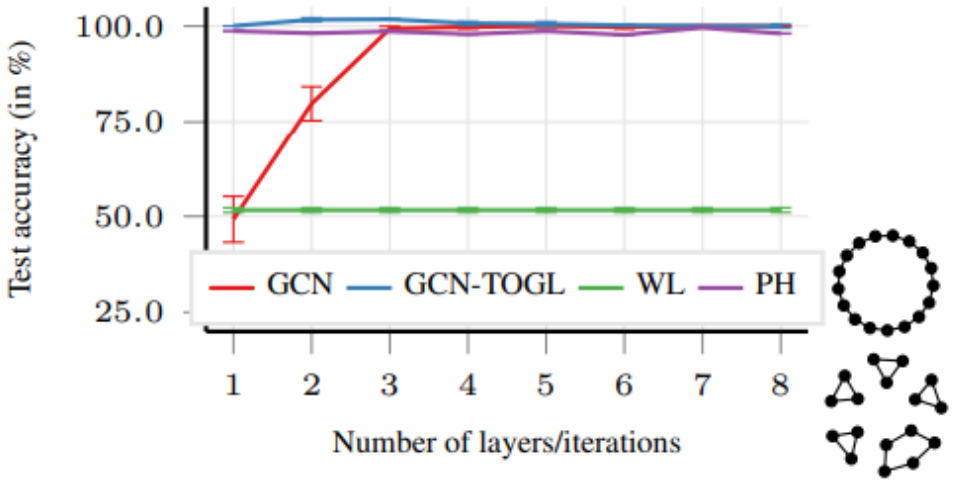
주로 persistent homology를 사용해 분석한다.

01 Introduction

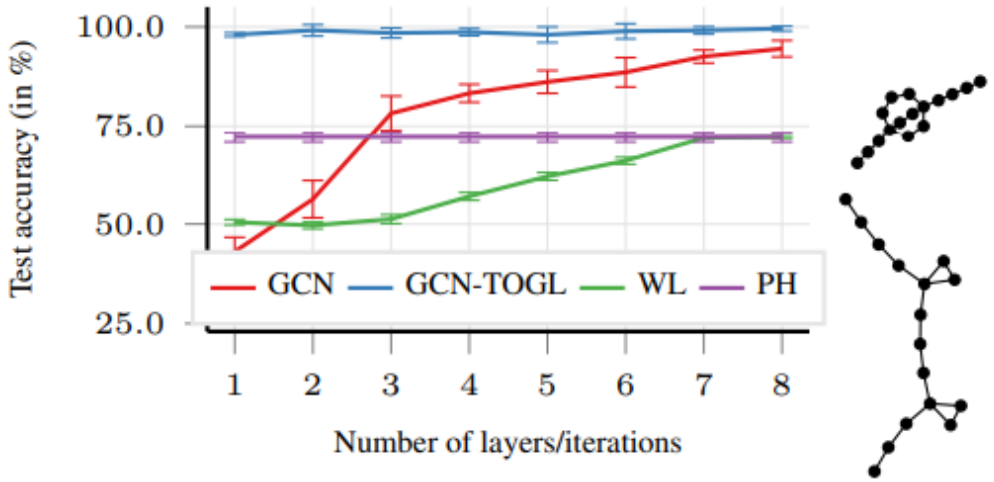
- 그래서 이걸 왜 하는지?

1. 대부분의 GNN은 iterative MP 방법을 이용해 학습을 하는데, 'GIN'논문에서 나오듯이 1-WL test성능에 upper bound하며, cycle같은 특정한 topology structure를 capture하지 못한다.
2. 그래서 topological information을 잘 활용하고자 제안하고자 함 [motivation]

   ➔ TDA에 기반을 두고 있으며, 위상적 차이를 modelonng하기 위해 persistent homology사용



(a) CYCLES



(b) NECKLACES

01 Introduction

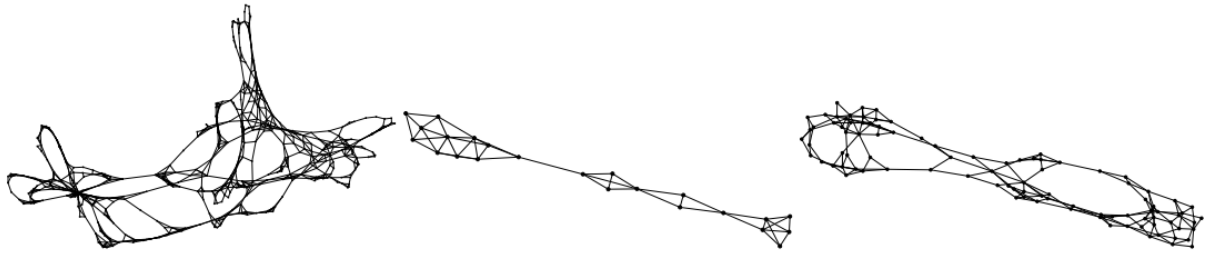
- Contribution

1. 모든 GNN에 통합 가능한 TDA 개념 기반의 novel layer를 제안

   1) 더 적은 layer에서 좋은 성능 (non-isomorphim)

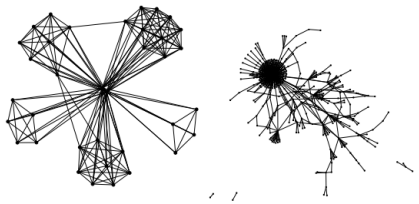
   2) 복잡한 topology structure 있는경우 learnable topological representation이 고정된 위상 구조를 가진 경우보다 좋은 성능을 보여줌

2. TOGL은 미분가능하고 graph의 contrasting topological representation을 학습 가능
3. TOGL은 multi scale topology information으로 작업할 수 있는 ability를 incorporate하여 expressivity를 강화시킨다.
4. Topological information이 각 task와 관련있을때 몇몇의 GNN 구조의 예측 성능을 향상



(a) DD (b) ENZYMES (c) PROTEINS

Figure S5: Example graphs of the benchmark data sets that describe molecular structures. These graphs give rise to complex topological structures that can be exploited.



(a) IMDB-BINARY (b) REDDIT-BINARY

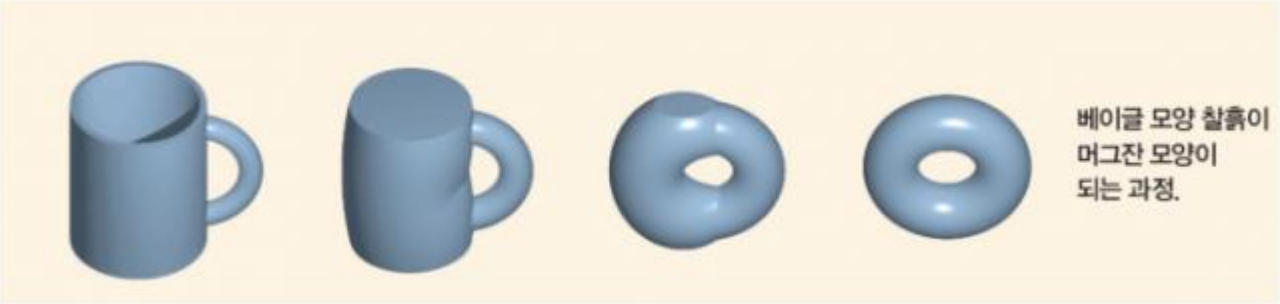
Figure S6: Example graphs of the benchmark data sets that correspond to extracted social networks. These graphs are best described in terms of clique connectivity; the existence of a single 'hub' node does not give rise to a complex topological structures that can be exploited *a priori*.

01 Introduction

수학은 기하학에서 시작되었다…!!!

- **우리 위상수학에 대해 알아볼까요?**

연결성이나 연속성등, 작은 변환에[invariant]에 의존하지 않는 기하학적 성질들을 다루는 수학 분야



https://www.dongascience.com/news.php?idx=31079

왜 위상동형(homeomorpic)이냐?

: 자르거나 붙이지 않고 늘이는 등의 변형을 가하면 서로 같은 모양이 되기 때문 [본질을 봐야합니다 본질을!]

임의의 두 가지 공간이 서로 homeomorpic 인지 판단하는 것은 위상수학에서 아주 근본적인 문제

➔ Bijective mapping(전단사상)의 존재 유무를 증명하면서 확인 [없다면 위상동형이 아닌것]

➔ 하지만 전단사상이 없음을 확인하는 건 너무 어렵기 때문에 유용한 topological invariant를 찾는 방법을 찾게 됨 (algebraic topology - 대수적 위상수학)

ex) betti's homology groups : homology는 위상공간 속에 존재하는 구멍들을 의미, 이런 구멍들을 이용하여 두 공간의 위상 동형 여부를 확인하는 것이 훨씬 다루기 쉽고 계산도 효율적임

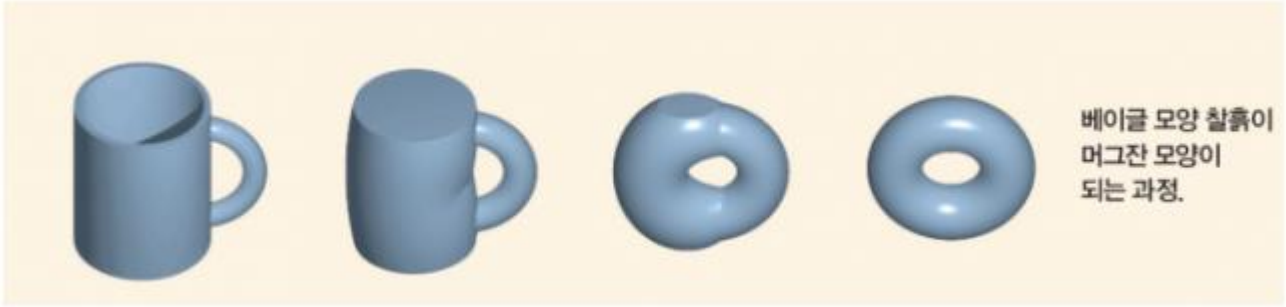
https://ko.wikipedia.org/wiki/%EC%9C%84%EC%83%81%EC%88%98%ED%95%99
https://jjycjnmath.tistory.com/150
http://koreascience.kr/article/JAKO201508449473670.pdf

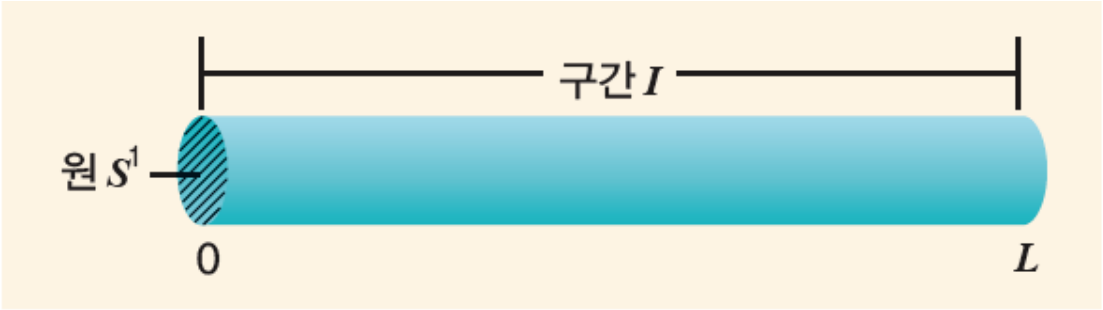
01 Introduction

수학은 기하학에서 시작되었다…!!!

- 우리 위상수학에 대해 알아볼까요?

빨대를 잘 조물조물(?) 하다보면 도넛도 되고 머그컵도 된다.. ➔ 구멍이 1개, 위상적으로 동형





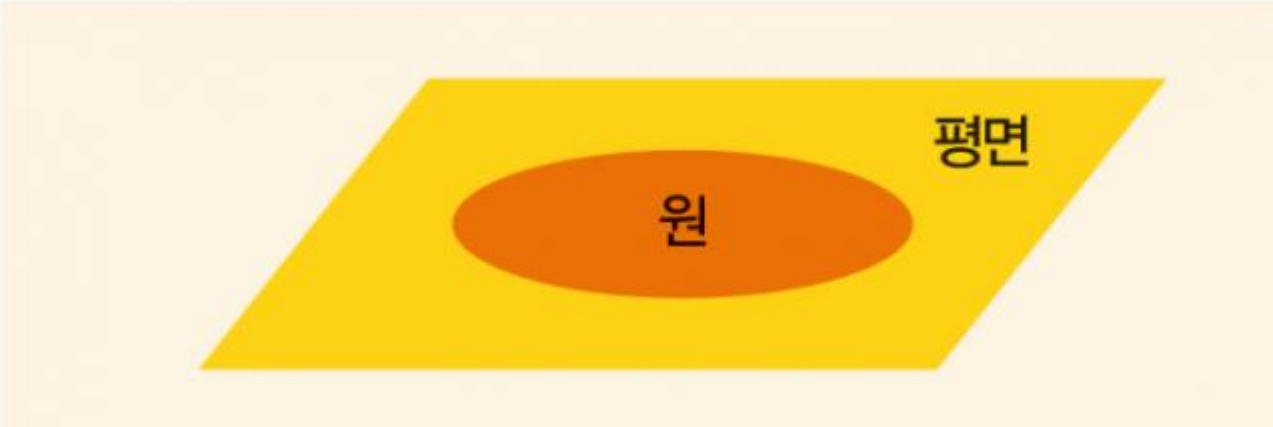
https://www.dongascience.com/news.php?idx=31079

https://ko.wikipedia.org/wiki/%EC%9C%84%EC%83%81%EC%88%98%ED%95%99
https://jjycjnmath.tistory.com/150
http://koreascience.kr/article/JAKO201508449473670.pdf

01 Introduction

수학은 기하학에서 시작되었다…!!!

- 우리 위상수학에 대해 알아볼까요? : 구멍이라는 게 대체 무엇인가



구멍 : 1차원의 선으로 둘러싸인 2차원의 대상

➔ 구멍의 수는 **폐곡선(closed)**으로 둘러싸인 공간의 수

➔ 구간의 모습과 상관없이 구멍의 수가 결정 됨 : 그래서 빨대와 도넛이 위상동형이라는 것

https://ko.wikipedia.org/wiki/%EC%9C%84%EC%83%81%EC%88%98%ED%95%99
https://jjycjnmath.tistory.com/150
http://koreascience.kr/article/JAKO201508449473670.pdf

01 Introduction

- **우리 위상수학에 대해 알아볼까요?**

Simplicial Complexes (단체 복합체)

- simplicial homology를 구성, complex는 topological space를 단순화하여 수학적으로 표현한 것을 말함

  ex) 뇌 신경망 분석 : 뇌의 각 영역[점], 영역간 상호관계[선]

- simplex (단체) : 조각(점, 선, 삼각형,…)들이 위상적으로 단순하고, 그들 사이의 교집합 혹은 맞닿은 면(face)이 같은 종류의 낮은 차원 조각들로 나타내어질 때, 이러한 조건을 만족하는 조각을 다른 말로 단체라 한다.

➔ 단체들의 집합 : simplicial complex (0차원 단체 : 꼭짓점(vertex) 1차원 단체 : 선)

Homology Groups (호몰로지군)

- 호몰로지군은 위상공간에서 다차원 '구멍'에 대해 다룬다 : 한 공간에서 공간을 구성하는 원소들끼리 어떤 형태로 이어져 있는가!
- **'이어진 형태(chain)'**를 통해 위상 공간에서 '구멍'을 찾는 것이 호몰로지군의 접근 방식
- **호몰로지군을 계산해서 위상적 차이를 볼 수 있다 (논문의 핵심)** : betti number는 위상 공간의 호몰로지 군의 계수, k차원의 구멍의 수
- Dimesion-0 [0차원 betti number] : number of connected components
- Dimesion-1 [1차원 betti number] : number of holes or cycles without boundary

http://koreascience.kr/article/JAKO201508449473670.pdf

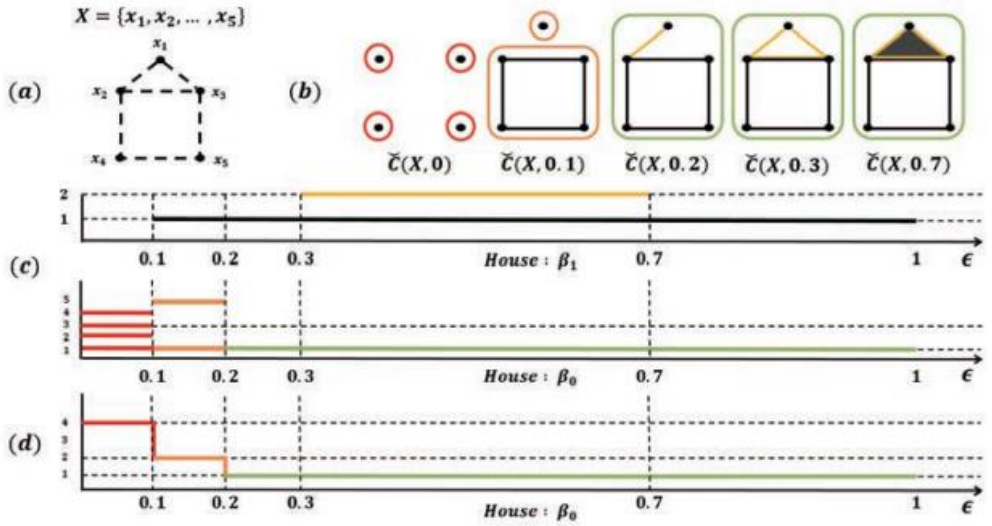
01 Introduction

▪ 우리 위상수학에 대해 알아볼까요?

Persistent homology

- 데이터를 처리하는 과정에서 중요 신호도 있겠지만 잡음도 존재하는데, 이를 구분하는것은 관점에 따라 달라짐

  ex) 뇌 신경망에서의 유의미한 correlation threshold value 설정
- 그래서 이러한 실용적인 문제를 해결하기 위해 연구되었고, 대수적 방법이나 함수나 도형의 위상적인 특징을 측정하여 데이터의 위상적 잡음을 제거하고자 하는 것이 주된 전략이다.
- 뇌 신경망을 예를 들자면, 신경망에 존재하는 임의의 한 구조에 **대해 임계값을 점진적으로 늘리거나, 줄여서** 그 구조가 상당히 많은 변화 동안 **꾸준히 존재한다면 (long persistence), 중요한 구조**이고, 약간의 변화만으로 구조가 없어진다면 (short persistence) 이는 위상적 잡음이라고 볼 수 있다.



〈그림 5〉 Persistent Homology의 간단한 예시

http://koreascience.kr/article/JAKO201508449473670.pdf

02

Background : computational topology & Related work

02 Background : computational topology & Related work

- Background

$$G = (V, E)$$

❖ Simplicial homology

- Simplicial complexes are the central concept in algebraic topology
- Graph는 0-simplices(vertices)와 1-simplices(edges)을 포함한 low-dimensional simplicial complex로 볼 수 있다.

❖ Betti numbers

- To fully close the loop, the sequence of Betti numbers b_0, b_d of a d-dimension simplicial complex is commonly used as a complexity measure, and they can be used to discriminate manifolds.
- 하지만 Betti numbe는 coarse하게 feature count를 하기 때문에, 한계점이 있는데, 이러한 제한된 표현력은 persistent homology의 개발을 촉진하게 됨

02 Background : computational topology & Related work

- Background

$$G = (V, E)$$

❖ Persistent homology

- Persistent homology is an extension of simplicial homology, which employs filtrations to imbue a simplicial complex K with scale information

$$\emptyset = \mathrm{K}^{(0)} \subseteq \mathrm{K}^{(1)} \subseteq \cdots \subseteq \mathrm{K}^{(m-1)} \subseteq \mathrm{K}^{(m)} = \mathrm{K},$$

- Simplicial homology와는 대조적으로, filtration은 변화를 추적할 수 있기 때문에 잠재적으로 더 많은 정보를 보유!
- At its core, persistent homology is 'just' a way of tracking topological features, representing each one by a creation and destruction value (f(i) , f(j)) ∈ R^2 , with i ≤ j.
- 이를 tuple로 표현을 함

02 Background : computational topology & Related work

- Background

$$G = (V, E)$$

- Basic topological features of Graph 'G'

β_0 *: the number of connected components [0-dimensional Betti number]*

β_1: *the number of cycles* $[1 - dimensional\ Betti\ number]$

- Betti number는 graph isomorphism에서 invariant하다 [중요 -> 1-WL test이 분류 못하는 걸 해낸다.]
- Betti number의 expressivity는 graph filtration을 사용함면서 커질 수 있다.
- Filtraction은 persistence와 관련이 깊다

$$\emptyset = G^{(0)} \subseteq G^{(1)} \subseteq G^{(2)} \subseteq \cdots \subseteq G^{(n-1)} \subseteq G^{(n)} = G.$$

- Topological feature는 tuple(i, j)로 표현되며, persistent diagram D에서 collecting
- first time : $G^{(i)}$, second time : G^{j}, topology feature가 G^{j} 에서는 사라지는 경우 ➔이 feature를 j-i의 persistent로 assign한다.
- 그래서 0~n까지 topology가 persistent한지 분석을 할 수 있는데, 이 **전체 과정이 persistent homolgy**인 것이고, 이 논문에서는 k view로 표현하는 것 같음.
- Core concept중 하나가 filtration function $f : V -> R^d$ 인데, 이는 tuple (i, j)를 tuple(f_i, f_j)로 replace를 해준다.
- 따라서 적절한 filtration function 선택이 중요하다고, 다른 field에서 언급하고 있음

02 Background : computational topology & Related work

▪ Background : notation

$$G = (V, E)$$

- Persistent diagrams 계산 : $ph(G, f)$ [G : graph, f : some filtration function]

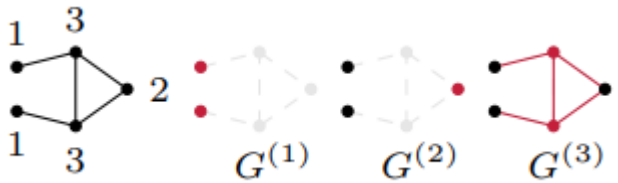
  ➔ 특정 핵심 구조를 강조

- 두 개의 persistence diagrams이 나오게 됨 : $D^{(0)}, D^{(1)}$
- $D^{(0)}$: dimension-0 (connected components) $D^{(1)}$: dimension-1 (cycles)

❖ Examples of graph filtrations (그래서 filtration function이 대체 뭐고, 왜 중요한가?)

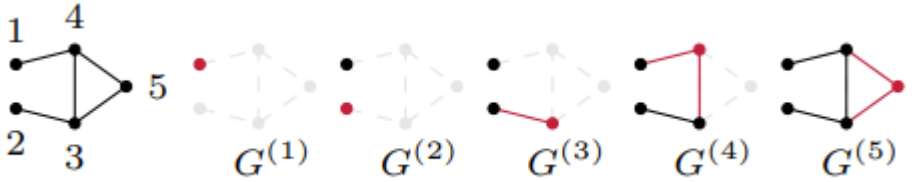
- Filtrations are most conveniently thought of as arising from a function f : G → R, which assigns a scalar-valued function value to each node and edge of the graph by means of setting f(u, v) := max{f(u), f(v)} for an edge (u, v).
- Red : added node and edges | black : already exists at this step

▪ a degree-based filtration of a simple graph



$$\mathcal{D}_0 = \{(1, \infty), (1, 3), (2, 3), (3, 3), (3, 3)\}$$
$$\mathcal{D}_1 = \{(3, \infty)\}$$

▪ a structural role in terms of a heat kernel



$$\mathcal{D}_0 = \{(1, \infty), (3, 3), (2, 4), (4, 4), (5, 5)\}$$
$$\mathcal{D}_1 = \{(5, \infty)\}.$$

➔ 새롭게 생성되면, 빨간색, 그리고 기존에 존재하는 node, edge는 검은색이며, 만약 기존에 존재하는 node와 edge연결이 있었다면, 빨간색으로 연결됨

02 Background : computational topology & Related work

- Related work

  - Graph representation learning has received a large amount of attention by the machine learning community

  1) 'Graph kernel method' : embedding in RKHS

  ➔ powerful하지만, 이웃간 partial 유사성 capture를 하지 못함 : GNN은 가능하다

  2) Recent GNN : additional substructures, higher-order message passing schemes 정의,

- Related work : Topological feature learning

  - Our method : **TDA에 근거하며, persistent homology를 employ해서 topological feature를 계산한다.**

    ➔ persistent homology는 본질적으로 discrete하기 때문에 미분가능하게 modeling해야하고 저자들은 end-to-end로 topological feature의 complementary view를 통합하여 표현력을 높였다고 함

  - 우리 model의 theoretical framework는 *'Hofer(2020)'*를 따른다고 함

    1) GNN output값으로 filtration function학습 가능 2) filtration function은 미분가능

02 Background : computational topology & Related work

- Related work : Topological feature learning
  - GFL (topological readout function) : GNN output만을 이용해 single filtration을 사용함 -> 말 그대로 readout
  - TOGL (our method) : multiple filtration을 사용하고, node feature hidden representation에 topological information을 포함시켜서 topological signal의 중요도를 network가 학습을 통해 변경할 수 있다.

❖ Closest to the scope of TOGL

A) Zhao (2020) : TOGL과 유사, 하지만 단순히 MP과정에서 scalar weigh로 제공되며, static vertorization eechique사용해 small neighbourhoods에서만 계산됨

B) Wong and Vong (2021) : 3D shape segmentation위한 모델로, non-learnable topological featur를 사용

C) TOGL (ours*) : end-to-end로 미분가능하기 때문에, general 하며, graph-level feature를 포함한 모든 scale에서 topological features 계산 가능하고, 임의의 GNN과 통합이 가능하다.

03

TOGL : A Topological Graph Layer

03 TOGL : A Topological Graph Layer

- Algorithmic details

TOGL
: input graph에서 multi scale topological information을 활용할 수 있는 새로운 novel layer

- The image of f_i is finite and results in a set of node values $a_i^{(1)} < \cdots < a_i^{(n)}$ such that the graph G is filtered according to

$$\emptyset = G_i^{(0)} \subseteq G_i^{(1)} \subseteq \cdots \subseteq G_i^{(n)} = G, \text{ where } G_i^{(j)} = \left(V_i^{(j)}, E_i^{(j)}\right), \text{ with } V_i^{(j)} := \left\{v \in V \mid f_i\left(x^{(v)}\right) \leq a_i^{(j)}\right\},$$

$$\text{and } E_i^{(j)} := \left\{v, w \in E \mid \max\left\{f_i\left(x^{(v)}\right), f_i\left(x^{(w)}\right)\right\} \leq a_i^{(j)}\right\}.$$

- Given this filtration, we calculate a set of persistence diagrams [fix l = 1 : connected components and cycles ➔ computational issue and performance]

$$\mathbf{ph}(G, f_i) = \left\{\mathcal{D}_i^{(0)}, \ldots, \mathcal{D}_i^{(l)}\right\}.$$

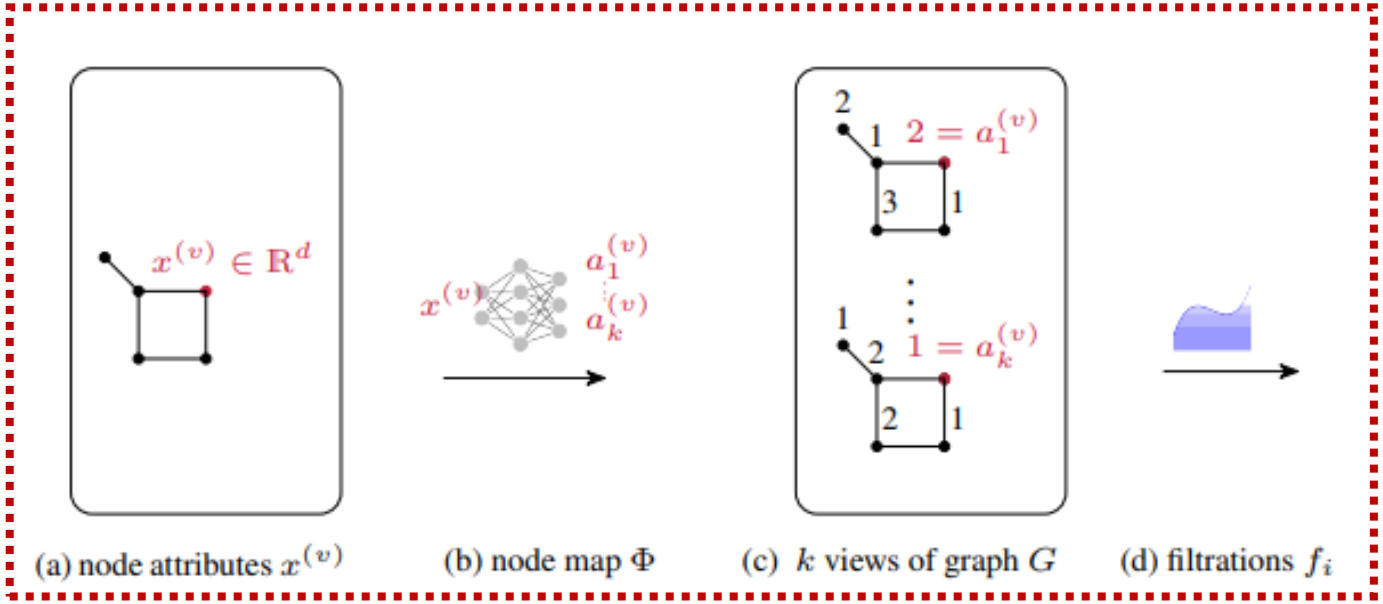
- Trainable end-to-end의 representation의 benefit을 위해 embedding function $\Psi^{(l)}: \left\{\mathcal{D}_1^{(l)}, \ldots, \mathcal{D}_k^{(l)}\right\} \to \mathbb{R}^{n' \times d}$

- Finally, output $\tilde{x}^v$라는 new node representation augmented with multi-scale topological information을 얻게 됨
- 이러한 과정을 통해 topological feature를 node feature로 활용하면서, TOGL을 어떤 임의의 GNN과도 통합시킬 수 있게 됨

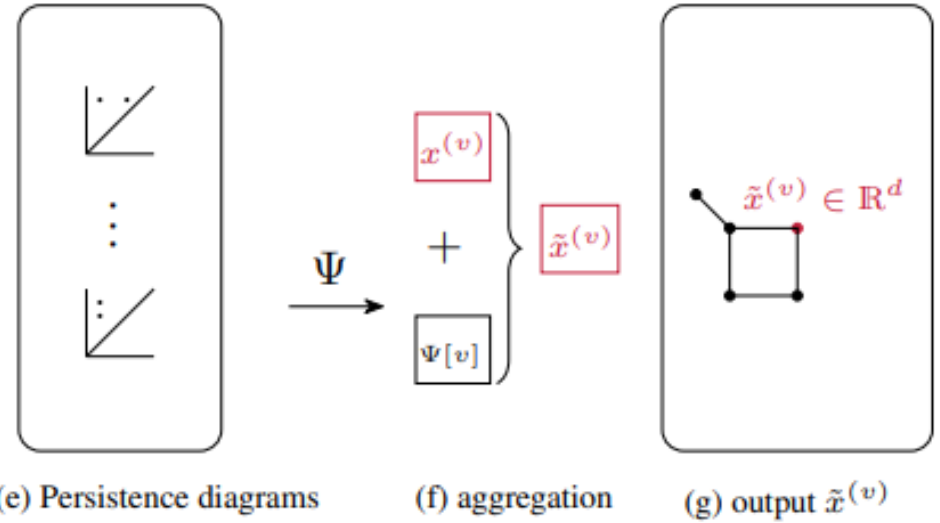
03 TOGL : A Topological Graph Layer

- Algorithmic details

❖ **Filtration** : each filtration function f_i can focus on **different properties** of the graph



(a) node attributes $x^{(v)}$ (b) node map Φ (c) k views of graph G (d) filtrations f_i



(e) Persistence diagrams (f) aggregation (g) output $\tilde{x}^{(v)}$

```
self.filtration_modules = torch.nn.ModuleList([
    torch.nn.Sequential(
        torch.nn.Linear(self.features_in, self.filtration_hidden),
        torch.nn.ReLU(),
        torch.nn.Linear(self.filtration_hidden, 1),
        final_filtration_activation
    ) for _ in range(num_filtrations)
])
```

$\Phi\colon \mathbb{R}^d \to \mathbb{R}^k$, an MLP with a single hidden layer, such that $f_i := \pi_i \circ \Phi$

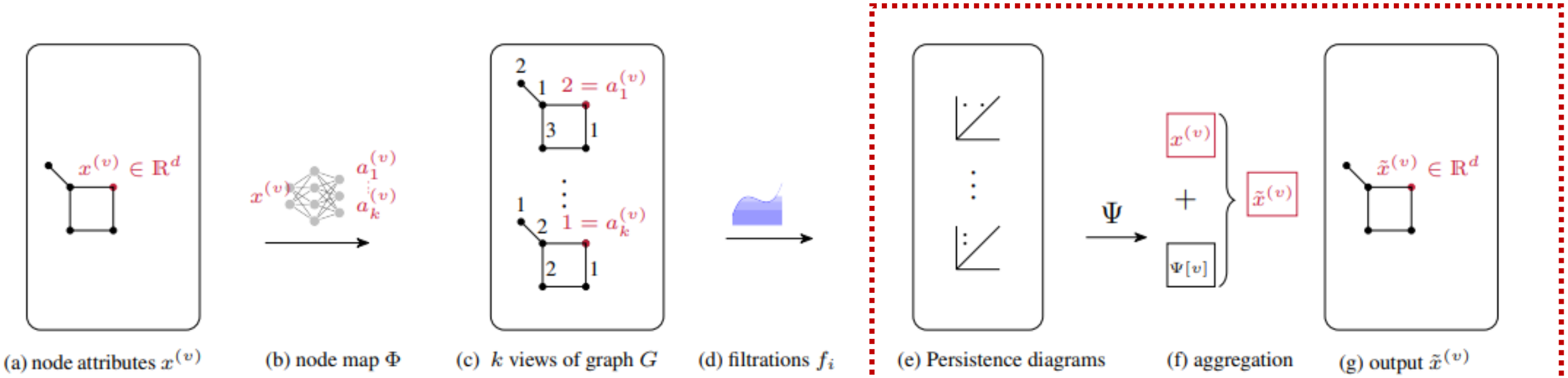
- K view : multi scale as a differently properties of graph

03 TOGL : A Topological Graph Layer

For dimension 0($D^{(0)}$), we have a bijective mapping of tuples in the persistence diagram to the vertices of the graph, which was previously exploited in topological representation learning

- Algorithmic details

❖ Output generation



(a) node attributes $x^{(v)}$ (b) node map Φ (c) k views of graph G (d) filtrations f_i (e) Persistence diagrams (f) aggregation (g) output $\tilde{x}^{(v)}$

$\Psi^{(0)}$ (i.e the topological embedding of vertex v).

$\Psi^{(1)}$ is pooled into a graph-level representation.

- Residual fashion : aggregation with the original node attribute vector $x^{(v)}$ $\tilde{x}^{(v)} = x^{(v)} + \Psi^{(0)}\left(\mathcal{D}_1^{(0)}, \ldots, \mathcal{D}_k^{(0)}\right)[v]$

03 TOGL : A Topological Graph Layer

The expressive power of TOGL hinges on the expressive power of the filtration—making it crucial that we can learn it…

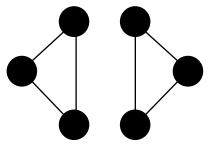
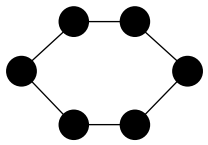
❖ *Differentiability & Expressive Power*

Theorem 1. *Let f_θ be a vertex filtration function $f_\theta : V \to \mathbb{R}$ with continuous parameters θ, and let Ψ be a differentiable embedding function of unspecified dimensions. If the vertex function values of f_θ are distinct for a specific set of parameters θ', i.e. $f_\theta(v) \neq f_\theta(w)$ for $v \neq w$, then the map $\theta \mapsto \Psi\left(\mathbf{ph}(G, f_\theta)\right)$ is differentiable at θ'.*

- Embedding function과 persistent diagram은 미분 가능하다 [자세한 증명 과정은 Hofer 논문과 appendix참고]

Theorem 2. *Persistent homology is* at least *as expressive as WL[1], i.e. if the WL[1] label sequences for two graphs G and G' diverge, there exists an injective filtration f such that the corresponding 0-dimensional persistence diagrams $\mathcal{D}_0$ and $\mathcal{D}'_0$ are not equal.*

- TOGL은 최소 1-WL test의 성능을 boundar로 가지고 있으며, 1-WL test가 구별하는 graph는 다 구별 가능하고, 1-WL test가 구별 못하는 non-isomorphism graph도 구별 가능하다.
- Filtration function이 injective를 반드시 만족하는 건 아니여서 $\tilde{f}$를 f로 근사 하는 걸 보인다고 함 [Appendix B]

G  G` 

04

Experiments

04 Experiments

❖ Experimental setup

Baseline : GCN, GAT, Gated-GCN, GIN, WL WL-OA

TOGL setup : 기존 존재하는 GNN 구조에 적용가능하고 second layer를 TOGL로 대체 [replace layer는 dataset에 따라 다름]

➔ 이 setup은 original, modified architectrue를 same parameter 수에 근사되게 같이 보장할 수 있음

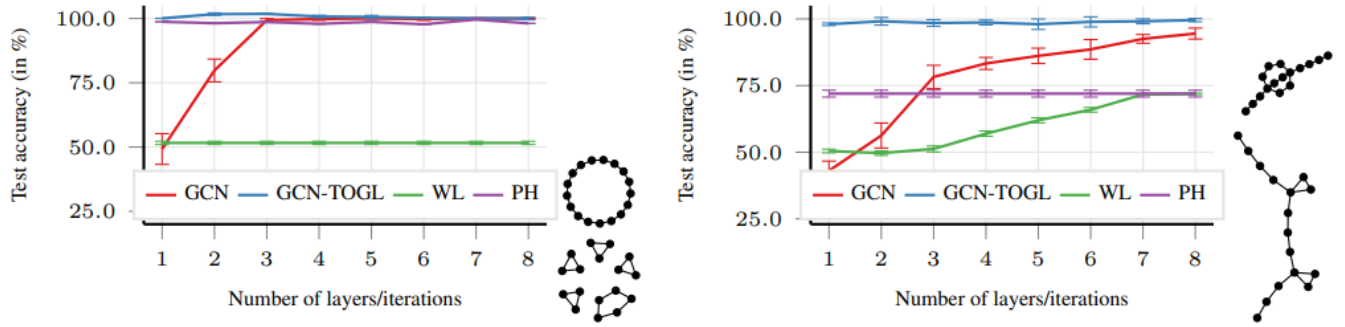
Table S6: The set of hyperparameters that we use to train TOGL, along with their respective value ranges. Notice that 'dropout' could be made configurable, but this would make our setup incomparable to the setup proposed by Dwivedi et al. (2020) for benchmarking GNNs.

NAME	VALUE(S)
`DeepSet`	{True, False}
Depth	{3, 4}
Dim1	True (by default, we always use cycle information)
Dropout	0.
Early Topo	(True, False)
Static	{True, False} (to evaluate the static variant)
Filtration Hidden Dimension	32
Hidden Dimension	138–146
No. coordinate functions	3
No. filtrations	8
Residual and Batch Norm	True
Share filtration parameters	True

04 Experiments

1. Performance on synthetic data sets

- Two synthetic balanced 2-class data set of 1000 graphs : **CYCLES** data set, **NECKLACES** data set
- CYCLES dataset은 connected component나 cycle수가 다르기 때매 어떤 topological 방법으로도 쉽게 구별
- 하지만 CYCLES의 경우 GCN은 최소 4개의 layer필요하고, NECKLACES는 더 많은 layer를 요구한다.
- WL은 애초에 실패하는 경우가 많음
- Cycle, connected component 둘다 중요하다



(a) CYCLES (b) NECKLACES

Figure 1: As a motivating example, we introduce two topology-based data sets whose graphs can be easily distinguished by humans; the left data set can be trivially classified by all topology-based methods, while the right data set necessitates *learnable* topological features. We show the performance of (i) a GCN with k layers, (ii) our layer TOGL (integrated into a GCN with $k-1$ layers): GCN-TOGL, (iii) the Weisfeiler–Lehman (WL) graph kernel using vertex degrees as the node features, and (iv) a method based on static topological features (PH). Next to the performance charts, we display examples of graphs of each class for each of the data sets.

04 Experiments

2. Structure-based graph and node classification performance

- 2020 survey논문을 보면 node feats가 substantial information을 가져다 준다고 얘기함 -> 하지만 graph structure자체에 의해 전달되는 신호를 어느정도 억제(?)
- 그래서 node feats에 uninformative한 정보를 넣어서 topological 정보만으로 잘 되는지 확인하고자 함 : TOGL의 장점을 보여줄 수 있다.
- 하지만 Proteins, Pattern dataset은 TOGL활용시 성능은 떨어짐, 특히 GAT에서 많이 떨어지는데 filteration 구성이 backbon에 의존하기 때문에 놀라운 결과는 아님 [논문에서 GAT 학습이 어려웠다고 말함...]
- 분류 성능 향상을 위해 추가 구조정보를 사용하는데 있어 TOGL의 유용성 보여줌

Table 1: Results for the structure-based experiments. We depict the test accuracy obtained on various benchmark data sets when only considering structural information (i.e. the network has access to *uninformative* node features). For MOLHIV, the ROC-AUC is reported. Graph classification results are shown on the left, while node classification results are shown on the right. We compare three architectures (GCN-4, GIN-4, GAT-4) with corresponding models where one layer is replaced with TOGL and highlight the winner of each comparison in **bold**.

	Graph classification					*Node classification*
METHOD	DD	ENZYMES	MNIST	PROTEINS	MOLHIV	PATTERN
GCN-4	68.0±3.6	22.0±3.3	76.2±0.5	68.8±2.8	66.4±1.8	85.5±0.4
GCN-3-TOGL-1	**75.1±2.1**	**30.3±6.5**	**84.8±0.4**	**73.8±4.3**	**69.4±1.8**	**86.6±0.1**
GIN-4	75.6±2.8	21.3±6.5	83.4±0.9	**74.6±3.1**	**68.7±0.9**	84.8±0.0
GIN-3-TOGL-1	**76.2±2.4**	**23.7±6.9**	**84.4±1.1**	73.9±4.9	65.1±6.2	**86.7±0.1**
GAT-4	63.3±3.7	21.7±2.9	63.2±10.4	67.5±2.6	51.8±5.6	**73.1±1.9**
GAT-3-TOGL-1	**75.7±2.1**	**23.5±6.1**	**77.2±10.5**	**72.4±4.6**	**68.6±1.7**	59.6±3.3

04 Experiments

2. Structure-based graph and node classification performance

- TOGL은 layer가 적어도 성능이 좋다 : layer 적으니까 over smoothing risk 가 줄어듦
- Appendix 1에 persistence diagrams을 embedding하는 functio비교 분석을 했다고 함 : 논문에서는 deepset 사용

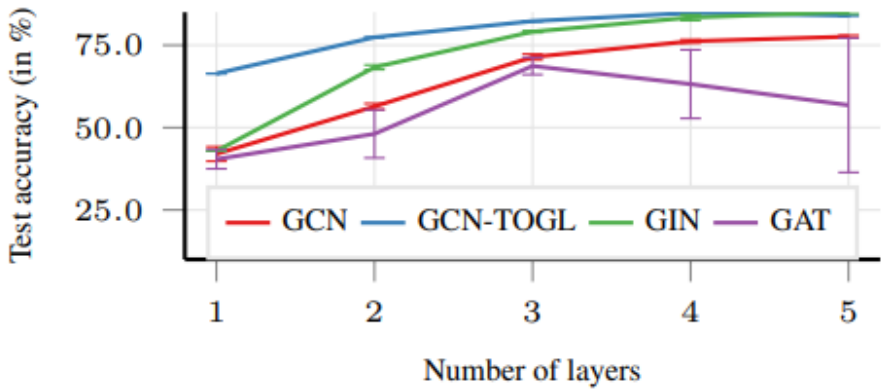
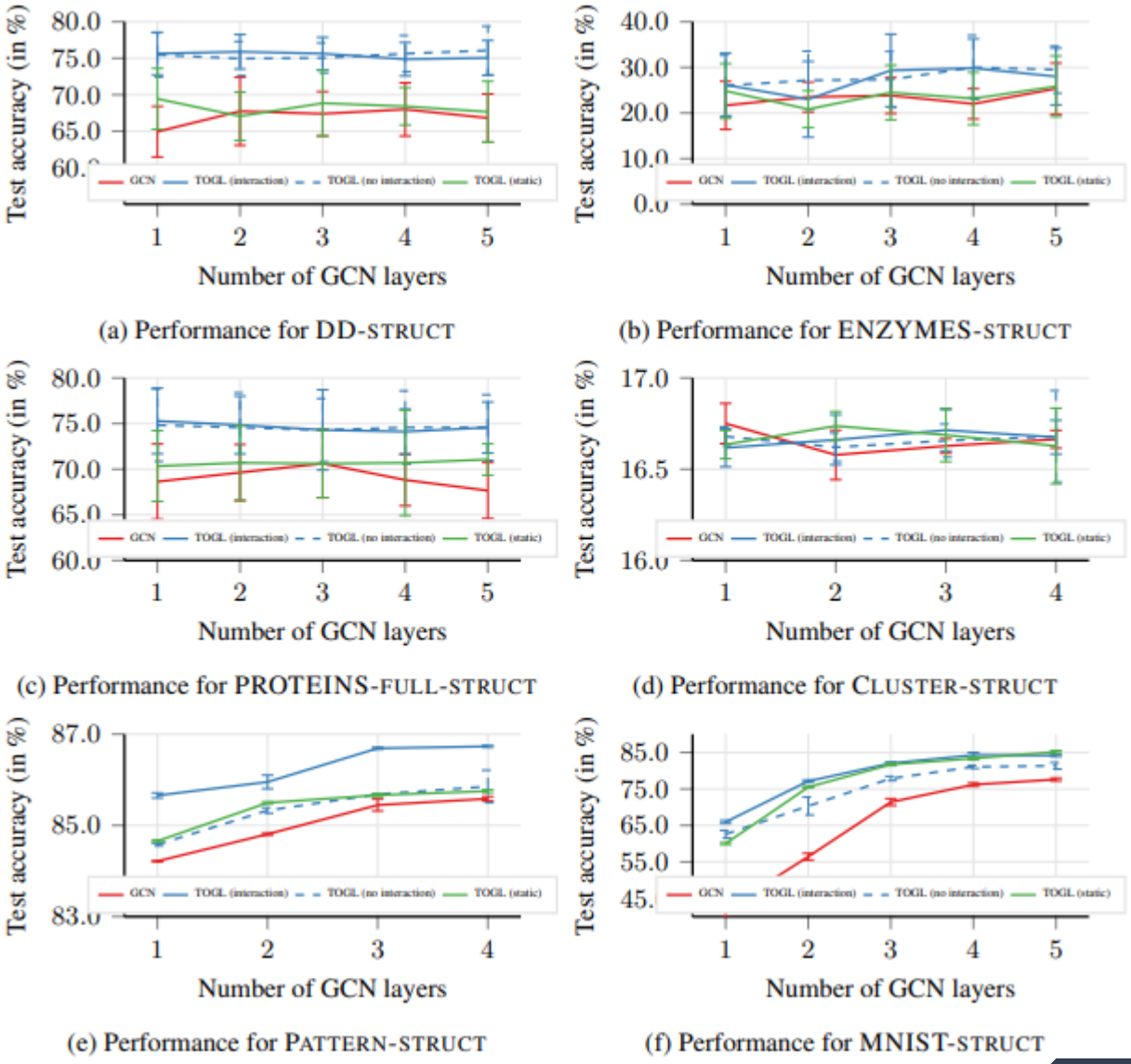


Figure 3: Classification performance when analysing the structural variant of MNIST.



(a) Performance for DD-STRUCT

(b) Performance for ENZYMES-STRUCT

(c) Performance for PROTEINS-FULL-STRUCT

(d) Performance for CLUSTER-STRUCT

(e) Performance for PATTERN-STRUCT

(f) Performance for MNIST-STRUCT

04 Experiments

3. Performance on benchmark data sets

- Standard graph and node classification task결과 분석 [well-known benchmarks dataset]
- GCN-4, GIN-4, GAT-4보다 대부분 좋은 결과를 보여줌 : 다만 ENZYMES성능에 대해서느 overfitting 영향이 좀 있다고 함, 떨어지는 결과도 있음
- GAT는 variance가 심하다. 그래도 TOGL추가시 대부분 올르긴 함
- Grey : cited from the literature, black : run in our setup

Table 2: Test accuracy on benchmark data sets (following standard practice, we report weighted accuracy on CLUSTER and PATTERN). Methods printed in black have been run in our setup, while methods printed in grey are cited from the literature, i.e. Dwivedi et al. (2020), Morris et al. (2020) for IMDB-B and REDDIT-B, and Borgwardt et al. (2020) for WL/WL-OA results. Graph classification results are shown on the left; node classification results are shown on the right. Following Table 1, we take existing architectures and replace their second layer by TOGL; we use *italics* to denote the winner of each comparison. A **bold** value indicates the overall winner of a column, i.e. a data set.

	Graph classification							*Node classification*
METHOD	CIFAR-10	DD	ENZYMES	MNIST	PROTEINS-FULL	IMDB-B	REDDIT-B	CLUSTER
GATED-GCN-4	**67.3±0.3**	72.9±2.1	65.7±4.9	**97.3±0.1**	**76.4±2.9**	—	—	**60.4±0.4**
WL	—	77.7±2.0	54.3±0.9	—	73.1±0.5	71.2±0.5	78.0±0.6	—
WL-OA	—	**77.8±1.2**	58.9±0.9	—	73.5±0.9	74.0±0.7	87.6±0.3	—
GCN-4	54.2±1.5	72.8±4.1	**65.8±4.6**	90.0±0.3	*76.1±2.4*	68.6±4.9	**92.8±1.7**	57.0±0.9
GCN-3-TOGL-1	*61.7±1.0*	*73.2±4.7*	53.0±9.2	95.5±0.2	76.0±3.9	*72.0±2.3*	89.4±2.2	*60.4±0.2*
GIN-4	54.8±1.4	70.8±3.8	*50.0±12.3*	*96.1±0.3*	72.3±3.3	72.8±2.5	81.7±6.9	58.5±0.1
GIN-3-TOGL-1	*61.3±0.4*	*75.2±4.2*	43.8±7.9	96.1±0.1	*73.6±4.8*	**74.2±4.2**	89.7±2.5	*60.4±0.2*
GAT-4	57.4±0.6	71.1±3.1	26.8±4.1	94.1±0.3	71.3±5.4	*73.2±4.1*	44.2±6.6	56.6±0.4
GAT-3-TOGL-1	*63.9±1.2*	*73.7±2.9*	*51.5±7.3*	*95.9±0.3*	*75.2±3.9*	70.8±8.0	*89.5±8.7*	*58.4±3.7*

04 Experiments

4. Comparison to other topology-based algorithms

- 다른 topology-based 방법들과 비교
- Detail은 Appendix D
- Static : topological features를 고려하지 않고 paramters수 측면에서 filtration 계산을 mimics [정확히 무슨 의미인지는 모르겠음]←→ dynamic one based on persistent homology

-> 무작위로 edge를 선택하는데, 이는 엄격한 topology 기반 대신 node features values에 기반한 graph-level information을 학습하는 효과가 있다고 함

-> persistent homology가 빠진것은 아니고 edge 선택 방법이 다른 것임

Table 3: Test accuracy when comparing TOGL (integrated into a simplified architecture) with existing topology-based embedding functions or `readout` functions. Results shown in grey are cited from the respective papers (Carrière et al., 2020; Hofer et al., 2020). For GFL, we cite degree-based results so that its performance values pertain to the same scenario.

METHOD	REDDIT-5K	IMDB-MULTI	NCI1	REDDIT-B	IMDB-B
GFL	55.7±2.1	49.7±2.9	71.2±2.1	90.2±2.8	**74.5±4.6**
PersLay	55.6	48.8	73.5	—	71.2
GCN-1-TOGL-1	**56.1±1.8**	**52.0±4.0**	**75.8±1.8**	90.1±0.8	74.3±3.6
GCN-1-TOGL-1 (static)	55.5±1.8	48.3±4.9	75.1±1.2	**90.4±1.4**	72.2±2.1

05

Conclusion

05 Conclusion

- We presented TOGL, a generically-applicable layer that incorporates topological information into any GNN architecture
- We proved that TOGL, due to its filtration functions (i.e. input functions) being learnable, is more expressive than WL[1], the Weisfeiler-Lehman test for graph isomorphism, and therefore also increases expressivity of GNNs
- We also saw that the choice of function for embedding topological descriptors is crucial, with embedding functions that can handle interactions between individual topological features typically performing better than those that cannot.
- 하지만, topological information은 때때로 overfitting을 불러오기 때문에 추가적인 regularization이 필요하며, 이는 future work
- 추가적으로 다른 유형의 filtraion과 advanced persistent homology algorithms을 사용하면, 예측 성능에 도움이 될 거라는 가설을 세움