GNN Explainer : Generating Explanations for Graph Neural Networks

탁해성 (tok33@pusan.ac.kr)
ALgorithm and Data Engineering Lab.

1. Graph Neural Network

- Graph Data

Normal DataFrame

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
2	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
3	Jordan Mickey	Boston Celtics	NaN	PF	21.0	6-8	235.0	LSU	1170960.0
4	Terry Rozier	Boston Celtics	12.0	PG	22.0	6-2	190.0	Louisville	1824360.0
5	Jared Sullinger	Boston Celtics	7.0	C	NaN	6-9	260.0	Ohio State	2569260.0
6	Evan Turner	Boston Celtics	11.0	SG	27.0	6-7	220.0	Ohio State	3425510.0

Independence

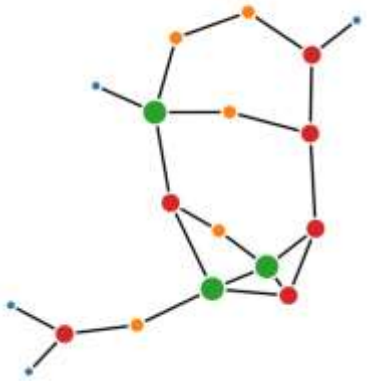
+

Adjacency Matrix

	A	B	C	D	E	F
A	0	1	0	1	1	0
B	1	0	1	0	1	0
C	0	1	0	0	0	1
D	1	0	0	0	0	0
E	1	1	0	0	0	1
F	0	0	1	0	1	0

=

Graph Data

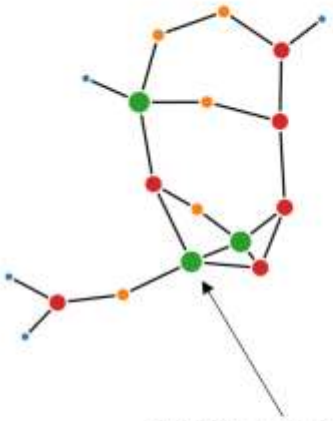


Dependence

1. Graph Neural Network

- Graph Data로 할 수 있는 것은?

Task 1: Node Classification

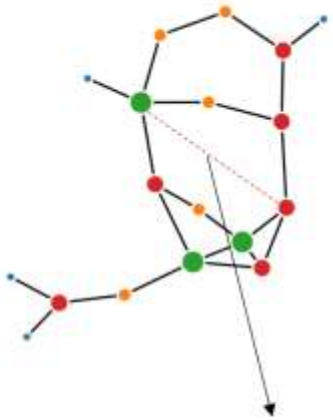


초록색의 Label은 무엇일까요?

사용 예시:

Social Network에서 각 사용자의 중요도 / 역할

Task 2: Link (Edge) Prediction



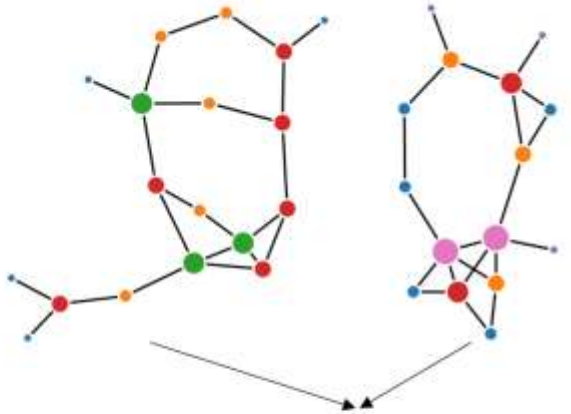
- 해당 노드들은 연결 되어 있을까요?
- 해당 노드들 간의 Edge Value는 어떻게 되는가?

⇒ 0이라면 연결 X / 0 이상이라면 연결

사용 예시:

추천 시스템의 사용자와 상품과의 새로운 연결

Task 3: Graph Classification



해당 그래프 자체의 Label은 어떻게 될까요?

사용 예시:

단백질 구조를 관찰하여 어떤 단백질인지 예측

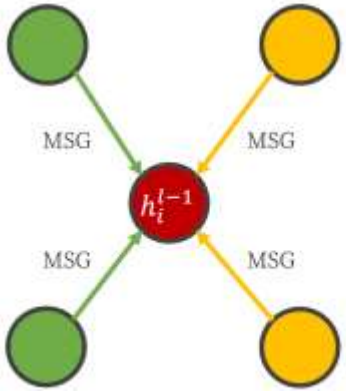
1. Graph Neural Network

- Basic GNN Operation

1) Message

이웃이 중심 노드에게 전달하는 정보

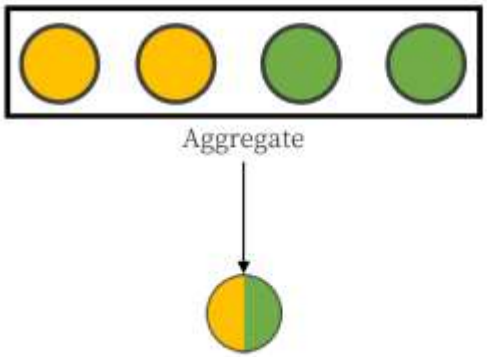
$$m_{ij}^{l} = MSG(h_i^{l-1}, h_j^{l-1}, r_{ij})$$



2) Aggregate

이웃들로부터 온 정보들을 취합

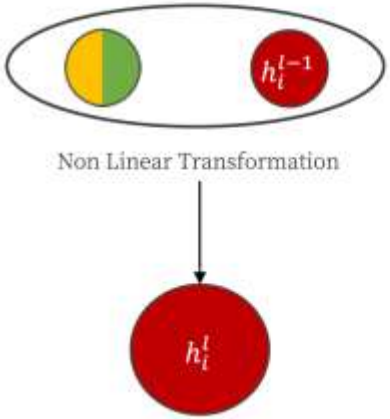
$$M_i^{l} = AGG(\{m_{ij}^{l} | v_j \in \mathcal{N}_{v_i}\})$$



3) Update

이웃 정보와 자기 자신의 정보를 합쳐 Embedding Update

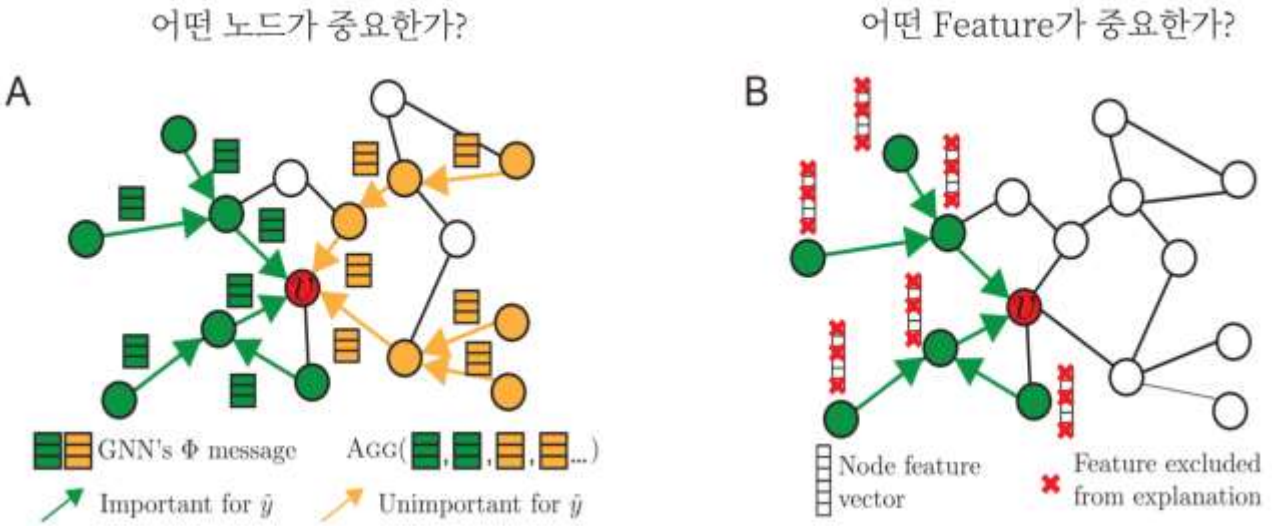
$$h_i^{l} = UPDATE(M_i^{l}, h_i^{l-1})$$



2. GNNExplainer

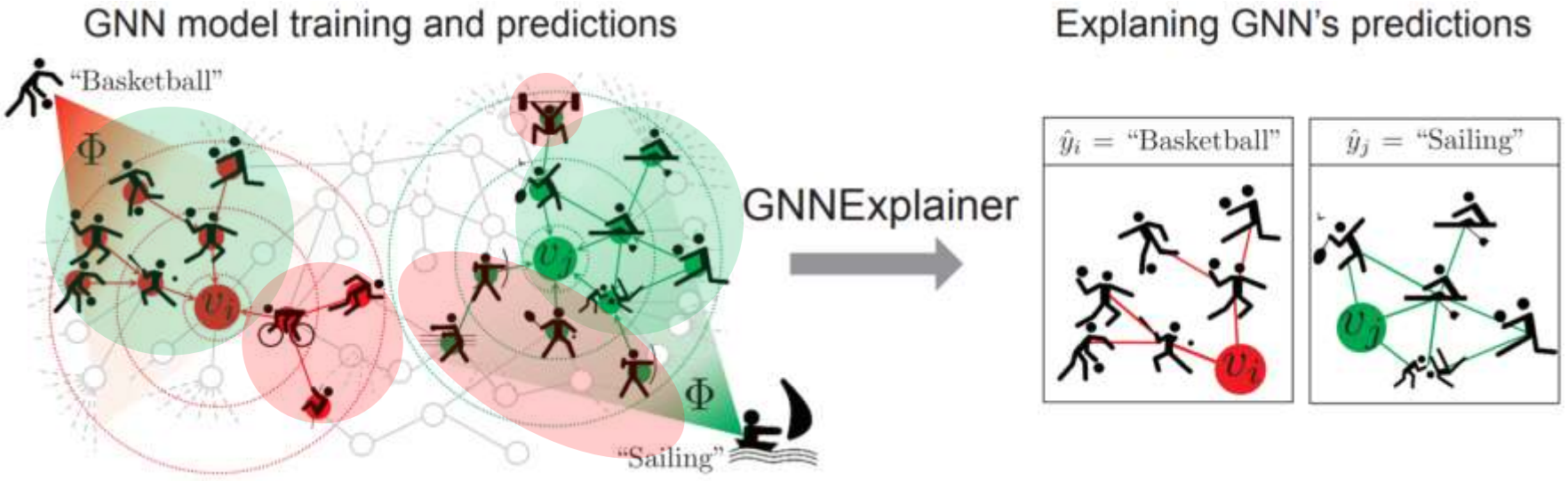
- GNNExplainer는 GNN의 Prediction을 설명하기 위한 기법
  - 훈련된 GNN과 Prediction을 가지고 특정 노드나 Class에 대하여 Subgraph를 구성
  - 노드를 특정하는 것 이외에, 노드의 어떤 Feature의 영향력이 큰지에 대해 분석 가능

- GNNExplainer의 해결 목표
  - 하나의 노드를 구성함에 있어 어떤 노드들이 영향을 미쳤는지? (Single-instance explanations)
  - 한 Label을 갖는 노드들에 대해, 어떤 노드들이 영향을 미쳤는지? (Multi-instance explanations)
  - 노드 내의 어떤 Feature가 영향을 미쳤는지?



2. GNNExplainer

- 특정 노드가 특정 Label로 분류되었을 때, 주위의 어떤 노드가 영향을 미쳤는가?

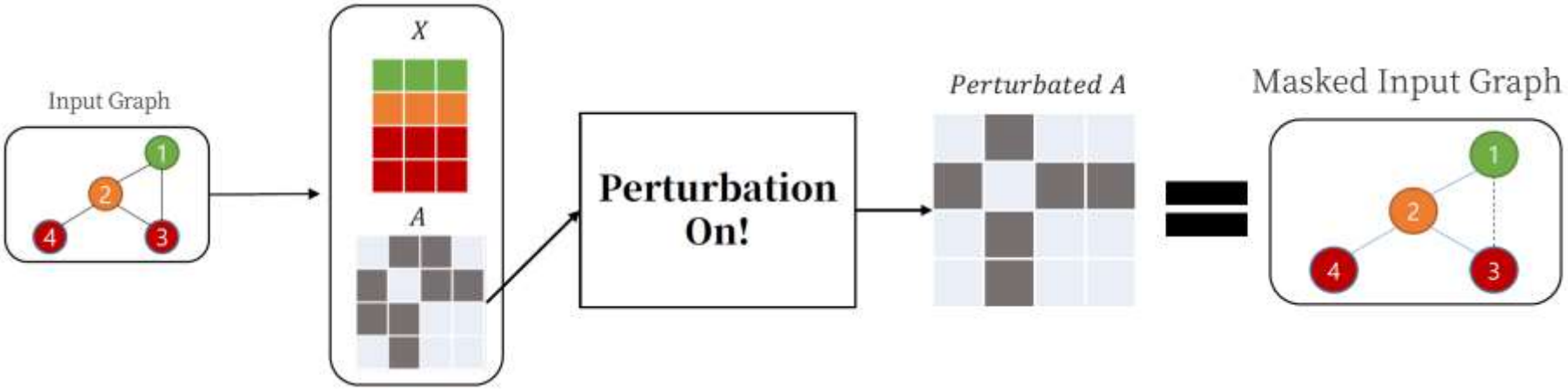


Pre-Trained Model의 Prediction

Prediction에 큰 영향을 주는 이웃 노드
Explanation = Subgraph

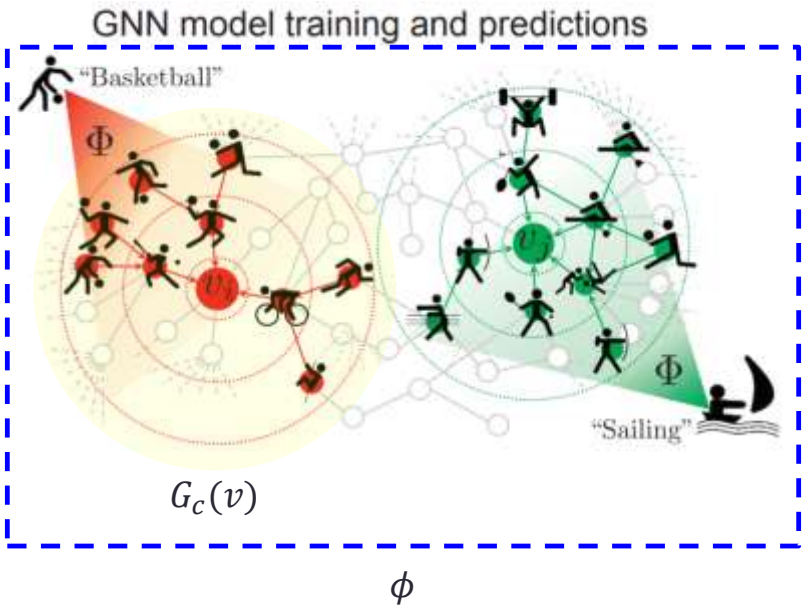
2. GNNExplainer

- Subgraph를 만드는 방법
  - 기존 Graph에 대해 Perturbation Mask를 적용하여 일부 에지를 삭제



3. GNNExplainer

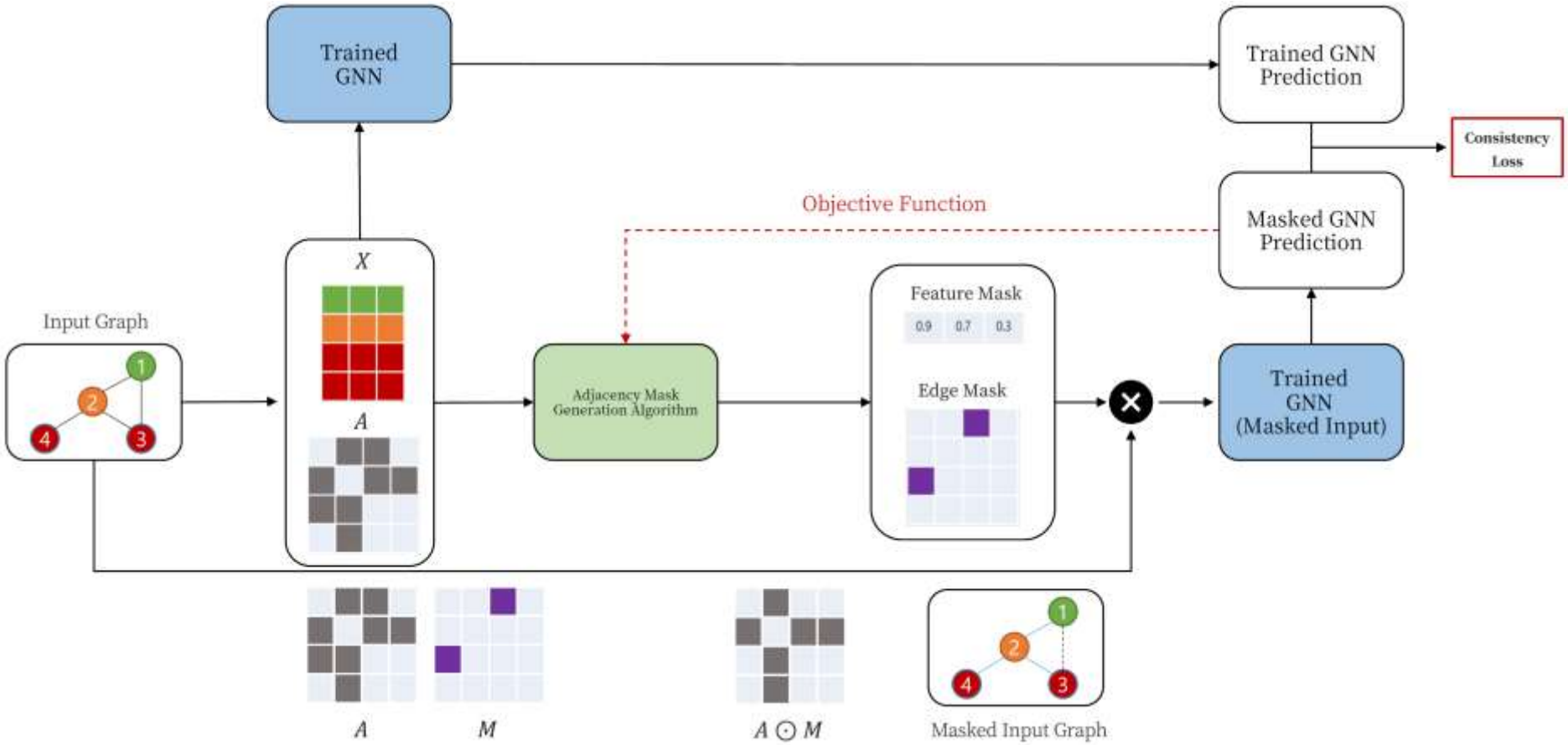
- Problem Formulation



Notation	Meaning
$G_c(v)$	Node v를 구성하기 위한 Computation Graph
$A_c(v) \in \{0,1\}^{n \times n}$	v와 연결되어 있는 노드들의 Adjacency Matrix
$X_c(v) = \{x_j \mid v_j \in G_c(v)\}$	Neighbors Feature Set
Φ	(Trained) GNN Model
$P_\Phi(Y \mid G_c, X_c)$	Computation Graph를 통해 구성한 모델의 Label에 대한 확률
Y	(Trained) Model Predicted Y

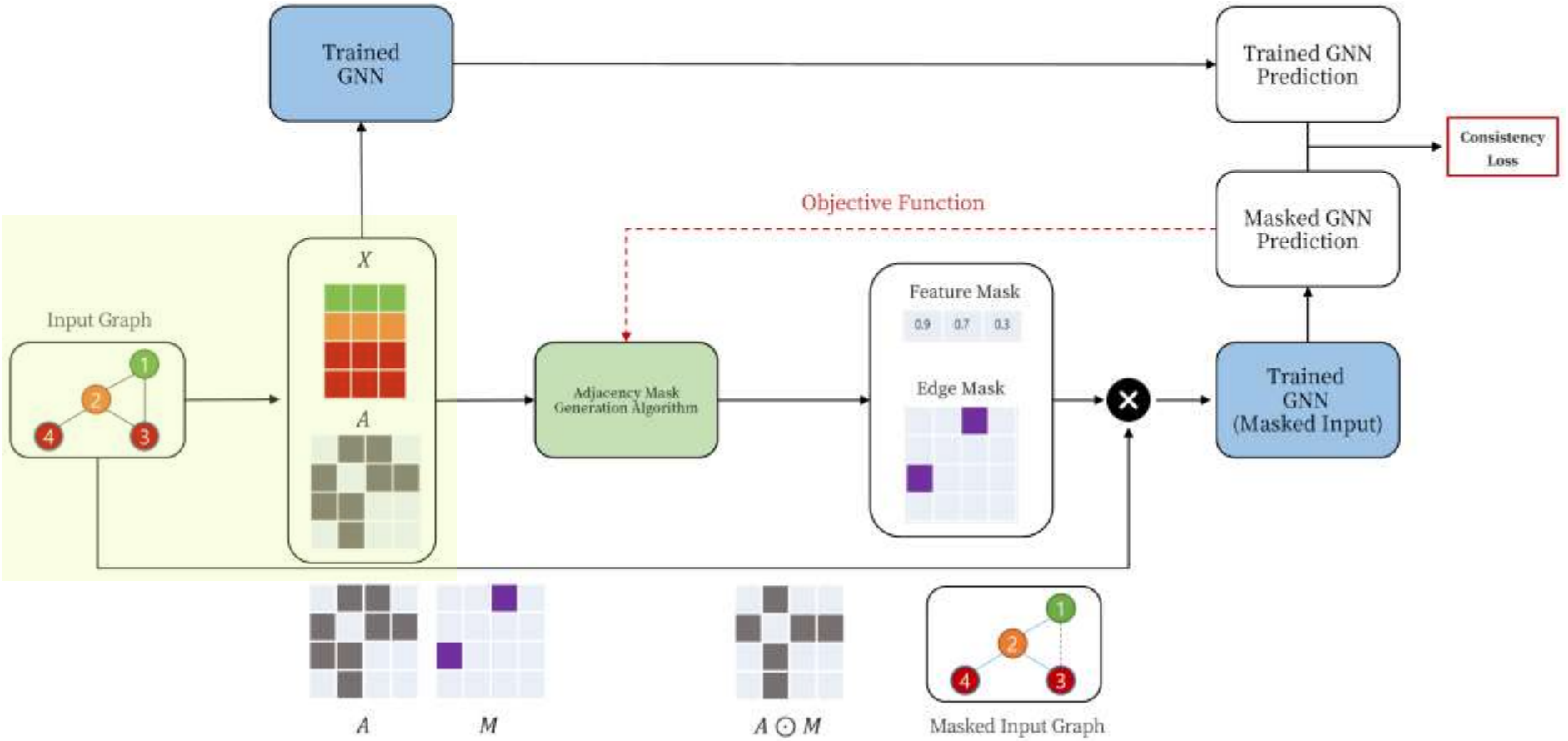
4. GNNExplainer - Overview

- 모델 학습 과정



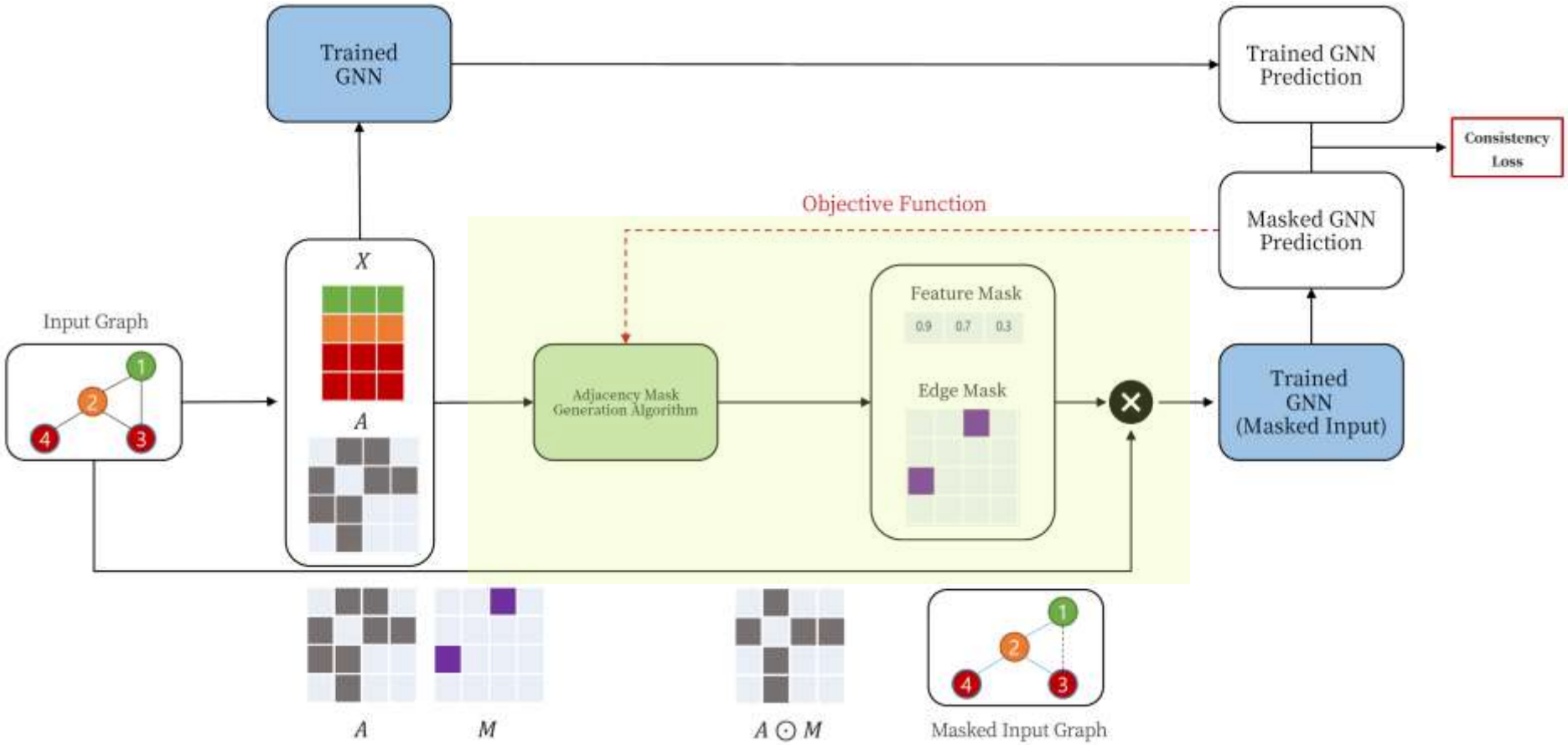
4. GNNExplainer - Overview (1)

- Input Graph의 X / A 준비



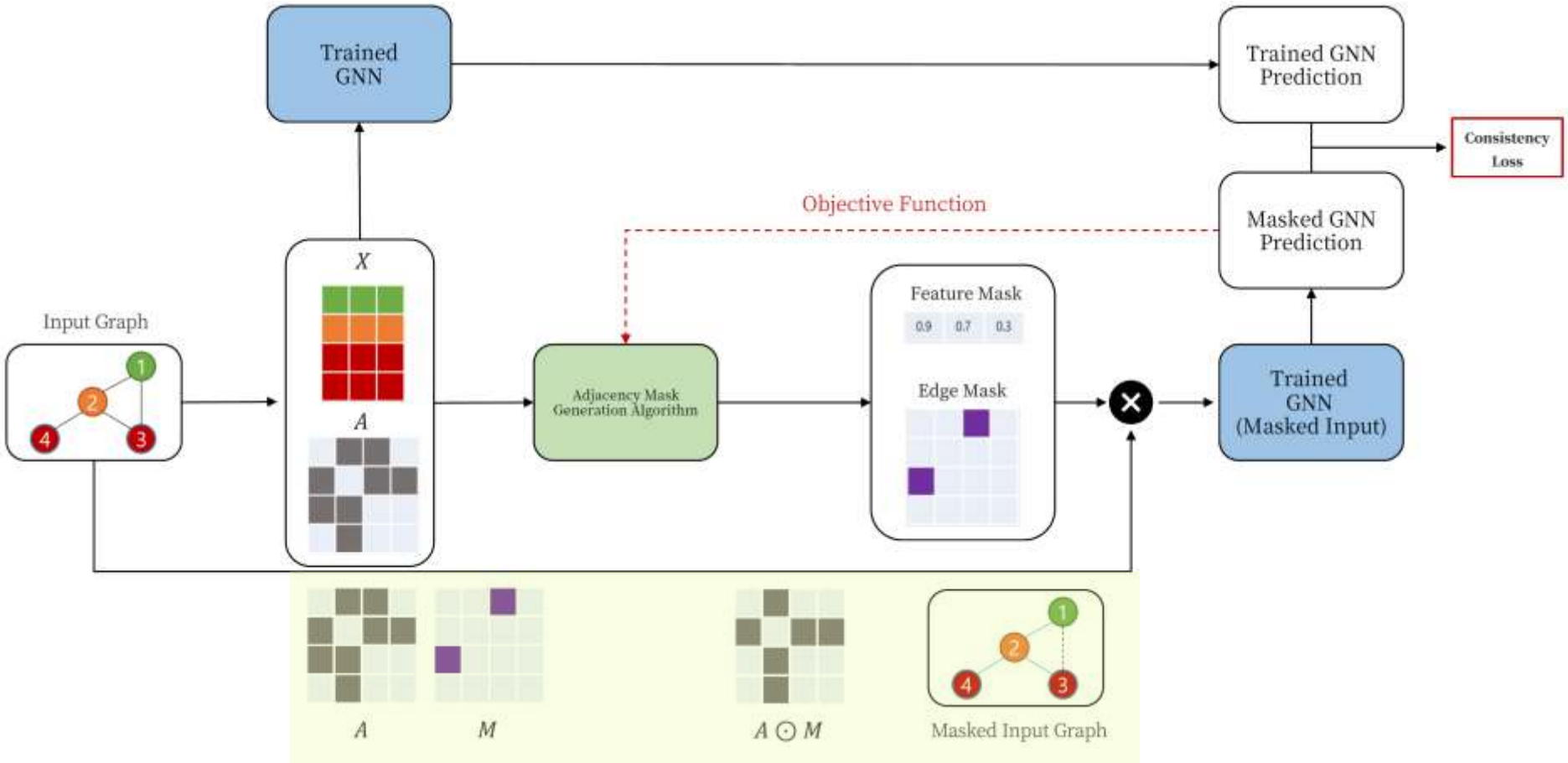
4. GNNExplainer - Overview (2)

- Adjacency / Feature Mask 생성



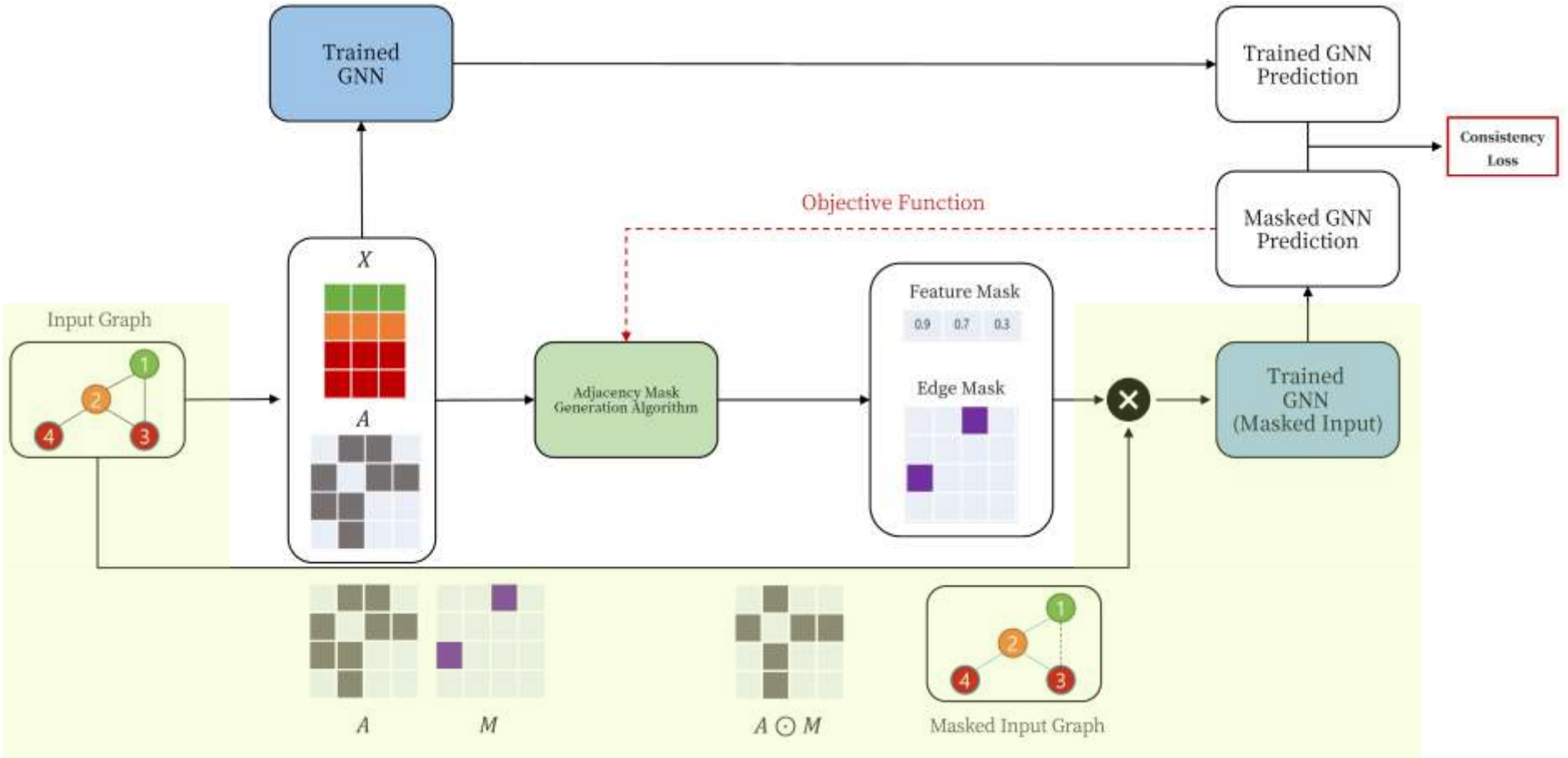
4. GNNExplainer - Overview (3)

- Adjacency Mask를 통해 Masked Input Graph 생성



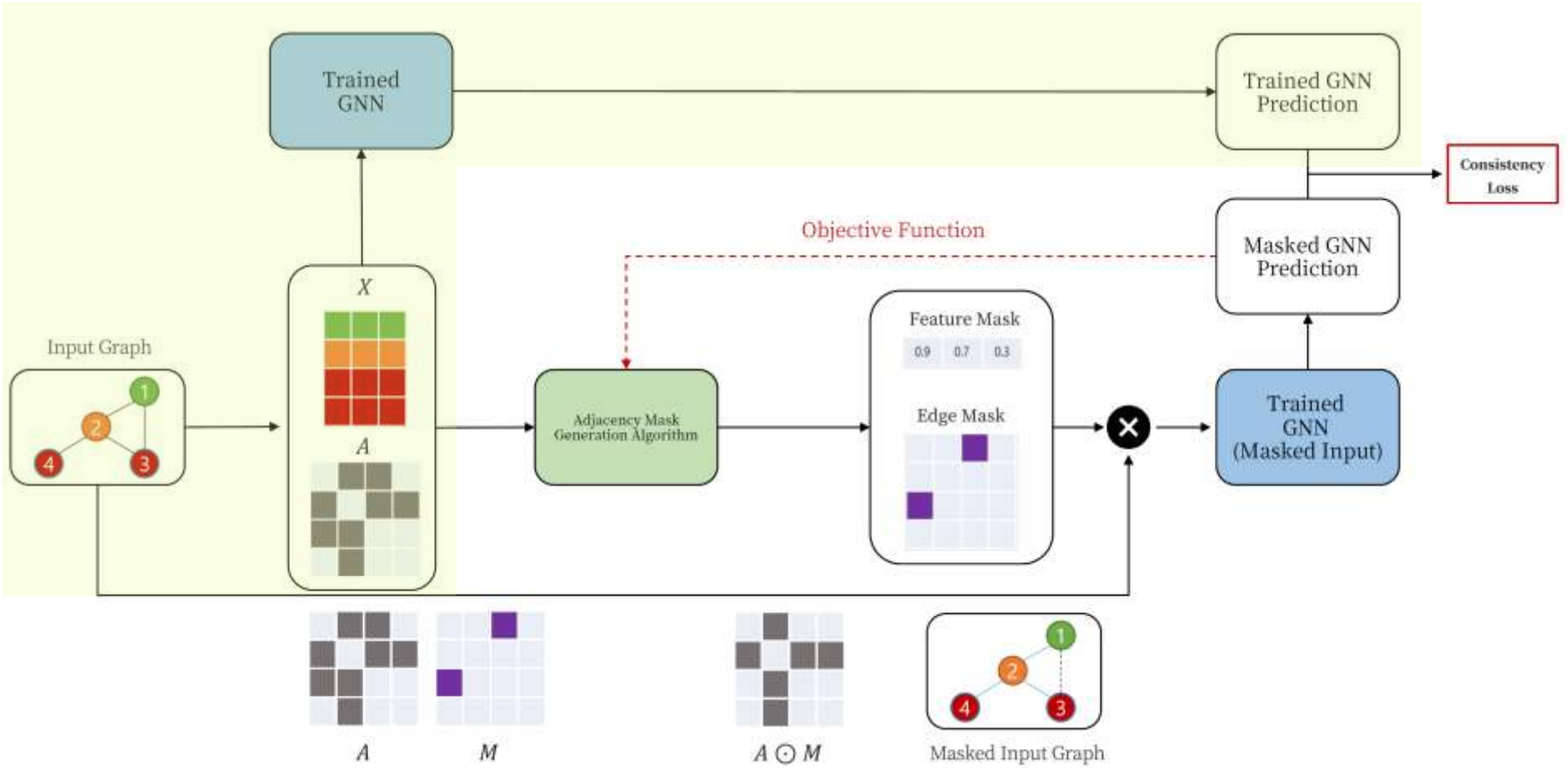
4. GNNExplainer - Overview (4)

- Masked Input Graph를 통해 Prediction 획득



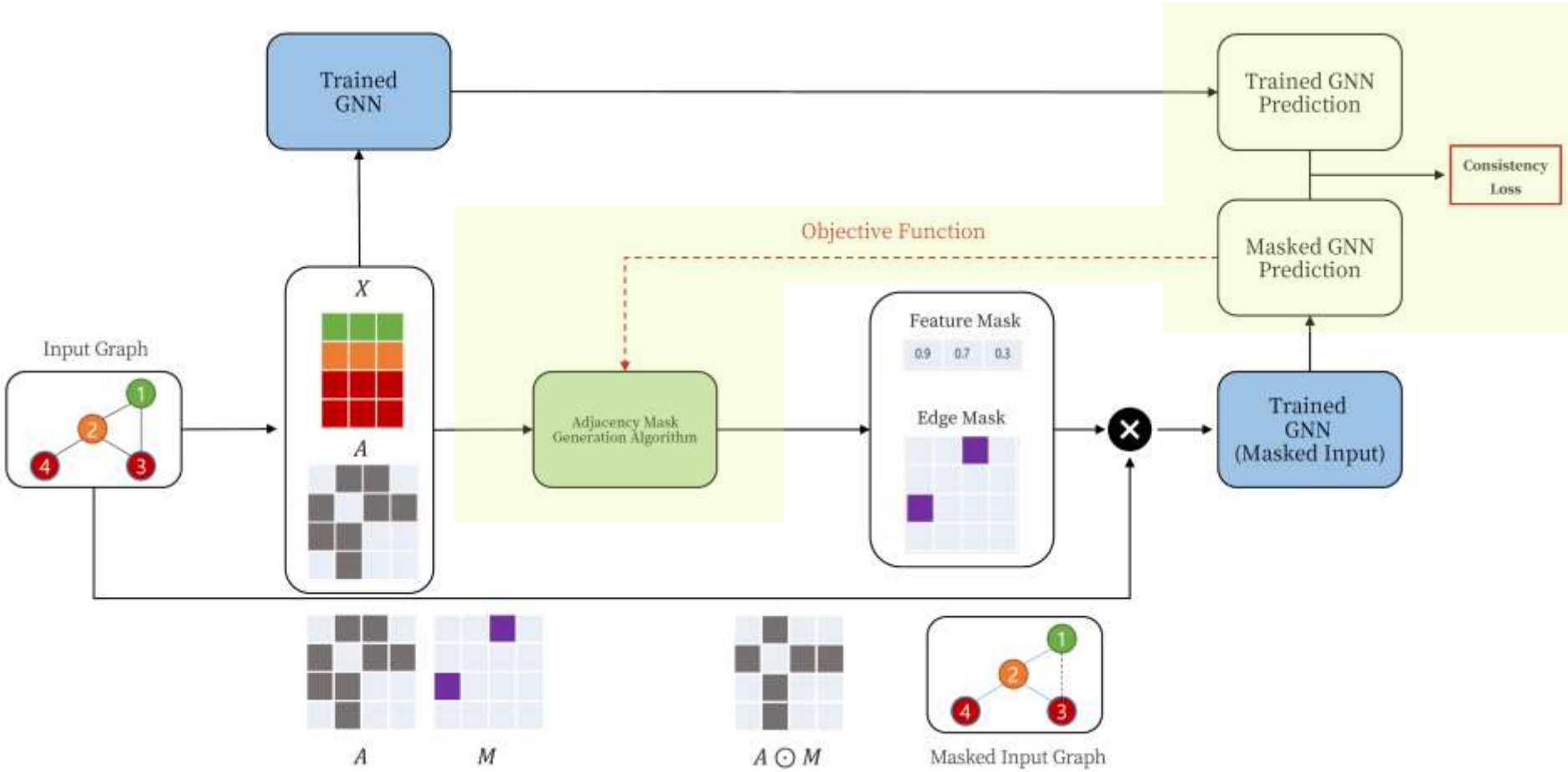
4. GNNExplainer - Overview (5)

- 기존 Input을 Trained GNN에 넣어 Prediction 획득



4. GNNExplainer - Overview (6)

- Objective Function을 통해 Mask 생성 알고리즘을 갱신



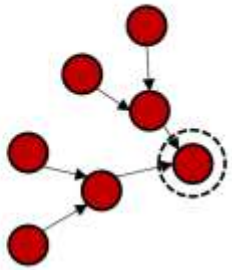
5. How Many Instances

- Adjacency Mask Generation Algorithm
  - Single Instance explanations
    - 단일 Instance에 대한 Subgraph 탐색
  - Multi Instance explanations
    - 하나의 Label c를 갖는 모든 Instance에 대한 Subgraph 탐색

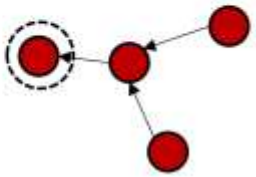
6. Single Instance explanations

- 단일 노드 v에 대하여,
  - Computation Graph G_c내에 포함된 Subgraph G_s 중 Mutual Information이 가장 큰 G_s 선택

G_s Candidate #1



G_s Candidate #2



$$\max_{G_S} MI\big(Y,(G_S,X_S)\big) = H(Y) - H(Y|G=G_S,X=X_S)$$

Mutual Information 의미
- 두 변수의 상호 종속 여부
- MI (a, b) : 독립일때와 종속일때의 두 변수의 결합확률 차이
- MI가 클수록 종속적이며 관계성이 큼

$$\min H(Y|G=G_S,X=X_S) = -E_{(Y|G_S,X_S)}[logP_\Phi(Y|G=G_S,X=X_S)]$$

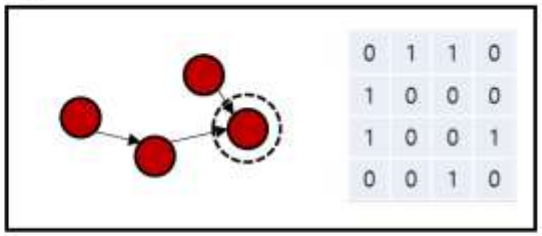
Entropy를 최소화하는 것은 Model ϕ의 불확실성을 줄이는 것
- $H(Y|G=G_S,X=X_S)$ 는 Subgraph G_s / Feature X_s 를 가질 때 Y의 불확실성

G_s의 사이즈를 K_M으로 한정하여 적당한 크기로 유지

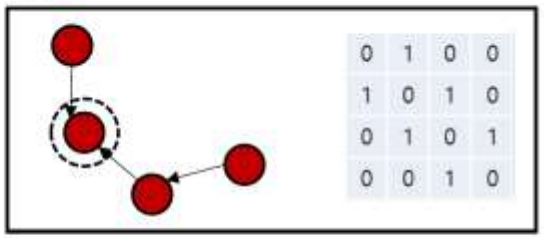
6. Single Instance explanations

- $\min H(Y|G = G_S, X = X_S) = -E_{(Y|G_S, X_S)}[logP_\Phi(Y|G = G_S, X = X_S)]$
  - 문제점
    - Computation Graph G_c내에 포함된 Subgraph G_s 후보가 매우 많아, Optimization 불가

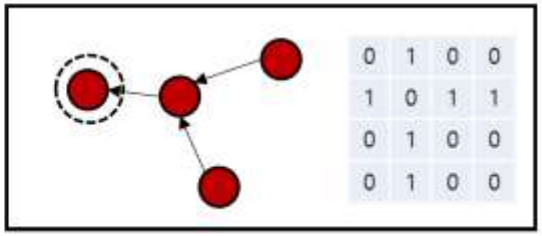
G_s Candidate #1



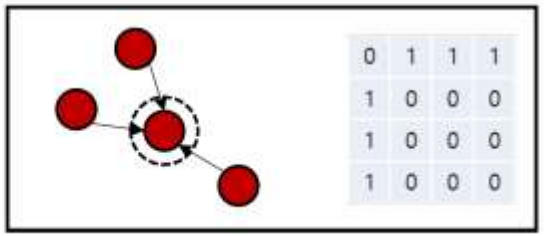
G_s Candidate #3



G_s Candidate #2

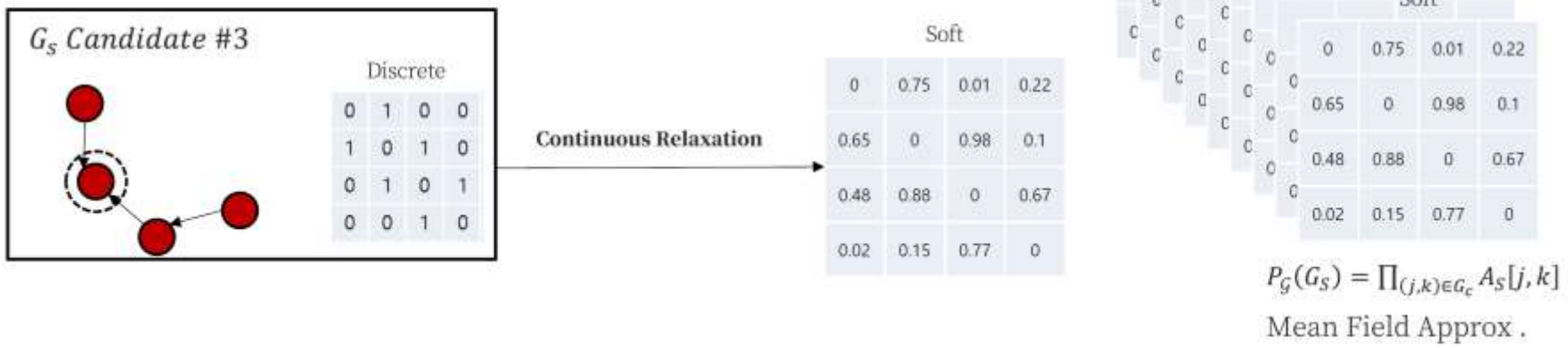


G_s Candidate #4



7. GNNExplainer Optimization Framework

- $H(Y|G = G_S, X = X_S) = -E_{(Y|G_S, X_S)}[logP_\Phi(Y|G = G_S, X = X_S)]$
  - 문제점
    - Computation Graph G_c내에 포함된 Subgraph G_s 후보가 매우 많아, Optimization 불가
  - $\min_{\mathcal{G}} E_{G_S \sim \mathcal{G}} H(Y|G = G_S, X = X_S) \leq \min_{\mathcal{G}} H(Y|G = E_{\mathcal{G}}[G_S], X = X_S)$
    - 개별적 G_s의 Entropy 기대 값 대신, G_s의 기대 값을 생성



7. GNNExplainer Optimization Framework

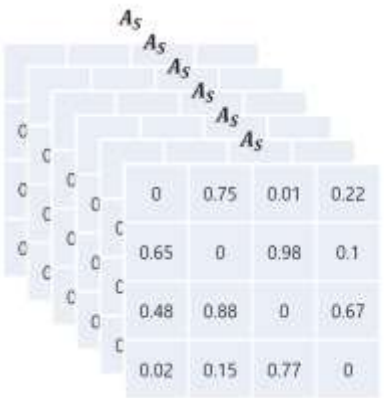
- Mean Field Variational Approximation
  - Computation Graph 에서 Random Subgraph들을 Multivariate Bernoulli 분포로 표현
  - Mean Field Variational Approximation으로 독립적인 를 모두 곱해, 결합 분포 구성
  - 결합 분포의 기대 값을 구하는 과정을 Masking으로 대체

Mean Field Variational Approximation Expectation

$$E[\log(q(z_{1:m}))] = \sum_{j=1}^{m} E_j[\log q(z_j)]$$

$$P_{\mathcal{G}}(G_S) = \prod_{(j,k)\in G_c} A_S[j,k]$$

$$\min_{\mathcal{G}} H(Y|G = E_{\mathcal{G}}[G_S], X = X_S) \longrightarrow \min_{\mathcal{G}} H(Y|G = A_c \odot \sigma(M), X = X_S)$$



G_S

0	0.75	0.01	0.22
0.65	0	0.98	0.1
0.48	0.88	0	0.67
0.02	0.15	0.77	0

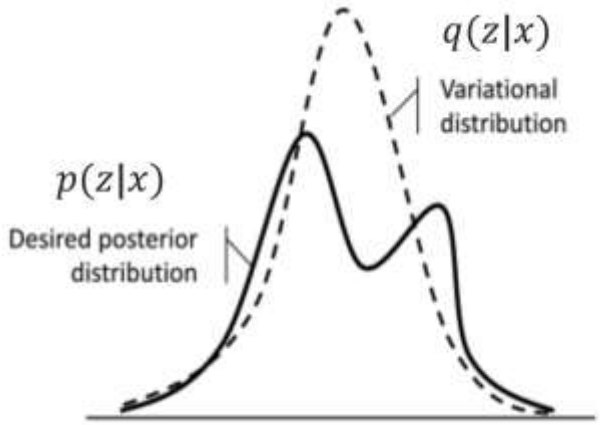
A_c

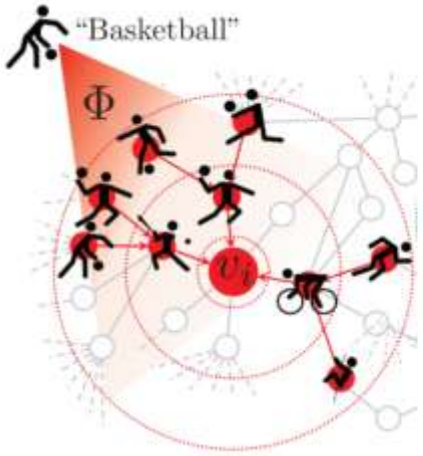
0	1	0	0
1	0	1	0
0	1	0	1
0	0	1	0

M

7. GNNExplainer Optimization Framework

- 왜 Variational Inference가 포함되어 있을까?
  - Variational Inference
    - 원 데이터 분포의 Intractable Latent Distribution $p(z|x)$를 찾기 위해,
    - Tractable Latent Distribution $q(z|x)$를 탐색하는 과정
  - GNNExplainer
    - Computation Graph G_c의 최적의 Explanation을 찾기 위해,
    - Tractable한 G_s를 탐색하는 과정





7. GNNExplainer Optimization Framework

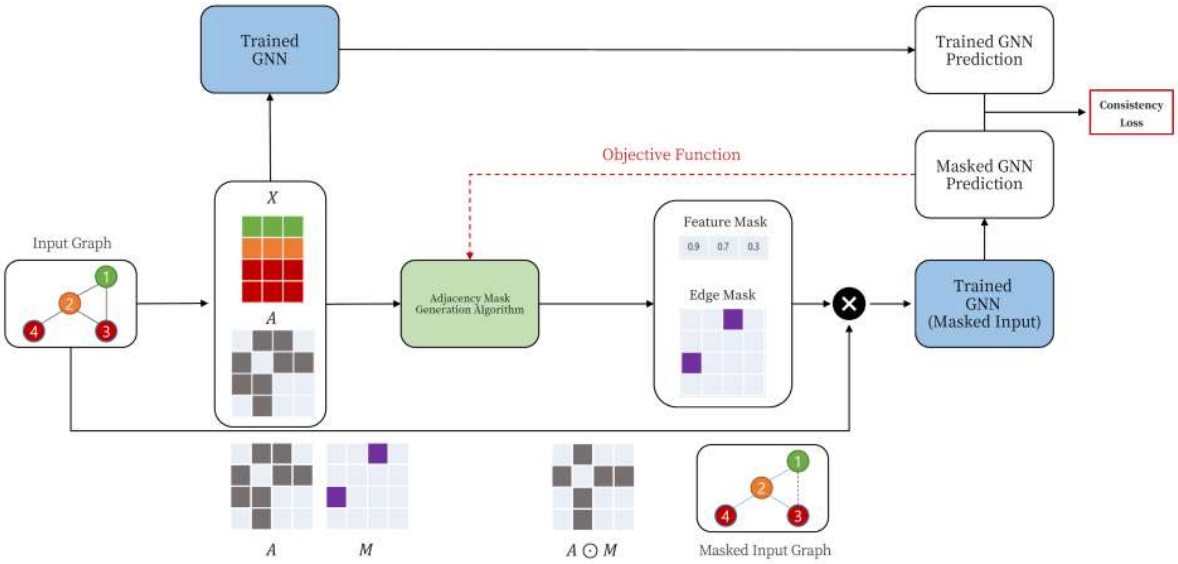
Single Instance explanations : 단일 Instance에 대한 Subgraph 탐색

- Model Prediction VS Masked Prediction
  - 모델이 특정 Prediction으로 예측한 이유가 무엇인가?

$$\min_{\mathcal{G}} H(Y|G = A_c \odot \sigma(M), X = X_S)$$

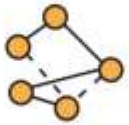
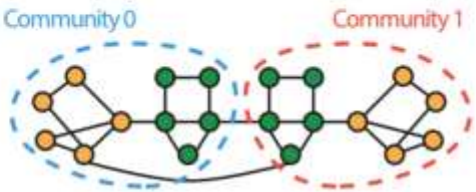
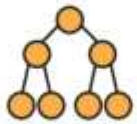
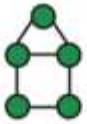
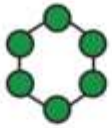
- Model Prediction VS Masked Prediction
  - 모델이 특정 Prediction으로 예측한 이유가 무엇인가?

$$\min_{M} -\sum_{c=1}^{C} 1[y = c] log P_{\Phi}(Y = y|G = A_c \odot \sigma(M), X = X_c)$$



8. Dataset (1)

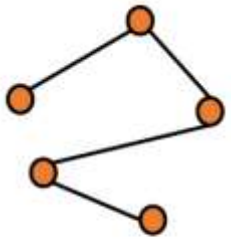
- Synthetic Datasets
  - Evaluation만을 위해 제작한 데이터셋
  - Node Classification에 활용

	BA-Shapes	BA-Community	Tree-Cycles	Tree-Grid
Base				
Motif				
Node Features	None	$\mathcal{N}(\mu_l, \sigma_l)$ where l = community ID	None	None
Explanation content	Graph structure	Graph structure Node feature information	Graph structure	Graph structure

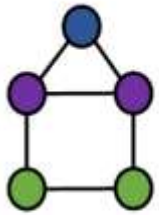
8. Dataset (2)

- Synthetic Datasets
  - Random Graph 기반 데이터 셋 (BA-Shaples / BA-Community)

1. BA Graph를 생성하고 Motif를 구성



Graph – 300 Nodes

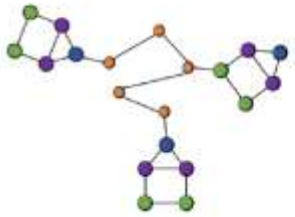


Motif - House Shaped

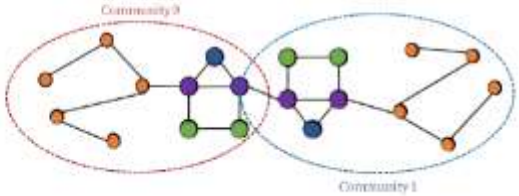
위치별 Class 상이

2. BA Graph에 Motif를 랜덤으로 추가

BA-Shapes



BA-Community

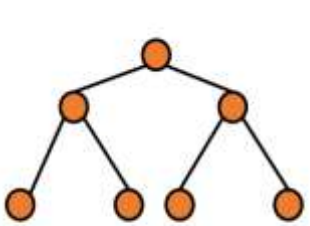


3. Graph 내 노드의 Classification 수행

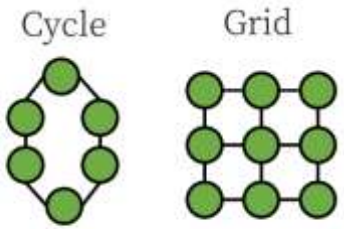
8. Dataset (3)

- Synthetic Datasets
  - Tree Graph 기반 데이터 셋 (Tree-Cycles / Tree-Grid)

1. Tree Graph를 생성하고 Motif를 구성

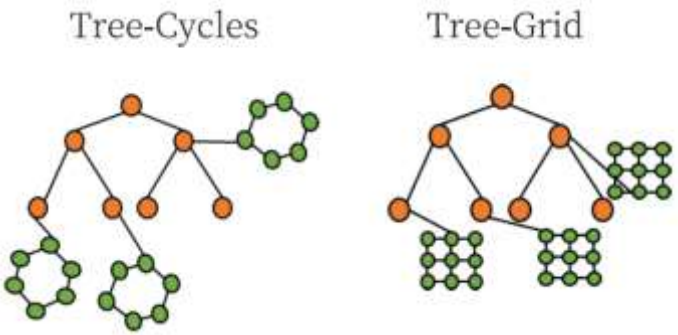


Tree Graph
– 8 level binary tree



Motif - House Shaped
위치별 Class 동일

2. Tree Graph에 Motif를 랜덤으로 추가



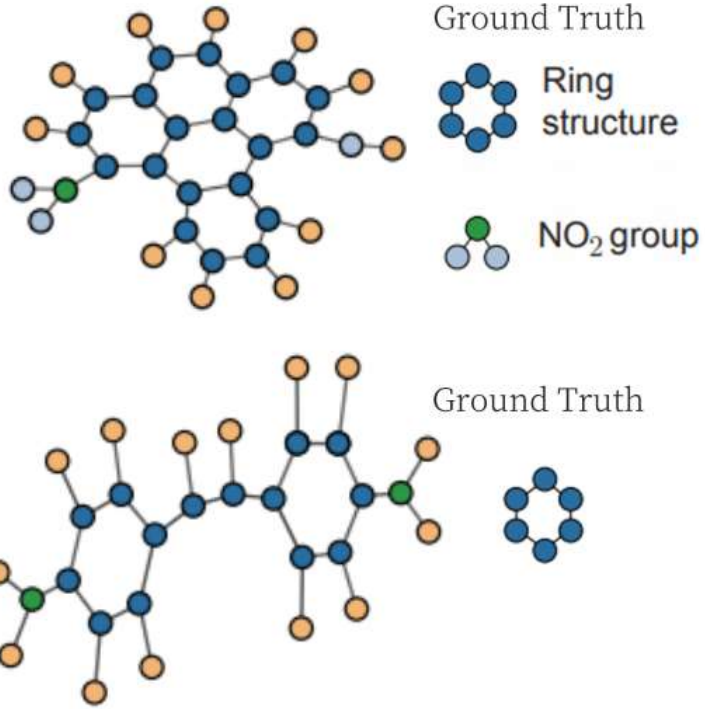
3. Graph 내 노드의 Classification 수행

8. Datasets (4)

- Real World Datasets
  - 실제로 존재하는 벤치마크 데이터셋
  - Graph Classification에 활용

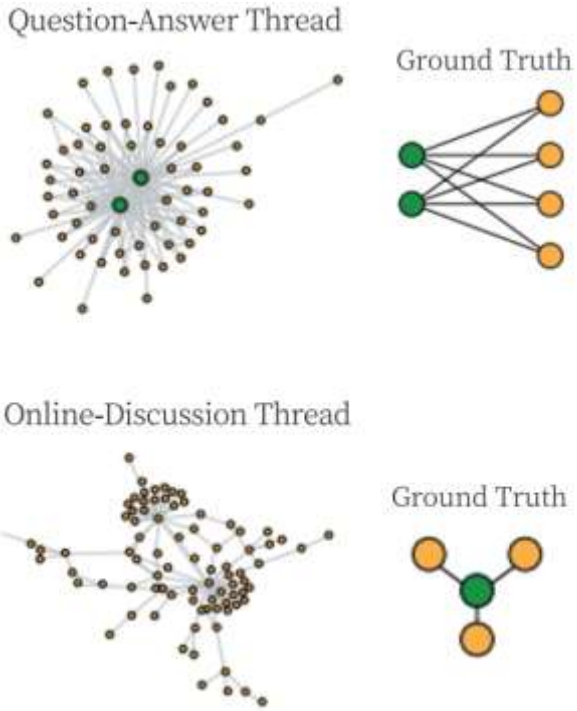
MUTAG

- 4,337개의 분자 그래프가 박테리아에 노출될 때의 유전 변화
- Data : Molecule Graph
- Label : Mutagenic Effect



Reddit-Binary

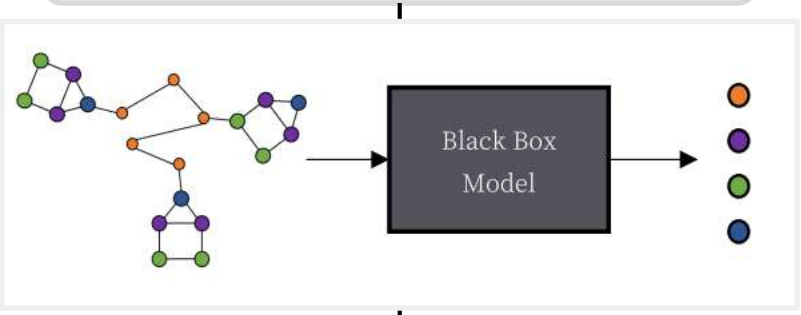
- Reddit 토론 게시판
- Node : User
- Edge : Comment



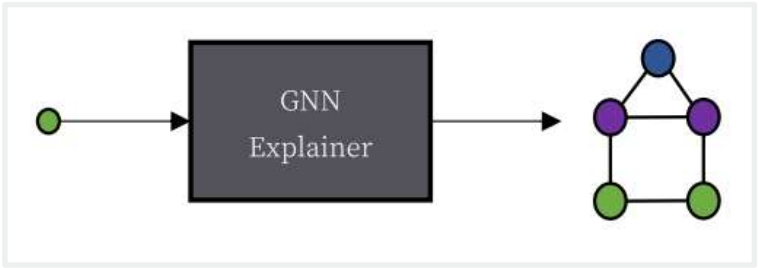
9. Experiments (1)

- Node Classification
  - Datasets : Synthetic Dataset

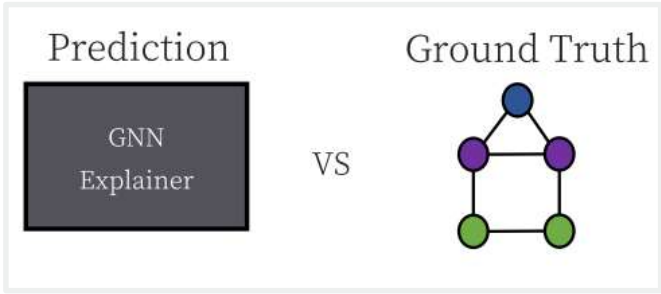
1. Node Classification Model 구성



2. 특정 노드 선택 -> GNNExplainer

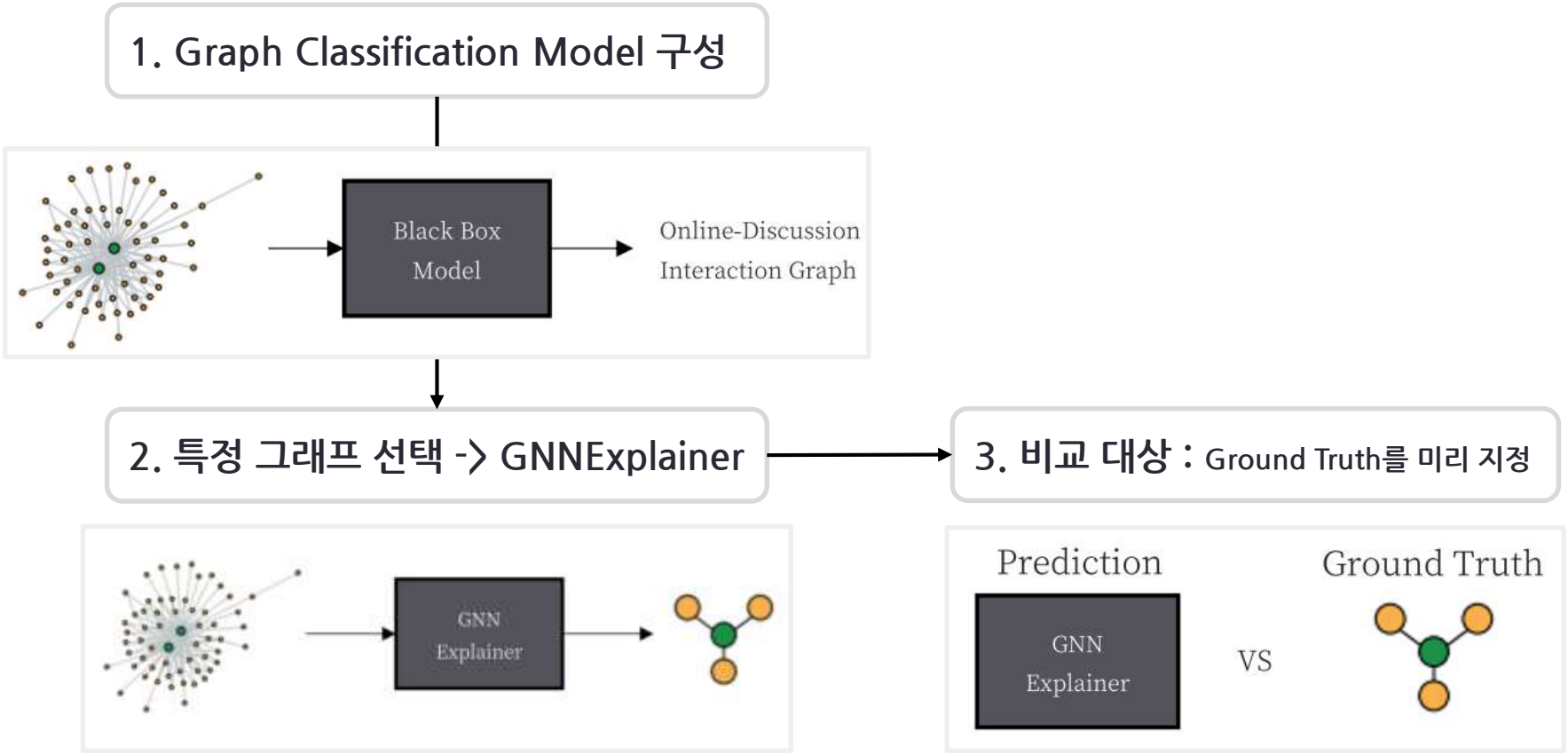


3. 비교 대상 :
Node 분류에 활용되는 것이 Motif이므로 이를 Ground Truth로 설정



9. Experiments (2)

- Graph Classification
  - Datasets : Real World Dataset



9. Experiments (3)

- 비교 시스템

GRAD

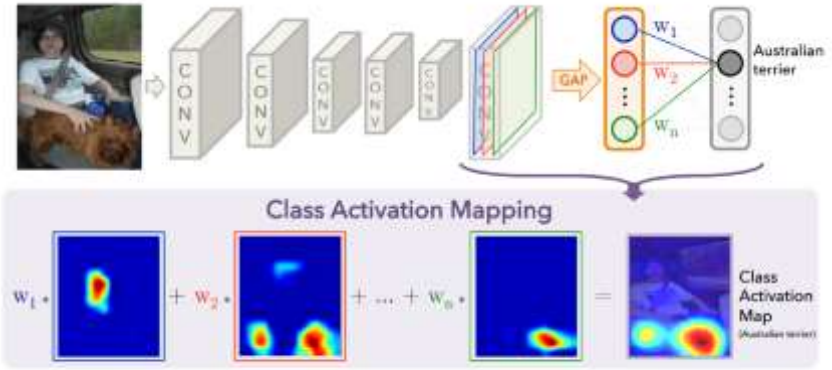
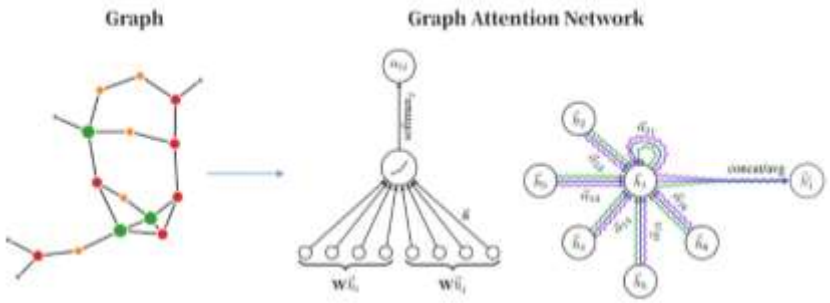


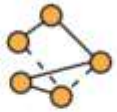
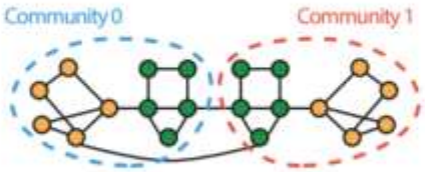
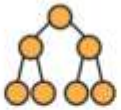
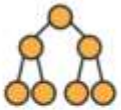
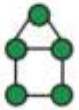
Figure 2. Class Activation Mapping: the predicted class score is mapped back to the previous convolutional layer to generate the class activation maps (CAMs). The CAM highlights the class-specific discriminative regions.

ATT (using GAT)



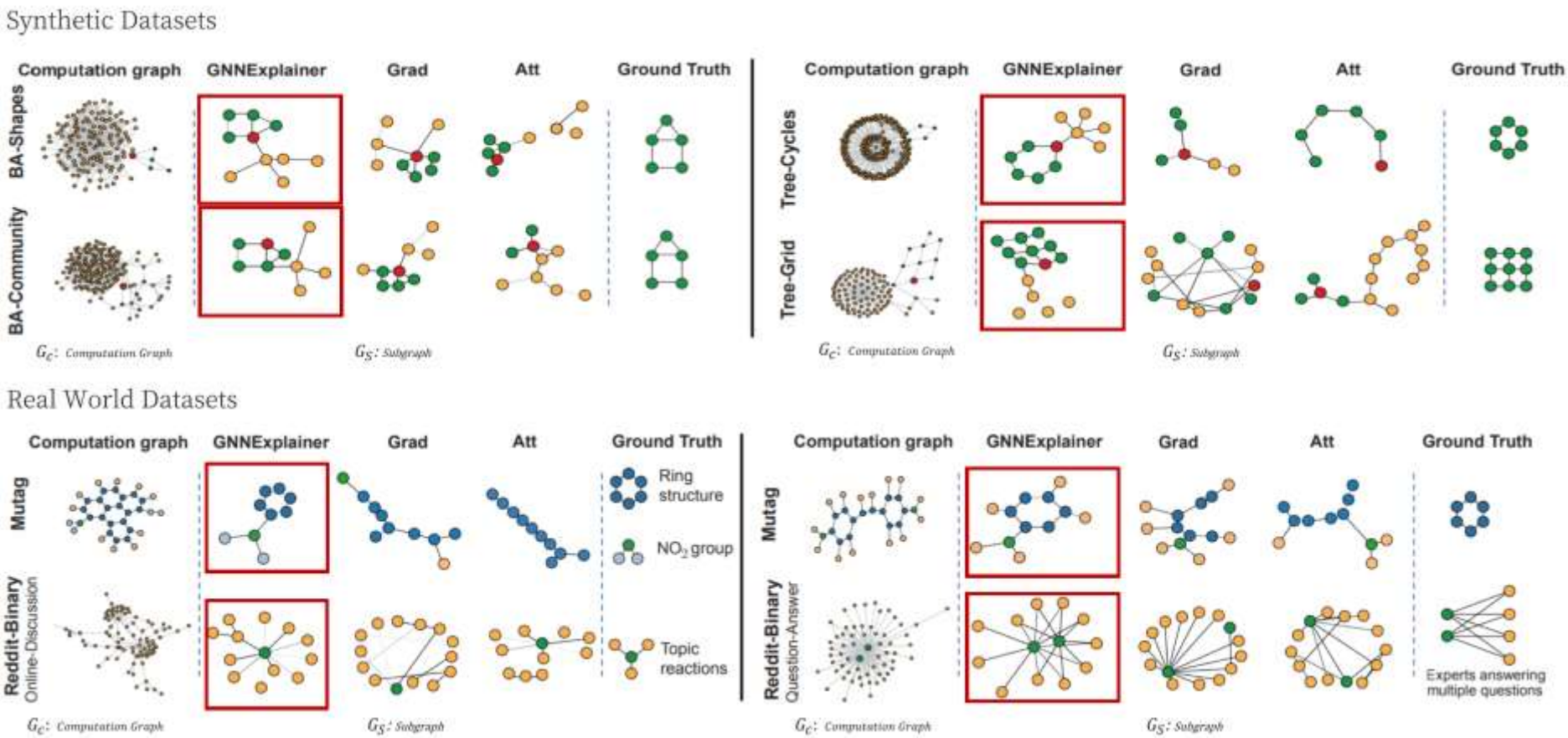
9. Experiments (4)

- Quantitative Analysis

	BA-Shapes	BA-Community	Tree-Cycles	Tree-Grid
Base				
Motif				
Node Features	None	$\mathcal{N}(\mu_l, \sigma_l)$ where l = community ID	None	None
Explanation content	Graph structure	Graph structure Node feature information	Graph structure	Graph structure
Explanation accuracy				
Att	0.815	0.739	0.824	0.612
Grad	0.882	0.750	0.905	0.667
GNNExplainer	**0.925**	**0.836**	**0.948**	**0.875**

9. Experiments (5)

- Qualitative Analysis



10. Pytorch-Geometric에서의 실습

- GNNExplainer_Example_(Pytorch_Geometric_Version).ipynb 참고