GNN 기초부터 논문까지

Basic to Paper

0. Intro

GNN(Graph Neural Network)은 그래프 데이터를 직접 분석할 수 있기에 많은 관심을 받고 있는 기술이다. 그래프에 대한 더 나은 이해를 위해 약간의 그래프 이론과 그래프 분석론으로 시작하여 해당 방법론의 한계와 이를 극복하기 위해 등장한 GNN의 형태 및 원리, 활용을 알아보자.

1. 그래프

그래프의 정의 및 표현 방법

그래프: 점과 점 사이를 잇는 선으로 이루어진 데이터 구조. 관계나 상호 작용을 나타내는 데이터를 분석할 때 주로 사용된다.

일반적으로 그래프는 G=(V,E)로 정의하며, V는 점(vertex) 집합, E는 선(edge)

$G = (V, E)$

G = ({1,2,3},{{1,2},{2,3},{1,3}})

1. 그래프

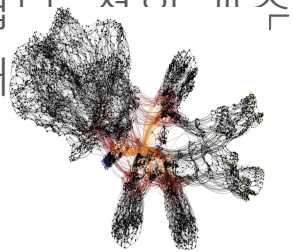
그래프는 주로 인접행렬(adjacency matrix)로 표현된다. 점의 개수가 n개일 때 인접행렬의 크기는 n*n이다.

머신러닝에서 그래프를 다룰 땐 점들의 특징을 묘사한 feature matrix로 표현하기도 하는데, feature의 개수가 f일 때 feature matrix의 차원은 n*f이다.

1. 그래프

그래프를 분석하기 어려운 이유

1) 그래프는 Euclidean space에 있지 않다. 즉, 좌표계로 표현이 불가능하다. 시계열, 음성, 이미지 등의 데이터는 비정형일지라도 2,3차원의 유클리드 공간에 쉽게 매핑이 가능하지만, 그래프의 데이터는 그렇지 않아 분석과 해석이 비교적 어렵다.
2) 그래프는 고정형태가 아니다. 인접행렬이 같더라도 그래프의 모양이 다를 수 있고, 그 반대 또한 존재한다.
3) 그래프는 사람이 해석할 수 있도록 시각화하는 것이 어렵다. 정의 개수가 늘어날 수록 사람의 인지능력이 이를 인지하기란 점점 어



2. 그래프를 사용하는 이유

그렇다면 왜 그래프를 사용할까?

그래프는 각 점(샘플) 사이의 유기적인 관계를 파악하는데에 용이하며, 이를 통해 다양한 연구를 수행할 수 있다.

관계, 상호작용과 같은 추상적인 개념을 다루기에 적합하며, 복잡한 문제를 간단히 표현할 수 있다.

점 사이의 관계를 G=(V,E) 형태로 표현할 수 있기 때문에 컴퓨터의 연산능력을 활용하기 용이하며 이것은 그래프이론, 더 나아가 GNN의 등장 배경이 되었다.

3. 기존 그래프 분석 방법

전통적인 그래프 분석 방법은 검색(BFS, DFS), 최단경로(Dijkstra, A*), 신장 트리(Prim, Kruskal), 클러스터링 알고리즘 등을 기반으로 한다.

이러한 방법은 입력 그래프에 대한 사전 지식(도메인 지식)이 필요하다는 단점을 가지며, 따라서 그래프 자체를 연구하는 것이 불가능했다.

또한 여러 개의 그래프가 있을 때 다양한 그래프의 정보를 이용하기가 어려웠다.

4. Graph Neural Network

GNN은 그래프에서 직접 적용할 수 있는 신경망으로, 점, 선, 그래프 레벨에서의 예측 작업에 사용된다.

점의 정의는 이웃과의 연결에 의해 발생하며, 이러한 연결 상태를 이용하여 각 점의 상태를 업데이트하는 과정을 학습이라고 본다.

예측은 이러한 점의 마지막 상태를 이용해 수행되며, 이러한 상태를 Node Embedding이라고 부른다.

5. Recurrent Graph Neural Network

Recurrent GNN은 Banach Fixed-Point Theorem을 기초로 만들어졌다.

>> k가 매우 커지면, x에 매핑 T를 k번 적용한 값과 k+1번 적용한 값이 거의 같아진다.

즉, 일정 k가 초과되면 학습이 종료된다는 의미라고 할 수 있다.

Recurrent GNN은 입력과 출력이 아래와 같은 함수 f_w를 정의하여 점의 상태를 구

$$\boldsymbol{x}_n = f_{\boldsymbol{w}}(\boldsymbol{l}_n, \boldsymbol{l}_{\mathrm{co}[n]}, \boldsymbol{x}_{\mathrm{ne}[n]}, \boldsymbol{l}_{\mathrm{ne}[n]})$$

5. Recurrent Graph Neural Network

l_n은 점 x_n의 feature

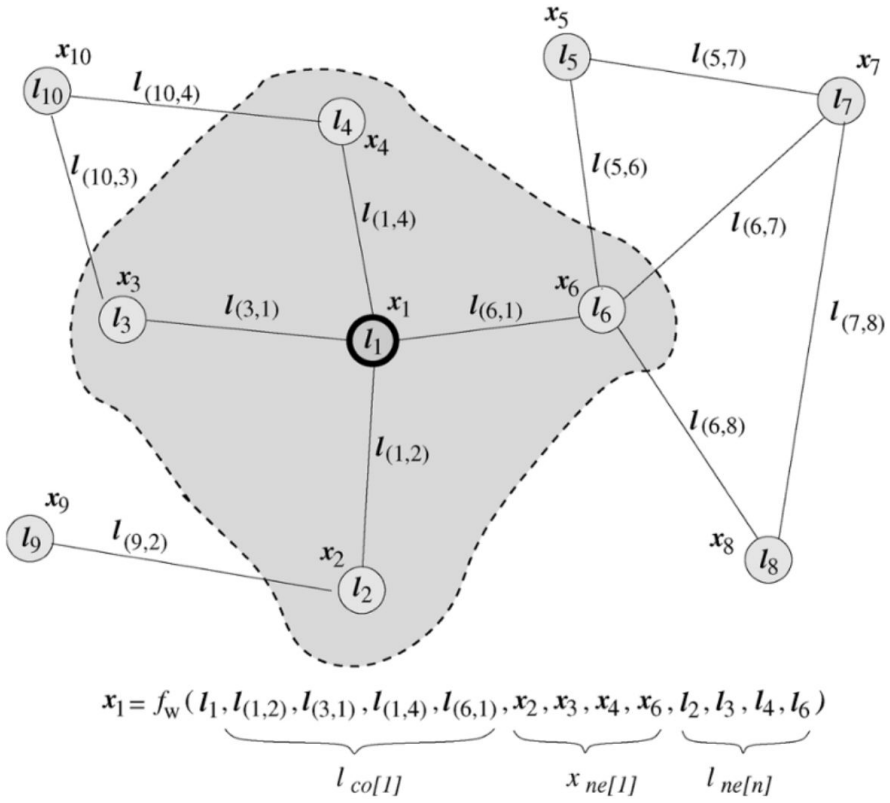
l_{co[n]}은 점 x_n과 연결된 선들의 featu

l_{ne[n]}은 점 x_n과 연결된 점들의 featu

x_{ne[n]}은 점 x_n과 연결된 점을 의미

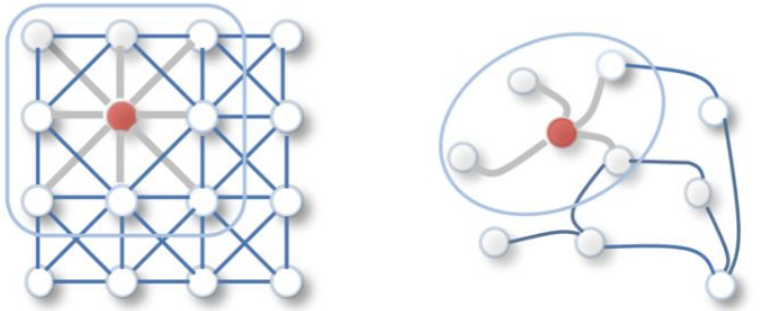
>> k번 반복을 통한 업데이트 후
마지막 상태(x_n)와 특징(l_n)을 사용하여
결과값(o_n)을 출력한다.

$$o_n = g_w(x_n, l_n)$$



6. Spatial Convolution Network

Spatial Convolution Network는 CNN과 비슷한 아이디어로, 필터를 통해 주변 픽셀을 합치는 것처럼 연결된 점의 특징을 이용한다.



(a) 2D Convolution. Analogous to a graph, each pixel in an image is taken as a node where neighbors are determined by the filter size. The 2D convolution takes the weighted average of pixel values of the red node along with its neighbors. The neighbors of a node are ordered and have a fixed size.

(b) Graph Convolution. To get a hidden representation of the red node, one simple solution of the graph convolutional operation is to take the average value of the node features of the red node along with its neighbors. Different from image data, the neighbors of a node are unordered and variable in size.

Fig. 1: 2D Convolution vs. Graph Convolution.

7. Spectral Convolutional Network

Spectral Convolutional Network는 그래프 신호 처리 이론을 기반으로 고안됐다.

인접행렬 * feature matrix = 인접 점(자신 포함) feature를 모두 더한 행렬 H

현재 대부분의 Convolutional GNN이 이러한 방식

점의 정보를 공유하고 업데이트하기 위한 전달방식은 함수에 따라 다르며, 이 함수를 message-passing 함수라고 한다.

- A Massage Passing function: $m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_u^t, e_{uv})$
- A Node Update function: $h_v^{l+1} = U_t(h_v^t, m_v^{t+1})$
- A Readout function (for graph classification): $\hat{y} = R(\{h_v^T | v \in G\})$

8. GNN이 할 수 있는 일

1) **Node Classification**
   Node Embedding을 통해 점을 분류. 일반적으로 그래프 일부만 레이블 된 상황에서 semi-supervised learning을 한다.
   ex) 인용 네트워크, Reddit 게시물, Youtube 동영상
2) **Link Prediction**
   그래프의 점들 사이 관계를 파악해 그 사이 연관성을 예측. 연결되지 않은 쌍 중 연결될 가능성이 높은 쌍을 찾아 **추천 시스템에 응용 가능**
   ex) 페이스북 친구 추천, 비디오 스트리밍 서비스 영상 추천
3) **Graph Classification**
   그래프 전체를 여러 카테고리로 분류. 자연과학 계열에 많이 사용

9. 실제 응용 사례

Scene graph generation by iterative message passing

CNN으로 탐지된 물체들 간의 scene graph를 만들어 관계를 파악

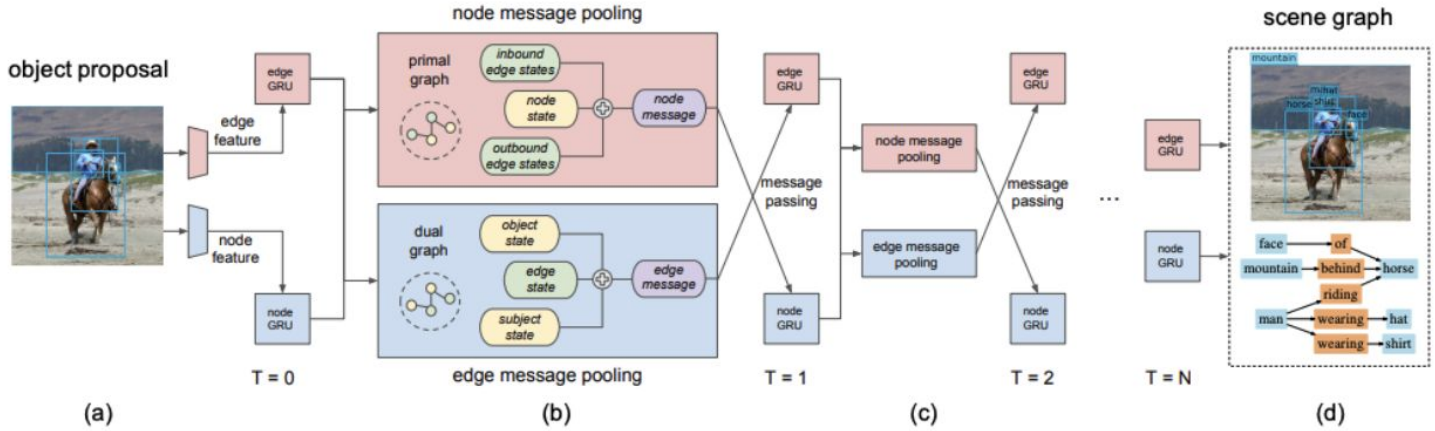


Figure 3. An illustration of our model architecture (Sec. 3). The model first extracts visual features of nodes and edges from a set of object proposals, and edge GRUs and node GRUs then take the visual features as initial input and produce a set of hidden states (a). Then a node message pooling function computes messages that are passed to the node GRU in the next iteration from the hidden states. Similarly, an edge message pooling function computes messages and feed to the edge GRU (b). The $\oplus$ symbol denotes a learnt weighted sum. The model iteratively updates the hidden states of the GRUs (c). At the last iteration step, the hidden states of the GRUs are used to predict object categories, bounding box offsets, and relationship types (d).

9. 실제 응용 사례

Image generation from scene graph

위와 반대로, scene graph에서 이미지를 생성

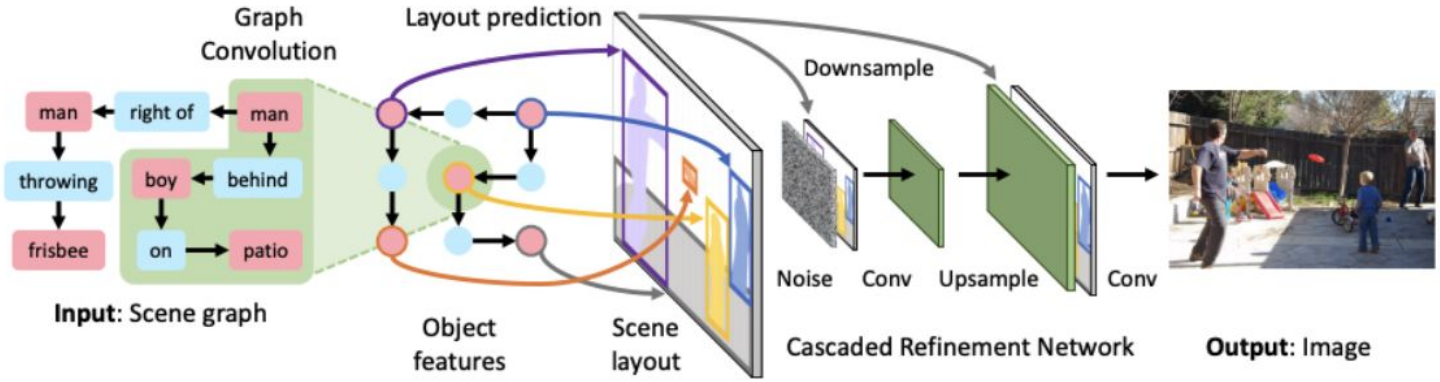


Figure 2. Overview of our image generation network f for generating images from scene graphs. The input to the model is a *scene graph* specifying objects and relationships; it is processed with a *graph convolution network* (Figure 3) which passes information along edges to compute embedding vectors for all objects. These vectors are used to predict bounding boxes and segmentation masks for objects, which are combined to form a *scene layout* (Figure 4). The layout is converted to an image using a *cascaded refinement network* (CRN) [6]. The model is trained adversarially against a pair of *discriminator networks*. During training the model observes ground-truth object bounding boxes and (optionally) segmentation masks, but these are predicted by the model at test-time.

9. 실제 응용 사례

Graph-Structured Representations for Visual Question Answering

Visual Question Answering 문제에 그래프를 도입. Scene graph와 question graph를 만든 후 pooling과 GRU를 적용

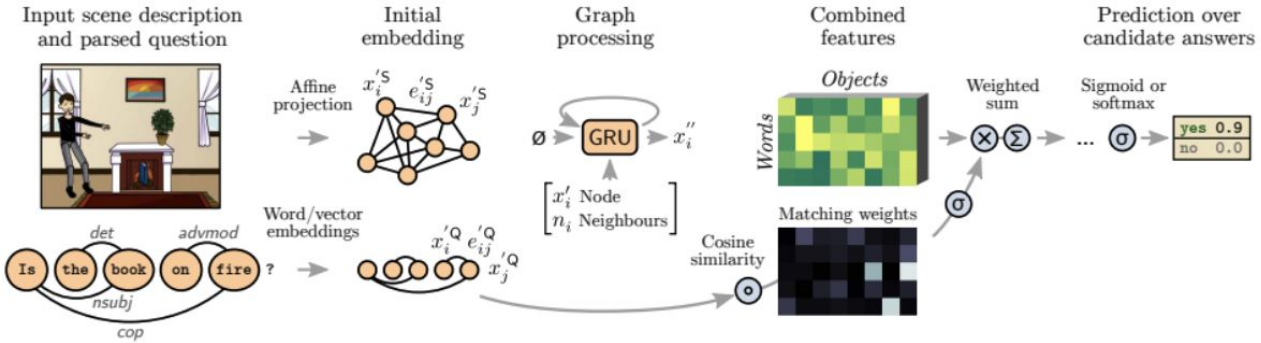


Figure 2. Architecture of the proposed neural network. The input is provided as a description of the scene (a list of objects with their visual characteristics) and a parsed question (words with their syntactic relations). The scene-graph contains a node with a feature vector for each object, and edge features that represent their spatial relationships. The question-graph reflects the parse tree of the question, with a word embedding for each node, and a vector embedding of types of syntactic dependencies for edges. A recurrent unit (GRU) is associated with each node of both graphs. Over multiple iterations, the GRU updates a representation of each node that integrates context from its neighbours within the graph. Features of all objects and all words are combined (concatenated) pairwise, and they are weighted with a form of attention. That effectively matches elements between the question and the scene. The weighted sum of features is passed through a final classifier that predicts scores over a fixed set of candidate answers.

9. 실제 응용 사례

Learning Generalizable Perceptual Representations of Small Molecules

후각 인지 분야에서 ML이 사용된 사례로, 분자 구조를 그래프로 변환하고 GNN을 거치면 138개의 향기를 예측할 수 있다고 한다.

기존 fingerprint 기반 분자 구조 예측에서 발전된 형태

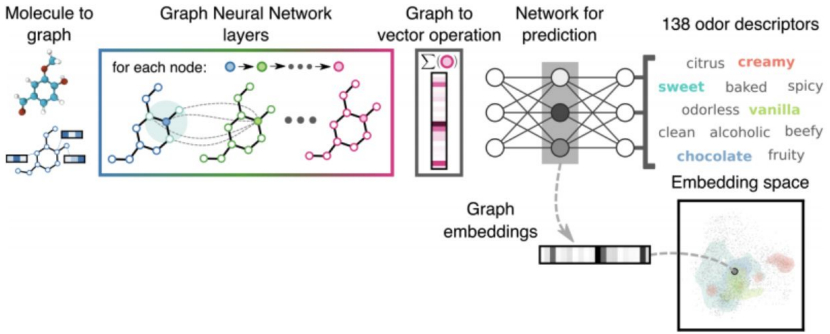


Figure 2: **Model Schematic.** Each molecule is first featurized by its constituent atoms, bonds, and connectivities. Each Graph Neural Network (GNN) layer, here represented as different colors, transforms the features from the previous layer. The outputs from the final GNN layer is reduced to a vector, which is then used for predicting odor descriptors via a fully-connected neural network. We retrieve graph embeddings from the penultimate layer of the model. An example of the embedding space representation for four odor descriptors is shown in the bottom right; the colors of the regions in this plot correspond to the colors of odor descriptors in top right.

9. 실제 응용 사례

Graph Convolutional Matrix Completion

유저-아이템 평점 행렬이 있을 때, 기존 평점을 기반으로 message passing 함수를 사용해서 아직 평가가 없는 평

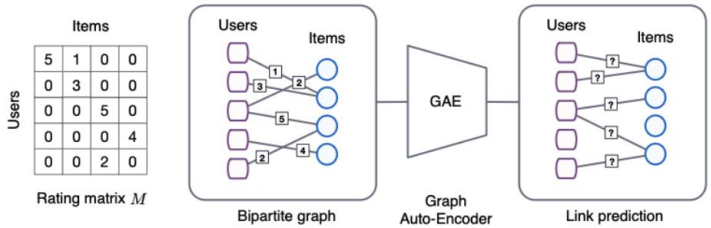


Figure 1: *Left*: Rating matrix M with entries that correspond to user-item interactions (ratings between 1-5) or missing observations (0). *Right*: User-item interaction graph with bipartite structure. Edges correspond to interaction events, numbers on edges denote the rating a user has given to a particular item. The matrix completion task (i.e. predictions for unobserved interactions) can be cast as a link prediction problem and modeled using an end-to-end trainable graph auto-encoder.

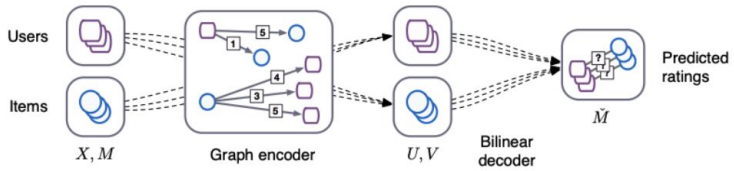


Figure 2: Schematic of a forward-pass through the GC-MC model, which is comprised of a graph convolutional encoder $[U, V] = f(X, M_1, \ldots, M_R)$ that passes and transforms messages from user to item nodes, and vice versa, followed by a bilinear decoder model that predicts entries of the (reconstructed) rating matrix $\check{M} = g(U, V)$, based on pairs of user and item embeddings.

10. 마무리

‘블랙박스’로 칭해지는 머신러닝에 그래프가 도입되면서 논리를 좀더 뒷받침할 수 있게 되었으며 더욱 자연스러운 문제 해결이 가능해졌다.

GNN은 여전히 비교적 새로운 분야이며, 더 많은 주목을 받을 가치가 있다.

11. 참고자료

1. F. Scarselli, M. Gori, “The graph neural network model,” *IEEE Transactions on Neural Networks, 2009*
2. Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, Philip S. Yu, “A Comprehensive Survey on Graph Neural Networks”, arXiv:1901.00596
3. T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in *Proc. of ICLR*, 2017
4. J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, “Neural Message Passing for Quantum Chemistry”, *in Proc. of ICML*, 2017
5. D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei, “Scene graph generation by iterative message passing,” in *Proc. of CVPR*, 2017
6. J. Johnson, A. Gupta, and L. Fei-Fei, “Image generation from scene graphs,” in *Proc. of CVPR*, 2018
7. D. Teney, L. Liu and A. van den Hengel, “Graph-Structured Representations for Visual Question Answering”, in *Proc. of CVPR*, 2017
8. B. Sanchez-Lengeling, J. N. Wei, B. K. Lee, R. C. Gerkin, A. Aspuru-Guzik, and A. B. Wiltschko, “Machine Learning for Scent: Learning Generalizable Perceptual Representations of Small Molecules”, arXiv: 1910.10685
9. R. van den Berg, T. N. Kipf, and M. Welling, “Graph Convolutional Matrix Completion”, arXiv:1706.02263