Module 5 assignment

1. Describe the various real-life applications of Heterogeneous graphs.

Ans)

A large set of real-world datasets are stored as heterogeneous graphs, motivating the introduction of specialized functionality for them in Pytorch Geometric (PyG). For example, most graphs in the area of recommendation, such as social graphs, are heterogeneous, as they store information about different types of entities and their different types of relations. This tutorial introduces how heterogeneous graphs are mapped to PyG and how they can be used as input to Graph Neural Network models.

Heterogeneous graphs come with different types of information attached to nodes and edges. Thus, a single node or edge feature tensor cannot hold all node or edge features of the whole graph, due to differences in type and dimensionality. Instead, a set of types need to be specified for nodes and edges, respectively, each having its own data tensors. As a consequence of the different data structure, the message passing formulation changes accordingly, allowing the computation of message and update function conditioned on node or edge type.

Example Graph[ℑ](https://pytorch-geometric.readthedocs.io/en/latest/notes/heterogeneous.html#example-graph)

As a guiding example, we take a look at the heterogenous [ogbn-mag](https://ogb.stanford.edu/docs/nodeprop) network from the [OGB datasets](https://ogb.stanford.edu/):

The given heterogeneous graph has 1,939,743 nodes, split between the four node types **author**, **paper**, **institution** and **field of study**. It further has 21,111,007 edges, which also are of one of four types:

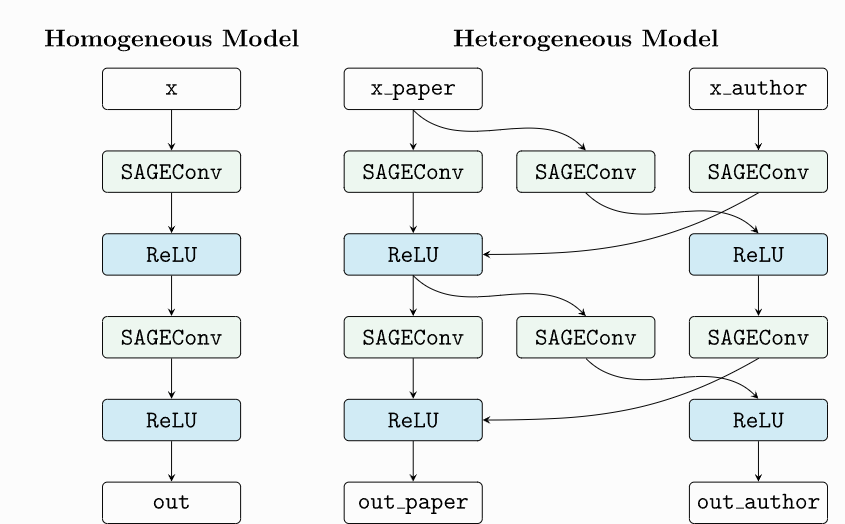
* **writes**: An author *writes* a specific paper
* **affiliated with**: An author is *affiliated with* a specific institution
* **cites**: A paper *cites* another paper
* **has topic**: A paper *has a topic* of a specific field of study

The task for this graph is to infer the venue of each paper (conference or journal) given the information stored in the graph.

### Automatically Converting GNN Models[ℑ](https://pytorch-geometric.readthedocs.io/en/latest/notes/heterogeneous.html#automatically-converting-gnn-models)

Pytorch Geometric allows to automatically convert any PyG GNN model to a model for heterogeneous input graphs, using the built in functions **torch\_geometric.nn.to\_hetero()** or **torch\_geometric.nn.to\_hetero\_with\_bases()**

The process takes an existing GNN model and duplicates the message functions to work on each edge type individually, as detailed in the following figure.



As a result, the model now expects dictionaries with node and edge types as keys as input arguments, rather than single tensors utilized in homogeneous graphs. Note that we pass in a tuple of **in\_channels** to [**SAGEConv**](https://pytorch-geometric.readthedocs.io/en/latest/modules/nn.html#torch_geometric.nn.conv.SAGEConv) in order to allow for message passing in bipartite graphs.

2. What are the key features of Graph neural networks and highlight its applications with detail?

Ans)

Graph Neural Networks (GNNs) are a class of deep learning methods designed to perform inference on data described by graphs. GNNs are neural networks that can be directly applied to graphs, and provide an easy way to do node-level, edge-level, and graph-level prediction tasks. GNNs can do what Convolutional Neural Networks (CNNs) failed to do.

Why do Convolutional Neural Networks (CNNs) fail on graphs?

CNNs can be used to make machines visualize things, and perform tasks like image classification, image recognition, or object detection. This is where CNNs are the most popular. The core concept behind CNNs introduces hidden convolution and pooling layers to identify spatially localized features via a set of receptive fields in kernel form. CNN on an image | Source How does convolution operate on images that are regular grids? We slide the convolutional operator window across a two-dimensional image, and we compute some function over that sliding window. Then, we pass it through many layers. Our goal is to generalize the notion of convolution beyond these simple two-dimensional lattices. The insight allowing us to reach our goal is that convolution takes a little sub-patch of the image (a little rectangular part of the image), applies a function to it, and produces a new part (a new pixel). What happens is that the center node of that center pixel aggregates information from its neighbors, as well as from itself, to produce a new value. It’s very difficult to perform CNN on graphs because of the arbitrary size of the graph, and the complex topology, which means there is no spatial locality. There’s also unfixed node ordering. If we first labeled the nodes A, B, C, D, E, and the second time we labeled them B, D, A, E, C, then the inputs of the matrix in the network will change. Graphs are invariant to node ordering, so we want to get the same result regardless of how we order the nodes.

**Graph Neural Networks and its Applications**

The power of machine learning is being leveraged to solve increasingly complex problems across a range of different areas. Models are required to recognise and understand more abstract concepts and objects, and in many cases make non-linear decisions. Although powerful in their own right, the more traditional types of machine learning models lack the ability to accurately map and process some of the most complex problems.

Graph Neural Networks have a range of applications, for example:

• Generating user recommendations based on relationships between different products previously purchased

• Recommending new connections in social media networks.

• Predicting shared interests of users within ecommerce or social media networks.

• Predicting changes in ecosystems or interconnected diseases, or the mutation of viruses.

• Classification of objects, or the labelling of unlabelled data through different nodes assigned a different object label.

• Processing modelled scientific systems and structures, with applications in physics, chemistry and biology.

• Processing of knowledge graphs which map relationships between events, locations, concepts or entities.

• Processing user interactions with other users, items and systems. • Labelling of sequences and the extraction and translation of text.

**Some other applications of GNN are classified as:**

1. Node Classification: the task here is to determine the labeling of samples (represented as nodes) by looking at the labels of their neighbors. Usually, problems of this type are trained in a semi-supervised way, with only a part of the graph being labeled.

2. Graph Classification: the task here is to classify the whole graph into different categories. It’s like image classification, but the target changes into the graph domain. The applications of graph classification are numerous and range from determining whether a protein is an enzyme or not in bioinformatics, to categorizing documents in NLP, or social network analysis.

3. Graph visualization: is an area of mathematics and computer science, at the intersection of geometric graph theory and information visualization. It is concerned with the visual representation of graphs that reveals structures and anomalies that may be present in the data and helps the user to understand the graphs.

4. Link prediction: here, the algorithm has to understand the relationship between entities in graphs, and it also tries to predict whether there’s a connection between two entities. It’s essential in social networks to infer social interactions or to suggest possible friends to the users. It has also been used in recommender system problems and in predicting criminal associations.

5. Graph clustering: refers to the clustering of data in the form of graphs. There are two distinct forms of clustering performed on graph data. Vertex clustering seeks to cluster the nodes of the graph into groups of densely connected regions based on either edge weights or edge distances. The second form of graph clustering treats the graphs as the objects to be clustered and clusters these objects based on similarity.

**3. Discuss the use of Graph neural networks in classifying the Images?**

Ans)

Graph convolutional network/ gated graph neural network

Image classification is a basic computer vision task. Most of the models provide attractive results when given a huge training set of labeled classes. The focus now is towards getting these models to perform well on zero-shot and few-shot learning tasks. For that, GNN appears quite appealing. Knowledge graphs can provide the necessary information to guide the ZSL (Zero-shot learning) task

Using regular CNNs, machines can distinguish and identify objects in images and videos. Although there is still much development needed for machines to have the visual intuition of a human. Yet, GNN architectures can be applied to image classification problems.

One of these problems is scene graph generation, in which the model aims to parse an image into a semantic graph that consists of objects and their semantic relationships. Given an image, scene graph generation models detect and recognize objects and predict semantic relationships between pairs of objects.

However, the number of applications of GNNs in computer vision is still growing. It includes human-object interaction, few-shot image classification, and more.

**4. How GNNs can be used in researching existing molecular structures and discovering new structures in Chemical sciences?**

Ans)

5) GNN in text classification?

Ans)

A classic application of GNNs in NLP is Text Classification. GNNs utilize the inter-relations of documents or words to infer document labels. GCN and GAT models are applied to solve this task. They convert text to graph-of-words, and then use graph convolution operations to convolve the word graph. They show through experiments that the graph-of-words representation of texts has the advantage of capturing non-consecutive and long-distance semantics

In NLP, we know that the text is a type of sequential data which can be described by an RNN or an LSTM. However, graphs are heavily used in various NLP tasks, due to their naturalness and ease of representation.

Recently, there has been a surge of interest in applying GNNs for a large number of NLP problems like text classification, exploiting semantics in machine translation, user geolocation, relation extraction, or question answering.

We know that every node is an entity and edges describe relations between them. In NLP research, the problem of question answering is not recent. But it was limited by the existing database. Although, with techniques like GraphSage (Hamilton et al.), the methods can be generalized to previously unseen nodes.