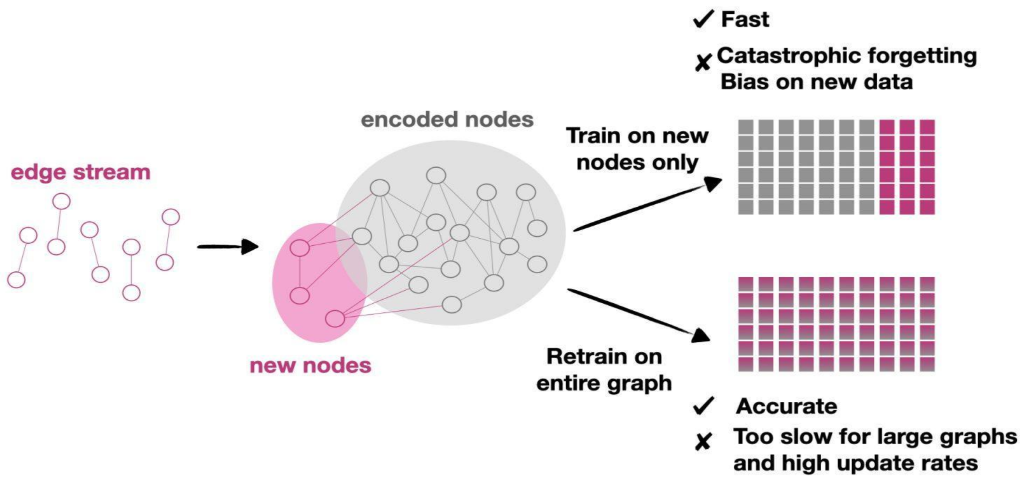
Online training of streaming Graph Neural Networks

Introduction

Graph representation learning is a methodology for understanding and analyzing complex graphs and their latent properties. It involves learning the encoding for vertices, edges, and their features into low-dimensional spaces, such as a set of vectors, also known as node *embeddings*. Node embeddings enable the application of Machine Learning techniques, such as node classification, on graphs and have been used in recommender systems, social network analysis, chemical synthesis, and financial networks. Many graphs in real-world problems, though, are generated incrementally as streams. For example, in a social network ,a vertex is added for a new user and an edge is created for new friendship relationships. In this project, you will develop a system for online training of streaming GNNs that do not fit in memory.

Background

Given a stream of graph changes, the streaming graph embedding problem seeks to continuously incorporate the latest graph snapshot into the current model. To achieve efficient retraining of Graph Neural Networks (GNNs) when the graph stream characteristics evolve over time, the online training algorithm needs to periodically generate *rehearsal samples*: small representative sets of vertices and edges that can be used to update the model parameters. A simple approach is Random-Based Rehearsal (RBR): At every time step t, the RBR strategy selects a set of past vertices $S_t \subset V_t$ at random.



In fact, RBR does not simply maintain a uniform random sample of the graph seen so far, as training on a single sample would bias the model towards the selected nodes. Instead, a new sample needs to be used for each backpropagation step, so that the model can be trained on multiple, possibly overlapping, random samples of fixed size to achieve good performance. However, as the graph stream grows in size, maintaining it in memory to compute samples becomes infeasible.

Project goal

The goal of this project is to design, implement, and evaluate a system for online training of streaming GNNs that do not fit in memory. You will assume that while the model parameters are small and can fit in memory, the input graph stream can be possibly unbounded and cannot be stored in memory in an accessible manner. Instead, you will need to modify the RBR algorithm to generate uniform random samples on-the-fly, as the input stream arrives at the system. Your initial solution will integrate GNN inference with Flink and modify RBR to use Reservoir Sampling (https://en.wikipedia.org/wiki/Reservoir_sampling) and maintain a fixed number of samples in memory. As samples might be overlapping, you will need to pay attention to avoid duplicate information and minimize the memory footprint of your solution. You will evaluate the accuracy and performance of your model on the node classification problem using 1-2 datasets and compare it with (1) no-rehearsal and (2) re-training.

Stretch Goals

1. Extend your model to work on a different graph ML task, such as link prediction.
2. Design an online sampling algorithm for Priority-Based-Rehearsal (RBR).

Required skill set

- Familiarity with Python and PyTorch.
- Familiarity with Machine Learning and/or Graph representation learning.

Where to start

- Read the Online GNN paper:
  https://sites.bu.edu/casp/files/2021/12/online-gnn-SAC-preprint.pdf
- Clone the Online GNN repository and run example:
  https://github.com/MassimoPerini/online-gnn-learning
- Find the code that implements RBR, understand how it works, and design a solution that uses reservoir sampling instead.