Reverse Engineering Graph Structures from LLM Attention Maps

Social Network Analysis Final Project

Logan Bolton

April 2025

Abstract

This study investigates whether properties of graphs formatted as text prompts can be reverse engineered from the attention maps of Large Language Models (LLMs). To investigate this, I train a Multi-Layer Perceptron (MLP) to reconstruct the adjacency matrices of undirected graphs from LLM attention values. This evaluation uses both traditional machine learning metrics and social network analysis tools to reveal the extent to which graph properties can be recovered from the inner workings of LLMs. The results of this study show that the structural information of graphs is roughly preserved in the LLM's attention values, providing insights into the interpretability of LLMs.

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities in understanding and generating text. However, their inner workings are not widely understood. The attention mechanism that enables the training of an LLM can be roughly explained as the model's understanding of how every word in an input relates to every other word in the input sequence. This attention map can be described as a complete, directed graph with self loops where each node is a word in an input sequence and each edge is the amount of attention between words.

This project explores whether it is possible to reverse engineer properties of a graph network based on the attention map of an LLM. Specifically, I focus on undirected graphs represented as text inputs and attempt to reconstruct these graphs from the resulting attention patterns.

2 Methodology

2.1 Overview

My approach consists of three main steps:

1. Feed undirected graphs to an LLM in a simple text format
2. Train an MLP on the attention values from the prompts to recreate the original graphs
3. Compare the generated graphs to the original graphs using machine learning and social network analysis metrics

2.2 Data Generation

2.2.1 Random Graphs

To create the input graphs, I generate a series of independent random graphs with the following parameters.

Parameter	**Value**
Number of Graphs	1,000
Maximum Number of Nodes	7
Edge Probability	25%

To transform each graph into a format the LLM can understand, I create a text prompt that describes each of the edges such as the example in Fig. 1. In order to reduce position bias in the prompt and to also increase the amount of training data, I generate multiple variations of the same prompt that describe one graph. These text prompts cover every possible rotation orientation of the edges in the prompt. This results in $\sum_G E_G$ total prompts where G is the total number of unique graphs and E_G is the number of edges in a graph G.

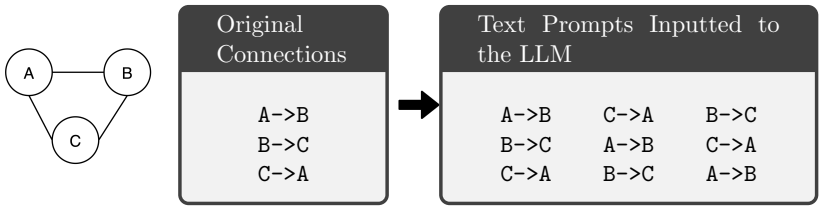


Figure 1: Each unique graph has `(Num of Edges)` number of text prompts describing it.

2.2.2 Extracted Attention Matrices

I then input each of these prompts into the Large Language Model `Llama-3.2-1B`. The attention values across all 32 heads of the model from the first layer are then extracted. I do not include other layers due to computational constraints and due to empirical observations that the attention values of deeper layers become increasingly noisy. After extracting all 32 attention maps, I average

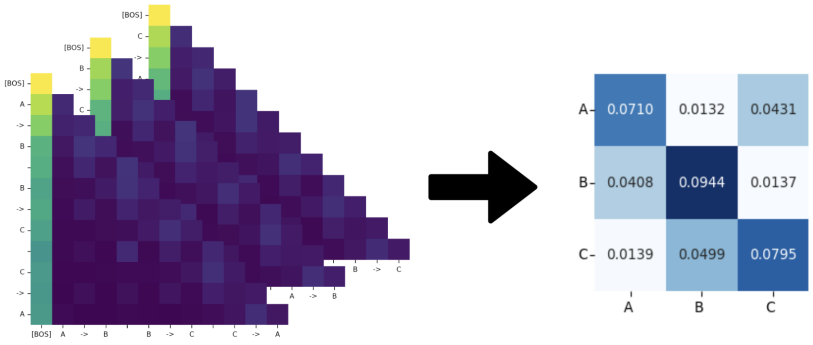


Figure 2: The amount of attention that each node attends to another node is averaged across every equivalent graph to generate a simplified aggregated attention matrix for each of the nodes.

them all into a single matrix of dimensions [# Tokens, # Tokens]. Every attention matrix extracted from equivalent graph text prompts are then averaged together.

In order to simplify the training process, I convert this token attention matrix into a matrix of size [# Nodes, # Nodes] (Fig. 2). This new matrix contains the average value that every node token attends to every other node token in the graph and disregards other tokens like the newline token, the $|BOS|$ token, etc.

2.3 Training Procedure

The combined dataset of all graphs was split into train, test and val sets at a ratio of 70%, 20% and 10% respectively. The MLP was trained to minimize the binary cross-entropy loss between the constant sized predicted adjacency matrix and the ground truth adjacency matrix.

2.4 Training Parameters

Parameter	**Value**
Epochs	200
Learning Rate	0.001
Batch Size	128
Hidden Channels	4096
Number of Layers	5

Table 1: Training parameters for the MLP trained on the weighted attention adjacency matrices from the LLM.

2.5 Evaluation Metrics

3 Results

I evaluated my approach using both traditional machine learning metrics and social network analysis tools.

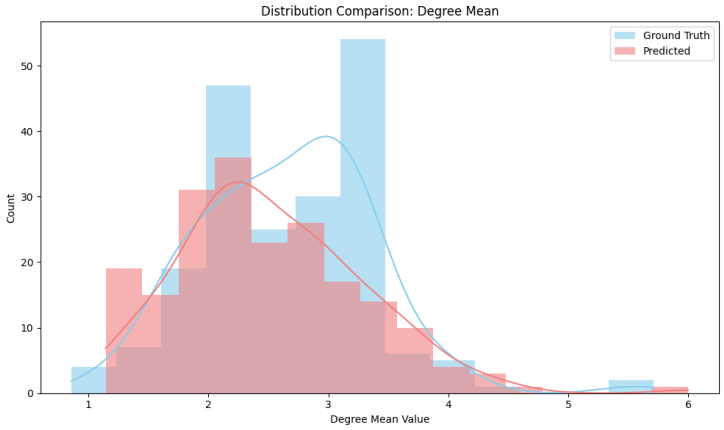
3.1 Machine Learning Metrics

Provided with the input of node attention values, the model predicts the original adjacency matrix of the graph. The MLP was able to achieve very high accuracy in terms of traditional machine learning metrics with an accuracy of 90.64% and an F1 score of 88.16%.

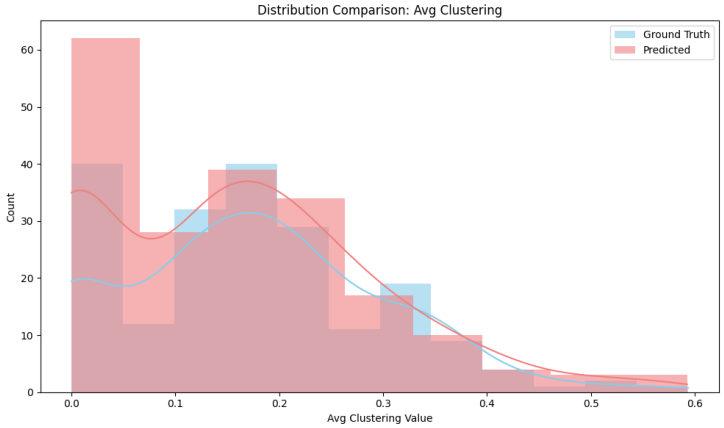
Parameter	Value
Accuracy	0.9064
F1 Score	0.8816

3.2 Network Analysis Metrics

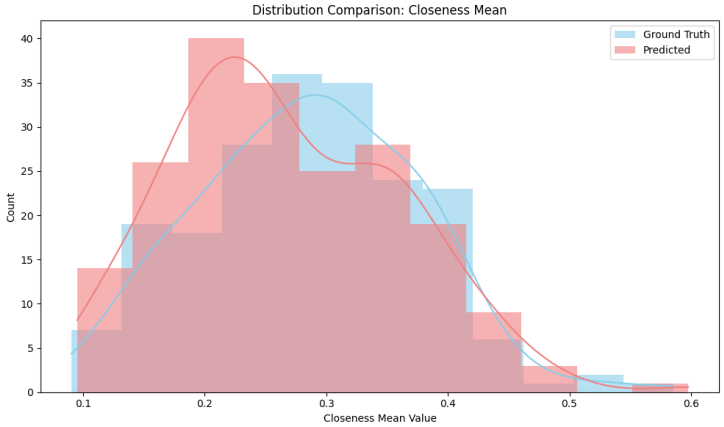
3.2.1 Degree Count



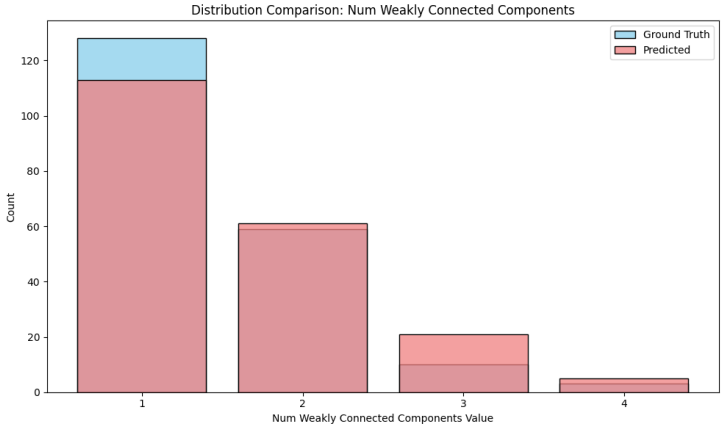
3.2.2 Clustering Coefficient



3.2.3 Closeness



3.2.4 Number of Weakly Connected Components



4 Discussion

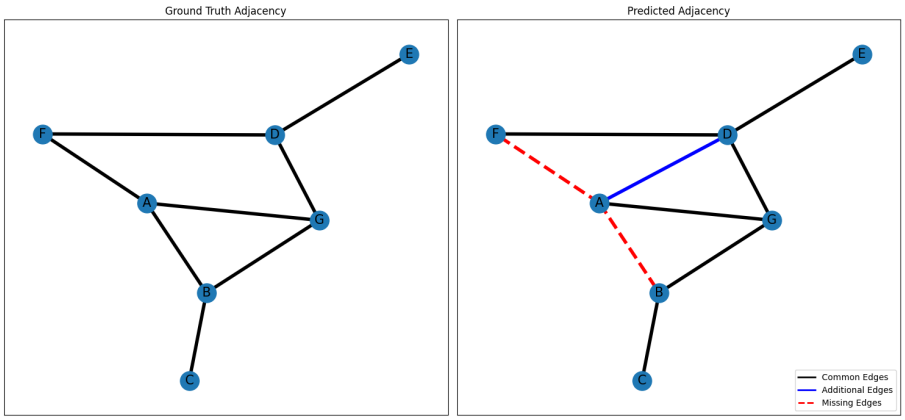


Figure 3: Example of a failure case where the MLP predicts a non-existent edge (blue) and does not accurately predict existing edges (red).

The loss function to train this model was optimized to generate high accuracy of predictions for the existence of edges between nodes. For future work, it could be valuable to explore if a more effective way to reconstruct a graph input based off LLM attention values would likely be to create a new loss function that instead optimizes for a quantitative measure of graph structural relationships instead of a more traditional loss function.

5 Conclusion

The MLP's predicted adjacency matrices had very high accuracy and F1 score. Additionally, the social network metrics from these predicted graphs roughly aligned with the ground truth characteristics. While the degree count of the predicted graphs did follow a different distribution, the metrics for the average clustering coefficient, closeness, and number of weakly connected components closely aligned. The results of this study suggest that the attention values of even a very small 1B parameter LLM can still hold rich information.