GraphMaskExplainer

1. Introduction

GraphMaskExplainer was first introduced in october 2020 in a scientific paper *"Interpreting graph neural networks for NLP with differentiable edge masking"*. This paper introduced a post-hoc interpretation method for GNNs, focusing on identifying unnecessary edges within the model. Using a fully differentiable approach, a simple classifier is trained to predict whether each edge in every layer can be dropped. Stochastic gates and sparsity encouragement through the expected L_0 norm are employed. This technique serves as an attribution method, shedding light on information flow in GNN models for tasks like question answering and semantic role labeling. Results demonstrate that a significant number of edges can be dropped without compromising model performance, allowing for focused analysis on the retained edges to interpret model predictions.

2. Simple explanation

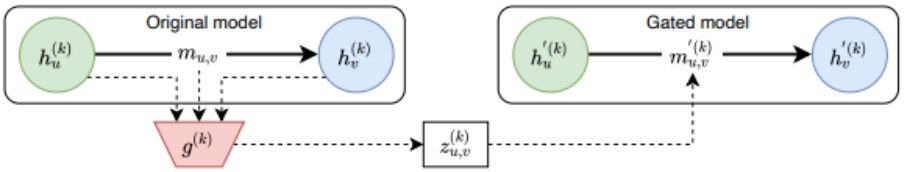

Figure 1: GRAPHMASK uses vertex hidden states and messages at layer k (left) as input to a classifier g that predicts a mask $z^{(\ell)}$. We use this to mask the messages of the kth layer and re-compute the forward pass with modified node states (right). The classifier g is trained to mask as many hidden states as possible without changing the output of the gated model.

Fig 1. The principle behind GraphMaskExplainer

When applied to Graph Neural Networks (GNNs), erasure search involves identifying the largest subgraph that can be entirely discarded. - Not tractable, Hindsight bias

GRAPHMASK can be conceptualized as a differentiable version of subset erasure. Instead of determining an optimal subset for erasure in each example, we train an erasure function to predict, for every edge (h_u, v_i) at each layer k, whether the connection should be maintained. For a given example graph G, our approach yields a subgraph $G^{(k)}_S$ for each layer k. This subgraph ensures that no edges outside $G^{(k)}_S$ impact the model's predictions. To facilitate gradient-based optimization for the erasure function, we leverage sparse stochastic gates.

3. Formulation

The equation provided in the text illustrates how the message $m^{(k)}_{u,v}$ for a particular edge is computed based on this binary choice and the learned baseline:

$$\widetilde{m}^{(k)}_{u,v} = z^{(k)}_{u,v} \cdot m^{(k)}_{u,v} + b^{(k)} \cdot (1 - z^{(k)}_{u,v}) \; .$$

Fig 2. The Equation for message with our version of subset erasure

Essentially, if $z^{(k)}_{u,v} = 1$, the original message $m^{(k)}_{u,v}$ is retained; otherwise, it is replaced with the learned baseline $b^{(k)}$. This mechanism allows the model to dynamically adjust the contribution of each edge during the inference process.

To calculate $z^{(k)}_{u,v}$ we use a simple function g with parameters denoted by π, learned once for each task across data points. This function, implemented as a single-layer neural network, computes $z^{(k)}_{u,v}$ based on input features $h_u^{(k-1)}$, $h_v^{(k-1)}$, and $m^{(k)}_{u,v}$.

$$z_{u,v}^{(k)} = g_\pi(h_u^{(k-1)}, h_v^{(k-1)}, m_{u,v}^{(k)}) ,$$

Fig 3. Equation for z binary choice variable

4. Example

https://graph-explainability.streamlit.app/

- **CORA Dataset (Multi-Class Classification):**

The model is a 2-layer Graph Convolutional Network (GCN) trained on the CORA dataset for node classification.

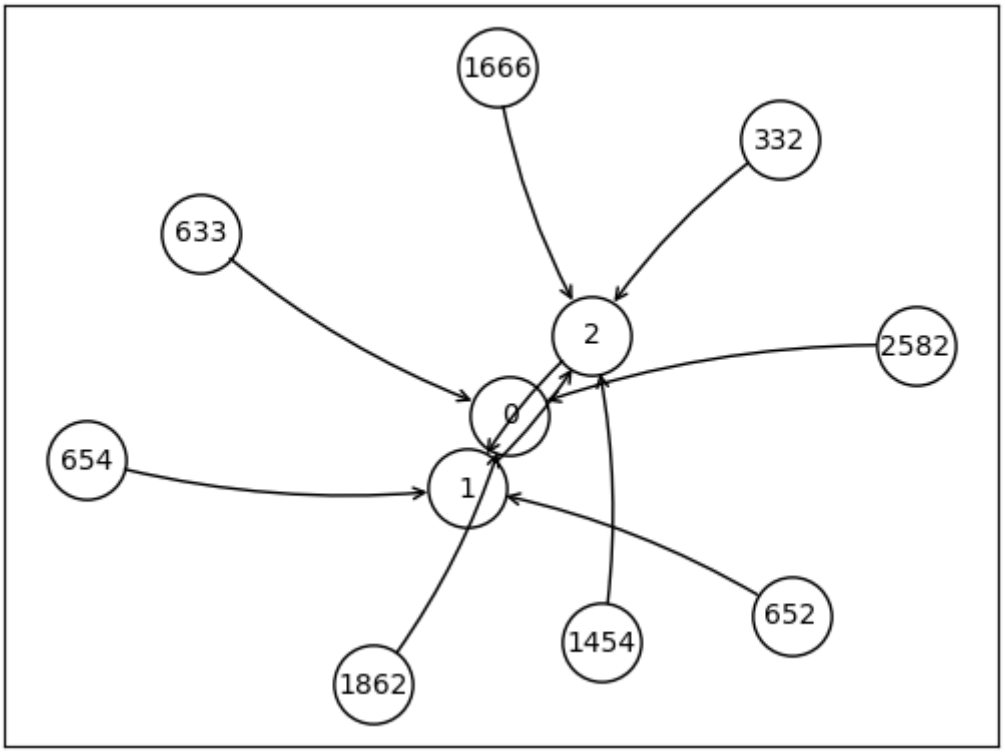

Fig 4. GCN model - The output of the explainer in case of the prediction for node 0.

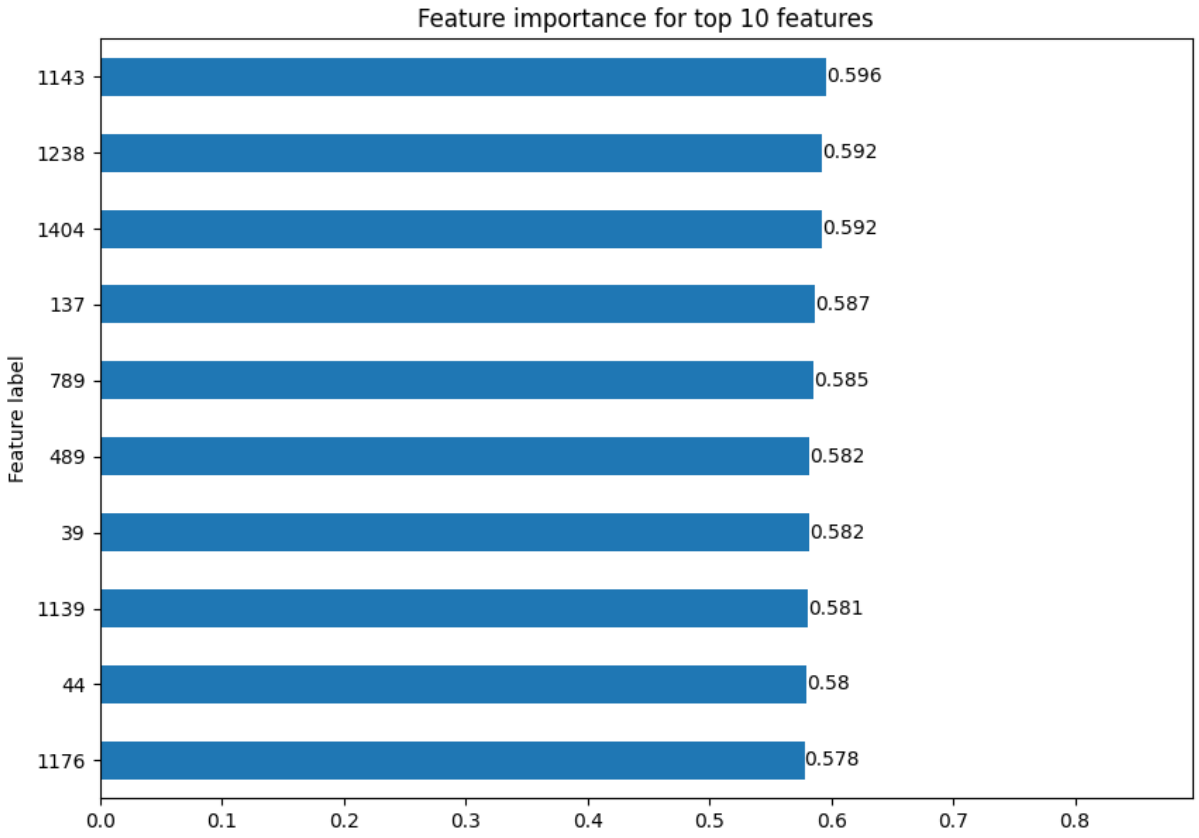

Fig 5. Feature importance in case of the prediction for node 0.

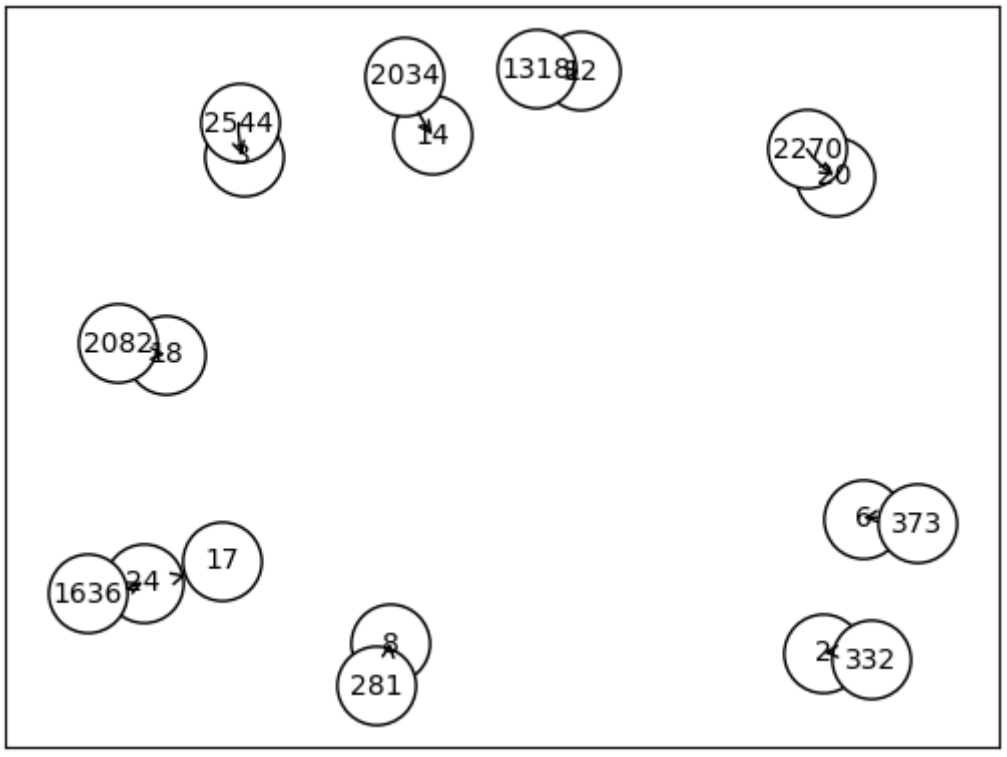

Fig 6. GAT model - The output of the explainer in case of the prediction for node 0.

- **Minesweeper Dataset (Binary Classification):**

The model is a GCN designed for binary classification on the Minesweeper dataset, where nodes represent cells in a Minesweeper grid.

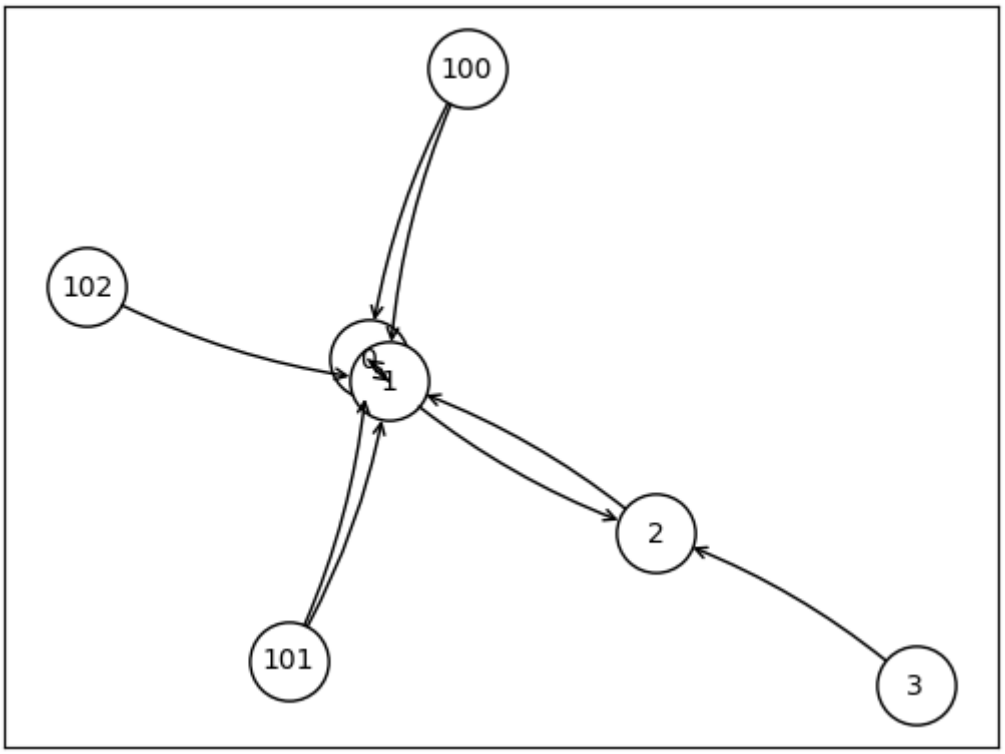

Fig 7. The output of the explainer in case of the prediction for node 0.

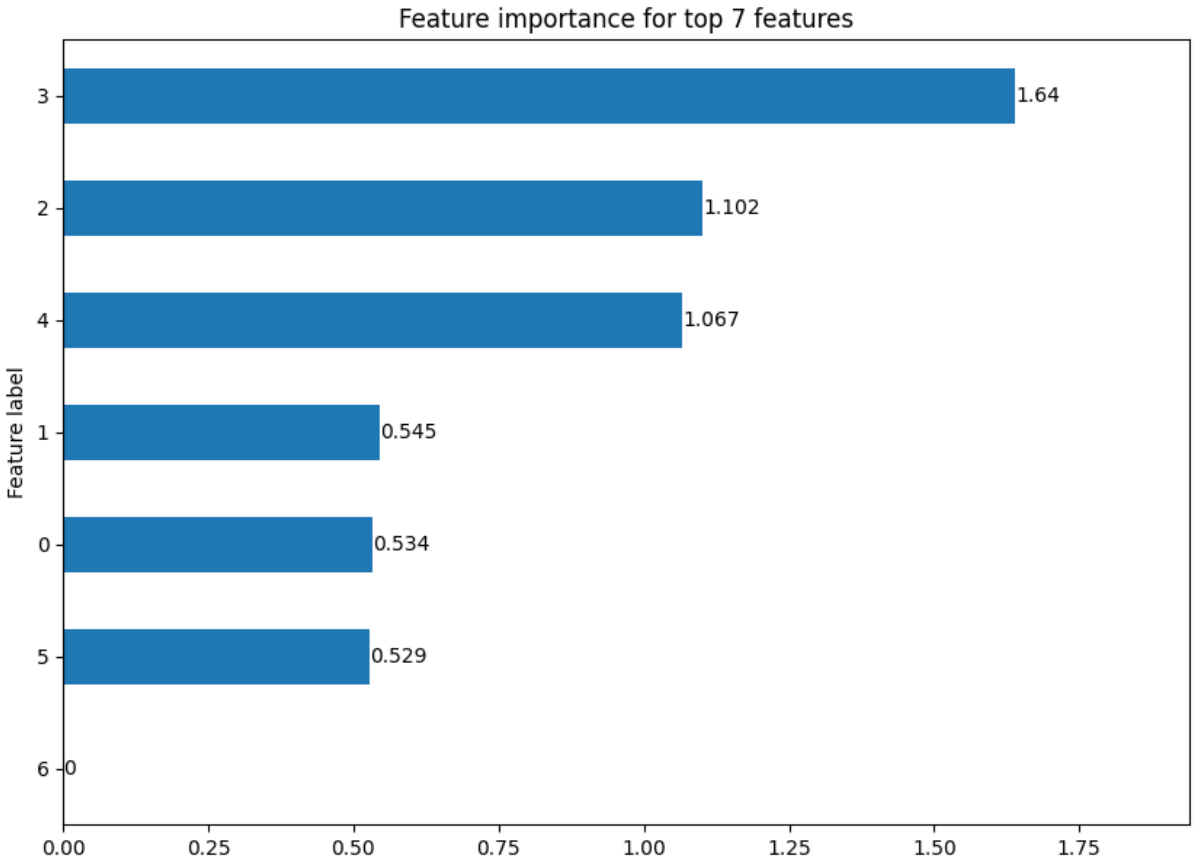

Fig 8. Feature importance in case of the prediction for node 0.

5. References

- [2010.00577] Interpreting Graph Neural Networks for NLP With Differentiable Edge Masking
- https://arxiv.org/pdf/2010.00577.pdf
- torch_geometric.explain.algorithm.GraphMaskExplainer — pytorch_geometric documentation
- https://github.com/pyg-team/pytorch_geometric/blob/master/examples//explain/graph mask_explainer.py