INTERPRETING GRAPH NEURAL NETWORKS FOR NLP WITH DIFFERENTIABLE EDGE MASKING

ICLR 2021

Agenda

1. Abstract
2. Introduction
3. Method(Graphmask)
4. experiments
    - synthetic data
    - QA (WikiHop)
5. conclusion & future work

Abstract

GRAPHMASK automatically prunes the clutter of unnecessary edges in a GNN, leaving only a handful of essential paths and turning the model into an interpretable post-hoc explainer—applied here to real NLP tasks

Introduction & Motivation

GNN models yield string performance
but **it's difficult to understand the "reasoning" behind their prediction**

For NLP practitioners, it is highly desirable to know **which linguistic information a given model encodes and how that encoding happens**

need **interpletation and explainability**

Introduction

A simple way to perform interpretation is to use **"erasure search"**
Erasure search is an approach wherein attribution happens by searching for a maximal subset of features that can be removed without affecting model predictions.
but it is note suited for GNN in some points:

1. combinatorial explosion of edge subsets
2. non-differentiable, discrete operations
3. high cost
4. **hindsight bias**

   ...

→they prppose **GRAPHMASK**

Introduction

GRAPHMASK is post-hoc interpretability technique for GNN.

It's goal is to **reveal which edges at every layer truly matter for model's prediction**

Preliminary

Graph Neural Networks

- Graph $G = (V, E)$、 node $u \in V$ node feature $h_u^{(0)}$
- **message culculation**

$$m_{u,v}^{(k)} = M^{(k)}\big(h_u^{(k-1)},\ h_v^{(k-1)},\ r_{u,v}\big)$$

- **aggregation & update **

$$h_v^{(k)} = A^{(k)}\Big(\{\, m_{u,v}^{(k)} \mid u \in \mathcal{N}(v)\}\Big)$$

- Layer $k = 1, \ldots, L$, update node features and get final representation.
- M is message function, A is aggregation fuction , $r_{u,v}$ iindicates the relation type between node u and v, $\mathcal{N}(v)$ is the set of neighbour nodes of v

Proposed Method

GRAPHMASK's goal is to

reveal which edges at every layer truly matter for model's prediction

Proposed Method

GRAPHMASK

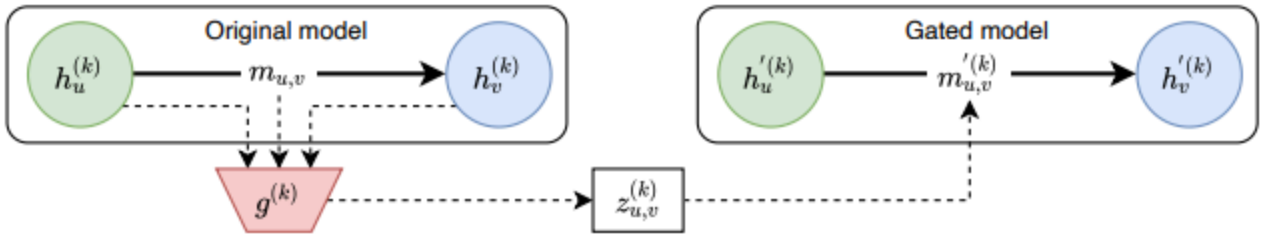

Figure 1: GRAPHMASK uses vertex hidden states and messages at layer k (left) as input to a classifier g that predicts a mask $z^{(\ell)}$. We use this to mask the messages of the kth layer and re-compute the forward pass with modified node states (right). The classifier g is trained to mask as many hidden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

Proposed Method

They call the messages associated with edges that can be ignored without changing the model's prediction **superfluous**.
Because GNNs are highly sensitive to modifications in graph structure, removing edges while preserving predictions is difficult.
Instead of explicitly searching for removable edges, our method uses a binary gate $z_{u.v}^{(k)} \in 0, 1$
z to decide whether an edge's message should be replaced by a learned baseline vector.
The replaced message is given by

$$\tilde{m}_{u,v}^{(k)} = z_{u,v}^{(k)} \, m_{u,v}^{(k)} + \left(1 - z_{u,v}^{(k)}\right) b^{(k)}$$

Proposed Method

GNNs are extremely sensitive to structural edits; brute-force edge erasure is **intractable** and prone to **hindsight bias**.

What is hindsight bias

If an explainer, after seeing the model's prediction for a test instance, searches for the smallest subset of features that still yields that prediction, the resulting subset can differ from the information the original model actually relied on—this mismatch is called **hindsight bias**.

Proposed Method

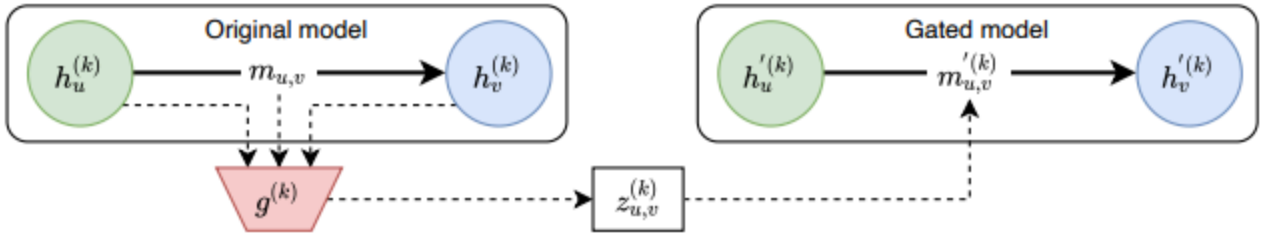

Figure 1: GRAPHMASK uses vertex hidden states and messages at layer k (left) as input to a classifier g that predicts a mask $z^{(\ell)}$. We use this to mask the messages of the kth layer and re-compute the forward pass with modified node states (right). The classifier g is trained to mask as many hidden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

To avoid hindsight bias, this paper use **amortising parameter learning**

$$\tilde{m}_{u,v}^{(k)} = z_{u,v}^{(k)}\, m_{u,v}^{(k)} + \left(1 - z_{u,v}^{(k)}\right) b^{(k)}$$

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

g_π is single-layer NN

parameter π is shared across the entire dataset → amortised learning

Parameter optimization

optimzie this and learn parameters

$$\max_{\lambda} \min_{\pi,\, b} \sum\nolimits_{(G,X) \in \mathcal{D}} \left(\sum_{k=1}^{L} \sum_{(u,v) \in E} \mathbf{1}\left[z_{u,v}^{(k)} \neq 0 \right] \right) + \lambda \left(\mathcal{D}\big(f(G, X) \parallel f(G_S, X) \big) - \beta \right)$$

D is divergence, GNN is represented as f

Synthetic Experiment

- problem settings
  graph shape: star graph G : one centroid v_0 and 6-12 leaf vertices.
  edge : each edge (u, v_0) is assigned one of 6 colors $c_{u,v} \in C$
  Query : Query $< x, y > \in C * C$
  task: predict whether the number of edges coloured x is greater than the number coloured y (binary output)
  evaluation goal : **an interpretation method should keep exactly the x and y edges (high recall) while discarding the rest (high precision), yielding near-perfect F1 if it is truly faithful.**
- model
  1- layer R-GCN + MLP → trained to 100% accuracy

Synthetic Experiment

Method	Prec.	Recall	F_1
Erasure search*	100.0	16.7	28.6
Integrated Gradients	88.3	93.5	90.8
Information Bottleneck	55.3	51.5	52.6
GNNExplainer	100.0	16.8	28.7
Ours (non-amortized)	96.7	26.2	41.2
Ours (amortized)	98.8	100.0	**99.4**

Table 1: Comparison using the faithfulness gold standard on the toy task. *as in Li et al. (2016).

-Take aways

1. Only amortised GraphMask recovers the exact ground-truth subgraph.
2. Per-instance optimisers (erasure, non-amortised) over-prune, confirming hindsight bias.
3. IG assigns dense scores; a single global threshold cannot cleanly separate useful vs. superfluous edges even in this toy task.

Question answering

- task
  a reasoning quiz taht stitches together evidence across multiple documents to answer question.
  eg.) Q : "what is the home country of the auther of Naruto?"
  Doc A → Doc B → Answer:Japan

Why is it hard?

- The answer isn’t contained in a single document—the model must “hop” between sources.
- Many entities and tangled evidence paths make it tough to track the relevant chain.

Question answering

graph structure in WikiHop Multihop QA model

nodes :

• Mention nodes – every occurrence of an entity name in the query or any support document.

• Query node – a special node that stores a sentence-level embedding of the question and is concatenated to each mention’s vector at layer 0.

Question answering

there are 4 edge types in this graph.
All edges are inserted bidirectionally, so information can flow both ways during message passing.

- task
  A multi-hop QA task where the question and multiple documents are represented as a graph, and the model must **select the single most appropriate entity as the answer**

2-layer BiLSTM + 3-layer R-GCN is trained to 59% accuracy

Question answering

Edge Type	k = 0	k = 1	k = 2
MATCH (8.1%)	9.4%	11.1%	8.9%
DOC-BASED (13.2%)	5.9%	17.7%	10.7%
COREF (4.2%)	4.4%	0%	0%
COMPLEMENT (73.5%)	31.9%	0%	0%
Total (100%)	51.6%	28.8%	19.6%

Table 2: Retained edges for De Cao et al.'s (2019) question answering GNN by layer (k) and type.

Table 2 reports the edge statistics after applying GRAPHMASK—i.e., it counts only the edges whose gates remained open

Question answering

GRAPHMASK worked as followed and produce key results

1. Gate-based edge pruning
2. Minimal subgraph extraction

- only 27% of edges were retained across all layers, meaning 73 were safely pruned with **no retaining** of the base mdoel

3. Accuracy preserved

- original accuracy 59%
- after masking **58.6%(-0.4%)**

Question answering

4. Layer-wise retention patterns

   Layer 0 (bottom): 51.6 % of edges kept

   Layer 1: 28.8 %

   Layer 2 (top): 19.6 %

5. Edge-type dynamics

- Layer 0 dominated by COMPLEMENT edges—used for broad context distribution.
- Layers 1–2 increasingly favor MATCH and DOC-BASED edges—used for precise, high-confidence evidence routing.

Question answering

-take aways

GraphMask distilled the complex, multi-relational QA graph down to just 27 % of its edges —without hurting performance—and revealed exactly **which layers, edge types, and multi-hop routes** the model actually uses for reasoning.

conclusion and future work

- GraphMask enables edge-level and path-level explanations.
- Model accuracy stays virtually unchanged after masking → the method isolates the truly essential information-flow paths.

Next steps:

- Apply GraphMask to GNNs in other domains (e.g., chemistry, transportation).
- Extend the approach to attention-based GNN architectures.
- Develop a user-friendly visualization toolkit for practitioners.