GNN IS A COUNTER? REVISITING GNN FOR QUESTION ANSWERING

ICLR 2022

인턴연구생 강규란

Introduction

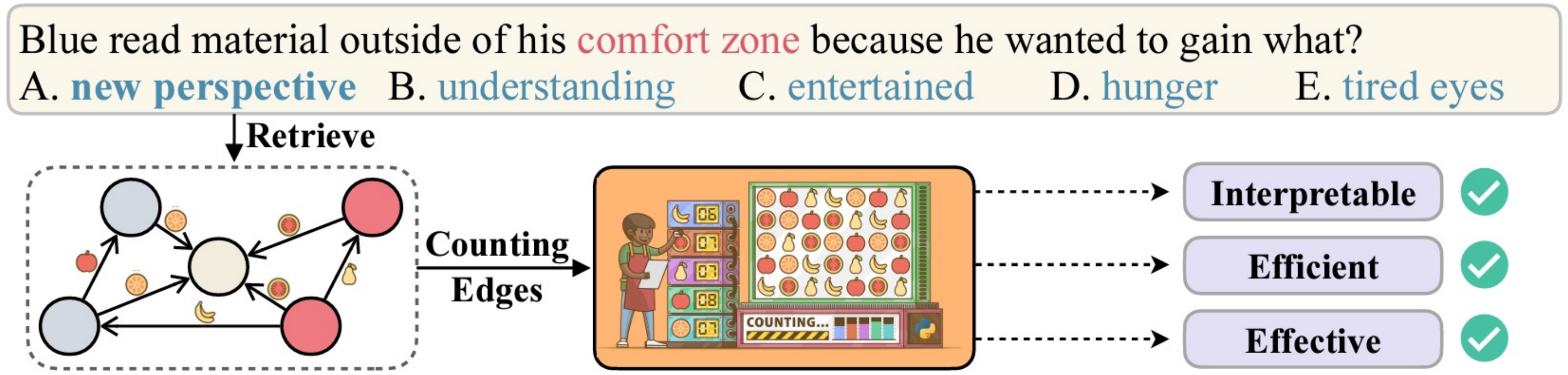
Questions about Today's QA Systems

- Are current GNN-based modules under- or overcomplicated for QA?
- What is the essential role they play in reasoning over knowledge?

Introduction

Analysis + Proposed Model: “GSC”

- Analysis of existing GNN modules: SparseVD as a diagnostic tool to analyze the importance of various parts of SOTA knowledge-aware GNN modules → GNN modules are over-complicated!
- Importance of edge counting: counting of edges in the graph plays a crucial role in knowledge-aware reasoning
- Design of GSC module:
    - proposing Graph Soft Counter (GSC), less than 1% trainable parameters compared to existing GNN modules
    - outperforms complex GNN modules on two benchmark

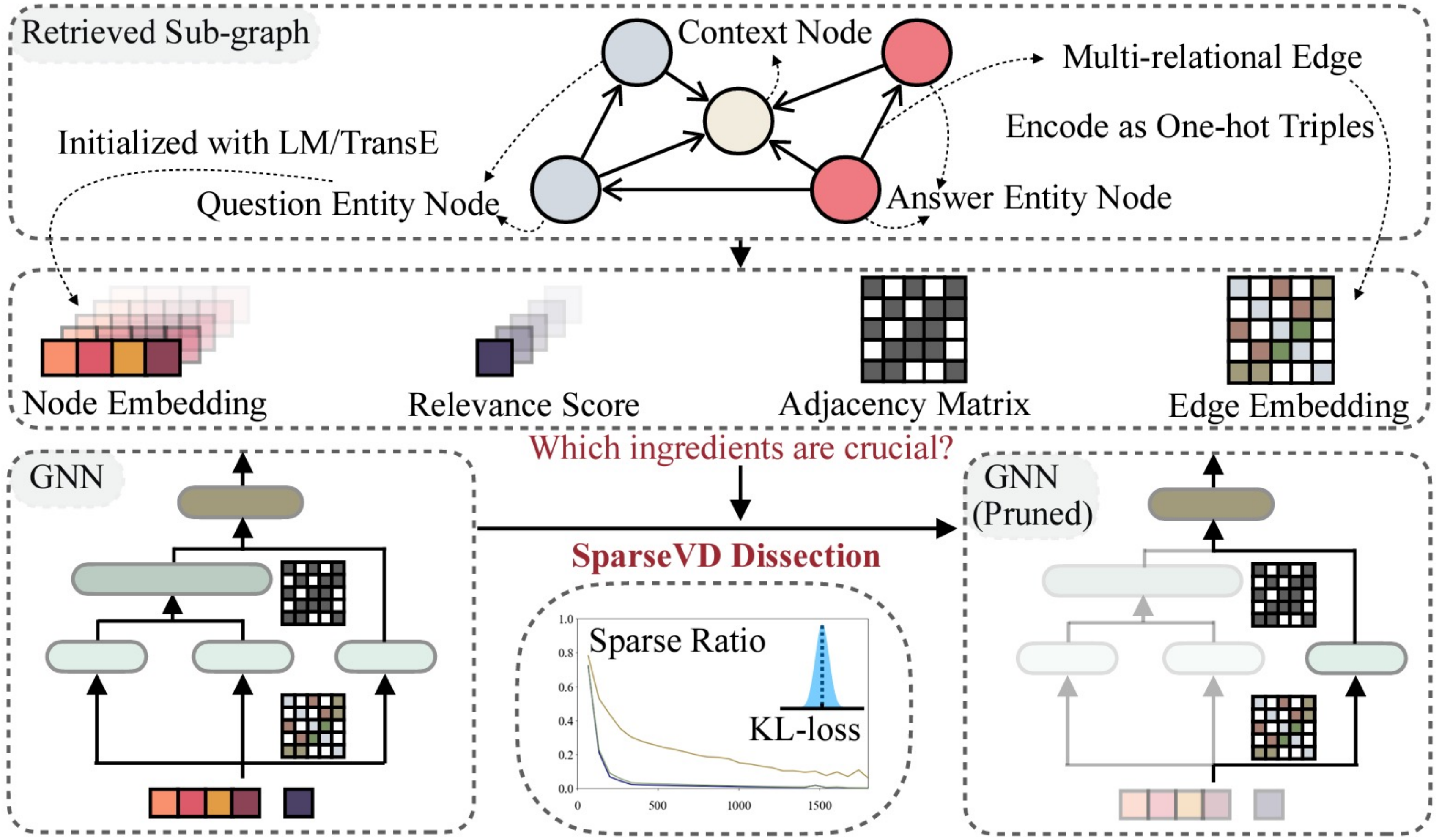


Analysis

Preliminaries

- LM : pool embedding of the start token as the sentence embedding + MLP to map the sentence embedding to the score for the choice
- KG subgraph of triplets of concepts from QA (same as Lin and Yasunaga)
- Node embeddings: existing works initialize node embedding with external embeddings
- Relevance score: extra embedding of relevance score in the input node feature to estimate the importance of KG nodes relative to QA context
- Adj matrix: typically converted to be symmetric before feeding into the GNN module
- Edge embeddings: concatenated one-hot vector to encode edge triplets

$$[u_s, e_{st}, u_t]$$



Analysis

Dissection

- SparseVD
  - A neural model pruning method
  - Use to investigate which parts of GNN can be pruned out (sparse ratio to zero) without loss of accuracy → meaning that part of the model is redundant

	w/o SparseVD		w/ SparseVD	
Methods	**IHdev-Acc.** (%)	**IHtest-Acc.** (%)	**IHdev-Acc.** (%)	**IHtest-Acc.** (%)
KagNet (Lin et al., 2019)	73.47 (±0.22)	69.01 (±0.76)	75.18 (±1.05)	70.48 (±0.77)
MHGRN (Feng et al., 2020)	74.45 (±0.10)	71.11 (±0.81)	77.15 (±0.32)	72.66 (±0.61)
QAGNN (Yasunaga et al., 2021)	76.54 (±0.21)	73.41 (±0.92)	77.64 (±0.50)	73.57 (±0.48)

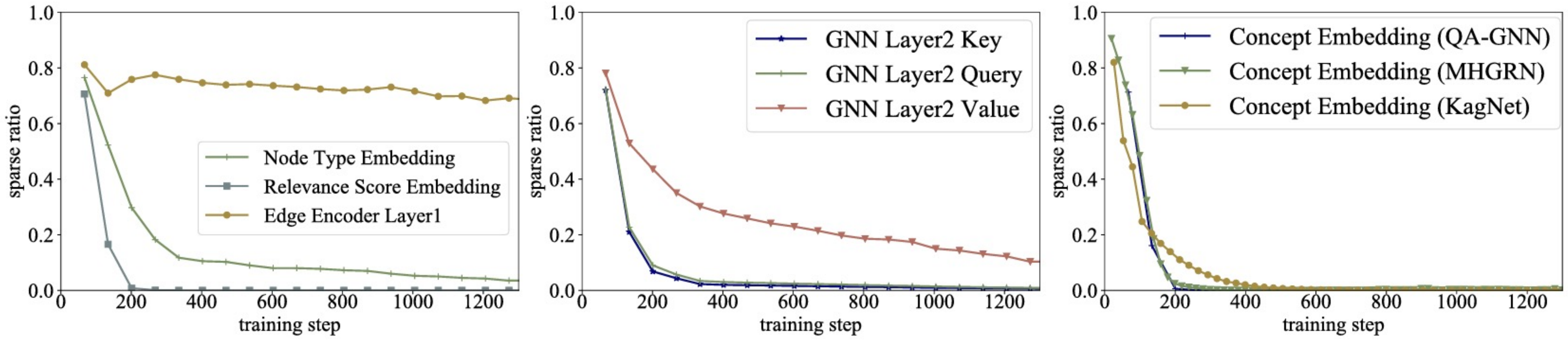
Table 1: To preserve the reasoning ability for analysis, our SparseVD tool prunes the GNN-based models without loss of accuracy on *CommonsenseQA* dataset. As the official test is hidden, here we report the in-house dev (IHdev) and test (IHtest) accuracy, following the data split of Lin et al. (2019).

Analysis

Obsercation and Hypothesis

1) Edge encoder that encodes edge triplets preserves a relatively higher sparse ratio, Node embedding can be fully pruned as well as relevance score
2) Layers inside GNN: all sparse ratios are low, while the value layer has relatively higher sparse ratio than key/query layers
3) Initial node embeddings can be completely discared

1) Node embeddings and relevance score are dispensable
2) Edge encoder layers are hard to prune → edge/node type info are essential to reasoning
3) Message passing layers can be pruned to a very low sparse ratio → GNN may be over parameterized, we only need fewer params for these layers
4) Graph pooler : key/query layers inside the pooler can be pruned out; graph pooler reduced to a linear transformation



Proposed Model: Graph Soft Counter

GSC Architecture

- **Only TWO basic components in GSC**
  - **Edge encoder**
  - **GSC layers**
  - Get QA choice score by simply summing up the graph score and context score

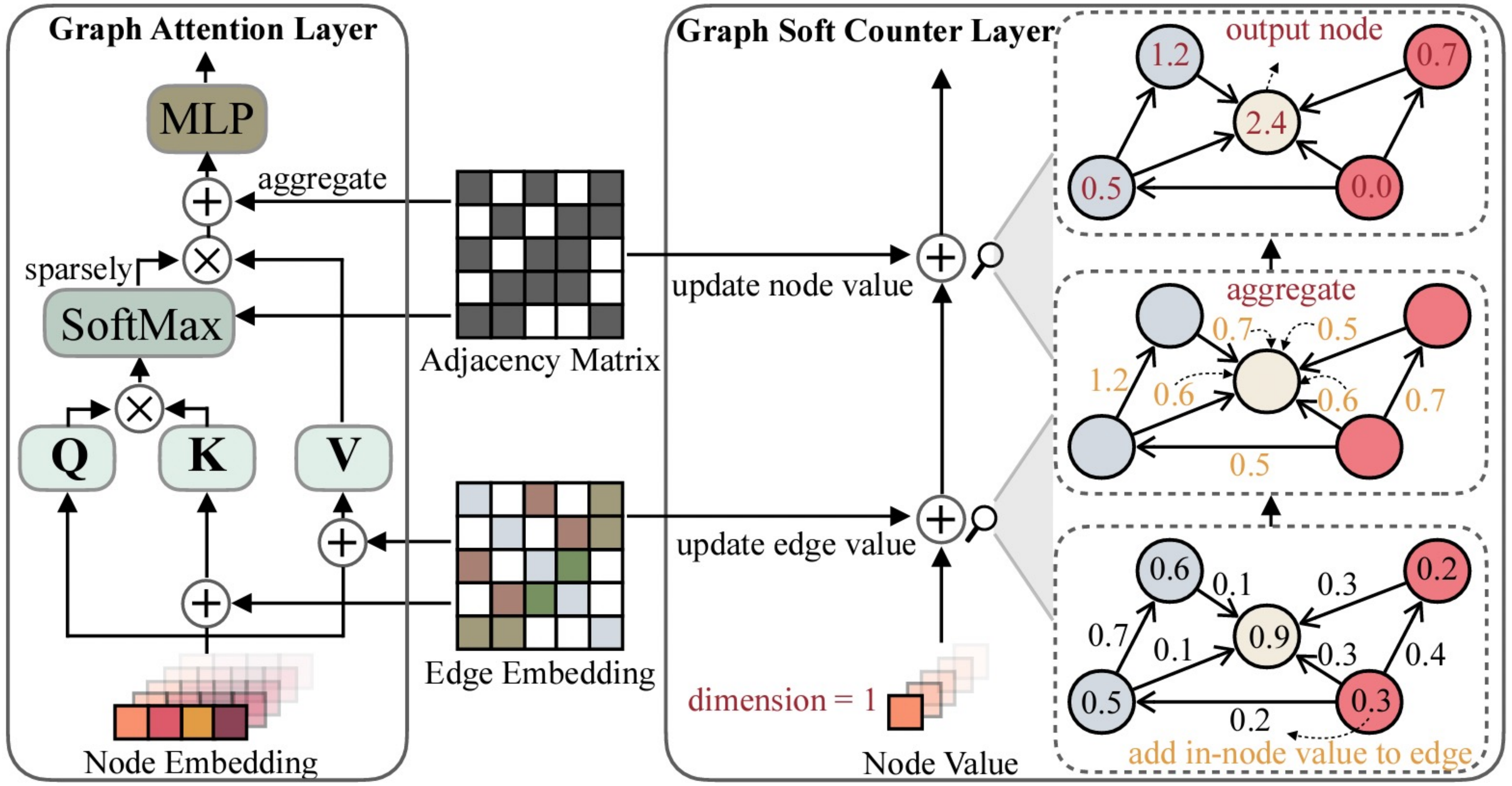
Algorithm 1 PyTorch-style code of GSC

```
# qa_context: question answer pair context
# adj: edge index with shape 2 x N_edge
# edge_type: edge type with shape 1 x N_edge
# node_type: node type with shape 1 x N_node
edge_emb = edge_encoder(adj, edge_type, node_type)
node_emb = torch.zeros(N_node)
for i_layer in range(num_gsc_layers):
   # propagate from node to edge
   edge_emb += node_emb[adj[1]]
   # aggregate from edge to node
   node_emb = scatter(edge_emb, adj[0])
graph_score = node_emb[0]
context_score = fc(roberta(qa_context))
qa_score = context_score + graph_score
```

Proposed Model: Graph Soft Counter

GSC Architecture

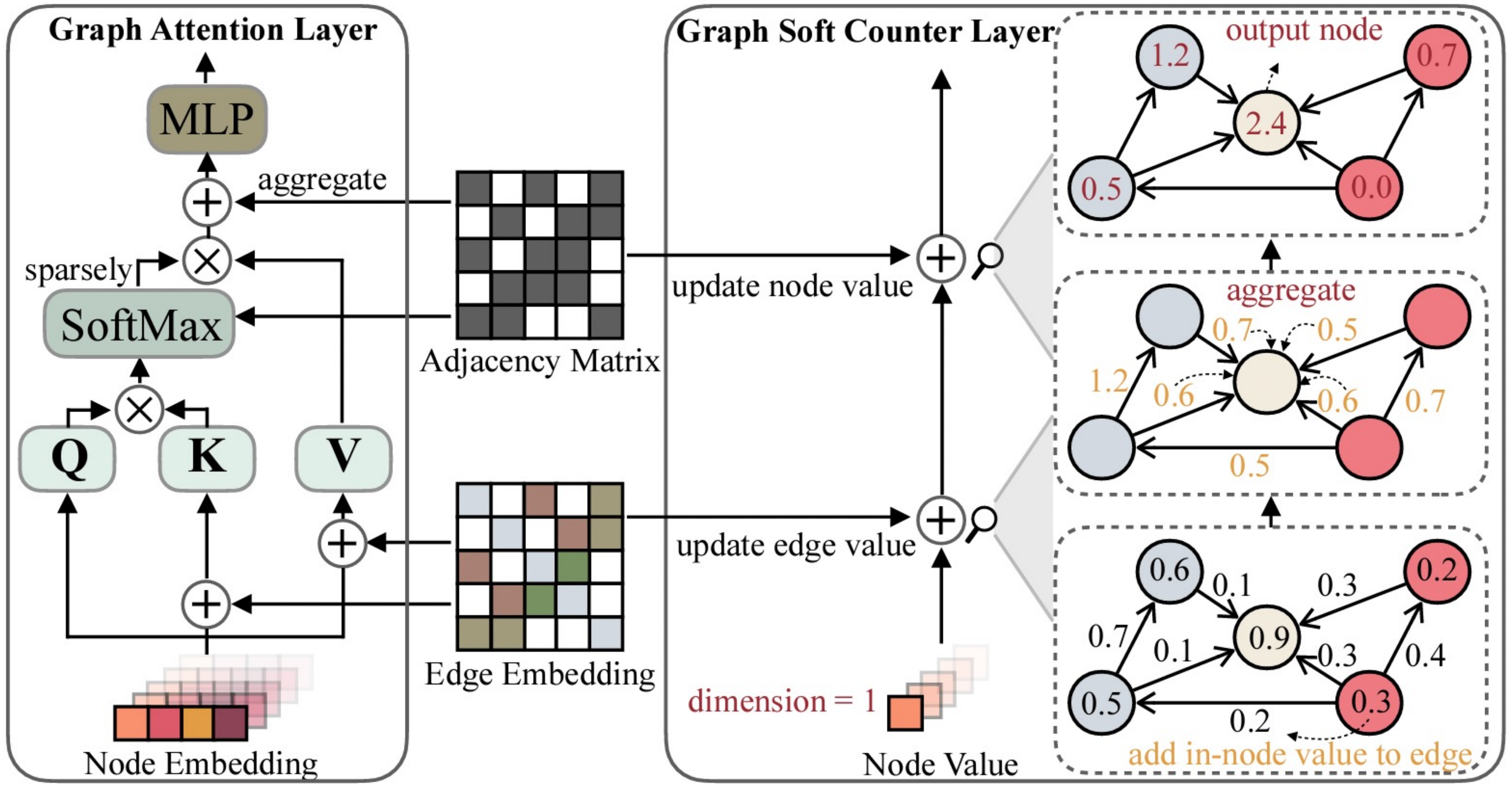
- **Only TWO basic components in GSC**
  - **Edge encoder**
    - two-layer MLP followed by a Sigmoid function. The triplets is represented as a concatenated one-hot vector $[u_s, e_{st}, u_t]$
  - **GSC layers**
  - Get QA choice score by simply summing up the graph score and context score



Proposed Model: Graph Soft Counter

GSC Architecture

- **Only TWO basic components in GSC**
  - **Edge encoder**
  - **GSC layers**
    - GSC layers are parameter-free, only keep basic graph operations; propagation and aggregation (2 layers):
      1) Update each edge embedding with incoming node(in-node) embeddings
      2) Update each node embedding by aggregating the edge embeddings.
  - Get QA choice score by simply summing up the graph score and context score



Proposed Model: Graph Soft Counter

GSC Architecture

- **Only TWO basic components in GSC**
  - **Edge encoder**
  - **GSC layers**
    - GSC layers are parameter-free, only keep basic graph operations; propagation and aggregation (2 layers):

      1) Update each edge embedding with incoming node(in-node) embeddings

      2) Update each node embedding by aggregating the edge embeddings.
  - Get QA choice score by simply summing up the graph score and context score

	KagNet	MHGRN	QAGNN	**GSC** (Ours)
Adj-matrix	✓	✓	✓	✓
Edge-type	✓	✓	✓	✓
Node-type	×	✓	✓	✓
Node-embedding	✓	✓	✓	×
Relevance-score	×	×	✓	×
#Learnable Param	700k	547k	2845k	**3k**
Model size	819M	819M	821M	**3k**

Model	**Time**	**Space**
$\mathcal{G}$ is a dense graph		
L-hop KagNet	$\mathcal{O}\left(\lvert\mathcal{R}\rvert^L \lvert\mathcal{V}\rvert^{L+1} L\right)$	$\mathcal{O}\left(\lvert\mathcal{R}\rvert^L \lvert\mathcal{V}\rvert^{L+1} L \cdot D\right)$
L-hop MHGRN	$\mathcal{O}\left(\lvert\mathcal{R}\rvert^2 \lvert\mathcal{V}\rvert^2 L\right)$	$\mathcal{O}\left(\lvert\mathcal{R}\rvert \lvert\mathcal{V}\rvert L \cdot D\right)$
L-layer QAGNN	$\mathcal{O}\left(\lvert\mathcal{V}\rvert^2 L\right)$	$\mathcal{O}\left(\lvert\mathcal{R}\rvert \lvert\mathcal{V}\rvert L \cdot D\right)$
L-layer GSC	$\mathcal{O}\left(\lvert\mathcal{V}\rvert L\right)$	$\mathcal{O}\left(\lvert\mathcal{R}\rvert \lvert\mathcal{V}\rvert L\right)$
$\mathcal{G}$ is a sparse graph with maximum node degree $\Delta \ll \lvert\mathcal{V}\rvert$		
L-hop KagNet	$\mathcal{O}\left(\lvert\mathcal{R}\rvert^L \lvert\mathcal{V}\rvert L \Delta^L\right)$	$\mathcal{O}\left(\lvert\mathcal{R}\rvert^L \lvert\mathcal{V}\rvert L \Delta^L \cdot D\right)$
L-hop MHGRN	$\mathcal{O}\left(\lvert\mathcal{R}\rvert^2 \lvert\mathcal{V}\rvert L \Delta\right)$	$\mathcal{O}\left(\lvert\mathcal{R}\rvert \lvert\mathcal{V}\rvert L \cdot D\right)$
L-layer QAGNN	$\mathcal{O}\left(\lvert\mathcal{V}\rvert L \Delta\right)$	$\mathcal{O}\left(\lvert\mathcal{R}\rvert \lvert\mathcal{V}\rvert L \cdot D\right)$
L-layer GSC	$\mathcal{O}\left(\lvert\mathcal{V}\rvert L\right)$	$\mathcal{O}\left(\lvert\mathcal{R}\rvert \lvert\mathcal{V}\rvert L\right)$

Experiment

Setup

- CommonsenseQA, OpenBookQA
- LM : CommonsenseQA(RoBERTa-Large), OpenBookQA(RoBERTa-Large, AristoRoBERTa)
- KG : ConceptNet

Results

Commonsense QA

Methods	IHdev-Acc. (%)	IHtest-Acc. (%)
RoBERTa-large (w/o KG)	73.07 (±0.45)	68.69 (±0.56)
+ RGCN (Schlichtkrull et al., 2018)	72.69 (±0.19)	68.41 (±0.66)
+ GconAttn (Wang et al., 2019)	72.61(±0.39)	68.59 (±0.96)
+ KagNet (Lin et al., 2019)	73.47 (±0.22)	69.01 (±0.76)
+ RN (Santoro et al., 2017)	74.57 (±0.91)	69.08 (±0.21)
+ MHGRN (Feng et al., 2020)	74.45 (±0.10)	71.11 (±0.81)
+ QAGNN (Yasunaga et al., 2021)	76.54 (±0.21)	73.41 (±0.92)
+ GSC (Ours)	**79.11** (±0.22)	**74.48** (±0.41)

Table 4: Performance comparison on *Commonsense QA*

Methods	Test
RoBERTa (Liu et al., 2019)	72.1
RoBERTa + FreeLB (Zhu et al., 2019) (ensemble)	73.1
RoBERTa + HyKAS (Ma et al., 2019)	73.2
RoBERTa + KE (ensemble)	73.3
RoBERTa + KEDGN (ensemble)	74.4
XLNet + GraphReason (Lv et al., 2020)	75.3
RoBERTa + MHGRN (Feng et al., 2020)	75.4
ALBERT + PG (Wang et al., 2020)	75.6
RoBERTa + QA-GNN (Yasunaga et al., 2021)	76.1
ALBERT (Lan et al., 2019) (ensemble)	76.5
UnifiedQA (11B)* (Khashabi et al., 2020)	**79.1**
RoBERTa + GSC (Ours)	76.2

Table 5: Test accuracy on *CommonsenseQA*'s official leaderboard. The previous top system, UnifiedQA (11B params) is 30x larger than our model.

Results

OpenBook QA

Methods	RoBERTa-large	AristoRoBERTa
Fine-tuned LMs (w/o KG)	64.80 (±2.37)	78.40 (±1.64)
+ RGCN	62.45 (±1.57)	74.60 (±2.53)
+ GconAtten	64.75 (±1.48)	71.80 (±1.21)
+ RN	65.20 (±1.18)	75.35 (±1.39)
+ MHGRN	66.85 (±1.19)	80.6
+ QAGNN	67.80* (±2.75)	82.77 (±1.56)
+ GSC (Ours)	**70.33** (±0.81)	**86.67** (±0.46)

Table 6: Test accuracy on *OpenBookQA*. Methods with AristoRoBERTa use the textual evidence by Clark et al. (2019) as an additional input to the QA context.

Discussion

Maximum number of retrieved node

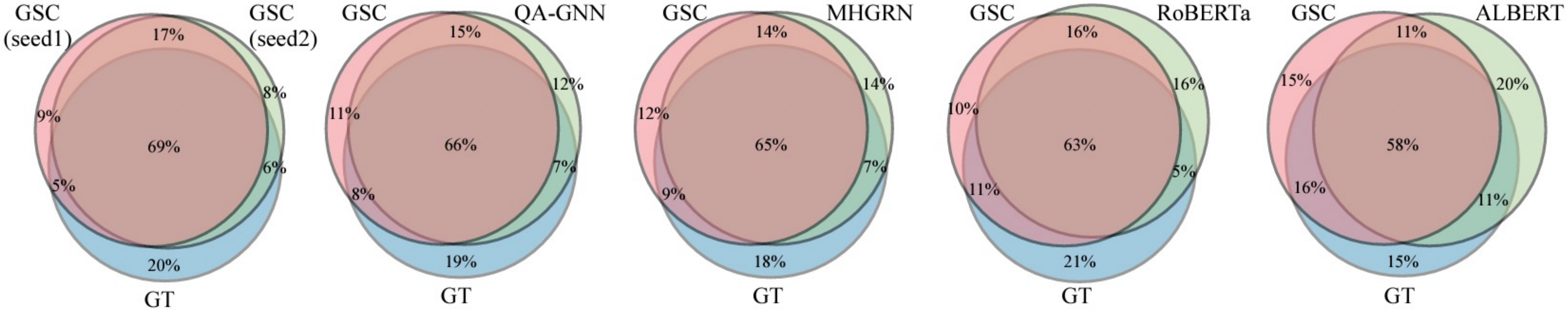
- 1-hop retrieval is adequate for GSC methods, and this could be done super efficiently than multi-hop retrieval.
- Handcrafted hard edge counting feature feeding into two-layer MLP
  - Hard counting model with 2-hop edge feature : achieve comparable performance, even outperforms other GNN baselines.
  - GSC is effective + counting is an essential functionality in knowledge-aware QA system.

Methods	IHdev-Acc. (%)	IHtest-Acc. (%)
MLP + Counter (1-hop)	78.02 (±0.05)	73.62 (±0.12)
MLP + Counter (2-hop)	78.30 (±0.09)	74.13 (±0.08)
GSC w/ QA nodes	79.11 (±0.22)	74.48 (±0.41)
w/ 32 nodes	78.52 (±0.58)	74.40 (±0.25)
w/ 64 nodes	78.53 (±0.77)	73.93 (±1.09)
w/ 128 nodes	78.50 (±0.96)	72.78 (±1.15)
w/ 256 nodes	78.32 (±0.60)	73.89 (±0.63)

Discussion

Prediction Overlap

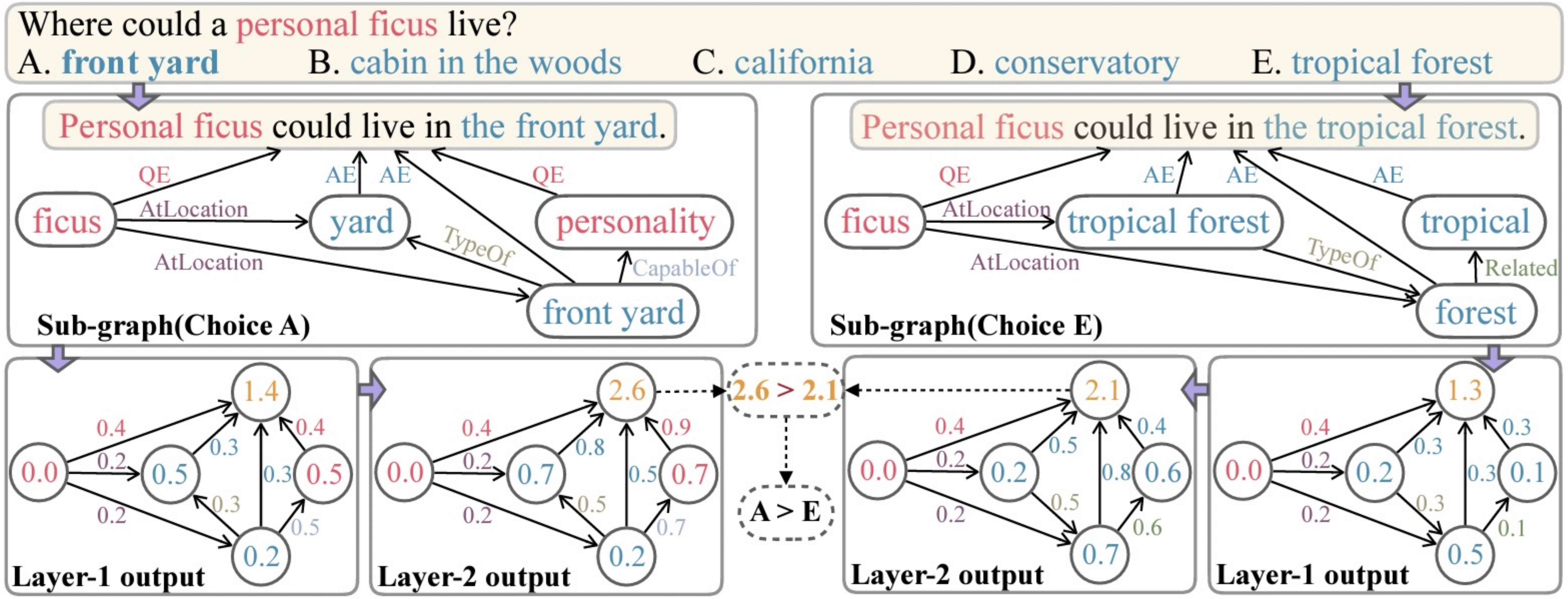
- Datasets are relatively noisy and there exists decent variance in the prediction
- GSC has a larger overlap for GNN-based systems
- Order of the percentage of overlap of left four exactly matches the order of the performance of each model: GSC > QA-GNN > MHGRN > RoBERTa. GSC has quite similar behaviors versus other GNN based systems, reasoning capability on par with existing GNN counter parts.



Discussion

Interpretability

- For retrieved sub-graph of each answer choice,
  - We can observe behavior of GSC by printing out output edge/node values of each layer.



Conclusion

- Current GNN-based QA systems are over-parameterized and over-complex
- Initial node embeddings and some GNN layers are dispensable
- GNN essentially works as a counter in the QA reasoning process

Thank you!

인턴연구생 강규란