Connecting the Dots: Explaining Human Reasoning on the Graph A Case Study on Deep Question Generation

Yong Liang Goh[*], **Yongbin Li**[*], **Yisong Miao** [*]
National University of Singapore, School of Computing
gyl@u.nus.edu, yongbinli@u.nus.edu, yisong@comp.nus.edu.sg

Abstract

Deep Question Generation (DQG) involves generating a complex question given an input passage. This task involves reasoning over multiple sources of information in the passage. To tackle this, Pan et al. (2020) leverages recent advances in graph neural networks (GNN) to represent the passage as a collection of phrases, where each phrase forms a node in a graph. Then, to reason over this graph, information is aggregated via a confined graph structure to combine them to generate meaningful questions. However, what is unclear at this moment is *why* is the graph important, *which* structures are important, and *if* this human imposed structure is even required. This paper looks deeper into answering these 3 main questions via the following approaches: (1) A linguistic heuristic approach to investigate the linguistic structures that are crucial for reasoning in DQG, which can be used as structural priors for building better reasoning models, (2) A GNN learnable approach to investigate the computational structure the model uses and if this aligns with the human one, (3) Generalizing the GNN model used to allow the model to learn its own computational graph structure for the reasoning task.

1 Introduction

How do humans compose a complex and natural sentence over multiple pieces of concepts, evidence, and beliefs? This question has puzzled computer and cognitive scientists for decades. Such composition is a complex process. It requires a broad range of abilities, including analyzing the syntax of the text, understanding the semantic relation among concepts, structurally combining multiple evidence together, etc. With the advent of natural language processing and graph learning techniques, we are rapidly approaching this unsolved question. Notably, a cohort of researchers is formalizing NLP tasks as reasoning on graphs, achieving superior results.

One notable progress is made by Pan et al. (2020), where they introduced the use of recent advances in GNNs together with encoder-decoder based frameworks for DQG. In a nutshell, DQG involves encoding word representations as a document, followed by constructing phrases from these words as a node. A graph is then constructed on these nodes, where they use either Dependency Parsing (DP) or Semantic Role Labeling (SRL). These graphs are built based on some form of human inductive bias, and passed into a gated graph attention network. The nodes get updated based on the structure and the edge types in the graph. Once done, these representations are used for a content selection task, whereby phrases that appear in the final generated question (ground truth) are considered as positive labels and otherwise. Additionally, these node representations are combined with a document-level representation, and fed into an attention decoder for question generation. The model is trained on both content selection and question generation. Readers are encouraged to read Pan et al. (2020)'s work for further details.

While the DQG model proposed demonstrates promising results, it has inherent limitations of using and interpreting the semantic graph:

- Node level: Their method fails to explain why a node is important. Although the authors did content selection analysis, the scalar attention score is still not explainable. The real explanation should elaborate on the intrinsic properties of a node and local subgraph structure.
- Edge level: Their method actually produces a "reasoning chain" (*c.f.* Figure 3 in (Pan et al. 2020)) by a few useful edges connecting together. However, they did not analyze and explain what edges are important.
- Subgraph level: From *c.f.* Figure 3 in (Pan et al. 2020) we can see certain subgraph structures are crucial for the reasoning. However, they did not explain the crucial subgraph structure and leave the irrelevant nodes being the noise for further steps.
- Learning level: The authors assume that the graphs take on a total of 3 relationships, as defined by human heuristics. However, reasoning for models might not necessarily be so; how humans construct graphs for reasoning might be entirely different from how machines do so.

These limitations in this particular DQG paper motivate us to reflect on what is being needed for explaining the reasoning on graphs. We have generated several key research questions (RQs) to facilitate the understanding of reasoning on the graph:

[*]Ordered alphabetically, work done during CS6208-2021.

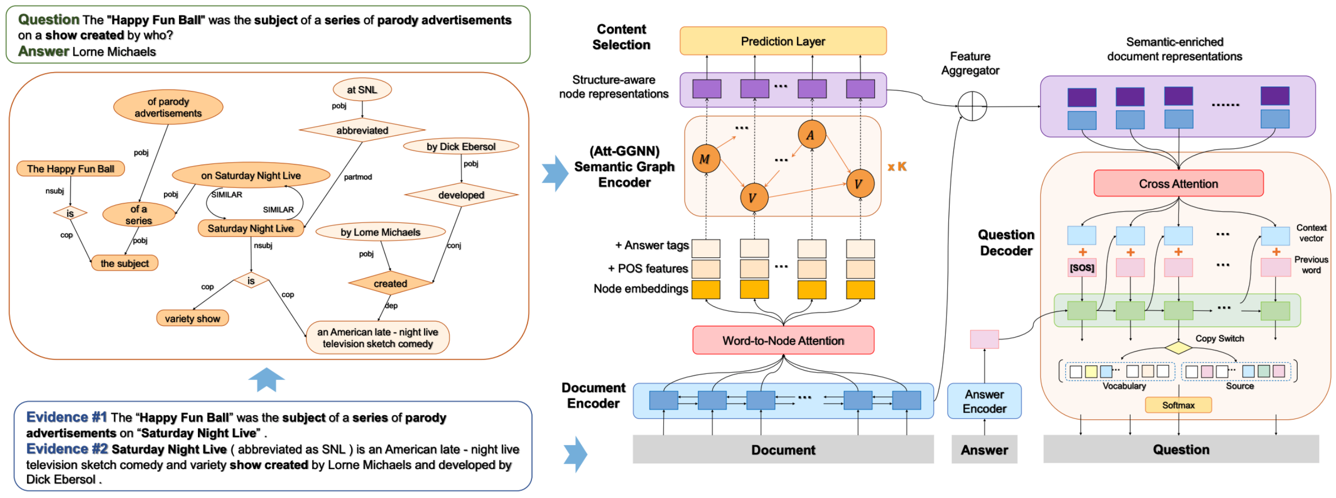
Question The "Happy Fun Ball" was the subject of a series of parody advertisements on a show created by who?
Answer Lorne Michaels
of parody advertisements
at SNL
abbreviated
by Dick Ebersol
The Happy Fun Ball
on Saturday Night Live
of a series
is
Saturday Night Live
developed
by Lorne Michaels
the subject
is
created
variety show
an American late - night live television sketch comedy
SIMILAR
pobj
nsubj
cop
partmod
conj
dep
Evidence #1 The "Happy Fun Ball" was the subject of a series of parody advertisements on "Saturday Night Live" .
Evidence #2 Saturday Night Live (abbreviated as SNL) is an American late - night live television sketch comedy and variety show created by Lorne Michaels and developed by Dick Ebersol .
Content Selection
Prediction Layer
Structure-aware node representations
(Att-GGNN) Semantic Graph Encoder
x K
+ Answer tags
+ POS features
Node embeddings
Word-to-Node Attention
Document Encoder
Document
Answer Encoder
Answer
Feature Aggregator
Semantic-enriched document representations
Cross Attention
Question Decoder
Context vector
Previous word
[SOS]
Copy Switch
Vocabulary
Source
Softmax
Question

Figure 1: The DQG framework of Pan et al. (2020)

- **RQ1:** What structures and innate priors are essential for reasoning on graphs?
- **RQ2:** Can we distill essential subgraph structures and are these substructures similar to human bias?
- **RQ3:** Do machines require human-level heuristics in learning?

RQ1 motivates a linguistic heuristic approach by looking at essential node features and edge features. It is crucial for us to understand the inherent process of human and machine reasoning on graphs, which will in turn help the NLP community build better reasoning models. RQ2 motivates a GNN learnable approach to find subgraph structures that are crucial for question generation. Specifically, we adapt GNNEXPLAINER (Ying et al. 2019) into our scenario. We investigate if the computational structure aligns with the human heuristics of reasoning. RQ3 motivates generalizing the DQG model such that it is allowed to freely learn the relationships between nodes. Specifically, we generalize the author's graph attention network (GAT) (Veličković et al. 2017) to a multi-headed one, where we postulate that if we have more heads than edge types (in (Pan et al. 2020)'s work, 3 edge types are defined), we should be capable for learning beyond the human interpretation of what the edges should be.

To this end, we arrive at the following conclusions:

- Based on linguistic heuristic, we identify crucial node features and edge features that are decisive for reasoning on graphs for DQG. It is intrinsically interesting because it brings empirical evidence to demystify the reasoning on graphs.
- With our developed framework, we are able to find an essential subgraph structure for any input node of DQG in a learnable approach. We find geometric features differ for different types of nodes. We also locate the overlap between heuristic and learnable approaches.
- Human-level interpretation, at this moment, is required. Allowing the model to freely learn the relationships appears to harm it more than help. We find that the content selection task begins to under-perform when generalized to a multi-headed attention approach. Consequently, the quality of question generation reduces when this happens.

2 Related Work

2.1 Graphs for NLP

The NLP community has made decades of efforts on incorporating graph structures to incorporate graph structures to improve the performance, robustness, and explainability, for different NLP tasks. These efforts can be categorized into two streams:

(1) Building knowledge graphs for NLP: WordNet (Strapparava, Valitutti et al. 2004) is the first large-scale knowledge graph describing the semantic relations between words. Its release has enabled a fast breakthrough in NLP in the 2000s. WordNet is then extended by ConceptNet (Speer, Chin, and Havasi 2017). Recently, more commonsense knowledge graphs are built to satisfy the deep understanding of NLP tasks (Sap et al. 2019). One notable recent effort is to distill a knowledge graph from scratch from BERT pretrained model (Wang, Liu, and Song 2020).

(2) Using knowledge graphs to facilitate NLP tasks: Knowledge graphs are shown to help in a wide range of NLP tasks, including question answering (Huang et al. 2019), natural language inference (Wang et al. 2019), discourse relation recognition (Dai and Huang 2019). A recent effort in (Lei et al. 2020) is using knowledge graph for conversational recommendation. The path reasoning on the graph keeps track of a conversation session. It brings improvement and explainability to the conversation.

2.2 Explanations for GNN

Graph neural networks have emerged recently, with the primary framework of neural message passing (Gilmer et al. 2017). Essentially, given a graph object, we can formulate local messages as linear projections of representations, followed by permutation-invariant functions to summarize local neighborhoods. This framework has found its way to a variety of tasks, such as key-point matching in images (Zhang and Lee 2019) (Fey et al. 2020), semantic role labeling (Marcheggiani and Titov 2017), question-answering via knowledge graphs (Saxena, Tripathi, and Talukdar 2020), etc. This variety of tasks showcases the flexibility and applicability of graph structures. At the same time, it suggests the innate priors and biases it contains based on how data is structured or generated. Consequently, our community has started reflecting on the success and expressiveness of GNN models. GNNEXPLAINER model by Ying et al. (2019) was proposed to find a compact subgraph structure that is essential for model decision. One very contemporary work by Schlichtkrull, Cao, and Titov (2021) designs a differentiable masking strategy to explain GNNs for NLP tasks. However, none of these works pay attention to explain the intrinsic process of reasoning on the graph for complex NLP tasks.

3 Method

Notations	Meaning
v	Node. In our case, nodes are tokens or entites. $v \in V$, the entire node set
$\mathbf{x}$	Feature of a node (aka node embedding)
e	Edge. In our cases, edges are dependencies between tokens and entities. $e \in E$, the entire edge set.
v^+	Relevant node, meaning the node is being used in the ground truth question.
G	The whole semantic graph, containing all v and e.
$G^{'}$	A subgraph produced by GNNEXPLAINER given one v^+.
$E^+(i)$	Incoming edges for a node v_i
$E^-(i)$	Outgoing edges for a node v_i

Table 1: Key notations in this paper.

Formally, a graph is defined by a tuple (V, E), where $v \in V$ is a set of vertices and $e \in E$ a set of edges. An edge e_{ij} denotes a connection between vertex v_i and v_j. In this work, graphs are defined as per (Pan et al. 2020), where we have a DP graph and a SRL graph. Each node is associated with a collection of words, $\{w_j\}_{j=m_v}^{n_v}$, where m_v and n_v denotes the starting / ending position of the text span. We leverage the implementations of Pan et al. (2020) for our work.

3.1 Linguistic Heuristic Approach

As we discussed before, reasoning on graphs to compose a deep question is an extremely complex process. As a first attempt to demystify this problem, we focus on the core component of the DQG model, which is the **content selection** module. The content selection module is designed to predict whether a node will be used in the final question generated. It is crucial for generating high-quality questions. In the meantime, content selection module is a **simple node classification** task. Such simplicity also drives us to focus on it in this initiative exploration.

In order to identify what innate priors are crucial for reasoning on graphs, we take inspiration from linguistics and design several probings:

- **Node Probings:** This probing aims to answer what node-level linguistic properties are essential for content selection? These linguistic properties include part of speech, whether the node is in the answer, etc. Our strategy is to mask a specific segment in the node embedding **x** and observe the performance changes for content selection.
- **Edge Probings:** This probing aims to answer what edge-level linguistic properties are essential for content selection? We adopt a simplistic edge masking solution. To be specific, we group the edges e in DQG by linguistic categories. We mask one type of edge each time and observe performance changes. A similar edge masking strategy is adopted in a contemporary ICLR spotlight paper (Schlichtkrull, Cao, and Titov 2021).

3.2 GNN Learnable Approach

In order to investigate how the machine works for a node classification problem, we use GNNEXPLAINER, which can provide interpretable explanations for predictions of a GNN-based model.

GNNEXPLAINER takes a trained GNN and its prediction(s) as input, and it generates an explanation in the form of a compact subgraph structure and a small set of node features that are important for this task, as shown in Figure 2.

Given a node v, GNNEXPLAINER uses n-hop neighborhoods of the given node to find the subgraph G_S with corresponding features X_S. The optimization framework of GNNEXPLAINER is:

$$\max_{G_S} MI\left(Y,(G_S,X_S)\right) = H(Y) - H\left(Y \mid G=G_S, X=X_S\right) \tag{1}$$

where MI is the mutual information and Y are labels. MI quantifies the change in the probability of prediction when $v's$ computation graph is limited in the subgraph. Since GNNEXPLAINER works for a trained GNN, the entropy term $H(Y)$ is constant. Maximizing the above equation is equivalent to minimizing conditional entropy $H\left(Y \mid G=G_S, X=X_S\right)$, which can be optimized using gradient descent after further transformation.

Although the author claims this approach is model-agnostic, it is not easy to combine the complex DQG model with GNNEXPLAINER directly. Therefore, we only consider the most simplified case, training GNNEXPLAINER's default pure GNN model with the data used in DQG. Then we use GNNEXPLAINER to explain all node $v^+ \in V^+$ to get all explained subgraphs.

3.3 Generalizing Learning of the Graph Structure

A key feature in (Pan et al. 2020) is the construction of the DP and SRL to form the graph structure. Then, it is assumed that there exist 3 key relationships in these graphs: predicate, subject/object, modifier. Each relationship is represented by its own learnable weight matrix $W^{t_{ij}}$, and neighboring node representations are aggregated according to these relationships. These weight matrices are combined with a common set of attention coefficients α to aggregate and attend to neighboring nodes accordingly. Based on the node types, the authors mask nodes such that they are of one of the 3 relationships. This suggests that there's a human bias imposed on this; only nodes of the same type can share a particular relationship and receive information from fellow nodes. The following equations (taken verbatim from (Pan et al. 2020)) showcase how the authors construct the necessary message passing and aggregation functions:

$$\mathbf{h}_{\mathcal{N}_{E^+}(i)} = \sum_{v_j \in \mathcal{N}_{E^+(i)}} \alpha_{ij}^{(k)} \mathbf{W}^{t_{e_{ij}}} \mathbf{h}_j^{(k)} \tag{2}$$

$$\mathbf{h}_{\mathcal{N}_{E^-}(i)} = \sum_{v_j \in \mathcal{N}_{E^-(i)}} \alpha_{ij}^{(k)} \mathbf{W}^{t_{e_{ij}}} \mathbf{h}_j^{(k)} \tag{3}$$

$$\alpha_{ij}^{(k)} = \frac{\exp(Attn(\mathbf{h}_i^{(k)}, \mathbf{h}_j^{(k)}))}{\sum_{t \in \mathcal{N}_{(i)}} \exp(Attn(\mathbf{h}_i^{(k)}, \mathbf{h}_t^{(k)}))} \tag{4}$$

where $Attn(\cdot, \cdot)$ is implemented as a single-layer neural network such that $\mathbf{a}^\top[\mathbf{W}^A \mathbf{h}_i^{(k)}; \mathbf{W}^A \mathbf{h}_j^{(k)}]$ in which $\mathbf{a}$ and $\mathbf{W}^A$ are learnable parameters, and $E^+(i)$ and $E^-(i)$ denoting the incoming and outgoing edges to node v_i respectively. Note that in calculating the attention scores, only neighbors that are of the same node type (out of the 3) are considered. Additionally, we found that only 1 set of attention weights are generated as the authors opted to combine the various aggregated node neighbors together before calculating the final attention score.

These 3 edge relationships are defined based on human-level understanding. However, they **might not necessarily** be a suitable relationship for computations. Hence, to explore if we still require this **human bias**, we propose to **generalize** the attention mechanism such that it is a multi-headed one, removing the need to filter the relationships accordingly. Effectively, we postulate that given a graph consisting of multiple relationships, can the model explicitly learn the useful ones? Compared to the original work, we do not ignore the potential of possible relationships that could exist across node types. Additionally, we explore if the model can learn a more efficient and effective relationship given complete freedom. To this end, we generalize the graph attention mechanism as such:

$$m_{\mathcal{N}_{E^+(i)}}^{(l)} = \sum_{v_j \in \mathcal{N}_{E^+(i)}} \alpha_+^{(l)} \mathbf{W} h_j^{(l)} \tag{5}$$

$$m_{\mathcal{N}_{E^-(i)}}^{(l)} = \sum_{v_j \in \mathcal{N}_{E^-(i)}} \alpha_-^{(l)} \mathbf{W} h_j^{(l)} \tag{6}$$

where $E^+(i)$ and $E^-(i)$ denotes the incoming and outgoing edges to node v_i respectively, α_+, α_- is a vector of learnable attention weights for each node, and m a single head attention head representation based on the direction of the edges. Notably, instead of assuming that a single set of attention coefficients suffice for the problem, we generalize the attention mechanism to take on a total of K heads. Hence, each set of attention coefficients is generated independently. In doing so, we ensure that $K > 3$, which allows to model to learn beyond the 3 predefined relationships. Here, each head could potentially represent one relationship. As such, by introducing multi-headed attention, we grant the model greater freedom to discover a set of possible relationships that can aid it in the prediction and generation task.

We then fuse all the various attention heads together by combining them using an affine transformation to achieve a similar final $h_{\mathcal{N}_{E+(i)}}$ and $h_{\mathcal{N}_{E-(i)}}$ set of representations. These are then passed onto the GRU cell to update the node's current representation. Concretely, we produce the following neighborhood representation in (Pan et al. 2020) by the following

$$h_{\mathcal{N}_{E^+(i)}}^{(l)} = \mathbf{W}\Big(\Big\|_{k \in K} m_{k, \mathcal{N}_{E^+(i)}}^{(l)}\Big) \tag{7}$$

$$h_{\mathcal{N}_{E^-(i)}}^{(l)} = \mathbf{W}\Big(\Big\|_{k \in K} m_{k, \mathcal{N}_{E^-(i)}}^{(l)}\Big) \tag{8}$$

where $\Big\|$ denotes the concatenation operation. We generate these hidden representations and combine them accordingly for the GRU cell update.

4 Experiment

4.1 Experiment Setup

Dataset: To evaluate the model's capabilities, we adopt the same training and testing dataset as Pan et al. (2020). The HotpotQA (Yang et al. 2018) contains approximately 100,000 crowd-sourced questions from Wikipedia articles. Each of these questions is accompanied by answers and requires some form of reasoning over the set of supporting facts. Following Pan et al. (2020), we retrieve the supporting documents and answers to generate a deep question. All training and validation splits are maintained as per Pan et al. (2020).

Evaluation Metrics: (1) Regarding the node classification task in the content selection module, we adopt the accuracy score (acc) as used in (Pan et al. 2020). We further adopt the F1 score of the positive class for more practical analysis. (2) To evaluate the output of the GNNExplainer module, we investigate the geometric properties of the subgraphs generated. Those geometric properties include subgraph size and density. For undirected simple graphs, the graph density is: $D = \frac{2|E|}{|V|(|V|-1)}$ (3)To evaluate the quality of the question generated, we adopt the BLEU score (Papineni et al. 2002), a popular metric in the natural language community. Specifically, our performances are mainly measured with BLEU4, meaning we take up to a 4-gram overlap. Generally, the BLEU score ranges from 0 to 1, with 1 indicating the perfect quality of generation.

Implementation Details: In our experiments, we maintain similar hyperparameters to the work done in (Pan et al. 2020). For both the linguistic heuristic approach and GNN learnable approach, we directly use their trained model for probing and explaining. For the multi-headed attention approach, we experiment with 2 settings, one where $K = 4$ and one where $K = 8$. For $K = 8$, we halved the batch size to 16 due to GPU memory limits.

4.2 Linguistic Heuristic Approach (RQ1)

As we discussed in Section 3.1, the node probing task examines what node features are crucial for content selection. To achieve that, we mask each segment of the embedding and observe the performance change. To be specific, there are three segments in node embeddings: $\mathbf{x} = x_{type} \oplus x_{show} \oplus x_{pos}$, indicating the node type (adjective, noun or verb), whether the node is shown in the answer, and the part-of-speech of the node, respectively.

	Original	Node Type	Show In Answer	Part-of-Speech
F1/Acc	67.6/84.7	47.7/41.5	64.9/82.4	63.7/83.7

Table 2: Node probing on the content selection module.

From Table 2 we observe that masking each segment has a negative impact on F1/acc score as compared to the original F1/acc. It shows that all segments of the node embeddings make positive contributions to the content selection module. Interestingly the node type segment has the biggest impact. One possible reason is that it only has three options (adjective, noun, or verb). Such a focused objective helps the model to leverage it. From a linguistic point, part-of-speech can be seen as a finer-grained taxonomy of the node type. It has 30 and more options, making the model harder to leverage on.

	Original	pobj	SIMILAR	nsubj
F1/acc	67.6/84.7	67.5/84.7	59.5/82.0	67.7/84.7
Frequency	N/A	752k	353k	283k
	cop	partmod	dobj	conj
F1/acc	67.6/84.6	60.6/82.4	67.6/84.7	67.6/84.7
Frequency	225k	122k	118k	109k

Table 3: Edge probing on content selection module.

Our edge probing task adopts a very straightforward approach. We mask one type of edge at each time and observe performance change on the F1/acc of the content selection module. Table 3 summarizes the results of masking the most common edges. We find that SIMILAR and partmod edges have the biggest impact on content selection. It is intuitive to understand the importance of SIMILAR edges, since SIMILAR edges connect different sentences. The partmod edges refer to the participial modifier of an NP or VP (De Marneffe and Manning 2008). It is not so intuitive and we leave the explanation for its importance to future work. Interestingly, we find that masking other edges have a little negative impact or even positive impact: nsubj has acc of 67.7, being higher than 67.6 as the original acc. This result indicates that there is possible redundancy in the graph design in the model of (Pan et al. 2020).

Table 4 (refer to Appendix A for reference on abbreviations) summarizes the impact of different edges on different nodes. We categorize nodes w.r.t. their part-of-speech and report the results from the most common categories. The results align well with Table 3, since SIMILAR and partmod have the largest negative impact on all nodes. We also observe positive effects of edge masking, for example, masking nsubj has 0.29 positive impact on the F1 of NNS. These results may indicate that some nodes and edges are unlikely to have synergies, providing clues for building simplified models in the future.

4.3 GNN Learnable Approach (RQ2)

As we discussed in Section 3.2, the GNNExplainer module takes a node v, its label and the whole graph G as input. It outputs a compact subgraph structure $G^{'}$ being crucial for its prediction.

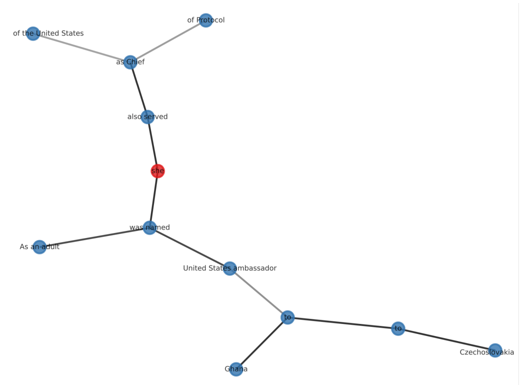

Figure 2: Random output from GNNExplainer

Human Observation: In order to assert the rationality of GNNExplainer on our corpus, we first manually examine 50 subgraphs produced by GNNExplainer. One random subgraph is shown in Figure 2. The node to be explained is "she". The darker an edge is, the higher weight this edge will receive. The produced subgraph shows that the person's identity ("chief", "adult", and "ambassador") is most closely related to the person and has the most importance. This is followed by the country name ("the United States", "Ghana", and "Czechoslovakia") related to the identity. There are also some important links between the relevant countries ("Ghana" and "Czechoslovakia"). This shows that the **computational chain** generated by GNNExplainer is **similar** to **human heuristics**. The most ideal case is to conduct human experiments to find if subgraphs make sense. However, it is difficult in this initiative study and is left for future work. We attach more of such subgraphs in Appendix D.
Intrinsic Evaluation: GNNExplainer model provides a compact subgraph structure given a node to be explained. It is natural to ask, what are the defining properties of such graphs? Since we are unable to manually count all substructures, we investigate the following geometric properties of the generated subgraphs: (1) the size of the graph (number

	NNP	NN	VBN	VBZ	NNS	IN	VBD	CD	PRP	VBG
pobj	-0.09	-0.12	0.05	0.07	-0.06	-0.27	0.02	-0.44	-0.23	0.32
SIMILAR	-7.74	-8.21	-8.96	-7.28	-8.03	-8.54	-8.71	-8.69	-6.82	-8.96
nsubj	0.12	-0.02	0.09	0.16	0.29	0.15	-0.05	-0.24	0.17	0.36
cop	0.04	-0.03	0.08	-0.11	0.34	0.1	0.05	-0.16	-0.07	0.16
partmod	-6.85	-7.15	-7.57	-5.72	-6.92	-7.08	-7.55	-8.13	-5.81	-7.61
dobj	0.05	-0.05	0.03	0.19	-0.02	0.03	0.04	-0.13	0.07	0.0
conj	-0.08	-0.02	0.02	0.02	0.07	-0.08	0.1	-0.24	0.17	0.24

Table 4: Innate Priors: Fine-grained analysis of the impact of edges on nodes. Results are in relative F1: masking pobj edges has -0.09 impact on the F1 of NNP nodes.

	All	Irrelevant	Relevant	NNP	NN	VBN	VBZ	NNS	IN	VBD	CD	PRP	VBG
# Node	21.9	21.3	23.4	21.5	21.9	21.5	20.7	24.4	24.2	19.8	17.2	23.6	26.5
# Edge	22	21.3	24	21.6	22.1	21.6	20.8	24.7	24.3	19.6	16.9	24.2	26.9
Density	0.137	0.139	0.134	0.139	0.138	0.143	0.161	0.123	0.112	0.142	0.161	0.143	0.107
# Subgraph	2502	1836	666	726	590	202	160	185	147	108	88	56	46

Table 5: Geometric Properties: We summarize the geometric properties of subgraph w.r.t. their input node types.

	NNP	NN	VBN	VBZ	NNS	IN	VBD	CD	PRP	VBG
pobj	1241.6	996.7	315.9	193.2	280.6	207.4	129.7	180	93	60.4
SIMILAR	460.4	383.9	126.7	68.1	73.4	23.6	31	26.1	39.3	28.8
nsubj	113.4	149.2	33.9	36	67.7	18.8	36.9	15.9	28.1	9
cop	535.5	487.2	118.8	142	52.3	52	76.5	56.8	32.9	20.7
partmod	180.5	130.9	90.2	35.6	26.8	14.9	19.5	22.8	9.4	18
dobj	133.9	139.4	37.3	29.8	57.9	17.6	40.1	22.3	19.6	21.4
conj	160.7	117.8	38.4	18.1	40.4	25.3	16	13.5	5	6.9
# Subgraph	726	590	202	160	185	147	108	88	56	46

Table 6: Edge Weight: We summarize the edge weight in the subgraphs generated for different types of nodes.

of edges and number of nodes), and the density of the subgraph.

Table 5 (refer to Appendix A for reference on abbreviations) summarizes these properties w.r.t. the type of the input node to be explained. We sampled 100 data instances, producing 2502 subgraphs. The mean number of nodes is 21.9, the mean number of edges is 22, and the mean density is 0.137. One interesting discovery is that the subgraph of relevant nodes are **larger** than irrelevant node ($23.4 > 21.3$ w.r.t. # nodes). It implies that relevant nodes may receive **more** information from their neighboring node to be used in the generated question. We also discover that verbs tend to have **higher** subgraph density. For example, VBN, VBZ, and VBD have density of 0.143, 0.161, and 0.142 respectively, which are all significantly higher than the mean density of 0.137. It aligns well with linguistics since verb is the root of a sentence or a clause.

Extrinsic Comparison with Linguistic Heuristic Approach: Since we have intuitive discoveries for the intrinsic properties of the subgraphs generated by GNNExplainer, we then conduct an extrinsic comparison with the linguistic heuristic approach. We aim to answer whether GNNExplainer's subgraphs align well with the node-edge importance discovered by the heuristic approach in Table 4 (refer to Appendix A for reference on abbreviations). To achieve this, we calculate the edge weight in the subgraphs and show the results in Table 6 (refer to Appendix A for reference on abbreviations). For a specific type of node, we find all the subgraphs generated for it, and sum up the edge weights. For example, among all 726 subgraphs for NNP nodes, the summed weight is 1241.6.

By comparing the linguistic heuristic approach (Table 4) and GNNExplainer (Table 6), we have the following discoveries: (1) The two approaches **overlap well** on some node-edge importance. For example, both linguistic heuristic and GNNExplainer assign **high importance** to SIMILAR edges for all nodes. This consensus again asserts the **importance** of SIMILAR edges that connect different sentences together. (2) The two approaches also disagree on some node-edge importance. For instance, linguistic heuristics find partmod edges are important for the content selection of all nodes. However, GNNExplainer assigns a **lower** weight for partmod. This inconsistency implies that the way GNN models reasons on the graph **might not obey the linguistic prior of humans**.

4.4 Generalizing Learning of Graph Structure (RQ3)

Figures 3 and 4 showcases the model accuracy and losses over the training period. Note that we stopped the model early (15 epochs or so) and only show the first 7 epochs

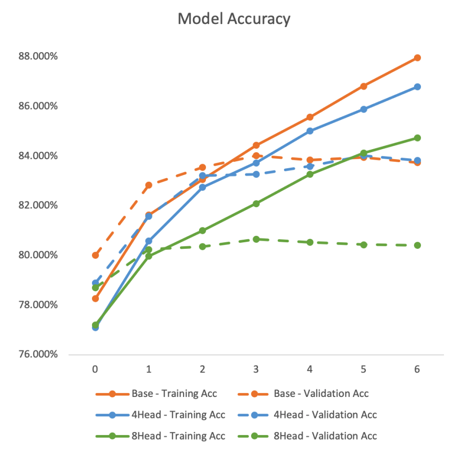

Figure 3: Training and validation accuracy of models

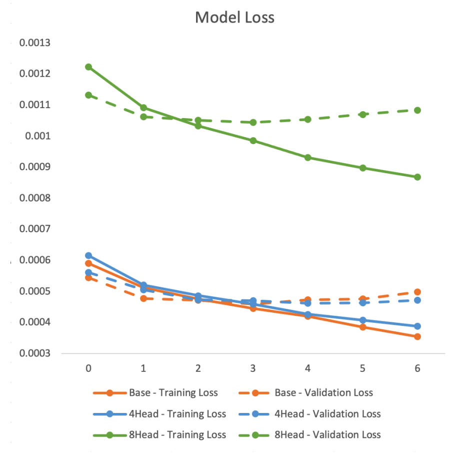

Figure 4: Training and validation loss of models

of training as we observed that the model started to overfit rather quickly. Both results showcase a rather interesting observation; increasing the number of heads of the graph attention model **does not** drastically help it. The implementation in (Pan et al. 2020) consists of a single headed attention, where we project the aggregations of 3 different relationships. However, all 3 relationships share the same attention weight. Conversely, our multi-headed approach allows for each head to learn its own attention weights. Technically, this is better as it allows the model to learn beyond just 1 form of attention, improving its generalization capabilities (Vaswani et al. 2017).

Model	BLEU4
Original	15.28486
Multi-headed Graph Attention (4 Heads)	15.26169
Multi-headed Graph Attention (4 Heads)	14.96905

Table 7: Comparison of BLEU4 scores produced by our models

Table 10 (in Appendix B) compares the number of parameters across our models. Our 4-headed attention is roughly 3% larger than the original, and the 8-headed attention is 9% larger. However, despite the larger model in 8-headed attention, we can see in Figure 3 that the accuracy in the content selection task drops drastically. We are unsure if this is due to the model being significantly larger and hence needing more data to learn, or the fact that we had to decrease our batch size to fit the computations on a GPU. Nevertheless, the 4-headed attention model also signifies that even if we provided slightly more compute power and freedom, our model does not do exceedingly better.

Table 7 shows the best BLEU4 scores of the models. Notably, the best achieved multi-headed attention model is still unable to beat the original model formulation. As discussed earlier, despite having more parameters, we are unable to achieve a better score.

Table 11 (in Appendix C) showcases a short and easy question as well as the generated answers from the models. From our analysis, when the question to be asked is clear, almost all 3 models are able to ask very similar questions. In longer and harder questions, the types of questions generated can be quite different. However, we are unable to determine if there's a clear pattern as to whether we learn significantly different questions. At the same time, we have not carried out any human evaluation study to evaluate the models qualitatively.

5 Conclusion and Future Work

In this work, we have examined the importance of graphs for the DQG task in a three-pronged approach. Linguistic heuristics provide a principled way of examining important information. However, this is limited in nature as there could be many possibilities and requires a human expert. We then showed that the computational graph shares similar traits with our human understanding with the use of GNNExplainer. At the same time, it is able to provide other insights that are counter-intuitive, suggesting a model's way of interpreting is slightly different. Thus, we explored what if we were to allow a model to decide freely on its own, without additional human bias beyond the graph construction. Interestingly, we found that models can not be allowed to run freely; the human interpretation and constraints are still highly valuable for such NLP tasks.

Our work is an initial effort on explaining reasoning on the graph for NLP, there are many loose ends. While the accuracy or BLEU score indicates poorer performance, we might be generating a more varied set of questions if the model is given total freedom. Nevertheless, our work can be furthered by including more human experts in evaluating the model. The subgraphs generated by GNNExplainer can be examined by human experts to assert their rationality, the final questions generated by different models can also be compared by humans. We can also try to generalize our discoveries on other NLP tasks that require graph reasonings. For example, since we have discovered the importance of certain node-edge relations in DQG, we can explore if such relations still hold in other NLP tasks like QA, discourse, and natural language inference.

Acknowledgments

We thank the authors of SG-DQG (especially Yuxi Xie) for their insightful comments and instructions on implementation.

References

Dai, Z.; and Huang, R. 2019. A regularization approach for incorporating event knowledge and coreference relations into neural discourse parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2967–2978.

De Marneffe, M.-C.; and Manning, C. D. 2008. Stanford typed dependencies manual. Technical report, Technical report, Stanford University.

Fey, M.; Lenssen, J. E.; Morris, C.; Masci, J.; and Kriege, N. M. 2020. Deep graph matching consensus. *arXiv preprint arXiv:2001.09621* .

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, 1263–1272. PMLR.

Huang, X.; Zhang, J.; Li, D.; and Li, P. 2019. Knowledge graph embedding based question answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 105–113.

Lei, W.; Zhang, G.; He, X.; Miao, Y.; Wang, X.; Chen, L.; and Chua, T.-S. 2020. Interactive path reasoning on graph for conversational recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2073–2083.

Marcheggiani, D.; and Titov, I. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826* .

Pan, L.; Xie, Y.; Feng, Y.; Chua, T.-S.; and Kan, M.-Y. 2020. Semantic Graphs for Generating Deep Questions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 1463–1475.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 311–318.

Sap, M.; Le Bras, R.; Allaway, E.; Bhagavatula, C.; Lourie, N.; Rashkin, H.; Roof, B.; Smith, N. A.; and Choi, Y. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3027–3035.

Saxena, A.; Tripathi, A.; and Talukdar, P. 2020. Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 4498–4507. Online: Association for Computational Linguistics. doi:10.18653/v1/2020.acl-main.412. URL https://www.aclweb.org/anthology/2020.acl-main.412.

Schlichtkrull, M. S.; Cao, N. D.; and Titov, I. 2021. Interpreting Graph Neural Networks for {NLP} With Differentiable Edge Masking. In *International Conference on Learning Representations*. URL https://openreview.net/forum?id=WznmQa42ZAx.

Speer, R.; Chin, J.; and Havasi, C. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Strapparava, C.; Valitutti, A.; et al. 2004. Wordnet affect: an affective extension of wordnet. In *Lrec*, volume 4, 40. Citeseer.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762* .

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* .

Wang, C.; Liu, X.; and Song, D. 2020. Language Models are Open Knowledge Graphs. *arXiv preprint arXiv:2010.11967* .

Wang, X.; Kapanipathi, P.; Musa, R.; Yu, M.; Talamadupula, K.; Abdelaziz, I.; Chang, M.; Fokoue, A.; Makni, B.; Mattei, N.; et al. 2019. Improving natural language inference using external knowledge in the science questions domain. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 7208–7215.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Ying, R.; Bourgeois, D.; You, J.; Zitnik, M.; and Leskovec, J. 2019. GNNExplainer: Generating Explanations for Graph Neural Networks. *Advances in Neural Information Processing Systems* 32: 9240–9251.

Zhang, Z.; and Lee, W. S. 2019. Deep graphical feature learning for the feature matching problem. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5087–5096.

Appendix A Linguistic Description

Since we have many linguistic abbreviations in the paper, here we detail their full names and give a brief description.

Abbreviation	Full Name or Description
NNP	Proper noun, singular
NN	Noun, singular
VBN	Verb, past participle
VBZ	Verb, present tense, third person singular
NNS	Plural common nouns
IN	Preposition or subordinating conjunction
VBD	Verb, past tense
CD	Cardinal number
PRP	Personal pronoun
VBG	Verb, gerund or present participle

Table 8: Part-of-speech: We details the full name or the description of the part-of-speech shown up in our paper.

Abbreviation	Full Name or Description	Description
pobj	Object of a preposition	The object of a preposition is the head of a noun phrase following the preposition, or the adverbs “here” and “there”.
nsubj	Nominal subject	A nominal subject is a noun phrase which is the syntactic subject of a clause.
cop	Copula	A copula is the relation between the complement of a copular verb and the copular verb.
partmod	Participial modifier	A participial modifier of an NP or VP is a participial verb form that serves to modify the meaning of the NP or VP.
dobj	Direct object	The direct object of a VP is the noun phrase which is the (accusative) object of the verb; the direct object of a clause is the direct object of the VP which is the predicate of that clause.
conj	Conjunct	A conjunct is the relation between two elements connected by a coordinating conjunction, such as “and”, “or”, etc.

Table 9: Dependencies: We detail the edges of dependency in our paper.

Appendix B Number of parameters across the models

Model	Number of Parameters
Original	57,741,719
Multi-headed Graph Attention (4 Heads)	59,718,545
Multi-headed Graph Attention (8 Heads)	62,876,561

Table 10: Comparison of parameters across models

Appendix C Generated questions across various models

Question	cooking light is an american monthly food and lifestyle magazine founded in 1987 . vibe is an american music and entertainment magazine founded by producer quincy jones .
Ground Truth	are both ” cooking light ” and ” vibe ” magazines ?

Model	Generated Answer
Original	are cooking light and vibe both magazines
4-Headed Attention	cooking light and vibe , are both magazines ?
8-Headed Attention	cooking light and vibe are both what ?

Table 11: Generated questions according to various models for short and easy texts

Appendix D Additional output examples from GNNExplainer

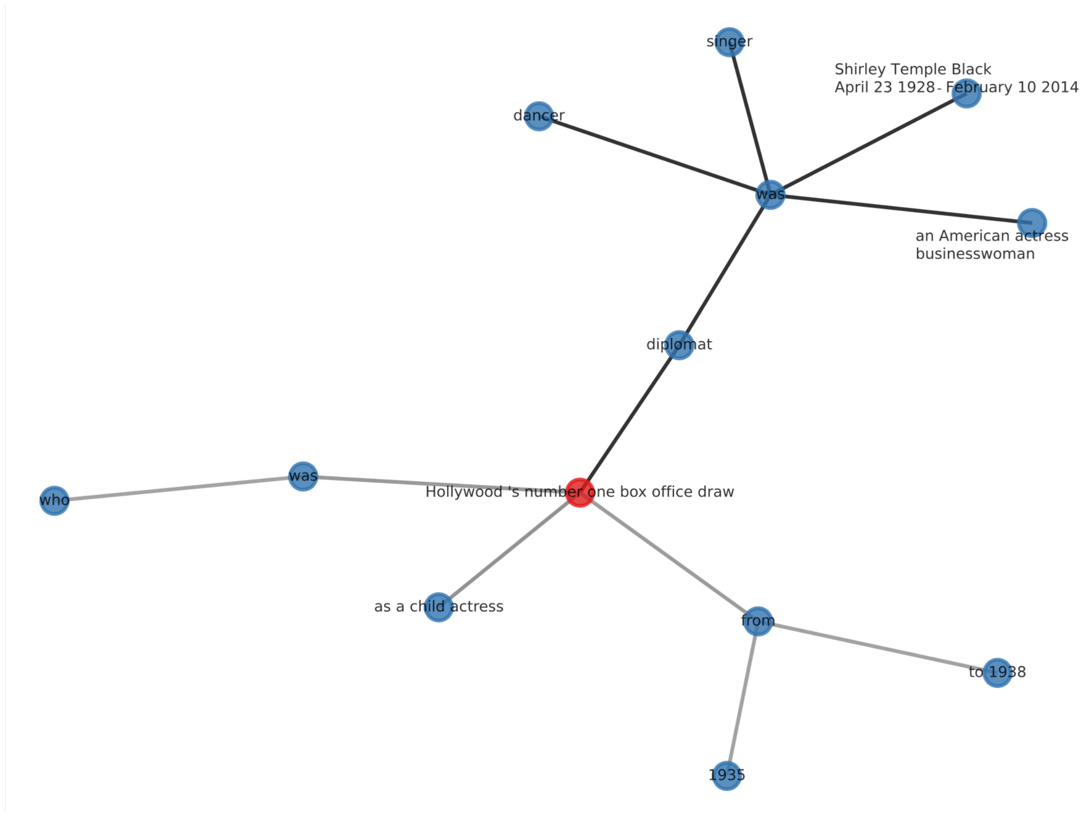

Figure 5: Additional example output 1 from GNNExplainer

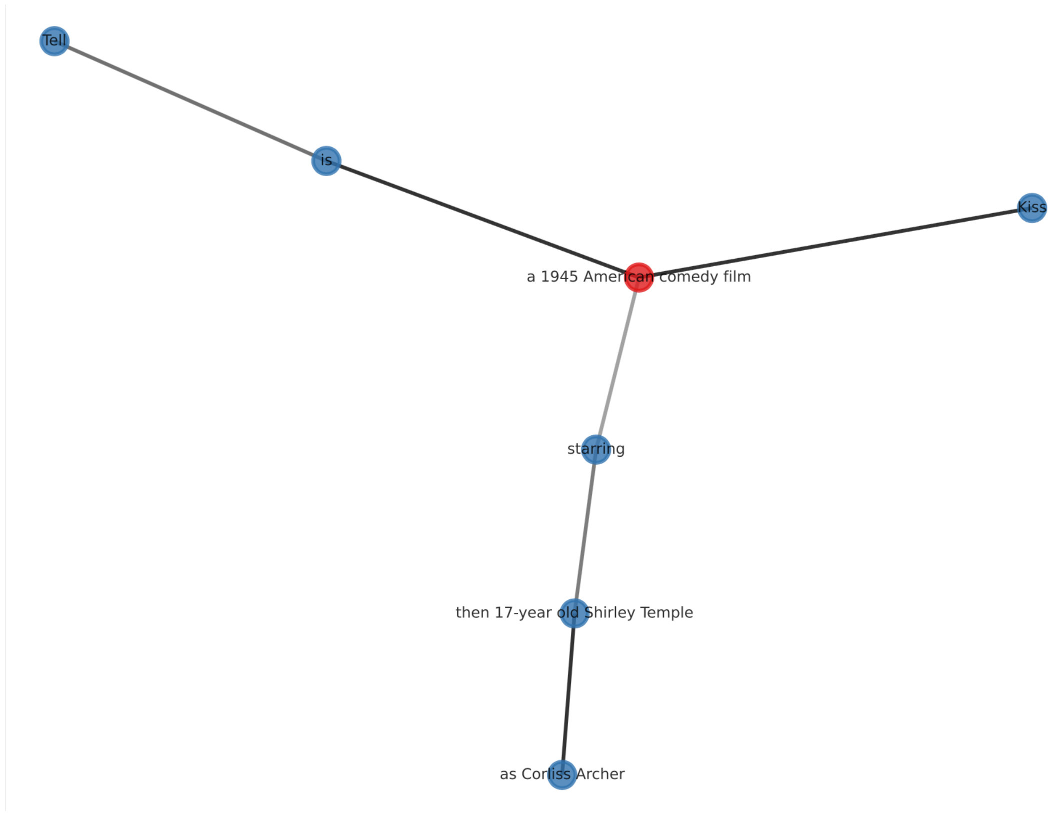

Figure 6: Additional example output 2 from GNNEXPLAINER