QA-GNN: Reasoning with Language Model and Knowledge Graph for Question Answering

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut
Percy Liang, Jure Leskovec
Stanford University
{myasu,hyren,antoineb,pliang,jure}@cs.stanford.edu

Abstract

The problem of answering questions using knowledge from pre-trained language models (LMs) and knowledge graphs (KGs) presents two challenges: given a QA context (question and answer choice), methods need to (i) identify relevant knowledge from large KGs, and (ii) perform joint reasoning over the QA context and KG. In this work, we propose a new model, QA-GNN, which addresses the above challenges through two key innovations: (i) relevance scoring, where we use LMs to estimate the importance of KG nodes relative to the given QA context, and (ii) joint reasoning, where we connect the QA context and KG to form a joint graph, and mutually update their representations through graph neural networks. We evaluate our model on QA benchmarks in the commonsense (CommonsenseQA, OpenBookQA) and biomedical (MedQA-USMLE) domains. QA-GNN outperforms existing LM and LM+KG models, and exhibits capabilities to perform interpretable and structured reasoning, e.g., correctly handling negation in questions. Our code and data are available at https://github.com/michiyasunaga/qagnn.

1 Introduction

Question answering systems must be able to access relevant knowledge and reason over it. Typically, knowledge can be implicitly encoded in large language models (LMs) pre-trained on unstructured text (Petroni et al., 2019; Bosselut et al., 2019), or explicitly represented in structured knowledge graphs (KGs), such as Freebase (Bollacker et al., 2008) and ConceptNet (Speer et al., 2017), where entities are represented as nodes and relations between them as edges. Recently, pre-trained LMs have demonstrated remarkable success in many question answering tasks (Liu et al., 2019; Raffel et al., 2020). However, while LMs have a broad coverage of knowledge, they do not empirically perform well on structured reasoning (e.g., handling negation) (Kassner and Schütze, 2020). On the other hand, KGs are more suited for structured reasoning (Ren et al., 2020; Ren and Leskovec, 2020) and enable explainable predictions e.g., by providing reasoning paths (Lin et al., 2019), but may lack coverage and be noisy (Bordes et al., 2013; Guu et al., 2015). How to reason effectively with both sources of knowledge remains an important open problem.

Combining LMs and KGs for reasoning (henceforth, LM+KG) presents two challenges: given a QA context (e.g., question and answer choices; Figure 1 purple box), methods need to (i) identify informative knowledge from a large KG (green box); and (ii) capture the nuance of the QA context and the structure of the KGs to perform joint reasoning over these two sources of information. Previous works (Bao et al., 2016; Sun et al., 2018; Lin et al., 2019) retrieve a subgraph from the KG by taking topic entities (KG entities mentioned in the given QA context) and their few-hop neighbors. However, this introduces many entity nodes that are semantically irrelevant to the QA context, especially when the number of topic entities or hops increases. Additionally, existing LM+KG methods for reasoning (Lin et al., 2019; Wang et al., 2019a; Feng et al., 2020; Lv et al., 2020) treat the QA context and KG as two separate modalities. They
If it is not used for hair, a round brush is an example of what? A. hair brush   B. bathroom   C. art supplies*

Individually apply LMs to the QA context and graph neural networks (GNNs) to the KG, and do not mutually update or unify their representations. This separation might limit their capability to perform structured reasoning, e.g., handling negation.

Here we propose QA-GNN, an end-to-end LM+KG model for question answering that addresses the above two challenges. We first encode the QA context using an LM, and retrieve a KG subgraph following prior works (Feng et al., 2020).

Our QA-GNN has two key insights: (i) **Relevance scoring**: Since the KG subgraph consists of all few-hop neighbors of the topic entities, some entity nodes are more relevant than others with respect to the given QA context. We hence propose KG node relevance scoring: we score each entity on the KG subgraph by concatenating the entity with the QA context and calculating the likelihood using a pre-trained LM. This presents a general framework to weight information on the KG: (ii) **Joint reasoning**: We design a joint graph representation of the QA context and KG, where we explicitly view the QA context as an additional node (QA context node) and connect it to the topic entities in the KG subgraph as shown in Figure 1. This joint graph, which we term the working graph, unifies the two modalities into one graph. We then augment the feature of each node with the relevance score, and design a new attention-based GNN module for reasoning. Our joint reasoning algorithm on the working graph simultaneously updates the representation of both the KG entities and the QA context node, bridging the gap between the two sources of information.

We evaluate QA-GNN on three question answering datasets that require reasoning with knowledge: CommonsenseQA (Talmor et al., 2019) and OpenBookQA (Mihaylov et al., 2018) in the commonsense domain (using the ConceptNet KG), and MedQA-USMLE (Jin et al., 2021) in the biomedical domain (using the UMLS and DrugBank KGs). QA-GNN outperforms strong fine-tuned LM baselines as well as the existing best LM+KG model (with the same LM) by 4.7% and 2.3% respectively. In particular, QA-GNN exhibits improved performance on some forms of structured reasoning (e.g., correctly handling negation and entity substitution in questions): it achieves 4.6% improvement over fine-tuned LMs on questions with negation, while existing LM+KG models are +0.6% over fine-tuned LMs. We also show that one can extract reasoning processes from QA-GNN in the form of general KG subgraphs, not just paths (Lin et al., 2019), suggesting a general method for explaining model predictions.

2 Problem statement

We aim to answer natural language questions using knowledge from a pre-trained LM and a structured KG. We use the term language model broadly to be any composition of two functions, $f_{\text{head}}(f_{\text{enc}}(x))$, where $f_{\text{enc}}$, the encoder, maps a textual input $x$ to a contextualized vector representation $h_{\text{LM}}$, and $f_{\text{head}}$ uses this representation to perform a desired task (which we discuss in §3.2). In this work, we specifically use masked language models (e.g., RoBERTa) as $f_{\text{enc}}$, and let $h_{\text{LM}}$ denote the output representation of a [CLS] token that is prepended to the input sequence $x$, unless otherwise noted. We define the knowledge graph as a multi-relational graph $G = (\mathcal{V}, \mathcal{E})$. Here $\mathcal{V}$ is the set of entity nodes in the KG; $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{R} \times \mathcal{V}$ is the set of edges that connect nodes in $\mathcal{V}$, where $\mathcal{R}$ represents a set of relation types.

Given a question $q$ and an answer choice $a \in \mathcal{C}$, we follow prior work (Lin et al., 2019) to link the entities mentioned in the question and answer choice to the given KG $G$. We denote $V_q \subseteq \mathcal{V}$ and $V_a \subseteq \mathcal{V}$ as the set of KG entities mentioned in the question (question entities; blue entities in Figure 1) and answer choice (answer choice entities; red entities in Figure 1), respectively, and use $V_{q,a} := V_q \cup V_a$ to denote all the entities that appear in either the question or answer choice, which we call topic entities. We then extract a subgraph from $G$ for a question-choice pair, $G_{q,a} = (V_{q,a}, E_{q,a})$, which comprises all nodes on the $k$-hop paths between nodes in $V_{q,a}$.

1We remove the superscript $q,a$ if there is no ambiguity.
3 Approach: QA-GNN

As shown in Figure 2, given a question and an answer choice a, we concatenate them to get the QA context \([q, a]\). To reason over a given QA context using knowledge from both the LM and the KG, QA-GNN works as follows. First, we use the LM to obtain a representation for the QA context, and retrieve the subgraph \(G_{\text{sub}}\) from the KG. Then we introduce a QA context node \(z\) that represents the QA context, and connect \(z\) to the topic entities \(V_{q,a}\) so that we have a joint graph over the two sources of knowledge, which we term the working graph, \(G_W\) (§3.1). To adaptively capture the relationship between the QA context node and each of the other nodes in \(G_W\), we calculate a relevance score for each pair using the LM, and use this score as an additional feature for each node (§3.2). We then propose an attention-based GNN module that does message passing on the \(G_W\) for multiple rounds (§3.3). Finally, we make the final prediction using the LM representation, QA context node representation and a pooled working graph representation (§3.4).

3.1 Joint graph representation

To design a joint reasoning space for the two sources of knowledge, we explicitly connect them in a common graph structure. We introduce a new QA context node \(z\) which represents the QA context, and connect \(z\) to each topic entity in \(V_{q,a}\) on the KG subgraph \(G_{\text{sub}}\) using two new relation types \(r_{z,q}\) and \(r_{z,a}\). These relation types capture the relationship between the QA context and the relevant entities in the KG, depending on whether the entity is found in the question portion or the answer portion of the QA context. Since this joint graph intuitively provides a reasoning space (working memory) over the QA context and KG, we term it working graph \(G_W = (V_W, E_W)\), where \(V_W = V_{\text{sub}} \cup \{z\}\) and \(E_W = E_{\text{sub}} \cup \{(z, r_{z,q}, v) | v \in V_q\} \cup \{(z, r_{z,a}, v) | v \in V_a\}\).

Each node in \(G_W\) is associated with one of the four types: \(T = \{Z, Q, A, O\}\), each indicating the context node \(z\), nodes in \(V_q\), nodes in \(V_a\), and other nodes, respectively (corresponding to the node color, purple, blue, red, gray in Figure 1 and 2). We denote the text of the context node \(z\) (QA context) and KG node \(v \in V_{\text{sub}}\) (entity name) as \(\text{text}(z)\) and \(\text{text}(v)\).

We initialize the node embedding of \(z\) by the LM representation of the QA context \(z^{\text{LM}} = f_{\text{enc}}(\text{text}(z))\), and each node on \(G_{\text{sub}}\) by its entity embedding (§4.2). In the subsequent sections, we will reason over the working graph to score a given (question, answer choice) pair.

3.2 KG node relevance scoring

Many nodes on the KG subgraph \(G_{\text{sub}}\) (i.e., those heuristically retrieved from the KG) can be irrelevant under the current QA context. As an example shown in Figure 3, the retrieved KG subgraph \(G_{\text{sub}}\) with few-hop neighbors of the \(V_{q,a}\) may include nodes that are uninformative for the reasoning process, e.g., nodes “holiday” and “river bank” are off-topic; “human” and “place” are generic. These irrelevant nodes may result in overfitting or introduce unnecessary difficulty in reasoning, an issue especially when \(V_{q,a}\) is large. For instance, we empirically find that using the ConceptNet KG (Speer et al., 2017), we will retrieve a KG with \(|V_{\text{sub}}| > 400\) nodes on average if we consider 3-hop neighbors.

In response, we propose node relevance scoring, where we use the pre-trained language model to score the relevance of each KG node \(v \in V_{\text{sub}}\) conditioned on the QA context. For each node \(v\), we concatenate the entity \(\text{text}(v)\) with the QA context
text(z) and compute the relevance score:
\[ \rho_t = f_{\text{head}}(f_{\text{enc}}(\text{text}(z); \text{text}(v))), \]
where \( f_{\text{head}} \circ f_{\text{enc}} \) denotes the probability of \( \text{text}(v) \) computed by the LM. This relevance score \( \rho_t \) captures the importance of each KG node relative to the given QA context, which is used for reasoning or pruning the working graph \( \mathcal{G}_W \).

### 3.3 GNN architecture

To perform reasoning on the working graph \( \mathcal{G}_W \), our GNN module builds on the graph attention framework (GAT) (Veličković et al., 2018), which induces node representations via iterative message passing between neighbors on the graph. Specifically, in a L-layer QA-GNN, for each layer, we update the representation \( h_t^{(l+1)} \in \mathbb{R}^D \) of each node \( t \in \mathcal{V}_W \) by
\[ h_t^{(l+1)} = f_h \left( \sum_{s \in \mathcal{N}_t} \alpha_{st} m_{st} \right) + h_t^{(l)}, \]
where \( \mathcal{N}_t \) represents the neighborhood of node \( t \), \( m_{st} \in \mathbb{R}^D \) denotes the message from each neighbor node \( s \) to \( t \), and \( \alpha_{st} \) is an attention weight that scales each message \( m_{st} \) from \( s \) to \( t \). The sum of the messages is then passed through a 2-layer MLP, \( f_s : \mathbb{R}^D \rightarrow \mathbb{R}^D \), with batch normalization (Ioffe and Szegedy, 2015). For each node \( t \in \mathcal{V}_W \), we set \( h_t^{(0)} \) using a linear transformation \( f_h \) that maps its initial node embedding (described in §3.1) to \( \mathbb{R}^D \).

Crucially, as our GNN message passing operates on the working graph, it will jointly leverage and update the representation of the QA context and KG. We further propose an expressive message \( (m_{st}) \) and attention \( (\alpha_{st}) \) computation below.

#### Node type, relation, and score-aware attention.

Attention captures the strength of association between two nodes, which is ideally informed by their node types, relations and node relevance scores. We first embed the relevance score of each node \( t \) by
\[ \rho_t = f_\rho(\rho_t), \]
where \( f_\rho : \mathbb{R} \rightarrow \mathbb{R}^{D/2} \) is an MLP. To compute the attention weight \( \alpha_{st} \) from node \( s \) to node \( t \), we obtain the query and key vectors \( q, k \) by
\[ q_s = f_q(h_t^{(l)}, u_s, \rho_s), \]
\[ k_t = f_k(h_t^{(l)}, u_t, \rho_t, r_{st}), \]
where \( f_q : \mathbb{R}^{2D} \rightarrow \mathbb{R}^D \) and \( f_k : \mathbb{R}^{3D} \rightarrow \mathbb{R}^D \) are linear transformations. The attention weight is then
\[ \alpha_{st} = \frac{\exp(\gamma_{st})}{\sum_{t' \in \mathcal{N}_s} \exp(\gamma_{s,t'})}, \gamma_{st} = \frac{q_s^T k_t}{\sqrt{D}}. \]

#### 3.4 Inference & Learning

Given a question \( q \) and an answer choice \( a \), we use the information from both the QA context and the KG to calculate the probability of it being the answer \( p(a \mid q) \propto \exp(\text{MLP}(z^{\text{LM}}, z^{\text{GNN}}, g)) \), where \( z^{\text{GNN}} = h_s^{(L)} \) and \( g \) denotes the pooling of \( \{h_s^{(L)} \mid v \in \mathcal{V}_{\text{sub}}\} \). In the training data, each question has a set of answer choices with one correct choice.

We optimize the model (both the LM and GNN components end-to-end) using the cross entropy loss.

#### 3.5 Computation complexity

We analyze the time and space complexity of our method and compare with prior works, KagNet (Lin et al., 2019) and MHGRN (Feng et al., 2020) in Table 1. As we handle edges of different relation types using different edge embeddings instead of designing an independent graph networks for each relation as in RGCN (Schlichtkrull et al., 2018) or MHGRN, the time complexity of our method is constant with respect to the number of relations and linear with respect to the number of nodes. We achieve the same space complexity as MHGRN (Feng et al., 2020).

### 4 Experiments

#### 4.1 Datasets

We evaluate QA-GNN on three question answering datasets: CommonsenseQA (Talmor et al., 2019), OpenBookQA (Mihaylov et al., 2018), and MedQA-OSMLE (Jin et al., 2021).

CommonsenseQA is a 5-way multiple choice QA task that requires reasoning with commonsense knowledge, containing 12,102 questions. The test set of CommonsenseQA is not publicly available, and model predictions can only be evaluated once
We use the original data splits from Jin et al. (2021). We perform main experiments on the in-house (IH) (Liu et al., 2020a). The knowledge graph ConceptNet (Speer et al., 2017), a general-domain (USMLE). The dataset contains 12,723 questions. With the relation set Computation complexity evance score computed in §3.2. Henceforth, in this section (§4) we use the term “KG” to refer to $G_{\text{sub}}$.

### 4.3 Implementation & training details

We set the dimension ($D = 200$) and number of layers ($L = 5$) of our GNN module, with dropout rate 0.2 applied to each layer (Srivastava et al., 2014). We train the model with the RAdam (Liu et al., 2020b) optimizer using two GPUs (GeForce RTX 2080 Ti), which takes ~20 hours. We set the batch size from {32, 64, 128, 256}, learning rate for the LM module from {5e-6, 1e-5, 2e-5, 5e-5}, and learning rate for the GNN module from {2e-4, 5e-4, 1e-3, 2e-3}. The above hyperparameters are tuned on the development set.

### 4.4 Baselines

**Fine-tuned LM.** To study the role of KGs, we compare with a vanilla fine-tuned LM, which does not use the KG. We use RoBERTa-large (Liu et al., 2019) for CommonsenseQA, and RoBERTa-large and AristoRoBERTa (Clark et al., 2019) for OpenBookQA. For MedQA-USMLE, we use a state-of-the-art biomedical LM, SapBERT (Liu et al., 2020a).

**Existing LM+KG models.** We compare with existing LM+KG methods, which share the same high-level framework as ours but use different modules to reason on the KG in place of QA-GNN (“yellow box” in Figure 2): (1) Relation Network (RN) (Santoro et al., 2017), (2) RGCN (Schlichtkrull et al., 2018), (3) GconAttn (Wang et al., 2019a), (4) KagNet (Lin et al., 2019), and (5) MHGRN (Feng et al., 2020). (1),(2),(3) are relation-aware GNNs for KGs, and (4),(5) further model paths in KGs. MHGRN is the existing top performance model under this LM+KG framework. For fair comparison, we use the same LM in all the baselines and our model. The key differences between QA-GNN and these are that they do not perform relevance scoring or joint updates with the QA context (§3).

### 4.5 Main results

Table 2 and Table 4 show the results on CommonsenseQA and OpenBookQA, respectively. On both datasets, we observe consistent improvements over fine-tuned LMs and existing LM+KG models, e.g., on CommonsenseQA, +4.7% over RoBERTa, and +2.3% over the prior best LM+KG system, MHGRN. The boost over MHGRN suggests that QA-GNN makes a better use of KGs to perform joint reasoning than existing LM+KG methods.

---

### Table 1: Computation complexity of different L-hop reasoning models on a dense/sparse graph $G = (V, E)$ with the relation set $R$.

<table>
<thead>
<tr>
<th>Model</th>
<th>Time</th>
<th>Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>L-hop KagNet</td>
<td>$O(</td>
<td>R</td>
</tr>
<tr>
<td>L-hop MHGRN</td>
<td>$O(</td>
<td>R</td>
</tr>
<tr>
<td>L-layer QA-GNN</td>
<td>$O(</td>
<td>V</td>
</tr>
</tbody>
</table>

$G$ is a dense graph

$G$ is a sparse graph with maximum node degree $\Delta < |V|$. |
We also achieve competitive results to other systems on the official leaderboards (Table 3 and 5). Notably, the top two systems, T5 (Raffel et al., 2020) and UnifiedQA (Khashabi et al., 2020), are trained with more data and use 8x to 30x more parameters than our model (ours has \( \sim 360M \) parameters). Excluding these and ensemble systems, our model is comparable in size and amount of data to other systems, and achieves the top performance on the two datasets.

Table 6 shows the result on MedQA-USMLE. QA-GNN outperforms state-of-the-art fine-tuned LMs (e.g., SapBERT). This result suggests that our method is an effective augmentation of LMs and KGs across different domains (i.e., the biomedical domain besides the commonsense domain).

### 4.6 Analysis

#### 4.6.1 Ablation studies

Table 7 summarizes the ablation study conducted on each of our model components (§3.1, §3.2, §3.3), using the CommonsenseQA IHdev set.

**Graph connection** (top left table): The first key component of QA-GNN is the joint graph that connects the \( z \) node (QA context) to QA entity nodes \( V_{q,a} \) in the KG (§3.1). Without these edges, the QA context and KG cannot mutually update their representations, hurting the performance: 76.5\% \( \rightarrow \) 74.8\%, which is close to the previous L+KG system, MHGRN. If we connected \( z \) to all the nodes in the KG (not just QA entities), the performance is comparable or drops slightly (-0.16\%).

**KG node relevance scoring** (top right table): We find the relevance scoring of KG nodes (§3.2) provides a boost: 75.56\% \( \rightarrow \) 76.54\%. As a
Attention visualization direction: BFS from Q

Where would you find a basement that can be accessed with an elevator?
A. closet  B. church  C. office building

Attention visualization direction: Q → O and A → O

Crabs live in what sort of environment?
A. salt water*  B. galapagos  C. fish market

Figure 4: Interpreting QA-GNN’s reasoning process by analyzing the node-to-node attention weights induced by the GNN. Darker and thicker edges indicate higher attention weights.

Variant of the relevance scoring in Eq. 1, we also experimented with obtaining a contextual embedding $w_v$ for each node $v \in V_{sub}$ and adding to the node features: $w_v = f_{enc}(text(z); text(v))$. However, we find that it does not perform as well (76.31%), and using both the relevance score and contextual embedding performs on par with using the score alone, suggesting that the score has a sufficient information in our tasks; hence, our final system simply uses the relevance score.

GNN architecture (bottom tables): We ablate the information of node type, relation, and relevance score from the attention and message computation in the GNN (§3.3). The results suggest that all these features improve the model performance. For the number of GNN layers, we find $L = 5$ works the best on the dev set. Our intuition is that 5 layers allow various message passing or reasoning patterns between the QA context ($z$) and KG, such as “$z \rightarrow 3$ hops on KG nodes $\rightarrow z$”.

4.6.2 Model interpretability

We aim to interpret QA-GNN’s reasoning process by analyzing the node-to-node attention weights induced by the GNN. Figure 4 shows two examples. In (a), we perform Best First Search (BFS) on the working graph to trace high attention weights from the QA context node (Z; purple) to Question entity nodes (blue) to Other (gray) or Answer choice entity nodes (orange), which reveals that the QA context $z$ attends to “elevator” and “basement” in the KG, “elevator” and “basement” both attend strongly to “building”, and “building” attends to “office building”, which is our final answer. In (b), we use BFS to trace attention weights from two directions: Z $\rightarrow$ Q $\rightarrow$ O and Z $\rightarrow$ A $\rightarrow$ O, which reveals concepts (“sea” and “ocean”) in the KG that are not necessarily mentioned in the QA context but bridge the reasoning between the question entity (“crab”) and answer choice entity (“salt water”). While prior KG reasoning models (Lin et al., 2019; Feng et al., 2020) enumerate individual paths in the KG for model interpretation, QA-GNN is not specific to paths, and helps to find more general reasoning structures (e.g., a KG subgraph with multiple anchor nodes as in example (a)).

4.6.3 Structured reasoning

Structured reasoning, e.g., precise handling of negation or entity substitution (e.g., “hair” $\rightarrow$ “art” in Figure 5b) in question, is crucial for making robust predictions. Here we analyze QA-GNN’s ability to perform structured reasoning and compare with baselines (fine-tuned LMs and existing LM+KG models).

Quantitative analysis. Table 9 compares model performance on questions containing negation words (e.g., no, not, nothing, unlikely), taken from the CommonsenseQA IH test set. We find that previous LM+KG models (KagNet, MHGRN) provide limited improvements over RoBERTa on questions with negation (+0.6%); whereas QA-GNN exhibits a bigger boost (+4.6%), suggesting its strength in structured reasoning. We hypothesize that QA-GNN’s joint updates of the representations of the QA context and KG (during GNN message passing) allows the model to integrate semantic nuances expressed in language. To further study this hypothesis, we remove the connections between $z$ and KG nodes from our QA-GNN (Table 9 bottom): now the performance on negation becomes close to the prior work, MHGRN, suggesting that the joint message passing helps for performing structured reasoning.

Qualitative analysis. Figure 5 shows a case study to analyze our model’s behavior for structured reasoning. The question on the left contains negation “not used for hair”, and the correct answer is “B. art supply”. We observe that in the 1st layer of QA-GNN, the attention from $z$ to question entities (“hair”, “round brush”) is diffuse. After multiples rounds of message passing on the working graph, $z$ attends strongly to “round brush” in the final layer of the GNN, but weakly to the negated entity “hair”.


Figure 5: Analysis of QA-GNN’s behavior for structured reasoning. Given an original question (left), we modify its negation (middle) or topic entity (right): we find that QA-GNN adapts attention weights and final predictions accordingly, suggesting its capability to handle structured reasoning.

Table 8: Case study of structured reasoning, comparing predictions by RoBERTa and our model (RoBERTa + QA-GNN). Our model correctly handles changes in negation and topic entities.

The model correctly predicts the answer “B. art supply”. Next, given the original question on the left, we (a) drop the negation or (b) modify the topic entity (“hair” → “art”). In (a), z now attends strongly to “hair”, which is not negated anymore. The model predicts the correct answer “A. hair brush”. In (b), we observe that QA-GNN recognizes the same structure as the original question (with only the entity swapped): z attends weakly to the negated entity (“art”) like before, and the model correctly predicts “A. hair brush” over “B. art supply”.  

Table 9: Performance on questions with negation in CommonsenseQA. (a) shows the difference with RoBERTa. Existing LM+KG methods (KagNet, MHGRN) provide limited improvements over RoBERTa (+0.6%); QA-GNN exhibits a bigger boost (+4.6%), suggesting its strength in structured reasoning.

the other hand, QA-GNN adapts predictions to the modifications correctly (except for double negation in the table bottom, which is a future work).

4.6.4 Effect of KG node relevance scoring

We find that KG node relevance scoring (§3.2) is helpful when the retrieved KG (G_sub) is large. Table 10 shows model performance on questions containing fewer (≤10) or more (>10) entities in the CommonsenseQA ITest set (on average, the former and latter result in 90 and 160 nodes in G_sub, respectively). Existing LM+KG models such as MHGRN achieve limited performance on questions with more entities due to the size and noisiness of retrieved KGs: 70.1% accuracy vs 71.5% accuracy on questions with fewer entities. KG node relevance scoring mitigates this bottleneck, reducing the accuracy discrepancy: 73.5% and 73.4% accuracy
on questions with more/fewer entities, respectively.

5 Related work and discussion

Knowledge-aware methods for NLP. Various works have studied methods to augment natural language processing (NLP) systems with knowledge. Existing works (Pan et al., 2019; Ye et al., 2019; Petroni et al., 2019; Bosselut et al., 2019) study pretrained LMs’ potential as latent knowledge bases. To provide more explicit and interpretable knowledge, several works integrate structured knowledge (KGs) into LMs (Mihaylov and Frank, 2018; Lin et al., 2019; Wang et al., 2019a; Yang et al., 2019; Wang et al., 2020b; Bosselut et al., 2021).

Question answering with LM+KG. In particular, a line of works propose LM+KG methods for question answering. Most closely related to ours are works by Lin et al. (2019); Feng et al. (2020); Lv et al. (2020). Our novelties are (1) the joint graph of QA context and KG, on which we mutually update the representations of the LM and KG; and (2) language-conditioned KG node relevance scoring. Other works on scoring or pruning KG nodes/paths rely on graph-based metrics such as PageRank, centrality, and off-the-shelf KG embeddings (Paul and Frank, 2019; Fadnis et al., 2019; Bauer et al., 2018; Lin et al., 2019), without reflecting the QA context.

Other QA tasks. Several works study other forms of question answering tasks, e.g., passage-based QA, where systems identify answers given or retrieved documents (Rajpurkar et al., 2016; Joshi et al., 2017; Yang et al., 2018), and KBQA, where systems perform semantic parsing of a given question and execute the parsed queries on knowledge bases (Berant et al., 2013; Yih et al., 2016; Yu et al., 2018). Different from these tasks, we approach question answering using knowledge available in LMs and KGs.

Knowledge representations. Several works study joint representations of external textual knowledge (e.g., Wikipedia articles) and structured knowledge (e.g., KGs) (Riedel et al., 2013; Toutanova et al., 2015; Xiong et al., 2019; Sun et al., 2019; Wang et al., 2019b). The primary distinction of our joint graph representation is that we construct a graph connecting each question and KG rather than textual and structural knowledge, approaching a complementary problem to the above works.

Graph neural networks (GNNs). GNNs have been shown to be effective for modeling graph-based data. Several works use GNNs to model the structure of text (Yasunaga et al., 2017; Zhang et al., 2018; Yasunaga and Liang, 2020) or KGs (Wang et al., 2020a). In contrast to these works, QA-GNN jointly models the language and KG. Graph Attention Networks (GATs) (Veličković et al., 2018) perform attention-based message passing to induce graph representations. We build on this framework, and further condition the GNN on the language input by introducing a QA context node (§3.1), KG node relevance scoring (§3.2), and joint update of the KG and language representations (§3.3).

6 Conclusion

We presented QA-GNN, an end-to-end question answering model that leverages LMs and KGs. Our key innovations include (i) Relevance scoring, where we compute the relevance of KG nodes conditioned on the given QA context, and (ii) Joint reasoning over the QA context and KGs, where we connect the two sources of information via the working graph, and jointly update their representations through GNN message passing. Through both quantitative and qualitative analyses, we showed QA-GNN’s improvements over existing LM and LM+KG models on question answering tasks, as well as its capability to perform interpretable and structured reasoning, e.g., correctly handling negation in questions.

Acknowledgment

We thank Rok Sosic, Weihua Hu, Jing Huang, Michele Catasta, members of the Stanford SNAP, P-Lambda and NLP groups and Project MOWGLI team, as well as our anonymous reviewers for valuable feedback. We gratefully acknowledge the support of DARPA under Nos. N660011924033 (MCS); Funai Foundation Fellowship; ARO under Nos. W911NF-16-1-0342 (MURI), W911NF-16-1-0171 (DURIP); NSF under Nos. OAC-1835598 (CINES), OAC-1934578 (HDR), CCF-1918940 (Expeditions), IIS-2030477 (RAPID); Stanford Data Science Initiative, Wu Tsai Neuro-sciences Institute, Chan Zuckerberg Biohub, Amazon, JP-Morgan Chase, Docomo, Hitachi, JD.com, KDDI, NVIDIA, Dell, Toshiba, and United Health Group. Hongyu Ren is supported by Masason Foundation Fellowship and the Apple PhD Fellowship. Jure Leskovec is a Chan Zuckerberg Biohub investigator.

Reproducibility

Code and data are available at https://github.com/michiyasunaga/qagnn. Experiments are available at https://worksheets.codalab.org/worksheets/0xf215deb05edf44a2ac353c711f52a25f.
References


Daniel Khoshabi, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirizi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. In Findings of EMNLP.


Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020b. On the variance of the adaptive learning rate and beyond. In International Conference on Learning Representations (ICLR).


An Yang, Quan Wang, Jing Liu, Kai Liu, Yajuan Lyu, Hua Wu, Qiaoqiao She, and Sujian Li. 2019. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In Association for Computational Linguistics (ACL).


Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In Association for Computational Linguistics (ACL).

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In Empirical Methods in Natural Language Processing (EMNLP).
