Connecting the Dots: A Knowledgeable Path Generator for Commonsense Question Answering

Peifeng Wang[1,3], **Nanyun Peng**[1,2,3], **Filip Ilievski**[3], **Pedro Szekely**[1,3], **Xiang Ren**[1,3]
[1]Department of Computer Science, University of Southern California
[2]Department of Computer Science, University of California, Los Angeles
[3]Information Sciences Institute, University of Southern California
{peifengw,xiangren}@usc.edu, violetpeng@cs.ucla.edu
{ilievski,pszekely}@isi.edu

Abstract

Commonsense question answering (QA) requires background knowledge which is not explicitly stated in a given context. Prior works use commonsense knowledge graphs (KGs) to obtain this knowledge for reasoning. However, relying entirely on these KGs may not suffice, considering their limited coverage and the contextual dependence of their knowledge. In this paper, we augment a general commonsense QA framework with a *knowledgeable path generator*. By extrapolating over existing paths in a KG with a state-of-the-art language model, our generator learns to connect a pair of entities in text with a dynamic, and potentially novel, multi-hop relational path. Such paths can provide structured evidence for solving commonsense questions without fine-tuning the path generator. Experiments on two datasets show the superiority of our method over previous works which fully rely on knowledge from KGs (with up to 6% improvement in accuracy), across various amounts of training data. Further evaluation suggests that the generated paths are typically interpretable, novel, and relevant to the task.[1]

1 Introduction

Solving commonsense QA tasks requires filling gaps with external knowledge. For instance, given the multiple-choice question in Figure 1, a system needs to know that *fungus* grows in moist environments, such as *caves*, and that a *cave* is a type of *geological feature*. Such commonsense knowledge is obvious for humans but most existing QA systems do not have it or cannot reason with it.

Although recent advances in pre-trained language models (LMs) have resulted in impressive performance on commonsense-related benchmarks (Zellers et al., 2018; Bhagavatula et al., 2019; Huang et al., 2019), it is unclear whether this is due to commonsense reasoning or to capturing spurious correlations in the data (Niven and Kao, 2019). Pre-trained LMs may answer a question correctly for wrong reasons, making them highly uninterpretable (Mitra et al., 2019).

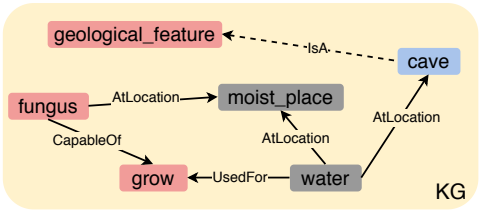

Figure 1: Our path generator learns to connect the question entities (in red) and choice entities (in blue). The dashed arrow indicates a missing link in a static KG.

Alternatively, a set of systems *retrieve* external knowledge either from large text corpora or knowledge graphs (KGs). A corpus, however, might not be an ideal source of commonsense knowledge, as such knowledge is seldom stated explicitly in text (Storks et al., 2019). In contrast, commonsense KGs, like ConceptNet (Speer et al., 2017) and ATOMIC (Sap et al., 2019), provide structured evidence about the relevant entities, thus enabling effective reasoning and higher interpretability. Existing systems retrieve knowledge from a KG in the form of: triplets (Mihaylov and Frank, 2018), multi-hop paths (Lin et al., 2019; Bauer et al., 2018), or subgraphs (Kapanipathi et al., 2019).

Despite the aforementioned benefits, exploiting these KGs poses the following challenges. Firstly, as KGs are known to suffer from *sparsity* (Li et al., 2016), they might not contain the knowledge needed to fill the gaps between the question and the answer. For example, a missing link *(cave, IsA, geological_feature)* in Figure 1 might prevent the QA system from choosing the correct answer. Recent

[1]The code is available at https://github.com/wangpf3/Commonsense-Path-Generator.

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 4129–4140

work on commonsense KG completion (Li et al., 2016; Bosselut et al., 2019; Bosselut and Choi, 2019) is limited to predicting the tail of a statement with known head and relation, or a single-hop relation between entities. Secondly, due to the large size and heterogeneity of modern KGs, *contextualization*—i.e., identifying a set of KG facts which are relevant or needed to answer a question—is also difficult (Fadnis et al., 2019). Simply retrieving all paths could introduce noisy information and potentially harm reasoning.

To address this gap between LMs and KGs, we propose a knowledgeable path generator (PG) that generalizes over the facts stored in a KG, rather than only retrieving them. We call our method *neural KG* due to its neural generalization over structured KGs, and, in contrast, we use the term *static KG* for methods which rely exclusively on existing facts in a KG. Our PG connects a pair of question and answer entities with a (novel) multi-hop path, which may not exist in the KG, allowing for missing facts like *(cave, IsA, geological_feature)* in Figure 1 to be considered during inference.

To learn such a generator, we: (1) sample a set of random walk instances from a static commonsense KG based on rules and constraints for informativeness and relevance (§3.1); (2) fine-tune a pre-trained language model — GPT-2 (Radford et al., 2019) on the sampled paths (§3.2). By doing so, we transfer the rich knowledge encoded in GPT-2 to our PG. This is expected to both enhance the generalization ability of the PG and combat the sparsity of KGs. Also, by generating high-quality missing links between the question and answer entities, we contextualize the task with relevant commonsense knowledge. To understand the impact of our multi-hop PG on downstream commonsense QA tasks, we integrate the PG in an augmented version of a general QA framework (§3.3).

We run experiments on two benchmark datasets *CommonsenseQA* (Talmor et al., 2018) and *OpenBookQA* (Mihaylov et al., 2018). The results show that out method performs better than previous systems augmented with static KGs by up to **6**% in accuracy, which also reveals its potential as a plug-in module for various datasets and as a vital complement to existing KG structures. In the low-resource setting, the accuracy gain over the baselines grows as the training data decreases, indicating a larger inductive bias of our generator. We also assess the quality and interpretability of our paths through both automatic and human evaluation.

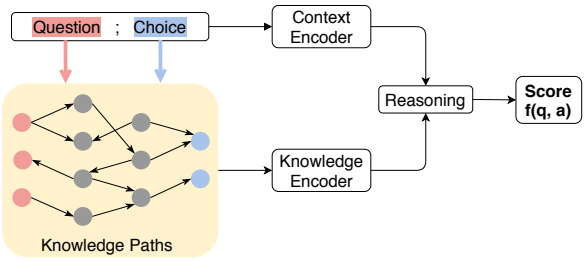

Figure 2: **Our KG-augmented QA Framework**. The reasoning module leverages both the unstructured context and structured knowledge to answer a question.

To summarize, our key contributions are:

1. We propose a method to generate task-relevant knowledge paths that may not exist in the original KG, thus addressing the contextualization and sparsity challenges of KGs.
2. We design and implement a framework with three variants of our PG, to understand the role of local and global graph information.
3. Extensive experiments on two benchmark datasets demonstrate the effectiveness of our method compared to previous methods, as well as its robustness to limited training data.

2 Preliminaries

Our multiple-choice commonsense QA setup follows prior work (Talmor et al., 2018; Mihaylov et al., 2018; Bisk et al., 2020): given a question q, a system selects exactly one of the choices a as an answer. To experiment with contextualized background knowledge, we adopt a general framework (Figure 2) consisting of a context module, a knowledge module and a reasoning module. The context module encodes both the question q and a choice a as *unstructured evidence*, while the knowledge module encodes external facts as *structured evidence*. Both the unstructured and the structured evidence are fed to the reasoning module, which produces a score for a question-choice pair. The choice with a highest score would be the predicted answer. Next, we introduce each module in detail.

Context Module We concatenate a question q and one of its choices a with a special token, and feed the sequence into a contextual encoder. This encoder generates an embedding $\mathbf{c}$, which serves as an unstructured evidence to our system. As commonly done for textual input, we consider a bidirectional pre-trained language model (Devlin et al., 2018; Liu et al., 2019) as a contextual encoder.

Knowledge Module Given a commonsense KG $\mathcal{G} = (\mathcal{E}, \mathcal{R})$, where $\mathcal{E}$ is the entity set and $\mathcal{R}$ is the relation set, we seek a set of relevant knowledge facts for a question-choice pair $\{q, a\}$, which would serve as structured evidence to support reasoning. We employ an entity recognition system to extract relevant entity mentions in the question (denoted by $\mathcal{E}^q = \{e^q\}$) and one of the choices ($\mathcal{E}^a = \{e^a\}$). We connect each pair of question-choice entities with a multi-hop path, which can be done either by retrieving existing paths for now (as in previous methods) or by generating paths (see §3.3). Formally, a path is $p(e^q, e^a) = \{e^q, r_0, e_1, r_1, ..., r_{T-1}, e^a\}$ where T is the number of hops. Note that when $T = 1$, the path is a single triplet. The set of paths is denoted by $\mathcal{P} = \{p(e^q, e^a) | e^q \in \mathcal{E}^q, e^a \in \mathcal{E}^a\}$.

Naturally, we employ a Relational Network (RN) (Santoro et al., 2017) to aggregate the retrieved paths into a static knowledge embedding $\mathbf{k}$, which serves as structured evidence. In essence, a RN is a composite function over the set $\mathcal{P}$:

$$\mathbf{k} = f_\phi(\{g_\theta(p) | p \in \mathcal{P}\}), \quad (1)$$

where f_ϕ could be any aggregation function and g_θ could be any neural network which projects a discrete path p into a fixed-size continuous embedding $\mathbf{p}$. We expect that not all paths contribute equally to choosing the right answer. Therefore, we construct the function f_ϕ as an attention network:

$$\mathbf{k} = \sum_{p \in \mathcal{P}} \alpha_p \mathbf{p}. \quad (2)$$

We compute the attention weight α_p by using the context embedding $\mathbf{c}$ as a query:

$$\alpha_p = \frac{exp(\hat{\alpha}_p)}{\sum_{p'} \exp(\hat{\alpha}_{p'})}, \quad (3)$$

where the context embedding $\mathbf{c}$ guides (as an attention query) the encoding of the structured evidence:

$$\hat{\alpha}_p = \mathbf{c}^\top \tanh(\mathbf{W}_{att} \cdot \mathbf{p} + \mathbf{b}_{att}). \quad (4)$$

Here, the attention network is parameterized by ($\mathbf{W}_{att}$,$\mathbf{b}_{att}$) and $\tanh(\cdot)$ is a nonlinear activation function. Regarding the function g_θ, we employ its original formulation:

$$g_\theta(p) = \text{MLP}[\mathbf{e^q}; (\mathbf{r_0} \circ ... \circ \mathbf{r_{T-1}}); \mathbf{e^a}], \quad (5)$$

where $[;]$ is vector concatenation and $\circ$ stands for element-wise multiplication. The components (entities and relations) of a path are represented by their feature vectors.

Reasoning Module This module leverages the unstructured evidence (the context embedding $\mathbf{c}$) and the structured one (the knowledge embedding $\mathbf{k}$), to compute the plausibility of a question-choice pair. We concatenate $\mathbf{c}$ with $\mathbf{k}$ and feed them to the final classification layer, which is a linear transformation that scores a question-choice pair $\{q, a\}$:

$$f(q, a) = \mathbf{W}_{cls} \cdot [\mathbf{c}; \mathbf{k}] + \mathbf{b}_{cls}, \quad (6)$$

The linear classification layer is parameterized by $(\mathbf{W}_{cls}, \mathbf{b}_{cls})$. We get the final probability over all choices by normalizing with softmax.

3 Knowledgeable Path Generator

Extracting the structured evidence by *retrieving* paths (or subgraphs) from a static KG, as in prior work (Mihaylov et al., 2018; Lin et al., 2019; Kapanipathi et al., 2019), faces two key challenges: sparsity and contextualization (§1). We thus propose a knowledgeable path generator (PG), which learns to connect a question-choice entity pair (e^q, e^a) with a multi-hop path. The generated paths are used as structured evidence in the knowledge module. Next, we detail the construction of training data (§3.1), the learning of our path generator over this data (§3.2), and the integration of the generator into the reasoning module (§3.3). Figure 3 presents an overview of our adapted knowledge module.

3.1 Knowledge Path Sampling

We sample paths from a commonsense KG using random walks, in order to provide training data for our PG. Such paths are expected to contain useful knowledge for commonsense QA tasks. Given a KG $\mathcal{G} = (\mathcal{E}, \mathcal{R})$, each sampled path $p = \{e_0, r_0, e_1, r_1, ..., r_{T-1}, e_T\}$ is a random walk on the graph, where $e_t \in \mathcal{E}$ and $r_t \in \mathcal{R}$. The number of hops, T, is a hyperparameter in our method. To improve the quality of the paths, we adopt two heuristic strategies. For *relevance*, we define a subset of relation types that are useful for answering commonsense questions, e.g., *AtLocation* and *IsA*, and filter out the remaining ones, e.g., *RelatedTo*, prior to sampling (see Appendix B for the discarded relations). For *informativeness*, we require all relation types in a path to be distinct.

We explore two sampling strategies in order to select the starting node of the random walks:

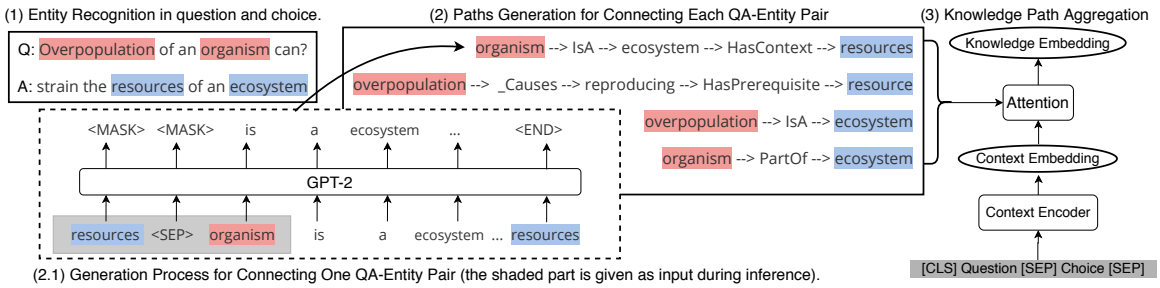

Figure 3: **Overview of our adapted knowledge module.** (1) Extraction of entities from a question and its answer choices. (2) Generation of a multi-hop knowledge path with our PG to connect each pair of question and answer entities. (3) Aggregation of the generated paths into a knowledge embedding.

Local Sampling. The random walks start from the entities that appear in the questions and answer choices of the training set of a benchmark. This strategy is expected to favor generation of paths that are tailored to the task.

Global Sampling. We conduct random walks starting from each entity in $\mathcal{E}$. This may divert our PG away from biasing on the local structure of the KG and enhance its generalizability to unseen data.

To include entities that are connected only with inverse triplets in a path, we add a reverse relation r^{-1} for each relation r. We also sample paths with a mixed number of hops T, so our generator can learn to connect entities using paths of variable length, when needed. The full path sampling procedure is described by Algorithm 1 in the Appendix.

3.2 Generating Paths to Connect Entities

We employ GPT-2 (Radford et al., 2019) as the backbone of our path generator. GPT-2 is a pre-trained language model that encodes rich unstructured knowledge from large text corpora. We foresee two benefits of combining a pre-trained model such as GPT-2 and a static KG: (1) the language model would be able to generate commonsense knowledge paths, by being enriched with relevant *structured* knowledge; (2) the *unstructured* knowledge encoded in the language model would help to alleviate the sparsity challenge of the static KGs.

Unlike COMET (Bosselut et al., 2019) which fine-tunes GPT (an earlier version of GPT-2) with independent triplets, we fine-tune GPT-2 with consecutive triplets that form paths (see Section 3.1). To do so, we first use GPT-2's Byte-Pair Encoding (Sennrich et al., 2016) to convert each symbolic path p to its textual form as a sequence $\{\mathbf{x}_0, \mathbf{y}_0, \mathbf{x}_1, \mathbf{y}_1, ..., \mathbf{y}_{T-1}, \mathbf{x}_T\}$, where $\mathbf{x}_t = \{x_t^1, x_t^2, ..., x_t^{|e_t|}\}$ are phrase tokens of the entity e_t and $\mathbf{y}_t = \{y_t^1, y_t^2, ..., y_t^{|r_t|}\}$ are phrase tokens of the relation r_t. The reverse relations are represented by adding a special prefix token "_". The resulting paths mimic natural language sentences to facilitate optimal usage of the knowledge encoded in the pre-trained language model. At inference time, in order to connect the question-choice entities, we also add the last entity phrase tokens $\mathbf{x}_T$ together with a separate token [SEP] at the beginning of each path sequence, which produces the final transformation $\mathbf{s}^p$. This informs the generator about the last entity it should output when generating a path. Table 1 provides an example path transformation.

Table 1: **Example Transformation of a Symbolic Path into Text.**

{predator, DistinctFrom, prey, IsA, animal}
→ { animal, [SEP], predator , distinct, from, prey, is, a, animal}

The PG learns to maximize the probability of the observed paths given the entity pairs. We use negative conditional log likelihood as a loss function:

$$\mathcal{L} = -\sum_{t=|\mathbf{x}_0|+|\mathbf{x}_T|+1}^{|\mathbf{s}^p|} \log P(s_t^p \mid s_{<t}^p), \quad (7)$$

where the conditional probability is defined as:

$$P(s_t^p \mid s_{<t}^p) = \text{softmax}(\mathbf{W}_{vocab} \cdot \mathbf{h_t}). \quad (8)$$

Here $\mathbf{h_t}$ denotes the final GPT-2 representation for s_t^p. $\mathbf{W}_{vocab}$ is the embedding matrix for the token-based vocabulary used by GPT-2, which generalizes well to unseen words.[2] During the inference, the target entity (e^a), the [SEP] token, and the starting entity (e^q) are fed to our generator (the shaded part in Table 1), and greedy decoding is used to generate a path connecting the two entities. Other constrained decoding strategies would be left as future work.

[2]This is because an unseen word of an entity or a relation may be split into several tokens that exist in the vocabulary.

3.3 Adapted Commonsense QA Framework

To facilitate integration of the structured evidence from our path generator instead of a static KG, we adapt the knowledge module from §2 slightly.

We construct the path set $\mathcal{P}$ by generating a multi-hop path $p(e^q, e^a)$ for each pair of a question entity e^q and a choice entity e^a with our PG and greedy decoding. To represent each path with an embedding, we perform mean pooling of the hidden states from the last layer of GPT-2 (before the softmax layer in Eq. 8) as a new formulation for the function g_θ:

$$g_\theta(p) = \text{MEAN}(\{\mathbf{h_1}, \mathbf{h_2}..., \mathbf{h}_{|\mathbf{s^p}|}\}). \quad (9)$$

Since GPT-2 has been pre-trained on a large corpus, we believe such representation should be sufficient for preserving the information of the paths. Then, the knowledge embedding obtained with the function f_ϕ of the RN (Eq. 2-4) is concatenated with the original static knowledge embedding as our new definition of $\mathbf{k}$.

The whole pipeline is optimized by minimizing its cross-entropy loss. The set of learnable parameters excludes the parameters of our proposed PG, because we observed that fixing their values yields optimal performance. This points to another advantage of our PG: after being fine-tuned on the sampled random walks from a KG, the PG could be integrated within an existing QA system with no further training.

4 Experiments

4.1 Datasets

We evaluate our method on two commonsense QA benchmarks: *CommonsenseQA* (Talmor et al., 2018) and *OpenBookQA* (Mihaylov et al., 2018). As the test set of *CommonsenseQA* is not publicly available, the predictions for it can only be evaluated once every two weeks via the official leaderboard. Thus, we report our test score on the leaderboard, and perform more extensive comparisons on the data split used in Lin et al. (2019). Besides questions and answers, *OpenBookQA* provides a collection of background facts in a textual form. We use the correspondence between these facts and their questions, prepared by Clark et al. (2019), as an additional input to the context module for all methods, except RoBERTa-large (see §4.5).

4.2 KG and Path Data Preparation

Entity Recognition We employ ConceptNet (Speer et al., 2017), a popular commonsense KG. As stated in §3.1, we disregard triplets that belong to a predefined set of relations (see Appendix). Similar to previous work (Lin et al., 2019), we use lexical matching to ground the entities mentioned in the question and the answer choices to our KG. One exception is that each answer choice in *CommonsenseQA* is treated as a single entity, as these tend to correspond directly to concepts in ConceptNet.

Path Sampling We sample a set of paths with varying lengths, ranging from 1 to 3 hops. Global sampling generates 2,825,692 paths, while local sampling results in 133,612 paths for CommonsenseQA and 105,155 for OpenBookQA. We split them into training/dev/test sets at a $90:5:5$ ratio.

4.3 Baselines

As baselines, we consider a fine-tuned LM, static KG-augmented models, and a 1-hop link predictor on the question and the answer entities.

Fine-tuned LM. To examine the role of the external knowledge, we compare to a "Fine-tuned LM" ablation of our QA framework without the knowledge module (§2).

Static KG Models. We compare to three static KG variants of our QA framework that model the knowledge module with path/graph encoders: (1) a RN degenerate version of our system, which computes a knowledge embedding by an attention mechanism over the retrieved paths for each question-choice entity pair; (2) Relational Graph Convolutional Networks (RGCN) (Schlichtkrull et al., 2018) which encode local graphs by using graph convolutional networks with relation-specific weight matrices; (3) GconAttn (Wang et al., 2019) which models the alignment between entities via attention and pools over all entity embeddings.

Link Prediction Model. This baseline predicts the relation between question and answer entities instead of creating or finding knowledge paths. Namely, we employ TransE (Bordes et al., 2013) to learn a representation for every entity and relation in ConceptNet, which is then leveraged to predict a 1-hop relation for each pair of question and answer entities. The representations for each resulting triplet are used as 1-hop path embeddings. The rest of this baseline is identical to our QA framework.

Table 2: **Test accuracy with varying proportions of *CommonsenseQA*** (using the data split in (Lin et al., 2019)). Results (as mean and standard deviation) are computed over 4 experimental runs with different random seeds (top score in boldface, second score underlined). Parts of the results for baselines are reported from our another work (Feng et al., 2020).

Methods	BERT-large			RoBERTa-large		
	20% Train	60% Train	100% Train	20% Train	60% Train	100% Train
Fine-tuned LM (w/o KG)	46.25 (±0.63)	52.30 (±0.16)	55.39 (±0.40)	55.28 (±0.35)	65.56 (±0.76)	68.69 (±0.56)
+ RN	45.12 (±0.69)	54.23 (±0.28)	<u>58.92</u> (±0.14)	61.32 (±0.68)	66.16 (±0.28)	69.59 (±3.80)
+ RGCN	48.67 (±0.28)	54.71 (±0.37)	57.13 (±0.36)	58.58 (±0.17)	68.33 (±0.85)	68.41 (±0.66)
+ GconAttn	47.95 (±0.11)	54.96 (±0.69)	56.94 (±0.77)	57.53 (±0.31)	68.09 (±0.63)	69.88 (±0.47)
+ Link Prediction	47.10 (±0.79)	53.96 (±0.56)	56.02 (±0.55)	60.84 (±1.36)	66.29 (±0.29)	69.33 (±0.98)
+ PG-Local	<u>50.20</u> (±0.31)	<u>55.68</u> (±0.07)	56.81 (±0.73)	61.56 (±0.72)	67.77 (±0.83)	70.43 (±0.65)
+ PG-Global	49.89 (±1.03)	55.47 (±0.92)	57.21 (±0.45)	<u>62.93</u> (±0.82)	<u>68.65</u> (±0.02)	<u>71.55</u> (±0.99)
+ PG-Full	**51.97** (±0.26)	**57.53** (±0.19)	**59.07** (±0.30)	**63.72** (±0.77)	**69.46** (±0.23)	**72.68** (±0.42)

Table 3: **Test accuracy on *OpenBookQA*.** Methods with AristoRoBERTa leverage the textual evidence by Clark et al. (2019) as an additional input to the context module.

Methods	RoBERTa-large	AristoRoBERTa
Fine-tuned LMs (w/o KG)	64.80 (±2.37)	78.40 (±1.64)
+ RN	65.20 (±1.18)	75.35 (±1.39)
+ RGCN	62.45 (±1.57)	74.60 (±2.53)
+ GconAtten	64.75 (±1.48)	71.80 (±1.21)
+ Link Prediction	66.30 (±0.48)	77.25 (±1.11)
+ PG-Local	<u>70.05</u> (±1.33)	<u>79.80</u> (±1.45)
+ PG-Global	68.40 (±0.31)	**80.05** (±0.68)
+ PG-Full	**71.20** (±0.96)	79.15 (±0.78)

Table 4: **Test accuracy on *CommonsenseQA*'s official leaderboard**. Note that the SOTA system, UnifiedQA is impractical (11B parameters) in an academic setting.

Methods	Single	Ensemble
RoBERTa (Liu et al., 2019)	72.1	72.5
RoBERTa+FreeLB (Zhu et al., 2019)	-	73.1
RoBERTa+HyKAS (Ma et al., 2019)	73.2	-
XLNet+DREAM	73.3	-
RoBERTa+KE	-	73.3
RoBERTa+KEDGN	-	74.4
XLNet+GraphReason (Lv et al., 2019)	75.3	-
Albert (Lan et al., 2019)	-	76.5
UnifiedQA* (Khashabi et al., 2020)	**79.1**	-
Albert+PG-Full	75.6	<u>78.2</u>

4.4 Model Variations

We experiment with three variants of our method which differ in terms of the knowledge embedding: (1) PG-Full: combination of our global PG and a static RN as detailed in §3.3; (2) PG-Local: a local PG which is trained on both local and global paths; (3) PG-Global: a global, data-independent PG which is trained on global paths only. We note that PG-Local and PG-Global do not include the static knowledge embedding.

4.5 Results

Main Results For all systems, we experiment with several encoders as a context module: BERT-large (Devlin et al., 2018) and RoBERTa-large (Liu et al., 2019) for *CommonsenseQA*, RoBERTa-large and AristoRoBERTa (Clark et al., 2019) for *OpenBookQA*. Tables 2 and 3 show the results for CommonsenseQA and OpenBookQA, respectively. On both datasets, we observe consistent improvements brought by our method with different context encoders. Our full model which, combines both generated and static knowledge, achieves the best performance overall, suggesting these two knowledge sources are complementary. Typically, either our local or global variant yields second best results, demonstrating the effectiveness of the generated paths as structured evidence and their superiority over the static KG methods. The comparable performance of Link Prediction to the static KG methods indicates that even predicting 1-hop knowledge paths helps to address the KG sparsity.

Furthermore, we report comparable results to the other systems on the official test sets, accessible via the leaderboards (Tables 4 and 5). Notably, the two best-performing systems, UnifiedQA (Khashabi et al., 2020) and TTTTT (Raffel et al., 2019), are based on the T5 language model (Raffel et al., 2019), which requires excessive computational resources and is impractical in an academic setting. Excluding these, our full method achieves the best performance on both datasets.

Less Labeled Data To compare the robustness of our model and the baselines to sparsity, we perform experiments with {20%, 40%, 60%, 80%, 100%} of the training data from both datasets. The results, displayed in Table 2 and Figure 4, show that our method (with RoBERTa) performs better or equal to the baselines with any amount of training data. The performance gain brought by either our Global or Full model is higher when less data is used, which shows that introducing structured evidence as inductive bias helps in a low-resource setting.

Table 5: **Test accuracy on *OpenBookQA* leaderboard**. All listed methods leverage the provided science facts as additional textual input. Note that the top 2 systems, UnifiedQA (11B parameters) and TTTTT (3B parameters) are computationally expensive and impractical in an academic setting.

Methods	Test
Careful Selection (Banerjee et al., 2019)	72.0
AristoRoBERTa	77.8
KF + SIR (Banerjee and Baral, 2020)	80.0
Albert + KB	81.0
TTTTT* (Raffel et al., 2019)	83.2
UnifiedQA* (Khashabi et al., 2020)	**87.2**
AristoRoBERTa + PG-Full	80.2
Albert + PG-Full	81.8

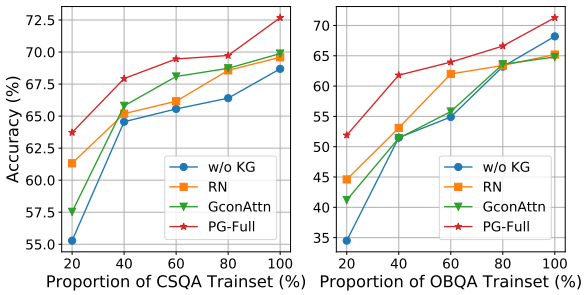

Figure 4: Test accuracy on *CommonsenseQA* (left) and *OpenBookQA* (right) with different proportions of training data.

Ablation Study We study the contribution of different strategies for learning our generator based on the performance of our Global and Local variants in Tables 2-3. We also include another variant by training our path generator from scratch, i.e. training a randomly-initialized model with the same architecture as GPT-2 instead of fine-tuning a pre-trained one. This *Scratch* variant achieves 68.75 and 65.50 accuracy on the *CommonsenseQA* and *OpenBookQA* test sets, respectively, with RoBERTa-large as the text encoder. Its performance thus resembles that of the static KG baselines while our *Full* method achieves 72.68 and 71.20. This demonstrates that learning paths from scratch approximates what a static KG has already, whereas the unstructured knowledge stored in a pre-trained GPT-2 helps to complement missing knowledge in a static KG. When coupled with a more powerful encoder like RoBERTa or Albert, our *Global* variant achieves comparable or better results than our *Local* variant, without fitting the paths to the task, and thus holds a promise to enhance generalization on a wider range of datasets.

4.6 Study of Path Quality & Interpretability

Automatic Evaluation We perform automatic evaluation of the validity and novelty of the generated paths from our *Global* and *Scratch* PG variants. To automatically measure *validity*, we analyze (1) the proportion of paths which successfully connect the head and the tail entities (`Connection`), (2) the proportion of entities/relations found in ConceptNet (`Valid Entity / Relation`). We also leverage a commonsense knowledge base completion model, *Bilinear AVG* (Li et al., 2016), which produces a score for a given triplet. This model reportedly achieves 92.5% accuracy on commonsense knowledge completion and has been used in previous work (Bosselut et al., 2019). We average the scores of all the triplets in a path which are missing in ConceptNet as its `Score`. We compute *novelty* as the proportion of paths which contain at least one triplet missing in ConceptNet (`Novelty`).

Table 6: **Automatic and Human Evaluation of the generated Paths on the task testset**. All scores are scaled to be percentage-based.

Metric	CommonsenseQA		OpenBookQA	
	Global	Scratch	Global	Scratch
Connection	97.33	91.16	96.03	96.01
Valid Entity	98.64	97.78	99.21	97.97
Valid Relation	100.00	100.00	100.00	100.00
Score	59.31	53.27	57.74	50.62
Novelty	75.82	58.18	78.93	53.81
H-Valid	89.20	60.13	84.93	53.73
H-Relevance	87.53	70.53	88.13	74.00

The results are presented in Table 6. Firstly, our two generator variants are able to connect a vast majority of the entity pairs with a valid path (over 90% `Connection`). For this purpose, our generators only use the relations in the relation set instead of other, out-of-KG phrases (100% `Valid Relation`). In addition, the novel paths from the *Global* generator are of higher quality compared with the ones from the *Scratch* generator, given that any fact with a score over 0.5 is classified as positive by *Bilinear AVG*, which is later confirmed by our human evaluation as well. The *Global* generator also has a higher `Novelty`, indicating the necessity of transferring knowledge from a pre-trained GPT-2 to complement a static KG.

Human Evaluation We also conduct human evaluation on two dimensions of the generated paths: (1) *validity* (How valid are the paths?) (2) *relevance* (How relevant are the paths to the question?). We randomly sample 50 paths from our *Global* and *Scratch* generator for different question-choice entity pairs in the test datasets. For each path, we provide the corresponding question and answer

Table 7: Paths from question to gold answer entities, with novel and valid triplets in boldface.

Q1: Where would you find magazines along side many other printed works? A: doctor. B^* : *bookstore*. C: market. D: train station. E: mortuary. PG-Global (2-hop): {magazine, IsA, book, AtLocation, bookstore} PG-Scratch: {magazine, _IsA, magazine, AtLocation, bookstore}
Q2: If you want harmony, what is something you should try to do with the world? A: take time. B: make noise. C: make war. D^* : *make peace*. E: make haste. PG-Global (2-hop): {**harmony, _MotivatedByGoal, make better world**, HasPrerequisite, make peace} PG-Scratch: {harmony, _UsedFor, committing perjury, Causes, make peace}
Q3: Janet was watching the film because she liked what? A: rejection. B: laughter. C^* : *being entertained*. D: fear. E: bordem. PG-Global (1-hop): {**film, _CausesDesire, being entertained**} PG-Scratch: {film, _HasContext, being entertained}

choices as the context. We ask three annotators to score each path from 1 (Not at all) to 5 (Very), resulting in a total of 150 scores for each dimension/-generator/dataset. The averages of these scores are reported as `H-Valid` and `H-Relevance` in Table 6. For both dimensions, our Global generator achieves higher scores, showing the ability of fine-tuning a pre-trained GPT-2 as our generator to learn the path distribution which is of high quality and relevant to commonsense QA.

Path Interpretability. In Table 7, we compare example paths generated by our *Global* and *Scratch* variants to connect the question entities to the gold answer entities. In Q1, our *Global* generator provides knowledge about the location of an entity with a 2-hop path, which helps with answering such "Where" questions. Although the path from our *Scratch* generator also contains the *AtLocation* relation, its first generated hop (*_IsA*) is less informative. In Q2, our *Global* generator is able to connect complex ideas about harmony and making peace with a 2-hop path, while the path from the *Scratch* variant contains incorrect information: *peace* is caused by *committing perjury*. In Q3, the path from our *Global* generator is able to predict the relevant property of an entity and realizes that a 1-hop relation suffices in this case. Our *Scratch* variant, however, predicts a less precise relation (*_HasContext*). These cases show the path generalization ability of the fine-tuned pre-trained GPT-2, owed to its unstructured knowledge. We refer readers to Table 12 in Appendix for more cases.

5 Related Work

Multi-hop Reasoning on KGs. Recent benchmarks for commonsense QA and related tasks like open domain QA (Yang et al., 2018) and reading comprehension (Welbl et al., 2018), require systems to conduct multi-hop reasoning. Existing systems typically employ entity linking to recognize the relevant entities, ground them to a KG, and retrieve the paths from the local graph neighborhood around the entities. The retrieved paths are scored or ranked using graph-based metrics (e,g., PageRank, centrality) (Paul and Frank, 2019; Fadnis et al., 2019; Bauer et al., 2018), handcrafted rules (Kapanipathi et al., 2019) or neural methods (e.g., attention mechanisms) (Kundu et al., 2018; Lin et al., 2019). Rather than relying on a static KG, our PG is able to generate knowledge paths dynamically, even when these are absent in the KG.

Dynamic Knowledge Path Generation. Several methods generate knowledge paths instead of extracting them from static KGs. Asai et al. (2019) learn reasoning paths by forming sequences of evidence documents, however, their approach relies on the inter-document hyperlinks to establish relations in the constructed KG. The extractor of Fu et al. (2019) retrieves missing facts in order to address the sparsity of KGs. Unlike our work, their setting is limited to knowledge graph completion, where both a query entity and a single query relation are given. The most similar existing work to ours is that by Bosselut and Choi (2019), which also leverages GPT-2 to dynamically generate knowledge paths. We see two key differences between this method and ours: (1) they expand their paths gradually by predicting the next entity one at a time, while we generate the paths in an end-to-end manner; (2) their method is restricted to a setting where the context could be treated as a single entity and the question - as a query relation, which is not a limitation to our method.

6 Conclusion

In this paper, we propose a generator of multi-hop knowledge paths, which provides structured evidence for answering commonsense questions. The generator, learned by fine-tuning GPT-2 on random walks sampled from ConceptNet, produces a path between each pair of question and answer entities. All generated paths are aggregated into a knowledge embedding and fused with a context embedding given by a text encoder for classification. Our QA framework enhanced with this generator outperformes both pre-trained language models and prior KG-augmented methods on two commonsense QA benchmarks. The accuracy gain increases with less training data. Furthermore, automatic- and human-based evaluations of the generated paths yield high scores for their validity,

novelty, and relevance. Future research should investigate how to optimally fuse the knowledge and the context embeddings. It should also address the ambiguity of the entity mentions in the questions, the answers, and the lexical nodes in ConceptNet.

Acknowledgments

We thank the anonymous reviewers for their insightful comments. This material is based upon work sponsored by the DARPA MCS program under Contract No. N660011924033 with the United States Office Of Naval Research.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2019. Learning to retrieve reasoning paths over wikipedia graph for question answering. *arXiv preprint arXiv:1911.10470*.

Pratyay Banerjee and Chitta Baral. 2020. Knowledge fusion and semantic knowledge ranking for open domain question answering. *arXiv preprint arXiv:2004.03101*.

Pratyay Banerjee, Kuntal Kumar Pal, Arindam Mitra, and Chitta Baral. 2019. Careful selection of knowledge to solve open book question answering. *arXiv preprint arXiv:1907.10738*.

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. *arXiv preprint arXiv:1809.06309*.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Wen-tau Yih, and Yejin Choi. 2019. Abductive commonsense reasoning. *arXiv preprint arXiv:1908.05739*.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Antoine Bosselut and Yejin Choi. 2019. Dynamic knowledge graph construction for zero-shot commonsense question answering. *arXiv preprint arXiv:1911.03876*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. *arXiv preprint arXiv:1906.05317*.

Peter Clark, Oren Etzioni, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, Niket Tandon, Sumithra Bhakthavatsalam, et al. 2019. From'f'to'a'on the ny regents science exams: An overview of the aristo project. *arXiv preprint arXiv:1909.01958*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kshitij Fadnis, Kartik Talamadupula, Pavan Kapanipathi, Haque Ishfaq, Salim Roukos, and Achille Fokoue. 2019. Heuristics for interpretable knowledge graph contextualization. *arXiv preprint arXiv:1911.02085*.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. *arXiv preprint arXiv:2005.00646*.

Cong Fu, Tong Chen, Meng Qu, Woojeong Jin, and Xiang Ren. 2019. Collaborative policy learning for open knowledge graph reasoning. *arXiv preprint arXiv:1909.00230*.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos qa: Machine reading comprehension with contextual commonsense reasoning. *arXiv preprint arXiv:1909.00277*.

Pavan Kapanipathi, Veronika Thost, Siva Sankalp Patel, Spencer Whitehead, Ibrahim Abdelaziz, Avinash Balakrishnan, Maria Chang, Kshitij Fadnis, Chulaka Gunasekara, Bassem Makni, et al. 2019. Infusing knowledge into the textual entailment task using graph convolutional networks. *arXiv preprint arXiv:1911.02060*.

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. *arXiv preprint arXiv:2005.00700*.

Souvik Kundu, Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Exploiting explicit paths for multi-hop reading comprehension. *arXiv preprint arXiv:1811.01127*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. 2016. Commonsense knowledge base completion. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1445–1455.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. *arXiv preprint arXiv:1909.02151*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2019. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. *arXiv preprint arXiv:1909.05311*.

Kaixin Ma, Jonathan Francis, Quanyang Lu, Eric Nyberg, and Alessandro Oltramari. 2019. Towards generalizable neuro-symbolic systems for commonsense question answering. *arXiv preprint arXiv:1910.14087*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.

Todor Mihaylov and Anette Frank. 2018. Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge. *arXiv preprint arXiv:1805.07858*.

Arindam Mitra, Pratyay Banerjee, Kuntal Kumar Pal, Swaroop Mishra, and Chitta Baral. 2019. Exploring ways to incorporate additional knowledge to improve natural language commonsense question answering. *arXiv preprint arXiv:1909.08855*.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. *arXiv preprint arXiv:1907.07355*.

Debjit Paul and Anette Frank. 2019. Ranking and selecting multi-hop knowledge paths to better predict human needs. *arXiv preprint arXiv:1904.00676*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. 2017. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4967–4976.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: an atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Robert Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Shane Storks, Qiaozi Gao, and Joyce Y Chai. 2019. Commonsense reasoning for natural language understanding: A survey of benchmarks, resources, and approaches. *arXiv preprint arXiv:1904.01172*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Xiaoyan Wang, Pavan Kapanipathi, Ryan Musa, Mo Yu, Kartik Talamadupula, Ibrahim Abdelaziz, Maria Chang, Achille Fokoue, Bassem Makni, Nicholas Mattei, 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, pages 7208–7215.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. *arXiv preprint arXiv:1808.05326*.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Thomas Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for language understanding. *arXiv preprint arXiv:1909.11764*.

A Algorithm for Path Sampling

Algorithm 1 Path Sampling

Input: $\mathcal{G} = (\mathcal{E}, \mathcal{R})$ and a set of all the question entities $\{e^q\}$
Output: A set of triplet paths $\{p\}$.

repeat
  if Do Global Sampling **then**
    current_node $u \leftarrow uniform\_sample(\mathcal{E})$
  else
    current_node $u \leftarrow uniform\_sample(\{e^q\})$
  end if
  $p \leftarrow \{u\}$
  for t = 1 to T **do**
    $N \leftarrow Neighbor(u)$
    next_node $v \leftarrow uniform\_sample(N)$
    $M \leftarrow All\_Relations(u, v)$
    while TRUE **do**
      $r \leftarrow uniform\_sample(M)$
      if r not in p **then**
        BREAK
      end if
    end while
    $p \leftarrow p \cup \{r, v\}$
    $u \leftarrow v$
  end for
until Maximum number of paths achieved.

B Discarded Relations

When sampling knowledge paths, we discard some relation types which are regarded to be uninformative and offer little help for answering the questions. They include *RelatedTo*, *Synonym*, *Antonym*, *DerivedFrom*, *FormOf*, *EtymologicallyDerivedFrom* and *EtymologicallyRelatedTo*.

Table 8: QA Dataset Statistics.

	Train	Dev	Test
CommonsenseQA (official)	9,741	1,221	1,140
CommonsenseQA (Lin et al.)	8,500	1,221	1,241
OpenBookQA	4,957	500	500

C Datasets Split

Both CommonsenseQA[3] and OpenbookQA[4] have their datasets available on their leaderboard pages. The dataset split used in (Lin et al., 2019) is also available by request and we have included it as a supplementary material.

[3] https://www.tau-nlp.org/commonsenseqa

[4] https://leaderboard.allenai.org/open_book_qa/submissions/public

Table 9: Learning rate of different context modules for *CommonsenseQA*.

	Learning Rate	Batch Size
BERT-large	2e-5	32
RoBERTa-large	2e-6	16
Albert-xxlarge-v2	1e-5	16

Table 10: Learning rate of different context modules for *OpenBookQA*.

	Learning Rate	Batch Size
Roberta-large	1e-5	32
AristoRoBERTa	2e-5	16
Albert-xxlarge-v2	1e-5	16

D Implementation Details

Path Generator Training We employ a pre-trained GPT2-base model (Radford et al., 2019) to initialize our generator. Then we fine-tune the generator with an initial learning rate of $1e-5$ and a batch size of 64. The learning rate is changed with a warm-up period of 500 mini batches and then linearly decayed. The training lasts until the loss on the development set no longer decreases for 2 epochs.

Training on the Task Datasets We search for the optimal hyper-parameters based on the classification accuracy on the development set. The learning rate for the context module is chosen from $\{2e-6, 5e-6, 1e-5, 2e-5, 5e-5\}$. The learning rate for the rest of the parameters is set to $1e-3$. The batch size is chosen from $\{8, 16, 32, 64, 128\}$. A large batch size is achieved by accumulating gradient through several small batches. The training lasts until the accuracy on the development set no longer increases for 2 epochs. The optimal hyper-parameters for both datasets are listed in Tables 9-10.

Model Size We list the model size of the major modules in our QA framework in Table 11. These include the different pre-trained LMs used as a context module, the backbone of our PG (GPT-2), and the RN used for the static knowledge module.

Table 11: Number of parameters of the major modules in our QA framework.

	# Parameters
BERT-large	340M
RoBERTa-large	355M
AristorRoBERTa	355M
Albert-xxlarge-v2	223M
GPT2-base	117M
RN	399K

Table 12: More Paths from questions to gold answer entities, with novel and valid triplets in boldface.

Q1: He spent all summer in his room playing video games, because of this it wasn't surprising for Mother to find a stack of dirty dishes in her what? A^*: son's room. B: party. C: dishwasher. D: restaurant kitchen. E: shoes PG-Global: {play_video, _UsedFor, **computer, AtLocation, son's room**} PG-Scratch: {play_video, _UsedFor, machine, _IsA, son's room}
Q2: What do people typically do while playing guitar? A: cry. B: hear sounds. C^*: singing. D: arthritis. E: making music. PG-Global: {guitar, Usedfor, **playing music, HasSubevent, singing**} PG-Scracth: {guitar, HasContext, music, _Causes, singing}
Q3: Blue read material outside of his comfort zone because he wanted to gain what? A^*: new perspective. B: entertained. C: understanding. D: hunger. E: tired eyes. PG-Global: {**reading material, _HasPrerequisite, learning about subject, Causes, new perspective**} PG-Scratch: {reading material, _HasSubevent, reading, Causes, new perspective}
Q4: Bob the lizard lives in a warm place with lots of water. Where does he probably live? A: rock. B^*: tropical rainforest. C: jazz club. D: new mexico. E: rocky places. PG-Global: {**warm place, _AtLocation, forest**, _IsA, tropical rainforest} PG-Scracth: {warm place, _AtLocation, tropical rainforest}
Q5: She was always helping at the senior center, it brought her what? A: satisfaction. B: heart. C: feel better. D: pay. E: happiness. PG-Global: {help, _UsedFor, giving assistance, Causes, happiness} PG-Scratch: {help, _HasSubevent, giving assistance, MotivatedByGoal, happiness}
Q6: What is likely to satisfy someone's curiosity? A^*: hear news. B: read book. C: see favorite show. D: comedy show. E: go somewhere. PG-Global: {curiosity, CausesDesire, find information, HasSubevent, read, _Hasprerequisite, hear news} PG-Scratch: {curiosity, CausesDesire, hear news}
Q7: Where would a person be doing when having to wait their turn? A: have patience. B: get in line. C: sing. D^*: stand in line. E: turn left. PG-Global: {wait, _HasPrerequisite, stand in line} PG-Scratch: {wait, _HasPrerequisite, stand in line}
Q8: It's easier for human's to survive in: A: a cave. B: the ocean. C^*: a town. D: alone. PG-Global: {survive _MotivatedByGoal, **live, _UsedFor, townhouse, AtLocation, town**} PG-Scratch: {survive, _HasProperty, town}
Q9: A man wanted to find the United States on a visual, where should he look? A: history book. B^*: atlas. C: tv channels. D: northern hemisphere. E: map. PG-Global: {**visual, _HasContext, map**, AtLocation, atlas} PG-Scratch: {visual, _IsA, atlas}
Q10: What leads to someone going to to bed? A: bad dreams. B: lazyness. C: get pregnant. D^*: sleepiness. E: rest. PG-Global: {bed, UsedFor, sleeping, Causes, sleepiness} PG-Scratch: {bed, UsedFor, sleepiness}