Exploiting Explicit Paths for Multi-hop Reading Comprehension

Souvik Kundu[†*] and **Tushar Khot**[‡] and **Ashish Sabharwal**[‡] and **Peter Clark**[‡]
[†]Department of Computer Science, National University of Singapore
[‡]Allen Institute for Artificial Intelligence, Seattle, WA, U.S.A.
souvik@comp.nus.edu.sg, {tushark,ashishs,peterc}@allenai.org

Abstract

We propose a novel, path-based reasoning approach for the multi-hop reading comprehension task where a system needs to combine facts from multiple passages to answer a question. Although inspired by multi-hop reasoning over knowledge graphs, our proposed approach operates directly over unstructured text. It generates potential paths through passages and scores them without any direct path supervision. The proposed model, named PathNet, attempts to extract implicit relations from text through entity pair representations, and compose them to encode each path. To capture additional context, PathNet also composes the passage representations along each path to compute a passage-based representation. Unlike previous approaches, our model is then able to explain its reasoning via these explicit paths through the passages. We show that our approach outperforms prior models on the multi-hop WikiHop dataset, and also can be generalized to apply to the OpenBookQA dataset, matching state-of-the-art performance.

1 Introduction

Many reading comprehension (RC) datasets (Rajpurkar et al., 2016; Trischler et al., 2017; Joshi et al., 2017) have been proposed recently to evaluate a system's ability to answer a question from a given text passage. However, most of the questions in these datasets can be answered by using only a single sentence or passage. As a result, systems designed for these tasks may not be able to compose knowledge from multiple sentences or passages, a key aspect of natural language understanding. To remedy this, new datasets (Weston et al., 2015; Welbl et al., 2018; Khashabi et al., 2018a; Mihaylov et al., 2018) have been proposed, requiring a system to combine information from multiple sentences in order to arrive at the answer, referred to as *multi-hop reasoning*.

Query: (always breaking my heart, record_label, ?)

Supporting Passages:
(p1) "Always Breaking My Heart" is the second single from Belinda Carlisle's A Woman and a Man album , released in 1996 (see 1996 in music) . It ...
(p2) A Woman and a Man is the sixth studio album by American singer Belinda Carlisle, released in the United Kingdom on September 23, 1996 by Chrysalis Records (then part of the EMI Group, ...

Candidates: chrysalis records, emi group, virgin records, ...

Answer: chrysalis records

Paths:
("Always Breaking My Heart" ... *single from* ... A Woman and a Man)
(A Woman and a Man ... *released* ... *by* ... Chrysalis Records)

Figure 1: Example illustrating our proposed path extraction and reasoning approach.

Multi-hop reasoning has been studied for question answering (QA) over structured knowledge graphs (Lao et al., 2011; Guu et al., 2015; Das et al., 2017). Many of the successful models explicitly identify paths in the knowledge graph that led to the answer. A strength of these models is high interpretability, arising from explicit path-based reasoning over the underlying graph structure. However, they cannot be directly applied to QA in the absence of such structure.

Consequently, most multi-hop RC models over unstructured text (Dhingra et al., 2017; Hu et al., 2018) extend standard attention-based models from RC by iteratively updating the attention to indirectly "hop" over different parts of the text. Recently, graph-based models (Song et al., 2018; Cao et al., 2018) have been proposed for the WikiHop dataset (Welbl et al., 2018). Nevertheless, these models still only implicitly combine knowl-

[*]Work performed while doing an internship at the Allen Institute for Artificial Intelligence.

edge from all passages, and are therefore unable to provide explicit reasoning paths.

We propose an approach[1] for multiple choice RC that explicitly extracts potential paths from text (without direct path supervision) and encodes the knowledge captured by each path. Figure 1 shows how to apply this approach to an example in the WikiHop dataset. It shows two sample paths connecting an entity in the question (*Always Breaking My Heart*) to a candidate answer (*Chrysalis Records*) through a singer (*Belinda Carlisle*) and an album (*A Woman and a Man*).

To encode the path, our model, named PathNet, first aims to extract implicit (latent) relations between entity pairs in a passage based on their contextual representations. For example, it aims to extract the implicit *single from* relation between the song and the name of the album in the first passage. Similarly, it extracts the *released by* relation between the album and the record label in the second passage. It learns to compose the extracted implicit relations such that they map to the main relation in the query, in this case *record_label*. In essence, the motivation is to learn to extract implicit relations from text and to identify their valid compositions, such as: (x, *single from*, y), (y, *released by*, z) → (x, *record_label*, z). Due to the absence of direct supervision on these relations, PathNet does not explicitly extract these relations. However, our qualitative analysis on a sampled set of instances from WikiHop development set shows that the top scoring paths in 78% of the correctly answered questions have implied relations in the text that could be composed to derive the query relations.

In addition, PathNet also learns to compose aggregated passage representations in a path to capture more global information: encoding(p1), encoding(p2) → (x, *record_label*, z). This passage-based representation is especially useful in domains such as science question answering where the lack of easily identifiable entities limits the effectiveness of the entity-based path representation. While this passage-based representation is less interpretable than the entity-based path representation, it still identifies the two passages used to select the answer, compared to a spread out attention over all documents produced by previous graph-based approaches.

We make three main contributions:

(1) A novel path-based reasoning approach for multi-hop QA over text that produces explanations in the form of explicit paths; (2) A model, PathNet, which aims to extract implicit relations from text and compose them; and (3) Outperforming prior models on the target WikiHop dataset[2] and generalizing to the open-domain science QA dataset, OpenBookQA, with performance comparable to prior models.

2 Related Work

We summarize related work in QA over text, semi-structured knowledge, and knowledge graphs.

Multi-hop RC. Recent datasets such as bAbI (Weston et al., 2015), Multi-RC (Khashabi et al., 2018a), WikiHop (Welbl et al., 2018), and OpenBookQA (Mihaylov et al., 2018) have encouraged research in multi-hop QA over text. The resulting multi-hop models can be categorized into state-based and graph-based reasoning models. State-based reasoning models (Dhingra et al., 2017; Shen et al., 2017; Hu et al., 2018) are closer to a standard attention-based RC model with an additional "state" representation that is iteratively updated. The changing state representation results in the model focusing on different parts of the passage during each iteration, allowing it to combine information from different parts of the passage. Graph-based reasoning models (Dhingra et al., 2018; Cao et al., 2018; Song et al., 2018), on the other hand, create graphs over entities within the passages and update entity representations via recurrent or convolutional networks. In contrast, our approach explicitly identifies paths connecting entities in the question to the answer choices.

Semi-structured QA. Our model is closer to Integer Linear Programming (ILP) based methods (Khashabi et al., 2016; Khot et al., 2017; Khashabi et al., 2018b), which define an ILP program to find optimal *support graphs* for connecting the question to the choices through a semi-structured knowledge representation. However, these models require a manually authored and tuned ILP program, and need to convert text into a semi-structured representation—a process that is often noisy (such as using Open IE tu-

[1]The source code is available at https://github.com/allenai/PathNet

[2]Other systems, such as by Zhong et al. (2019), have recently appeared on the WikiHop leaderboard (http://qangaroo.cs.ucl.ac.uk/leaderboard.html).

ples (Khot et al., 2017), SRL frames (Khashabi et al., 2018b)). Our model, on the other hand, is trained end-to-end, and discover relevant relational structure from text. Instead of an ILP program, Jansen et al. (2017) train a latent ranking perceptron using features from aggregated syntactic structures from multiple sentences. However, their system operates at the detailed (and often noisy) level of dependency graphs, whereas we identify entities and let the model learn implicit relations and their compositions.

Knowledge Graph QA. QA datasets on knowledge graphs such as Freebase (Bollacker et al., 2008), require systems to map queries to a single relation (Bordes et al., 2015), a path (Guu et al., 2015), or complex structured queries (Berant et al., 2013) over these graphs. While early models (Lao et al., 2011; Gardner and Mitchell, 2015) focused on creating path-based features, recent neural models (Guu et al., 2015; Das et al., 2017; Toutanova et al., 2016) encode the entities and relations along a path and compose them using recurrent networks. Importantly, the input knowledge graphs have entities and relations that are shared across all training and test examples, which the model can exploit during learning (e.g., via learned entity and relation embeddings). When reasoning with text, our model must learn these representations purely based on their local context.

3 Approach Overview

We focus on the multiple-choice RC setting: given a question and a set of passages, the task is to find the correct answer among a predefined set of candidates. The proposed approach can be applied to m-hop reasoning, as discussed briefly in the corresponding sections for path extraction, encoding, and scoring. Since our target datasets primarily need 2-hop reasoning[3] and the potential of semantic drift with increased number of hops (Fried et al., 2015; Khashabi et al., 2019), we focus on and assess the case of 2-hop paths ($m = 2$). As discussed later (see Footnote 4), our path-extraction step scales exponentially with m. Using $m = 2$ keeps this step tractable, while still covering almost all examples in our target datasets.

In WikiHop, a question $\mathcal{Q}$ is given in the form of a tuple $(h_e, r, ?)$, where h_e represents the head entity and r represents the relation between h_e and the unknown tail entity. The task is to select the unknown tail entity from a given set of candidates $\{c_1, c_2, \ldots c_N\}$, by reasoning over supporting passages $\mathcal{P} = p_1, \ldots, p_M$. To perform multi-hop reasoning, we extract multiple paths $\mathbb{P}$ (cf. Section 4) connecting h_e to each c_k from the supporting passages $\mathcal{P}$. The j-th 2-hop path for candidate c_k is denoted p_{kj}, where $\mathsf{p}_{kj} = h_e \rightarrow e_1 \rightarrow c_k$, and e_1 is referred to as the *intermediate entity*.

In OpenBookQA, different from WikiHop, the questions and candidate answer choices are plain text sentences. To construct paths, we extract all head entities from the question and tail entities from candidate answer choices, considering all noun phrases and named entities as entities. This often results in many 2-hop paths connecting a question to a candidate answer choice via the same intermediate entity. With $\{h_{e_1}, h_{e_2}, \ldots\}$ representing the list of head entities from a question, and $\{c_{k_1}, c_{k_2}, \ldots\}$ the list of tail entities from candidate c_k, the j-th path connecting c_{k_α} to h_{e_β} can be represented as: $\mathsf{p}_{kj}^{\alpha,\beta} = h_{e_\alpha} \rightarrow e_1 \rightarrow c_{k_\beta}$. For simplicity, we omit the notations α and β from path representation.

Next, the extracted paths are encoded and scored (cf. Section 5). Following, the normalized path scores are summed for each candidate to give a probability distribution over the candidate answer choices.

4 Path Extraction

The first step in our approach is extracting paths from text passages. Consider the example in Figure 1. Path extraction proceeds as follows:

(a) We find a passage p_1 that contains a head entity h_e from the question $\mathcal{Q}$. In our example, we would identify the first supporting passage that contains *always breaking my heart*.

(b) We then find all named entities and noun phrases that appear in the same sentence as h_e or in the subsequent sentence. Here, we would collect *Belinda Carlisle*, *A Woman and a Man*, and *album* as potential intermediate entity e_1.

(c) Next, we find a passage p_2 that contains the potential intermediate entity identified above. For clarity, we refer to the occurrence of e_1 in p_2 as $e_1\prime$. By design, (h_e, e_1) and $(e_1\prime, c_k)$ are located in different passages. For instance, we find the second passage that contains both *Belinda Carlisle* and *A Woman and a Man*.

[3] We found that most WikiHop questions can be answered with 2 hops and OpenBookQA also targets 2-hop questions.

(d) Finally, we check whether p_2 contains any of the candidate answer choices. For instance, p_2 contains *chrysalis records* and *emi group*.

The resulting extracted paths can be summarized as a set of entity sequences. In this case, for the candidate answer *chrysalis records*, we obtain a set of two paths: *(always breaking my heart → Belinda Carlisle → chrysalis records), (always breaking my heart → A Man and a Woman → chrysalis records)*. Similarly, we can collect paths for the other candidate, *emi group*.

Notably, our path extraction method can be easily extended for more hops. Specifically, for m-hop reasoning, steps (b) and (c) are repeated $(m-1)$ times, where the intermediate entity from step (c) becomes the head entity for the subsequent step (b). For larger values of m, maintaining tractability of this approach would require optimizing the complexity of identifying the passages containing an entity (steps (a) and (c)) and limiting the number of neighboring entities considered (step (b)).[4]

For one hop reasoning, i.e., when a single passage is sufficient to answer a question, we construct the path with e_1 as *null*. In this case, both h_e and c_k are found in a single passage. In this way, for a task requiring more hops, one only need to guess the maximum number of hops. If some questions in that task require less hops, our proposed approach can easily handle that by assigning the intermediate entity to *null*. For instance, in this work, our approach can handle 1-hop reasoning although it is developed for 2-hop.

5 PathNet: Path-based Multi-hop QA Model

Once we have all potential paths, we score them using the proposed model, named PathNet, whose overview is depicted in Figure 2. The key component is the path-scorer module that computes the score for each path p_{kj}. We normalize these scores across all paths, and compute the probability of a candidate c_k being the correct answer by summing the normalized scores of the paths associated with c_k:

$$\text{prob}(c_k) = \sum_j \text{score}(\mathsf{p}_{kj}). \quad (1)$$

Next, we describe three main model components, operating on the following inputs: question

[4] If the search step takes no more than s steps and identifies a fixed number k of passages, and we select up to e neighboring entities, our approach would have a time complexity of $\mathrm{O}\big((ke)^{m-1}s^m\big)$ for enumerating m-hop paths.

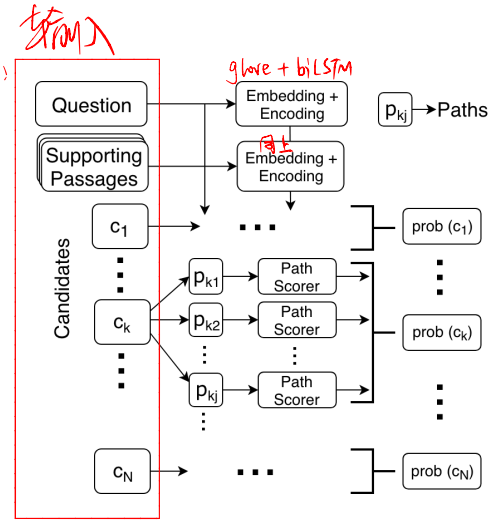

Figure 2: Architecture of the proposed model.

$\mathcal{Q}$, passages p_1 and p_2, candidate c_k, and the locations of h_e, e_1, e_1', c_k in these passages: (1) Embedding and Encoding (§ 5.1) (2) Path Encoding (§ 5.2) (3) Path Scoring (§ 5.3). In Figure 3, we present the model architecture for these three components used for scoring the paths.

5.1 Embedding and Encoding

We start by describing how we embed and contextually encode all pieces of text: question, supporting passages, and candidate answer choices. For word embedding, we use pretrained 300 dimensional vectors from GloVe (Pennington et al., 2014), randomly initializing vectors for out of vocabulary (OOV) words. For contextual encoding, we use bi-directional LSTM (BiLSTM) (Hochreiter and Schmidhuber, 1997).

Let T, U, and V represent the number of tokens in the p-th supporting passage, question, and k-th answer candidate, respectively. The final encoded representation for the p-th supporting passage can be obtained by stacking these vectors into $\mathbf{S}_p \in \mathbb{R}^{T\times H}$, where H is the number of hidden units for the BiLSTMs. The sequence level encoding for the question, $\mathbf{Q} \in \mathbb{R}^{U\times H}$, and for the k-th candidate answer, $\mathbf{C}_k \in \mathbb{R}^{V\times H}$, are obtained similarly. We use row vector representation (e.g., $\mathbb{R}^{1\times H}$) for all vectors in this paper.

5.2 Path Encoding

After extracting the paths as discussed in Section 4, they are encoded using an end-to-end neural network. This path encoder consists of two components: context-based and passage-based.

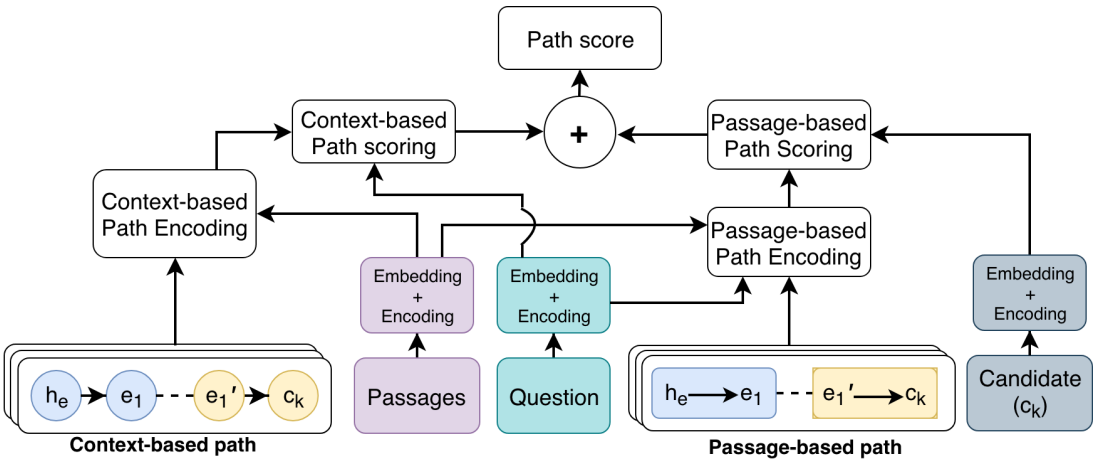

Figure 3: Architecture of the path scoring module, shown here for 2-hop paths.

5.2.1 Context-based Path Encoding

This component aims to implicitly encode the relation between h_e and e_1, and between $e_1\prime$ and c_k. These implicit relation representations are then composed together to encode a path representation for $h_e \rightarrow e_1 \ldots e_1\prime \rightarrow c_k$.

First, we extract the contextual representations for each of h_e, e_1, $e_1\prime$, and c_k. Based on the *locations* of these entities in the corresponding passages, we extract the boundary vectors from the passage encoding representation. For instance, if h_e appears in the p-th supporting passage from token i_1 to i_2 ($i_1 \leq i_2$), then the contextual encoding of h_e, $\mathbf{g}_{h_e} \in \mathbb{R}^{2H}$ is taken to be: $\mathbf{g}_{h_e} = \mathbf{s}_{p_1,i_1} \,||\, \mathbf{s}_{p_1,i_2}$, where $||$ denotes the concatenation operation. If h_e appears in multiple locations within the passage, we use the mean vector representation across all of these locations. The location encoding vectors $\mathbf{g}_{e_1}$, $\mathbf{g}_{e_1\prime}$, and $\mathbf{g}_{c_k}$ are obtained similarly.

Next, we extract the implicit relation between h_e and e_1 as $\mathbf{r}_{h_e,e_1} \in \mathbb{R}^H$, using a feed forward layer:

$$\mathbf{r}_{h_e,e_1} = \text{FFL}(\mathbf{g}_{h_e}, \mathbf{g}_{e_1}) \quad (2)$$

where FFL is defined as:

$$\text{FFL}(\mathbf{a}, \mathbf{b}) = \tanh(\mathbf{a}\mathbf{W}_a + \mathbf{b}\mathbf{W}_b) \; . \quad (3)$$

Here $\mathbf{a} \in \mathbb{R}^{H'}$ and $\mathbf{b} \in \mathbb{R}^{H''}$ are input vectors, and $\mathbf{W}_a \in \mathbb{R}^{H' \times H}$ and $\mathbf{W}_b \in \mathbb{R}^{H'' \times H}$ are trainable weight matrices. The bias vectors are not shown here for simplicity. Similarly, we compute the implicit relation between $e_1\prime$ and c_k as $\mathbf{r}_{e_1\prime,c_k} \in \mathbb{R}^H$, using their location encoding vectors $\mathbf{g}_{e_1\prime}$ and $\mathbf{g}_{c_k}$.

Finally, we *compose* all implicit relation vectors along the path to obtain a context-based path representation $\mathbf{x}_{\text{ctx}} \in \mathbb{R}^H$ given by:

$$\mathbf{x}_{\text{ctx}} = \text{comp}(\mathbf{r}_{h_e,e_1}, \mathbf{r}_{e_1\prime,c_k}) \quad (4)$$

For fixed length paths, we can use a feed forward network as the composition function. E.g., for 2-hop paths, we use $\text{FFL}(\mathbf{r}_{h_e,e_1}, \mathbf{r}_{e_1\prime,c_k})$. For variable length paths, we can use recurrent composition networks such as LSTM, GRU. We compare these composition functions in Section 6.3.

5.2.2 Passage-based Path Encoding

In this encoder, we use *entire passages* to compute the path representation. As before, suppose (h_e, e_1) and $(e_1\prime, c_k)$ appear in supporting passages p_1 and p_2, respectively. We encode each of p_1 and p_2 into a single vector based on passage-question interaction. As discussed below, we first compute a question-weighted representation for passage tokens and then aggregate it across the passage.

Question-Weighted Passage Representation: For the p-th passage, we first compute the attention matrix $\mathbf{A} \in \mathbb{R}^{T \times U}$, capturing the similarity between the passage and question words. Then, we calculate a question-aware passage representation $\mathbf{S}_p^{q_1} \in \mathbb{R}^{T \times H}$, where $\mathbf{S}_p^{q_1} = \mathbf{A}\mathbf{Q}$. Similarly, a passage-aware question representation, $\mathbf{Q}_p \in \mathbb{R}^{U \times H}$, is computed, where $\mathbf{Q}_p = \mathbf{A}^\top \mathbf{S}_p$.

Further, we compute another passage representation $\mathbf{S}_p^{q_2} = \mathbf{A}\mathbf{Q}_p \in \mathbb{R}^{T \times H}$. Intuitively, $\mathbf{S}_p^{q_1}$ captures important passage words based on the question, whereas $\mathbf{S}_p^{q_2}$ is another passage representation which focuses on the interaction with passage-relevant question words. The idea of encoding a passage after interacting with the question multiple times is inspired from the Gated Attention Reader model (Dhingra et al., 2017).

Aggregate Passage Representation: To derive a single passage vector, we first concatenate the two passage representations for each token, obtaining $\mathbf{S}_p^q = \mathbf{S}_p^{q_1} \,||\, \mathbf{S}_p^{q_2} \in \mathbb{R}^{T \times 2H}$. We then use an attentive pooling mechanism for aggregating the token representations. The aggregated vector $\tilde{\mathbf{s}}_p \in \mathbb{R}^{2H}$ for the p-th passage is obtained as:

$$a_t^p \propto \exp(\mathbf{s}_{p,t}^q \mathbf{w}^\top); \;\; \tilde{\mathbf{s}}_p = \mathbf{a}^p \mathbf{S}_p^q \tag{5}$$

where $\mathbf{w} \in \mathbb{R}^{2H}$ is a learned vector. In this way, we obtain the aggregated vector representations for both supporting passages p_1 and p_2 as $\tilde{\mathbf{s}}_{p_1} \in \mathbb{R}^{2H}$ and $\tilde{\mathbf{s}}_{p_2} \in \mathbb{R}^{2H}$, respectively.

Composition: We compose the aggregated passage vectors to obtain the passage-based path representation $\mathbf{x}_{\text{psg}} \in \mathbb{R}^H$ similar to Equation 4:

$$\mathbf{x}_{\text{psg}} = \text{comp}(\tilde{\mathbf{s}}_{p_1}, \tilde{\mathbf{s}}_{p_2}) \tag{6}$$

Similar to the composition function in context-based path encoding, this composition function can be a feed-forward network for fixed length or recurrent networks for variable length paths.

5.3 Path Scoring

Encoded paths are scored from two perspectives.

Context-based Path Scoring: We score context-based paths based on their interaction with the question encoding. First, we aggregate the question into a single vector. We take the first and last hidden state representations from the question encoding $\mathbf{Q}$ to obtain an aggregated question vector representation.

The aggregated question vector $\tilde{\mathbf{q}} \in \mathbb{R}^H$ is

$$\tilde{\mathbf{q}} = (\mathbf{q}_0 \,||\, \mathbf{q}_U)\, \mathbf{W}_q \tag{7}$$

where $\mathbf{W}_q \in \mathbb{R}^{2H \times H}$ is a learnable weight matrix. The combined representation $\mathbf{y}_{x_{\text{ctx}},q} \in \mathbb{R}^H$ of the question and a context-based path is computed as: $\mathbf{y}_{x_{\text{ctx}},q} = \text{FFL}(\mathbf{x}_{\text{ctx}}, \tilde{\mathbf{q}})$ Finally, we derive scores for context-based paths:

$$z_{\text{ctx}} = \mathbf{y}_{x_{\text{ctx}},q} \mathbf{w}_{\text{ctx}}^\top \tag{8}$$

where $\mathbf{w}_{\text{ctx}} \in \mathbb{R}^H$ is a trainable vector.

Passage-based Path Scoring: We also score paths based on the interaction between the passage-based path encoding vector and the candidate encoding. In this case, only candidate encoding is used since passage-based path encoding already uses the question representation. We aggregate the representation $\mathbf{C}_k$ for candidate c_k into a single vector $\tilde{\mathbf{c}}_k \in \mathbb{R}^H$ by applying an attentive pooling operation similar to Equation 5. The score for passage-based path is then computed as follows:

$$z_{\text{psg}} = \tilde{\mathbf{c}}_k \, \mathbf{x}_{\text{psg}}^\top \tag{9}$$

Finally, the unnormalized score for path p_{kj} is:

$$z = z_{\text{ctx}} + z_{\text{psg}} \tag{10}$$

and its normalized version, $score(\mathsf{p}_{kj})$, is calculated by applying the softmax operation over all the paths and candidate answers.

6 Experiments

We start by describing the experimental setup, and then present results and an analysis of our model.

6.1 Setup

We consider the standard (unmasked) version of the recently proposed WikiHop dataset (Welbl et al., 2018). WikiHop is a large scale multi-hop QA dataset consisting of about 51K questions (5129 Dev, 2451 Test). Each question is associated with an average of 13.7 supporting Wikipedia passages, each with 36.4 tokens on average.

We also evaluate our model on OpenBookQA (Mihaylov et al., 2018), a very recent and challenging multi-hop QA dataset with about 6K questions (500 Dev, 500 Test), each with 4 candidate answer choices. Since OpenBookQA does not have associated passages for the questions, we retrieve sentences from a text corpus to create single sentence passages.

We start with a corpus of 1.5M sentences used by previous systems (Khot et al., 2017) for science QA. It is then filtered down to 590K sentences by identifying sentences about generalities and removing noise. We assume sentences that start with a plural noun are likely to capture general concepts, e.g. "Mammals have fur", and only consider such sentences. We also eliminate noisy and irrelevant sentences by using a few rules such as root of the parse tree must be a sentence, it must not contain proper nouns. This corpus is also provided along with our code.

Next, we need to retrieve sentences that can lead to paths between the question q and an answer choice c. Doing so naively will only retrieve sentences that directly connect entities in q to c,

Model	Accuracy (%)	
	Dev	Test
Welbl et al. (2018)	-	42.9
Dhingra et al. (2018)	56.0	59.3
Song et al. (2018)	62.8	65.4
Cao et al. (2018)	64.8	67.6
PathNet	**67.4**†	**69.6**†

Table 1: Accuracy on the WikiHop dataset. †Statistically significant (Wilson, 1927)

Model	Accuracy (%)	
	Dev	Test
KER (OMCS)	54.4	52.2
KER (WordNet)	55.6	51.4
KER (OB + OMCS)	54.6	50.8
KER (OB + WordNet)	54.2	51.2
KER (OB + Text)	55.4	52.0
PathNet (OB + Text)	55.0	**53.4**

Table 2: Accuracy on the OpenBookQA dataset.

i.e., 1-hop paths. To facilitate 2-hop reasoning, we first retrieve sentences based on words in q, and for each retrieved sentence s_1, we find sentences that overlap with both s_1 and c. Each path is scored using $\text{idf}(q, s_1) \cdot \text{idf}(s_1, s_2) \cdot \text{idf}(s_2, c)$, where s_2 is the second retrieved sentence and $\text{idf}(w)$ is the idf score of token w based on the input corpus:

$$\text{idf}(x, y) = \frac{\sum_{w \in x \cap y} \text{idf}(w)}{\min(\sum_{w \in x} \text{idf}(w), \sum_{w \in y} \text{idf}(w))}$$

For efficiency, we perform beam search and ignore any chain if the score drops below a threshold (0.08). Finally we take the top 100 chains and use these sentences as passages in our model.

We use Spacy[5] for tokenization. For word embedding, we use the 840B 300-dimensional pre-trained word vectors from GloVe and we do not update them during training. For simplicity, we do not use any character embedding. The number of hidden units in all LSTMs is 50 ($H = 100$). We use dropout (Srivastava et al., 2014) with probability 0.25 for every learnable layer. During training, the minibatch size is fixed at 8. We use the Adam optimizer (Kingma and Ba, 2015) with learning rate 0.001 and clipnorm 5. We use cross entropy loss for training. This being a multiple-choice QA task, we use accuracy as the evaluation metric.

6.2 Main Results

Table 1 compares our results on the WikiHop dataset with several recently proposed multi-hop QA models. We show the best results from each of the competing entries. Welbl et al. (2018) presented the results of BiDAF (Seo et al., 2017) on the WikiHop dataset. Dhingra et al. (2018) incorporated coreference connections inside GRU network to capture coreference links while obtaining the contextual representation. Recently, Cao et al. (2018) and Song et al. (2018) proposed graph neural network approaches for multi-hop reading comprehension. While the high level idea is similar for these work, Cao et al. (2018) used ELMo (Peters et al., 2018) for a contextual embedding, which has proven to be very useful in the recent past in many NLP tasks.

As seen in Table 1, our proposed model PathNet significantly outperforms prior approaches on WikiHop. Additionally, we benefit from *interpretability*: unlike these prior methods, our model allows identifying specific entity chains that led to the predicted answer.

Table 2 presents results on the OpenBookQA dataset. We compare with the Knowledge Enhanced Reader (KER) model (Mihaylov et al., 2018). The variants reflect the source from which the model retrieves relevant knowledge: the open book (OB), WordNet subset of ConceptNet, and Open Mind Common Sense (OMCS) subset of ConceptNet, and the corpus of 590K sentences (Text). Since KER does not scale to a corpus of this size, we provided it with the combined set of sentences retrieved by our model for all the OpenBookQA questions. The model computes various cross-attentions between the question, knowledge, and answer choices, and combines these attentions to select the answer. Overall, our proposed approach marginally improved over the previous models on the OpenBookQA dataset[6]. Note that, our model was designed for the closed-domain setting where all the required knowledge is provided. Yet, our model is able to generalize on the open-domain setting where the retrieved knowledge may be noisy or insufficient to answer the question.

6.3 Effectiveness of Model Components

Table 3 shows the impact of context-based and passage-based path encodings. Performance of the model degrades when we ablate either of

[5] https://spacy.io/api/tokenizer

[6] Sun et al. (2018) used the large OpenAI fine-tuned language model (Radford et al., 2018) pre-trained on an additional dataset, RACE (Lai et al., 2017) to achieve a score of 55% on this task.

the two path encoding modules. Intuitively, in context-based path encodings, limited and more fine-grained context is considered due to the use of specific entity locations. On the contrary, the passage-based path encoder computes the path representations considering the entire passage representations (both passages which contain the head entity and tail entity respectively). As a result, even if the intermediate entity can not be used meaningfully, the model poses the ability to form an implicit path representation. Passage-based path encoder is more helpful on OpenBookQA as it is often difficult to find meaningful explicit context-based paths through entity linking across passages. Let us consider the following example taken from OpenBookQA development set where our model successfully predicted the correct answer.

Question: What happens when someone on top of a **bicycle** starts pushing it 's peddles in a circular motion ?
Answer: the *bike* accelerates
Best Path: (**bicycle**, pedal, *bike*)
p1: **bicycles** require continuous circular motion on pedals
p2: pushing on the pedals of a *bike* cause that bike to move.

In this case, the extracted path through entity linking is not meaningful as the path composition would connect bicycles to bike [7]. However, when the entire passages are considered, they contain sufficient information to help infer the answer.

Table 4 presents the results on WikiHop development set when different composition functions are used for Equation (4). Recurrent networks, such as LSTM and GRU, enable the path encoder to model an arbitrary number of hops. For 2-hop paths, we found that a simple feed forward network (FFL) performs slightly better than the rest. We also considered sharing the weights (FFL shared) when obtaining the relation vectors $\mathbf{r}_{h_e,e_1}$ and $\mathbf{r}_{e_1\prime,c_k}$. Technically, the FFL model is performing the same task in both cases: extracting implicit relations and the parameters could be shared. However, practically, the unshared weights perform better, possibly because it gives the model the freedom to handle answer candidates differently, especially allowing the model to consider the likelihood of a candidate being a valid answer to *any* question, akin to a prior.

[7] Entities in science questions can be phrases and events (e.g., "the bike accelerates"). Identifying and matching such entities are very challenging in case of the OpenBookQA dataset. We show that our entity-linking approach, designed for noun phrases and named entities, is still able to perform comparable to state-of-the-art methods on science question answering, despite this noisy entity matching.

Model	% Accuracy (Δ)	
	WikiHop	OBQA
PathNet	**67.4**†	**55.0**†
- context-based path	64.7 (2.7)	54.8* (0.2)
- passage-based path	63.2 (4.2)	46.2 (8.8)

Table 3: Ablation results on development sets. *Improvement over this is not statistically significant.

Model	Accuracy (%)	
	WikiHop	Δ
FFL (PathNet)	**67.4**	-
FFL Shared	66.7	0.7
LSTM	67.1	0.3
GRU	67.3	0.1

Table 4: Various composition functions to generate path representation ($\mathbf{x}_{\text{ctx}}$) on WikiHop development set.

6.4 Qualitative Analysis

One key aspect of our model is its ability to indicate the paths that contribute most towards predicting an answer choice. Table 5 illustrates the two highest-scoring paths for two sample WikiHop questions which lead to correct answer prediction. In the first question, the top-2 paths are formed by connecting *Zoo Lake* to *Gauteng* through the intermediate entities *Johannesburg* and *South Africa*, respectively. In the second example, the science fiction novel *This Day All Gods Die* is connected to the publisher *Bantam Books* through the author *Stephen R. Donaldson*, and the collection *Gap Cycle* for first and second paths, respectively.

We also analyzed 50 randomly chosen questions that are annotated as requiring multi-hop reasoning in the WikiHop development set and that our model answered correctly. In 78% of the questions, we found at least one meaningful path[8] in the top-3 extracted paths, which dropped to 62% for top-1 path. On average, 66% of the top-3 paths returned by our model were meaningful. In contrast, only 46% of three randomly selected paths per question made sense, even when limited to the paths for the correct answers. That is, a random baseline, even with oracle knowledge of the correct answer, would only find a good path in 46%

[8] A path is considered meaningful if it has valid relations that can be composed to conclude the predicted answer.

Question: (zoo lake, located_in_the_administrative_territorial_entity, ?)
Answer: gauteng
Rank-1 Path: (zoo lake, Johannesburg, gauteng)
Passage1: ... Zoo Lake is a popular lake and public park in Johannesburg , South Africa . It is part of the Hermann Eckstein Park and is ...
Passage2: ... Johannesburg (also known as Jozi , Joburg and eGoli) is the largest city in South Africa and is one of the 50 largest urban areas in the world . It is the provincial capital of Gauteng , which is ...
Rank-2 Path: (zoo lake, South Africa, gauteng)
Passage1: ... Zoo Lake is a popular lake and public park in Johannesburg , South Africa . It is ...
Passage2: ... aka The Reef , is a 56-kilometre - long north - facing scarp in the Gauteng Province of South Africa . It consists of a ...

Question: (this day all gods die, publisher, ?)
Answer: bantam books
Rank-1 Path: (this day all gods die, Stephen R. Donaldson, bantam books)
Passage1: ... All Gods Die , officially The Gap into Ruin : This Day All Gods Die , is a science fiction novel by Stephen R. Donaldson , being the final book of The Gap Cycle ...
Passage2: ... The Gap Cycle (published 19911996 by Bantam Books and reprinted by Gollancz in 2008) is a science fiction story , told in a series of 5 books , written by Stephen R. Donaldson . It is an ...
Rank-2 Path: (this day all gods die, Gap Cycle, bantam books)
Passage1: ... All Gods Die , officially The Gap into Ruin : This Day All Gods Die , is a science fiction novel by Stephen R. Donaldson , being the final book of The Gap Cycle ...
Passage2: ... The Gap Cycle (published 19911996 by Bantam Books and reprinted by Gollancz in 2008) is a science fiction story ...

Table 5: Two top-scoring paths for sample WikiHop Dev questions. In the Rank-1 path for the first question, the model composes the implicit *located in* relations between (Zoo lake, Johannesburg) and (Johannesburg, Gauteng).

of the cases. We also analyzed 50 questions that our model gets wrong. The top-scoring paths here were of lower quality (only 16.7% were meaningful). This provides qualitative evidence that our model's performance is correlated with the quality of the paths it identifies, and it does not simply *guess* using auxiliary information such as entity types, number of paths,[9] etc.

7 Conclusion

We present a novel, path-based, multi-hop reading comprehension model that outperforms previous models on WikiHop and OpenBookQA. Importantly, we illustrate how our model can explain its reasoning via explicit paths extracted across multiple passages. While we focused on 2-hop reasoning required by our evaluation datasets, the approach can be generalized to longer chains and to longer natural language questions.

Acknowledgment

We thank Johannes Welbl for helping us evaluating our model on the WikiHop test set. We thank Rodney Kinney, Brandon Stilson and Tal Friedman for helping to produce the clean corpus used for OpenBookQA. We thank Dirk Groeneveld for retrieving the sentences from this corpus using the 2-hop retrieval. Computations on beaker.org were supported in part by credits from Google Cloud.

[9] A model that returns the answer with the highest number of paths would score only 18.5% on the WikiHop development set.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of EMNLP*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of ACM SIGMOD international conference on Management of data*.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. In *NIPS*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2018. Question answering by reasoning across documents with graph convolutional networks. *CoRR*, abs/1808.09920.

Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2017. Chains of reasoning over entities, relations, and text using recurrent neural networks. In *Proceedings of EACL*.

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. 2018. Neural models for reasoning over multiple mentions using coreference. In *Proceedings of NAACL*.

Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. 2017. Gated-attention readers for text comprehension. In *Proceedings of ACL*.

Daniel Fried, Peter Jansen, Gustave Hahn-Powell, Mihai Surdeanu, and Peter Clark. 2015. Higher-order lexical semantic models for non-factoid answer reranking. *TACL*, 3:197–210.

Matt Gardner and Tom M. Mitchell. 2015. Efficient and expressive knowledge base completion using subgraph feature extraction. In *Proceedings of EMNLP*.

Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. In *Proceedings of EMNLP*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Minghao Hu, Yuxing Peng, Zhen Huang, Xipeng Qiu, Furu Wei, and Ming Zhou. 2018. Reinforced mnemonic reader for machine reading comprehension. In *Proceedings of IJCAI*.

Peter Jansen, Rebecca Sharp, Mihai Surdeanu, and Peter Clark. 2017. Framing qa as building and ranking intersentence answer justifications. *Computational Linguistics*, 43:407–449.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of ACL*.

Daniel Khashabi, Erfan Sadeqi Azer, Tushar Khot, Ashutosh Sabharwal, and Dan Roth. 2019. On the capabilities and limitations of reasoning for natural language understanding. *CoRR*, abs/1901.02522.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018a. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of NAACL*.

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, Peter Clark, Oren Etzioni, and Dan Roth. 2016. Question answering via integer programming over semi-structured knowledge. In *Proceedings of IJCAI*.

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2018b. Question answering as global reasoning over semantic abstractions. In *Proceedings of AAAI*.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2017. Answering complex questions using open information extraction. In *Proceedings of ACL*.

Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard H. Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. In *EMNLP*.

Ni Lao, Tom M. Mitchell, and William W. Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of EMNLP*.

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 *Proceedings of EMNLP*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Technical report, Technical report, OpenAI.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of EMNLP*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *Proceedings of ICLR*.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. 2017. Reasonet: Learning to stop reading in machine comprehension. In *Proceedings of KDD*.

Linfeng Song, Zhiguo Wang, Mo Yu, Yue Zhang, Radu Florian, and Daniel Gildea. 2018. Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. *CoRR*, abs/1809.02040.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. 2018. Improving machine reading comprehension with general reading strategies. *CoRR*, abs/1810.13441.

Kristina Toutanova, Victoria Lin, Wen tau Yih, Hoifung Poon, and Chris Quirk. 2016. Compositional learning of embeddings for relation paths in knowledge base and text. In *Proceedings of ACL*.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *TACL*, 6:287–302.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. Towards AI-complete question answering: A set of prerequisite toy tasks. *CoRR*, abs/1502.05698.

Edwin B. Wilson. 1927. Probable inference, the law of succession, and statistical inference. *JASA*, 22(158):209–212.

Victor Zhong, Caiming Xiong, Nitish Shirish Keskar, and Richard Socher. 2019. Coarse-grain fine-grain coattention network for multi-evidence question answering. In *Proceedings of ICLR*.