Question Answering by Reasoning Across Documents with Graph Convolutional Networks

Nicola De Cao
University of Edinburgh
University of Amsterdam
nicola.decao@uva.nl

Wilker Aziz
University of Amsterdam
w.aziz@uva.nl

Ivan Titov
University of Edinburgh
University of Amsterdam
ititov@inf.ed.ac.uk

Abstract

Most research in reading comprehension has focused on answering questions based on individual documents or even single paragraphs. We introduce a neural model which integrates and reasons relying on information spread within documents and across multiple documents. We frame it as an inference problem on a graph. Mentions of entities are nodes of this graph while edges encode relations between different mentions (e.g., within- and cross-document coreference). Graph convolutional networks (GCNs) are applied to these graphs and trained to perform multi-step reasoning. Our Entity-GCN method is scalable and compact, and it achieves state-of-the-art results on a multi-document question answering dataset, WIKIHOP (Welbl et al., 2018).

1 Introduction

The long-standing goal of natural language understanding is the development of systems which can acquire knowledge from text collections. Fresh interest in reading comprehension tasks was sparked by the availability of large-scale datasets, such as SQuAD (Rajpurkar et al., 2016) and CNN/Daily Mail (Hermann et al., 2015), enabling end-to-end training of neural models (Seo et al., 2016; Xiong et al., 2016; Shen et al., 2017). These systems, given a text and a question, need to answer the query relying on the given document. Recently, it has been observed that most questions in these datasets do not require reasoning across the document, but they can be answered relying on information contained in a single sentence (Weissenborn et al., 2017). The last generation of large-scale reading comprehension datasets, such as a NarrativeQA (Kocisky et al., 2018), TriviaQA (Joshi et al., 2017), and RACE (Lai et al., 2017), have been created in such a way as to address this shortcoming and to ensure that systems

Thorildsplan is a small park in Kristineberg in **Stockholm**, named in 1925 after the writer [..]

Stockholm is the capital of **Sweden** and the most populous city in [..]

query: country **Thorildsplan**
candidates: {Denmark, Finland, **Sweden**, Italy, ...}
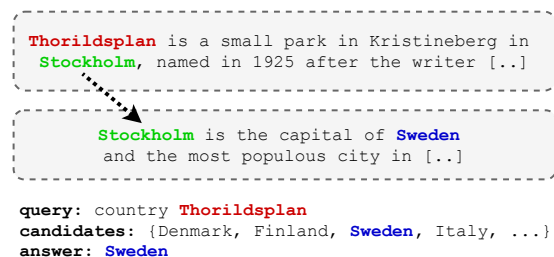**answer**: **Sweden**

Figure 1: A sample from WIKIHOP where multi-step reasoning and information combination from different documents is necessary to infer the correct answer.

relying only on local information cannot achieve competitive performance.

Even though these new datasets are challenging and require reasoning within documents, many question answering and search applications require aggregation of information across multiple documents. The WIKIHOP dataset (Welbl et al., 2018) was explicitly created to facilitate the development of systems dealing with these scenarios. Each example in WIKIHOP consists of a collection of documents, a query and a set of candidate answers (Figure 1). Though there is no guarantee that a question cannot be answered by relying just on a single sentence, the authors ensure that it is answerable using a chain of reasoning crossing document boundaries.

Though an important practical problem, the multi-hop setting has so far received little attention. The methods reported by Welbl et al. (2018) approach the task by merely concatenating all documents into a single long text and training a standard RNN-based reading comprehension model, namely, BiDAF (Seo et al., 2016) and FastQA (Weissenborn et al., 2017). Document concatenation in this setting is also used in Weaver (Raison et al., 2018) and MHPGM (Bauer et al., 2018). The only published paper which

2306

goes beyond concatenation is due to Dhingra et al. (2018), where they augment RNNs with jump-links corresponding to co-reference edges. Though these edges provide a structural bias, the RNN states are still tasked with passing the information across the document and performing multi-hop reasoning.

Instead, we frame question answering as an inference problem on a graph representing the document collection. Nodes in this graph correspond to named entities in a document whereas edges encode relations between them (e.g., cross- and within-document coreference links or simply co-occurrence in a document). We assume that reasoning chains can be captured by propagating local contextual information along edges in this graph using a graph convolutional network (GCN) (Kipf and Welling, 2017).

The multi-document setting imposes scalability challenges. In realistic scenarios, a system needs to learn to answer a query for a given collection (e.g., Wikipedia or a domain-specific set of documents). In such scenarios one cannot afford to run expensive document encoders (e.g., RNN or transformer-like self-attention (Vaswani et al., 2017)), unless the computation can be pre-processed both at train and test time. Even if (similarly to WIKIHOP creators) one considers a coarse-to-fine approach, where a set of potentially relevant documents is provided, re-encoding them in a query-specific way remains the bottleneck. In contrast to other proposed methods (e.g., (Dhingra et al., 2018; Raison et al., 2018; Seo et al., 2016)), we avoid training expensive document encoders.

In our approach, only a small query encoder, the GCN layers and a simple feed-forward answer selection component are learned. Instead of training RNN encoders, we use contextualized embeddings (ELMo) to obtain initial (local) representations of nodes. This implies that only a lightweight computation has to be performed online, both at train and test time, whereas the rest is preprocessed. Even in the somewhat contrived WIKIHOP setting, where fairly small sets of candidates are provided, the model is at least 5 times faster to train than BiDAF.[1] Interestingly, when we substitute ELMo with simple pre-trained word embeddings, Entity-GCN still performs on par

with many techniques that use expensive question-aware recurrent document encoders.

Despite not using recurrent document encoders, the full Entity-GCN model achieves over 2% improvement over the best previously-published results. As our model is efficient, we also reported results of an ensemble which brings further 3.6% of improvement and only 3% below the human performance reported by Welbl et al. (2018). Our contributions can be summarized as follows:

- we present a novel approach for multi-hop QA that relies on a (pre-trained) document encoder and information propagation across multiple documents using graph neural networks;

- we provide an efficient training technique which relies on a slower offline and a faster on-line computation that does not require expensive document processing;

- we empirically show that our algorithm is effective, presenting an improvement over previous results.

2 Method

In this section we explain our method. We first introduce the dataset we focus on, WIKIHOP by Welbl et al. (2018), as well as the task abstraction. We then present the building blocks that make up our Entity-GCN model, namely, an *entity graph* used to relate mentions to entities within and across documents, a *document encoder* used to obtain representations of mentions in context, and a *relational graph convolutional network* that propagates information through the entity graph.

2.1 Dataset and Task Abstraction

Data The WIKIHOP dataset comprises of tuples $\langle q, S_q, C_q, a^\star \rangle$ where: q is a query/question, S_q is a set of supporting documents, C_q is a set of candidate answers (all of which are entities mentioned in S_q), and $a^\star \in C_q$ is the entity that correctly answers the question. WIKIHOP is assembled assuming that there exists a corpus and a knowledge base (KB) related to each other. The KB contains triples $\langle s, r, o \rangle$ where s is a subject entity, o an object entity, and r a unidirectional relation between them. Welbl et al. (2018) used WIKIPEDIA as corpus and WIKIDATA (Vrandečić, 2012) as KB. The KB is only used for constructing WIKIHOP: Welbl

---

[1]When compared to the 'small' and hence fast BiDAF model reported in Welbl et al. (2018), which is 25% less accurate than our Entity-GCN. Larger RNN models are problematic also because of GPU memory constraints.

et al. (2018) retrieved the supporting documents S_q from the corpus looking at mentions of subject and object entities in the text. Note that the set S_q (not the KB) is provided to the QA system, and not all of the supporting documents are relevant for the query but some of them act as distractors. Queries, on the other hand, are not expressed in natural language, but instead consist of tuples $\langle s, r, ? \rangle$ where the object entity is unknown and it has to be inferred by reading the support documents. Therefore, answering a query corresponds to finding the entity $a^\star$ that is the object of a tuple in the KB with subject s and relation r among the provided set of candidate answers C_q.

Task The goal is to learn a model that can identify the correct answer $a^\star$ from the set of supporting documents S_q. To that end, we exploit the available supervision to train a neural network that computes scores for candidates in C_q. We estimate the parameters of the architecture by maximizing the likelihood of observations. For prediction, we then output the candidate that achieves the highest probability. In the following, we present our model discussing the design decisions that enable multi-step reasoning and an efficient computation.

2.2 Reasoning on an Entity Graph

Entity graph In an offline step, we organize the content of each training instance in a graph connecting mentions of candidate answers within and across supporting documents. For a given query $q = \langle s, r, ? \rangle$, we identify mentions in S_q of the entities in $C_q \cup \{s\}$ and create one node per mention. This process is based on the following heuristic:

1. we consider mentions spans in S_q exactly matching an element of $C_q \cup \{s\}$. Admittedly, this is a rather simple strategy which may suffer from low recall.

2. we use predictions from a coreference resolution system to add mentions of elements in $C_q \cup \{s\}$ beyond exact matching (including both noun phrases and anaphoric pronouns). In particular, we use the end-to-end coreference resolution by Lee et al. (2017).

3. we discard mentions which are ambiguously resolved to multiple coreference chains; this may sacrifice recall, but avoids propagating ambiguity.
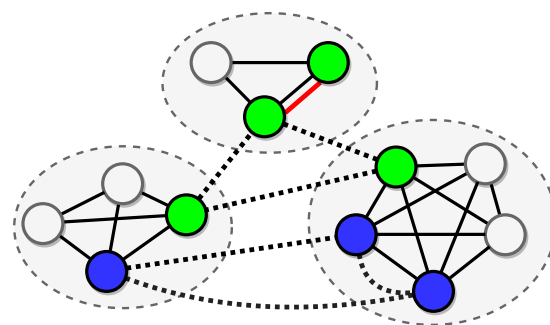

Figure 2: Supporting documents (dashed ellipses) organized as a graph where nodes are mentions of either candidate entities or query entities. Nodes with the same color indicates they refer to the same entity (exact match, coreference or both). Nodes are connected by three simple relations: one indicating co-occurrence in the same document (solid edges), another connecting mentions that exactly match (dashed edges), and a third one indicating a coreference (bold-red line).

To each node v_i, we associate a continuous annotation $\mathbf{x}_i \in \mathbb{R}^D$ which represents an entity in the context where it was mentioned (details in Section 2.3). We then proceed to connect these mentions i) if they co-occur within the same document (we will refer to this as DOC-BASED edges), ii) if the pair of named entity mentions is identical (MATCH edges—these may connect nodes across and within documents), or iii) if they are in the same coreference chain, as predicted by the external coreference system (COREF edges). Note that MATCH edges when connecting mentions in the same document are mostly included in the set of edges predicted by the coreference system. Having the two types of edges lets us distinguish between less reliable edges provided by the coreference system and more reliable (but also more sparse) edges given by the exact-match heuristic. We treat these three types of connections as three different types of relations. See Figure 2 for an illustration. In addition to that, and to prevent having disconnected graphs, we add a fourth type of relation (COMPLEMENT edge) between any two nodes that are not connected with any of the other relations. We can think of these edges as those in the complement set of the entity graph with respect to a fully connected graph.

Multi-step reasoning Our model then approaches multi-step reasoning by transforming node representations (Section 2.3 for details) with a differentiable message passing algorithm that propagates information through the entity

2308

graph. The algorithm is parameterized by a graph convolutional network (GCN) (Kipf and Welling, 2017), in particular, we employ relational-GCNs (Schlichtkrull et al., 2018), an extended version that accommodates edges of different types. In Section 2.4 we describe the propagation rule.

Each step of the algorithm (also referred to as a *hop*) updates all node representations in parallel. In particular, a node is updated as a function of messages from its direct neighbours, and a message is possibly specific to a certain relation. At the end of the first step, every node is aware of every other node it connects directly to. Besides, the neighbourhood of a node may include mentions of the same entity as well as others (e.g., same-document relation), and these mentions may have occurred in different documents. Taking this idea recursively, each further step of the algorithm allows a node to indirectly interact with nodes already known to their neighbours. After L layers of R-GCN, information has been propagated through paths connecting up to $L + 1$ nodes.

We start with node representations $\{\mathbf{h}_i^{(0)}\}_{i=1}^N$, and transform them by applying L layers of R-GCN obtaining $\{\mathbf{h}_i^{(L)}\}_{i=1}^N$. Together with a representation $\mathbf{q}$ of the query, we define a distribution over candidate answers and we train maximizing the likelihood of observations. The probability of selecting a candidate $c \in C_q$ as an answer is then

$$P(c|q, C_q, S_q) \propto \exp\left(\max_{i \in \mathcal{M}_c} f_o([\mathbf{q}, \mathbf{h}_i^{(L)}])\right) , \quad (1)$$

where f_o is a parameterized affine transformation, and $\mathcal{M}_c$ is the set of node indices such that $i \in \mathcal{M}_c$ only if node v_i is a mention of c. The max operator in Equation 1 is necessary to select the node with highest predicted probability since a candidate answer is realized in multiple locations via different nodes.

2.3 Node Annotations

Keeping in mind we want an efficient model, we encode words in supporting documents and in the query using only a pre-trained model for contextualized word representations rather than training our own encoder. Specifically, we use ELMo[2] (Peters et al., 2018), a pre-trained bi-directional lan-

guage model that relies on character-based input representation. ELMo representations, differently from other pre-trained word-based models (e.g., *word2vec* (Mikolov et al., 2013) or GloVe (Pennington et al., 2014)), are contextualized since each token representation depends on the entire text excerpt (i.e., the whole sentence).

We choose not to fine tune nor propagate gradients through the ELMo architecture, as it would have defied the goal of not having specialized RNN encoders. In the experiments, we will also ablate the use of ELMo showing how our model behaves using non-contextualized word representations (we use GloVe).

Documents pre-processing ELMo encodings are used to produce a set of representations $\{\mathbf{x}_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^D$ denotes the ith candidate mention in context. Note that these representations do not depend on the query yet and no trainable model was used to process the documents so far, that is, we use ELMo as a fixed pre-trained encoder. Therefore, we can pre-compute representation of mentions once and store them for later use.

Query-dependent mention encodings ELMo encodings are used to produce a query representation $\mathbf{q} \in \mathbb{R}^K$ as well. Here, $\mathbf{q}$ is a concatenation of the final outputs from a bidirectional RNN layer trained to re-encode ELMo representations of words in the query. The vector $\mathbf{q}$ is used to compute a query-dependent representation of mentions $\{\hat{\mathbf{x}}_i\}_{i=1}^N$ as well as to compute a probability distribution over candidates (as in Equation 1). Query-dependent mention encodings $\hat{\mathbf{x}}_i = f_x(\mathbf{q}, \mathbf{x}_i)$ are generated by a trainable function f_x which is parameterized by a feed-forward neural network.

2.4 Entity Relational Graph Convolutional Network

Our model uses a gated version of the original R-GCN propagation rule. At the first layer, all hidden node representation are initialized with the query-aware encodings $\mathbf{h}_i^{(0)} = \hat{\mathbf{x}}_i$. Then, at each layer $0 \le \ell \le L$, the update message $\mathbf{u}_i^{(\ell)}$ to the ith node is a sum of a transformation f_s of the current node representation $\mathbf{h}_i^{(\ell)}$ and transformations of its neighbours:

$$\mathbf{u}_i^{(\ell)} = f_s(\mathbf{h}_i^{(\ell)}) + \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \sum_{r \in \mathcal{R}_{ij}} f_r(\mathbf{h}_j^{(\ell)}) , \quad (2)$$

---

[2]The use of ELMo is an implementation choice, and, in principle, any other contextual pre-trained model could be used (Radford et al., 2018; Devlin et al., 2018).

where $\mathcal{N}_i$ is the set of indices of nodes neighbouring the ith node, $\mathcal{R}_{ij}$ is the set of edge annotations between i and j, and f_r is a parametrized function specific to an edge type $r \in \mathcal{R}$. Recall the available relations from Section 2.2, namely, $\mathcal{R} = \{\texttt{DOC-BASED}, \texttt{MATCH}, \texttt{COREF}, \texttt{COMPLEMENT}\}$.

A gating mechanism regulates how much of the update message propagates to the next step. This provides the model a way to prevent completely overwriting past information. Indeed, if all necessary information to answer a question is present at a layer which is not the last, then the model should learn to stop using neighbouring information for the next steps. Gate levels are computed as

$$\mathbf{a}_i^{(\ell)} = \sigma \left(f_a \left([\mathbf{u}_i^{(\ell)}, \mathbf{h}_i^{(\ell)}] \right) \right) , \qquad (3)$$

where $\sigma(\cdot)$ is the sigmoid function and f_a a parametrized transformation. Ultimately, the updated representation is a gated combination of the previous representation and a non-linear transformation of the update message:

$$\mathbf{h}_i^{(\ell+1)} = \phi(\mathbf{u}_i^{(\ell)}) \odot \mathbf{a}_i^{(\ell)} + \mathbf{h}_i^{(\ell)} \odot (1 - \mathbf{a}_i^{(\ell)}) , \quad (4)$$

where $\phi(\cdot)$ is any nonlinear function (we used $\tanh$) and $\odot$ stands for element-wise multiplication. All transformations f_* are affine and they are not layer-dependent (since we would like to use as few parameters as possible to decrease model complexity promoting efficiency and scalability).

3 Experiments

In this section, we compare our method against recent work as well as preforming an ablation study using the WIKIHOP dataset (Welbl et al., 2018). See Appendix A in the supplementary material for a description of the hyper-parameters of our model and training details.

WIKIHOP We use WIKIHOP for training, validation/development and test. The test set is not publicly available and therefore we measure performance on the validation set in almost all experiments. WIKIHOP has 43,738/ 5,129/ 2,451 query-documents samples in the training, validation and test sets respectively for a total of 51,318 samples. Authors constructed the dataset as described in Section 2.1 selecting samples with a graph traversal up to a maximum chain length of 3 documents (see Table 1 for additional dataset statistics). WIKIHOP comes in two versions, a

	Min	Max	Avg.	Median
# candidates	2	79	19.8	14
# documents	3	63	13.7	11
# tokens/doc.	4	2,046	100.4	91

Table 1: WIKIHOP dataset statistics from Welbl et al. (2018): number of candidates and documents per sample and document length.

standard (unmasked) one and a masked one. The masked version was created by the authors to test whether methods are able to learn lexical abstraction. In this version, all candidates and all mentions of them in the support documents are replaced by random but consistent placeholder tokens. Thus, in the masked version, mentions are always referred to via unambiguous surface forms. We do not use coreference systems in the masked version as they rely crucially on lexical realization of mentions and cannot operate on masked tokens.

3.1 Comparison

In this experiment, we compare our Enitity-GCN against recent prior work on the same task. We present test and development results (when present) for both versions of the dataset in Table 2. From Welbl et al. (2018), we list an oracle based on human performance as well as two standard reading comprehension models, namely BiDAF (Seo et al., 2016) and FastQA (Weissenborn et al., 2017). We also compare against Coref-GRU (Dhingra et al., 2018), MHPGM (Bauer et al., 2018), and Weaver (Raison et al., 2018). Additionally, we include results of MHQA-GRN (Song et al., 2018), from a recent arXiv preprint describing concurrent work. They jointly train graph neural networks and recurrent encoders. We report single runs of our two best single models and an ensemble one on the unmasked test set (recall that the test set is not publicly available and the task organizers only report unmasked results) as well as both versions of the validation set.

Entity-GCN (best single model without coreference edges) outperforms all previous work by over 2% points. We additionally re-ran BiDAF baseline to compare training time: when using a single Titan X GPU, BiDAF and Entity-GCN process 12.5 and 57.8 document sets per second, respectively. Note that Welbl et al. (2018) had to use BiDAF with very small state dimensionalities

Model	Unmasked		Masked	
	Test	Dev	Test	Dev
Human (Welbl et al., 2018)	74.1	–	–	–
FastQA (Welbl et al., 2018)	25.7	–	35.8	–
BiDAF (Welbl et al., 2018)	42.9	–	54.5	–
Coref-GRU (Dhingra et al., 2018)	59.3	56.0	–	–
MHPGM (Bauer et al., 2018)	–	58.2	–	–
Weaver / Jenga (Raison et al., 2018)	65.3	64.1	–	–
MHQA-GRN (Song et al., 2018)	65.4	62.8	–	–
Entity-GCN without coreference (single model)	**67.6**	**64.8**	–	70.5
Entity-GCN with coreference (single model)	**66.4**	**65.3**	–	–
Entity-GCN* (ensemble 5 models)	**71.2**	**68.5**	–	71.6

Table 2: Accuracy of different models on WIKIHOP closed test set and public validation set. Our Entity-GCN outperforms recent prior work without learning any language model to process the input but relying on a pre-trained one (ELMo – without fine-tunning it) and applying R-GCN to reason among entities in the text. * with coreference for unmasked dataset and without coreference for the masked one.

(20), and smaller batch size due to the scalability issues (both memory and computation costs). We compare applying the same reductions.[3] Eventually, we also report an ensemble of 5 independently trained models. All models are trained on the same dataset splits with different weight initializations. The ensemble prediction is obtained as $\arg\max_c \prod_{i=1}^{5} P_i(c|q, C_q, S_q)$ from each model.

3.2 Ablation Study

To help determine the sources of improvements, we perform an ablation study using the publicly available validation set (see Table 3). We perform two groups of ablation, one on the embedding layer, to study the effect of ELMo, and one on the edges, to study how different relations affect the overall model performance.

Embedding ablation We argue that ELMo is crucial, since we do not rely on any other context encoder. However, it is interesting to explore how our R-GCN performs without it. Therefore, in this experiment, we replace the deep contextualized embeddings of both the query and the nodes with GloVe (Pennington et al., 2014) vectors (insensitive to context). Since we do not have any component in our model that processes the documents, we expect a drop in performance. In other words, in this ablation our model tries to answer questions

without reading the context at all. For example, in Figure 1, our model would be aware that "Stockholm" and "Sweden" appear in the same document but any context words, including the ones encoding relations (e.g., "is the capital of") will be hidden. Besides, in the masked case all mentions become 'unknown' tokens with GloVe and therefore the predictions are equivalent to a random guess. Once the strong pre-trained encoder is out of the way, we also ablate the use of our R-GCN component, thus completely depriving the model from inductive biases that aim at multi-hop reasoning.

The first important observation is that replacing ELMo by GloVe (GloVe with R-GCN in Table 3) still yields a competitive system that ranks far above baselines from (Welbl et al., 2018) and even above the Coref-GRU of Dhingra et al. (2018), in terms of accuracy on (unmasked) validation set. The second important observation is that if we then remove R-GCN (GloVe w/o R-GCN in Table 3), we lose 8.0 points. That is, the R-GCN component pushes the model to perform above Coref-GRU still without accessing context, but rather by updating mention representations based on their relation to other ones. These results highlight the impact of our R-GCN component.

Graph edges ablation In this experiment we investigate the effect of the different relations available in the entity graph and processed by the R-GCN module. We start off by testing our stronger encoder (i.e., ELMo) in absence of edges connecting mentions in the supporting documents (i.e., us-

---

[3]Besides, we could not run any other method we compare with combined with ELMo without reducing the dimensionality further or having to implement a distributed version.

Model	unmasked	masked
full (ensemble)	**68.5**	**71.6**
full (single)	**65.1** ± 0.11	**70.4** ± 0.12
GloVe with R-GCN	59.2	11.1
GloVe w/o R-GCN	51.2	11.6
No R-GCN	62.4	63.2
No relation types	62.7	63.9
No DOC-BASED	62.9	65.8
No MATCH	64.3	67.4
No COREF	64.8	–
No COMPLEMENT	64.1	70.3
Induced edges	61.5	56.4

Table 3: Ablation study on WIKIHOP validation set. The *full model* is our Entity-GCN with all of its components and other rows indicate models trained without a component of interest. We also report baselines using GloVe instead of ELMo with and without R-GCN. For the *full model* we report mean ±1 std over 5 runs.

ing only self-loops – No R-GCN in Table 3). The results suggest that WIKIPHOP genuinely requires multihop inference, as our best model is 6.1% and 8.4% more accurate than this local model, in unmasked and masked settings, respectively.[4] However, it also shows that ELMo representations capture predictive context features, without being explicitly trained for the task. It confirms that our goal of getting away with training expensive document encoders is a realistic one.

We then inspect our model's effectiveness in making use of the structure encoded in the graph. We start naively by fully-connecting all nodes within and across documents without distinguishing edges by type (No relation types in Table 3). We observe only marginal improvements with respect to ELMo alone (No R-GCN in Table 3) in both the unmasked and masked setting suggesting that a GCN operating over a naive entity graph would not add much to this task and a more informative graph construction and/or a more sophisticated parameterization is indeed needed.

Next, we ablate each type of relations independently, that is, we either remove connections of mentions that co-occur in the same document (DOC-BASED), connections between mentions matching exactly (MATCH), or edges predicted by the coreference system (COREF). The

first thing to note is that the model makes better use of DOC-BASED connections than MATCH or COREF connections. This is mostly because i) the majority of the connections are indeed between mentions in the same document, and ii) without connecting mentions within the same document we remove important information since the model is unaware they appear closely in the document. Secondly, we notice that coreference links and complement edges seem to play a more marginal role. Though it may be surprising for coreference edges, recall that the MATCH heuristic already captures the easiest coreference cases, and for the rest the out-of-domain coreference system may not be reliable. Still, modelling all these different relations together gives our Entity-GCN a clear advantage. This is our best system evaluating on the development. Since Entity-GCN seems to gain little advantage using the coreference system, we report test results both with and without using it. Surprisingly, with coreference, we observe performance degradation on the test set. It is likely that the test documents are harder for the coreference system.[5]

We do perform one last ablation, namely, we replace our heuristic for assigning edges and their labels by a model component that predicts them. The last row of Table 3 (Induced edges) shows model performance when edges are not predetermined but predicted. For this experiment, we use a bilinear function $f_e(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j) = \sigma\left(\hat{\mathbf{x}}_i^\top \mathbf{W}_e \hat{\mathbf{x}}_j\right)$ that predicts the importance of a single edge connecting two nodes i, j using the query-dependent representation of mentions (see Section 2.3). The performance drops below 'No R-GCN' suggesting that it cannot learn these dependencies on its own.

Most results are stronger for the masked settings even though we do not apply the coreference resolution system in this setting due to masking. It is not surprising as coreferred mentions are labeled with the same identifier in the masked version, even if their original surface forms did not match (Welbl et al. (2018) used WIKIPEDIA links for masking). Indeed, in the masked version, an entity is always referred to via the same unique surface form (e.g., MASK1) within and across documents. In the unmasked setting, on the other hand, mentions to an entity may differ (e.g., "US" vs "United States") and they might not be retrieved by the coreference system we are employing, mak-

---

[4]Recall that all models in the ensemble use the same local representations, ELMo.

[5]Since the test set is hidden from us, we cannot analyze this difference further.

	Relation	Accuracy	P@2	P@5	Avg. $|C_q|$	Supports
	overall (ensemble)	68.5	81.0	94.1	20.4 ± 16.6	5129
	overall (single model)	65.3	79.7	92.9	20.4 ± 16.6	5129
3 best	member_of_political_party	85.5	95.7	98.6	5.4 ± 2.4	70
	record_label	83.0	93.6	99.3	12.4 ± 6.1	283
	publisher	81.5	96.3	100.0	9.6 ± 5.1	54
3 worst	place_of_birth	51.0	67.2	86.8	27.2 ± 14.5	309
	place_of_death	50.0	67.3	89.1	25.1 ± 14.3	159
	inception	29.9	53.2	83.1	21.9 ± 11.0	77

Table 4: Accuracy and precision at K (P@K in the table) analysis overall and per query type. Avg. $|C_q|$ indicates the average number of candidates with one standard deviation.

ing the task harder for all models. Therefore, as we rely mostly on exact matching when constructing our graph for the masked case, we are more effective in recovering coreference links on the masked rather than unmasked version.[6]

4 Error Analysis

In this section we provide an error analysis for our best single model predictions. First of all, we look at which type of questions our model performs well or poorly. There are more than 150 query types in the validation set but we filtered the three with the best and with the worst accuracy that have at least 50 supporting documents and at least 5 candidates. We show results in Table 4. We observe that questions regarding places (birth and death) are considered harder for Entity-GCN. We then inspect samples where our model fails while assigning highest likelihood and noticed two principal sources of failure i) a mismatch between what is written in WIKIPEDIA and what is annotated in WIKIDATA, and ii) a different degree of granularity (e.g., born in "London" vs "UK" could be considered both correct by a human but not when measuring accuracy). See Table 6 in the supplement material for some reported samples.

Secondly, we study how the model performance degrades when the input graph is large. In particular, we observe a negative Pearson's correlation (-0.687) between accuracy and the number of candidate answers. However, the performance does not decrease steeply. The distribution of the number of candidates in the dataset peaks at 5 and has an average of approximately 20. Therefore, the model

does not see many samples where there are a large number of candidate entities during training. Differently, we notice that as the number of nodes in the graph increases, the model performance drops but more gently (negative but closer to zero Pearson's correlation). This is important as document sets can be large in practical applications. See Figure 3 in the supplemental material for plots.

5 Related Work

In previous work, BiDAF (Seo et al., 2016), FastQA (Weissenborn et al., 2017), Coref-GRU (Dhingra et al., 2018), MHPGM (Bauer et al., 2018), and Weaver / Jenga (Raison et al., 2018) have been applied to multi-document question answering. The first two mainly focus on single document QA and Welbl et al. (2018) adapted both of them to work with WIKIHOP. They process each instance of the dataset by concatenating all $d \in S_q$ in a random order adding document separator tokens. They trained using the first answer mention in the concatenated document and evaluating exact match at test time. Coref-GRU, similarly to us, encodes relations between entity mentions in the document. Instead of using graph neural network layers, as we do, they augment RNNs with jump links corresponding to pairs of corefereed mentions. MHPGM uses a multi-attention mechanism in combination with external commonsense relations to perform multiple hops of reasoning. Weaver is a deep co-encoding model that uses several alternating bi-LSTMs to process the concatenated documents and the query.

Graph neural networks have been shown successful on a number of NLP tasks (Marcheggiani and Titov, 2017; Bastings et al., 2017; Zhang et al.,

---

[6]Though other systems do not explicitly link matching mentions, they similarly benefit from masking (e.g., masks essentially single out spans that contain candidate answers).

2018a), including those involving document level modeling (Peng et al., 2017). They have also been applied in the context of asking questions about knowledge contained in a knowledge base (Zhang et al., 2018b). In Schlichtkrull et al. (2018), GCNs are used to capture reasoning chains in a knowledge base. Our work and unpublished concurrent work by Song et al. (2018) are the first to study graph neural networks in the context of multi-document QA. Besides differences in the architecture, Song et al. (2018) propose to train a combination of a graph recurrent network and an RNN encoder. We do not train any RNN document encoders in this work.

6 Conclusion

We designed a graph neural network that operates over a compact graph representation of a set of documents where nodes are mentions to entities and edges signal relations such as within and cross-document coreference. The model learns to answer questions by gathering evidence from different documents via a differentiable message passing algorithm that updates node representations based on their neighbourhood. Our model outperforms published results where ablations show substantial evidence in favour of multi-step reasoning. Moreover, we make the model fast by using pre-trained (contextual) embeddings.

Acknowledgments

References

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967. Association for Computational Linguistics.

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4220–4230. Association for Computational Linguistics.

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*.

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. 2018. Neural models for reasoning over multiple mentions using coreference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 42–48, New Orleans, Louisiana. Association for Computational Linguistics.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1601–1611.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *Kinga, D., and J. Ba Adam. "A method for stochastic optimization." International Conference on Learning Representations (ICLR).*, 5.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations (ICLR)*.

Tomas Kocisky, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gabor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197. Association for Computational Linguistics.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph lstms. *Transactions of the Association for Computational Linguistics*, 5:101–115.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. *Technical report, OpenAI*.

Martin Raison, Pierre-Emmanuel Mazaré, Rajarshi Das, and Antoine Bordes. 2018. Weaver: Deep co-encoding of questions and documents for machine reading. *In Proceedings of the International Conference on Machine Learning (ICML)*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

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 *The Semantic Web*, pages 593–607, Cham. Springer International Publishing.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *International Conference on Learning Representations (ICLR)*.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. 2017. Reasonet: Learning to stop reading in machine comprehension. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1047–1055. ACM.

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. *arXiv preprint arXiv:1809.02040*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Denny Vrandečić. 2012. Wikidata: A new platform for collaborative data collection. In *Proceedings of the 21st International Conference on World Wide Web*, pages 1063–1064. ACM.

Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making neural qa as simple as possible but not simpler. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 271–280. Association for Computational Linguistics.

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.

Caiming Xiong, Victor Zhong, and Richard Socher. 2016. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604*.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018a. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215. Association for Computational Linguistics.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. 2018b. Variational reasoning for question answering with knowledge graph. *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*.

A Implementation and Experiments Details

A.1 Architecture

See table 5 for an outline of Entity-GCN architectural detail. Here the computational steps

1. ELMo embeddings are a concatenation of three 1024-dimensional vectors resulting in 3072-dimensional input vectors $\{\mathbf{x}_i\}_{i=1}^N$.

2. For the query representation $\mathbf{q}$, we apply 2 bi-LSTM layers of 256 and 128 hidden units to its ELMo vectors. The concatenation of the forward and backward states results in a 256-dimensional question representation.

3. ELMo embeddings of candidates are projected to 256-dimensional vectors, concatenated to the $\mathbf{q}$, and further transformed with a two layers MLP of 1024 and 512 hidden units in 512-dimensional query aware entity representations $\{\hat{\mathbf{x}}_i\}_{i=1}^N \in \mathbb{R}^{512}$.

4. All transformations f_* in R-GCN-layers are affine and they do maintain the input and output dimensionality of node representations the same (512-dimensional).

5. Eventually, a 2-layers MLP with [256, 128] hidden units takes the concatenation between $\{\mathbf{h}_i^{(L)}\}_{i=1}^N$ and $\mathbf{q}$ to predict the probability that a candidate node v_i may be the answer to the query q (see Equation 1).

During preliminary trials, we experimented with different numbers of R-GCN-layers (in the range 1-7). We observed that with WIKIHOP, for $L \geq 3$ models reach essentially the same performance, but more layers increase the time required to train them. Besides, we observed that the gating mechanism learns to keep more and more information from the past at each layer making unnecessary to have more layers than required.

A.2 Training Details

We train our models with a batch size of 32 for at most 20 epochs using the Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and a learning rate of 10^{-4}. To help against overfitting, we employ dropout (drop rate $\in 0, 0.1, 0.15, 0.2, 0.25$) (Srivastava et al., 2014) and early-stopping on validation accuracy. We report the best results of each experiment based on accuracy on validation set.

B Error Analysis

In Table 6, we report three samples from WIKI-HOP development set where out Entity-GCN fails. In particular, we show two instances where our model presents high confidence on the answer, and one where is not. We commented these samples explaining why our model might fail in these cases.

C Ablation Study

In Figure 3, we show how the model performance goes when the input graph is large. In particular, how Entity-GCN performs as the number of candidate answers or the number of nodes increases.
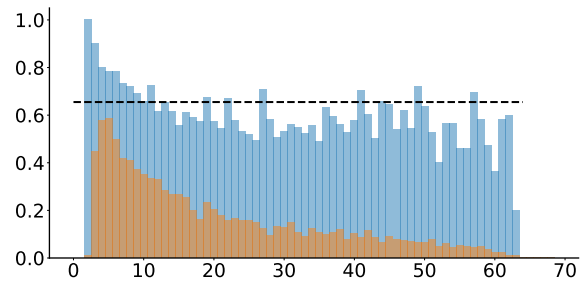
(a) Candidates set size (x-axis) and accuracy (y-axis). Pearson's correlation of -0.687 ($p < 10^{-7}$).
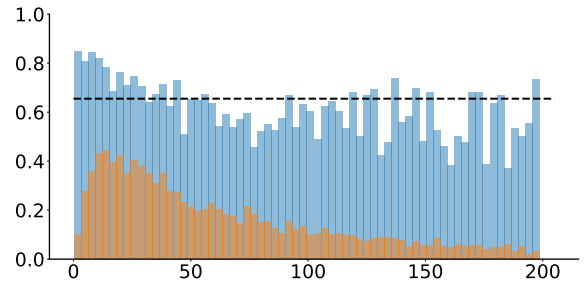
(b) Nodes set size (x-axis) and accuracy (y-axis). Pearson's correlation of -0.385 ($p < 10^{-7}$).

Figure 3: Accuracy (blue) of our best single model with respect to the candidate set size (on the *top*) and nodes set size (on the *bottom*) on the validation set. Rescaled data distributions (orange) per number of candidate (*top*) and nodes (*bottom*). Dashed lines indicate average accuracy.

Input - q, $\{v_i\}_{i=1}^N$	
query ELMo 3072-dim	candidates ELMo 3072-dim
2 layers bi-LSTM [256, 128]-dim	1 layer FF 256-dim
concatenation 512-dim	
2 layer FF [1024, 512]-dim: : $\{\hat{\mathbf{x}}_i\}_{i=1}^N$	
3 layers R-GCN 512-dim each (shared parameters)	
concatenation with **q** 768-dim	
3 layers FF [256,128,1]-dim	
Output - probabilities over C_q	

Table 5: Model architecture.

ID	WH_dev_2257	**Gold answer**	2003 ($p = 14.1$)
Query	inception (of) Derrty Entertainment	**Predicted answer**	2000 ($p = 15.8$)
Support 1	**Derrty Entertainment** is a record label founded by [...]. The first album released under **Derrty Entertainment** was Nelly 's **Country Grammar**.		
Support 2	**Country Grammar** is the debut single by American rapper Nelly. The song was produced by Jason Epperson. It was released in **2000**, [...]		

(a) In this example, the model predicts the answer correctly. However, there is a mismatch between what is written in WIKIPEDIA and what is annotated in WIKIDATA. In WIKIHOP, answers are generated with WIKIDATA.

ID	WH_dev_2401	**Gold answer**	Adolph Zukor ($p = 7.1\mathrm{e}{-}4\%$)
Query	producer (of) Forbidden Paradise	**Predicted answer**	Jesse L. Lask ($p = 99.9\%$)
Support 1	**Forbidden Paradise** is a [...] drama film produced by **Famous Players-Lasky** [...]		
Support 2	**Famous Players-Lasky** Corporation was [...] from the merger of **Adolph Zukor**'s Famous Players Film Company [..] and the **Jesse L. Lasky** Feature Play Company.		

(b) In this sample, there is ambiguity between two entities since both are correct answers reading the passages but only one is marked as correct. The model fails assigning very high probability to only on one of them.

ID	WH_dev_3030	**Gold answer**	Scania ($p = 0.029\%$)
Query	place_of_birth (of) Erik Penser	**Predicted answer**	Eslöv ($p = 97.3\%$)
Support 1	Nils Wilhelm **Erik Penser** (born August 22, 1942, in **Eslöv**, **Skåne**) is a Swedish [...]		
Support 2	**Skåne** County, sometimes referred to as " **Scania** County " in English, is the [...]		

(c) In this sample, there is ambiguity between two entities since the city Eslöv is located in the Scania County (English name of Skåne County). The model assigning high probability to the city and it cannot select the county.

Table 6: Samples from WIKIHOP set where Entity-GCN fails. p indicates the predicted likelihood.