

Complex Question Answering on knowledge graphs using machine translation and multi-task learning

Saurabh Srivastava, Mayur Patidar, Sudip Chowdhury,
Puneet Agarwal, Indrajit Bhattacharya, Gautam Shroff

TCS Research, New Delhi, India

{sriv.saurabh, patidar.mayur, sudip.chowdhury2,
puneet.a, gautam.shroff}@tcs.com

Abstract

Question answering (QA) over a knowledge graph (KG) is a task of answering a natural language (NL) query using the information stored in KG. In a real-world industrial setting, this involves addressing multiple challenges including entity linking, multi-hop reasoning over KG, etc. Traditional approaches handle these challenges in a modularized sequential manner where errors in one module lead to the accumulation of errors in downstream modules. Often these challenges are inter-related and the solutions to them can reinforce each other when handled simultaneously in an end-to-end learning setup. To this end, we propose a multi-task BERT based Neural Machine Translation (NMT) model to address these challenges. Through experimental analysis, we demonstrate the efficacy of our proposed approach on one publicly available and one proprietary dataset.

1 Introduction

Question answering on knowledge graphs (KGQA) has mainly been attempted on publicly available KGs such as Freebase [Bollacker et al. \(2008\)](#), DB-Pedia [Lehmann et al. \(2015\)](#), Yago [Suchanek et al. \(2007\)](#), etc. There is also a demand for questions answering on proprietary KGs created by large enterprises. For example, KGQA, on a) a KG that contains information related to retail products, can help the customers choose the right product for their needs, or b) a KG containing document catalogs (best practices, white papers, research papers) can help a knowledge worker find a specific piece of information, or c) a KG that stores profiles of various companies can be used to do preliminary analysis before giving them a loan, etc. Our motivating use-case comes from an enterprise system (referred to as *LOCA*) that is expected to answer users' questions about the R&D division of an enterprise.










S/N	Natural Language Question (NLQ)	Entity ID - Mentioned Entity (Entity Type)	Target Entity	Relations	is SP?	Topology
1	Who is the head of deep learning?	e1 - Deep Learning and AI (research area)	Head	Head	Y	
2	Who all have written papers on deep learning?	e2 - Deep Learning (keyword)	Researcher	Has paper, author	N	
3	Who are all the key people in John's research area?	e3 - John Dexter (Head)	Researcher	Head of, sub area, key person	N	
4	Who is head of the research area that has published papers in deep genomics?	e4 - Deep Genomics (Keyword)	Head	Has paper, explored in, belongs to, head	N	
5	Who is working in automated regulatory compliance and has published a paper in NLP?	e5 - Automated regulatory compliance (Subarea), e6 - NLP (keyword)	Researcher	key person; has paper, author	N	
6	How many papers have been published by Libby Farmer?	e7 - Libby Farmer (Researcher)	Paper	Author of	Y	
7	Show me the papers written by Libby and Rakesh on NLP	e7 - Libby Farmer (Researcher), e8 - Rakesh Gupta (Researcher), e6 - NLP (Keyword)	Paper/Title	Author of; author of; has paper	Y	
8	Has William written a paper on deep learning?	e9 - William Harmon (Researcher), e2 - deep learning (keyword)	Paper	Author of; has paper	N	
9	What is the vision of Robotics Program?	e10 - Cloud Robotics (Program)	Program/Vision	Vision	Y	

Figure 1: Example queries from a real-world dataset LOCA. Column 6 (is SP?) represents whether the queries can be answered via shortest path or not?, all the other columns are self-explanatory.

Sample questions from LOCA dataset are shown in Figure 1. The schema of the corresponding KG is shown in Figure 2. Answering such questions often requires a traversal of KG along multiple relations which may not form a directed chain graph, and may follow more complex topology as shown for question 5, 7 and 8 in Figure 1. It can also be observed that most often words of the natural language question (NLQ) and corresponding relations have a weak correlation. Most of the proposed approaches on the KGQA task [Bollacker et al. \(2008\)](#) parse the NLQ and convert it into a structured query and then execute the structured query on the KG to retrieve the factoid answers. Such conversion involves multiple sub-tasks: a) linking the mentioned entity with corresponding entity-node in the KG [Blanco et al. \(2015\)](#); [Pappu et al. \(2017\)](#), b) identification of the type of the answer entity [Ziegler et al. \(2017\)](#), c) identification of relations [Dubey et al. \(2018\)](#); [Weston et al. \(2013\)](#); [Hakkani-Tür et al. \(2014\)](#). These tasks are most often performed in a sequence [Both et al. \(2016\)](#); [Dubey et al. \(2016\)](#); [Singh et al. \(2018\)](#), or in parallel [Veyseh \(2016\)](#); [Xu et al. \(2014\)](#); [Park et al. \(2015\)](#), which results

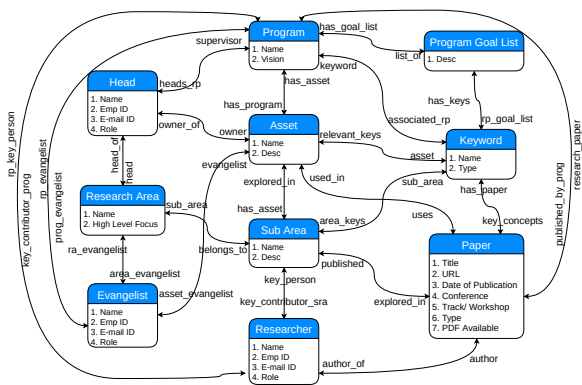


Figure 2: Schema of our proprietary KG LOCA.

in accumulation of errors [Dubey et al. \(2018\)](#). Further, most of the KGQA tasks are not as complex as LOCA. For example, a) All questions of SimpleQA [Bordes et al. \(2015\)](#) can be answered using single triple, b) NLQs for most of the datasets (e.g., SimpleQA, Meta QA) contain only one mentioned entity, and c) Even if multiple relations are required for answer entity retrieval, they are organized in a sequence, i.e., chain.

Our motivating example contains specific types of questions that pose many challenges with respect to each of the aforementioned tasks. Moreover, some of the questions can only be answered via a model that attempts more than one sub-tasks together. For example, the first two questions of [Figure 1](#) mention the same words, i.e., “deep learning” but they get associated with two different entity nodes of the KG. Additionally, the prior work could detect the set of relations when the schema sub-graph follows a specific topology, however, in our example, most of the questions follow a different topology. We demonstrate in [Section 5](#) that most of the prior art approaches fail to solve such challenges. We provide a summary of such challenges in [Section 2](#).

In this paper, we propose CQA-NMT, a novel transformer-based, NMT (neural machine translation) model to solve the aforementioned challenges by performing four tasks jointly using a single model, i.e., i) Detection of mentioned entities, ii) prediction of entity types of answer nodes, iii) prediction of topology and relations involved, and iv) question type classification such as ‘Factoid’, ‘Count’, etc. CQA-NMT not only performs the four sub-tasks but also helps downstream tasks of mentioned entity disambiguation and subsequent answer retrieval from the KG. The key contributions of this paper are:

- (i) We propose a multi-task model that performs all tasks for parsing of natural language question together, rather than the traditional approach of performing these tasks in a sequential manner, which also involves candidate generation based on upstream task and then short-listing them to make the final prediction. We also demonstrate that using such an approach newer types of challenges of the KGQA task can be solved, which have not been attempted by prior work so far.
- (ii) We propose the use of neural machine translation based approach to retrieve the variable number of relations involved in answering a complex NLQ against a KG.
- (iii) We also demonstrate that every sub-task of parsing an NLQ is complementary to other tasks and helps the model in performing better towards the final goal of KGQA. In [Table 3](#), we have demonstrated that via joint training on more than one task, the accuracy of individual tasks improves as compared to training them separately. For example, when trained separately, the best F1-score for detecting mentioned entity(s) was 83.3, and the best accuracy for the prediction of entity types of answer nodes was 75.7. When trained jointly, we get the corresponding metrics as 87.1 and 76.3. When trained jointly for all tasks, the results improve even further.
- (iv) CQA-NMT predicts the relations involved in a sub-graph of KG and also helps to predict the topology of the sub-graph, resulting in compositional reasoning via a neural network on the KG. However, the prior work predicts the relations for a specific topology only¹.
- (v) We also demonstrate that our approach outperforms the state-of-the-art approaches on the MetaQA dataset, and therefore we present a new baseline on this dataset. Our approach also performs better than standard approaches

¹Topology is a specific arrangement of how the mentioned entities, and answer entities are connected to each other via the predicted relations. Sample topologies are given in [Figure 1](#). Our approach can be used to answer questions of any topology, if adequate number of samples are included in the training data. The prior works have not attempted a dataset such as LOCA which contains many different topologies. To the best of our efforts we could not find another such dataset, which has led to our aforementioned belief.

as applicable to our dataset and helps us solve most of the real-world industrial challenges.

2 KGQA Problem and Challenges

For answering natural language questions (NLQ), we assume that the background knowledge is stored in a knowledge graph G , comprising of a set of nodes $V(G)$, and edges $E(G)$. Here, nodes represent entities, and edges represent the relationship between a pair of entities or connect an entity to one of its properties. An NLQ (q) is a sequence of words w_i of a natural language (e.g., English), i.e., $q = \{w_1, w_2, \dots, w_N\}$. We also assume that the NLQs can mention zero, one, or more entities present in G and enquire about another entity of G , which is connected with the mentioned entity(s). We pose the KGQA problem as a supervised learning problem and next, describe the labels assumed to be available for every question in the training data and that need to be predicted for every question in the test data.

Entity Linking Annotation Some of the n-grams (η_i) in an NLQ refer to entity(s) of KG. Such n-grams have been underlined in Figure 1. The entity-id (as shown in the third column of Figure 1) of the mentioned entity is also assumed to be available as part of label annotation for every question.

Answer Entity Type Annotation (AET), τ : We assume that every NLQ has an entity type (t_i) for the answer entities. These are shown in the middle column of Figure 1. We refer τ as a set of all entity types in the knowledge graph G .

Relation Sequence and Topology Annotation (path) Sequence of relations connecting the linked entities to the answer entities can be considered paths ($path_i$), each of which can contain one or more relations. These paths are connected to form a topology, as shown in Figure 1. This topology of the paths and relations are also assumed to be available for an NLQ in training data. These paths need not be the shortest paths between the linked entities and the answer entities. For example, the last three columns of Figure 1 indicate a) the set of paths separated by a semicolon (;), b) whether this is the shortest path, and c) topology of the paths connecting the linked entities to the answer entities.

Question Type Annotation (q_{type}) Some NLQs can be answered by a single triple of the knowledge graph ('Simple'), while some of them require traversal along with more complex topology as indicated earlier ('Factoid'), some questions require

an aggregate operation such as count ('Count', see question 6, in Figure 1), and finally, some questions perform existence check ('Boolean', see question 8, in Figure 1). Such information is also assumed to be available for every NLQ in training data.

We now describe the challenges that need to be addressed while performing the KGQA task. To the best of our efforts we could not find any prior work that covers all these challenges together.

1. **Incomplete Entity Mention:** In the NLQ users often do not mention the complete name of the intended entity [Huang et al. \(2019\)](#), e.g., only the first name of a person, short name of a group, etc., e.g., question 8 in Figure 1.
2. **Co-occurrence disambiguation:** For situations when a mentioned entity should be linked to KG entity with help of another mentioned entity in the question, e.g., in question 7 of Figure 1, there can be many people who have the same first name ('Libby') but there is only one of them who works on NLP, the models needs to use this information to conclusively resolve the mentioned entities [Mohammed et al. \(2017\)](#).
3. **Avoid un-intended match:** Some of the words in a sentence coincidentally match with an entity name but are not an intended mention of an entity, e.g., the word 'vision' may get matched with 'Computer Vision' which is not intended in question 9 of Figure 1.
4. **Duplicate KG Entity** The intended entity names may be different from the words used in the NLQ, and there can be more than one entity in the KG that has the same name [Shen et al. \(2019\)](#), for example, "Life Sciences" is the name of a research area, as well as a keyword (see KG schema given in Figure 2). The model needs to link the entity using other words, similar to how it is shown in questions 1 and 2 of Figure 1.
5. **Relation names mismatch:** Often the words of the KG relations and the words of the NLQ do not match, [Huang et al. \(2019\)](#), e.g., questions 2, 4, 6, etc. in Figure 1.
6. **Implicit Relations Indication:** Sometimes words of the NLQ do not even make any mention of the relations involved, however, they

need to be inferred Zhang et al. (2018). For example, in question 4 of Figure 1, some of the relations are not mentioned in the question.

Problem Definition: The objective of the proposed approach is to output 1) the mentioned entity(s) (s_i) in the query, 2) the answer entity type, 3) the path or set of predicates, $P_q = \{p_q^1, p_q^2, p_q^3, \dots, p_q^N\}$ where, each $p_i \in E(G)$ and, 4) the question type. The set P_q is a sequence of predicates, such that if traversed along these edges from the mentioned entity node(s), we can arrive at the answer entity(s) node(s). The final answer is then retrieved from KG and is post-processed as per the outputs of question type and the ‘answer entity type’ modules. We assume that we have N training samples $D_{Train} = \{(q_i, \eta_i, t_i, qt_i, path_i)\}_{i=1}^N$ available to us where, $q_i \in Q, s_i \subseteq V(G), t_i \in \tau, qt_i \in q_{type}$, and, $path_i \subseteq 2^{E(G)}$.

3 Related Work

In this section, we present a view of prior work, on the KGQA problem as an NLP task, and then on set of techniques used for this task.

3.1 KGQA Task

Berant et al. (2013); Berant and Liang (2014); Reddy et al. (2014); Luo et al. (2018) proposed an approach to perform KGQA by mapping a query to its logical form and then converting it to a formal query to extract answers. However, these are not joint learning tasks as proposed in our work.

Multi-Task based approaches Similar to us, many works like Lukovnikov et al. (2019); Huang et al. (2019); Shen et al. (2019) rely on the jointly learning multiple sub-tasks tasks of KGQA problem. However, all these approaches focus on single-hop relations only, and therefore we cannot take such approaches as a baseline for our model. In a more complex setting, Shen et al. (2019) proposed a joint learning task for entity linking, path prediction (chains topology only), and question type. However, their model does not predict answer entity type. We do not compare our approach with Shen et al. (2019) because they focus on the implicit mention of the entities in previous sentences of dialogue, and also because they do not attempt to predict non-chain topology or the answer entity type.

Non-Chain Multi-Hop Relations Agarwal et al. (2019) proposed an embedding based approach to predict non-chain multi-hop relation prediction (for

a fixed and small set of topologies). They perform only one task of relationship prediction.

3.2 Techniques used for KGQA

Transformers and Machine Translation: Transformer Vaswani et al. (2017) has proved to be one of the most exciting approaches for NLP research. They have shown dominating results in Vaswani et al. (2017); Devlin et al. (2018), etc. The paper Lukovnikov et al. (2019) closely resembles our approach as they proposed a joint-learning based multi-task model using Transformer. However, they handle only 1-hop questions and consider relation prediction as a classification task. In its current form it cannot be used to solve the variable length path prediction form, as required in our motivating example. In an extension to the work of using logical forms for KGQA Dong and Lapata (2016) proposed the usage of attention-based seq2seq model to generate the logical form of an input utterance. However, they use an LSTM model and not Transformer.

Graph-Based Approaches GraftNet Sun et al. (2018) and PullNet Sun et al. (2019) approached the problem of KGQA using graph-based solutions. The approach extracts a subgraph related to query and then performs reasoning over it to extract the final answer(s). KV-MEM Bordes et al. (2015) proposed a memory network-based approach for a single relationship prediction.

Embedding Based Approaches Approaches for KGQA using KG embeddings (such as TransE Bordes et al. (2013) and TransR Lin et al. (2015)) were used by Huang et al. (2019) when only one relation (i.e., one RDF triple) is involved. In Bordes et al. (2013); Socher et al. (2013); Dettmers et al. (2018) also one-hop KG modeling approaches were proposed. Recently, Saxena et al. (2020) presented an approach, EmbedKGQA, for joint learning, again using KG Embeddings, in the context of multi-hop relations. However, their approach is not truly a joint model as they perform answer candidate selection via the model, i.e., they arrive at the candidates before executing the model.

Our proposed approach has outperformed PullNet and EmbedKGQA on the MetaQA dataset, as shown in Section 5.

4 Proposed Architecture

In this section, we describe our proposed joint model (CQA-NMT) which is an encoder-decoder

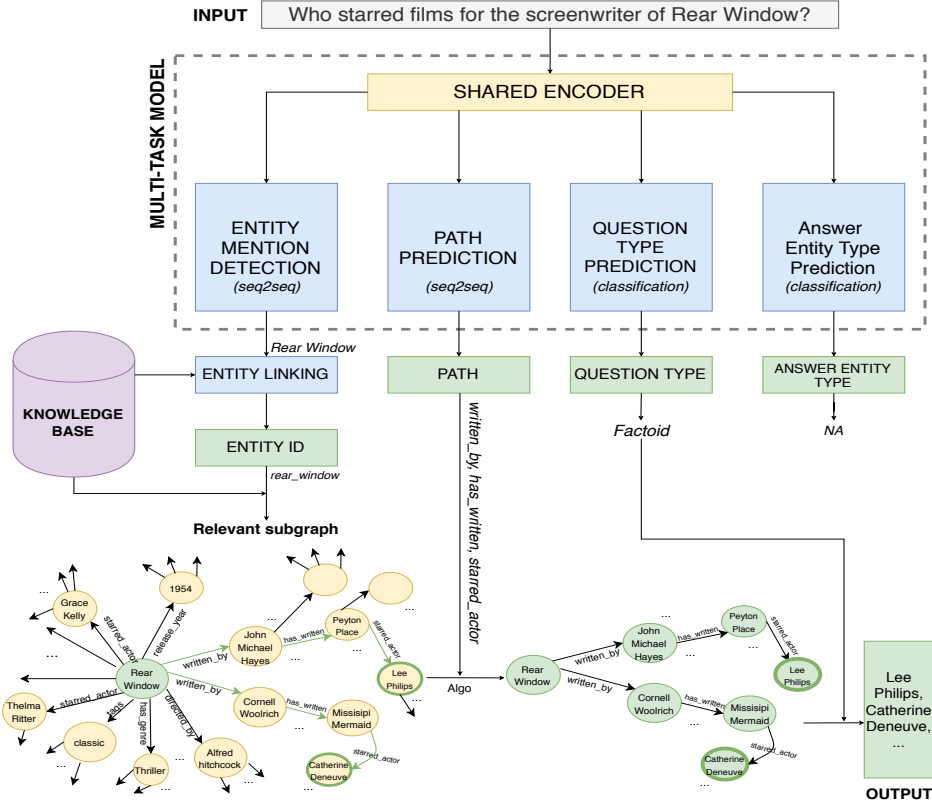


Figure 3: Proposed Multi-task model CQA-NMT. The query is passed through all the modules to (1) Extract the mentioned entity and link it to a KG node. (2) Generate the sequence of predicates required for traversal. (3) Predict the Question type and, (4) Predict AET. After extracting all the information from query, a corresponding formal query such as, SPARQL, is generated to retrieve the final answer.

based model where, BERT Devlin et al. (2018) is used as an Encoder and a Transformer Vaswani et al. (2017) as a decoder. Figure 3 illustrates a high-level view of the proposed model.

Joint Model for KGQA

In this paper, we extend BERT to generate path (or *inference chains*), perform sequence labeling and classification jointly. Details of each module are described next.

1. Entity Mention Detection Module: To extract the mentioned entity(s) from NL query, we performed a sequence labeling task using BERT’s hidden states (Figure 4). Sequence labeling is a seq2seq task that tags the input word sequence $x = (w_1, w_2, \dots, w_T)$ with the output label sequence $y_{seq} = (y_1, y_2, \dots, y_T)$. In this paper, we augmented CQA-NMT to jointly infer the type of the mentioned entity(s) along with its(their) ‘span’. We feed the final hidden states of the tokens h_2, h_3, \dots, h_{T-1} , into a softmax layer to generate output sequence. Also, we ignore the h_1 and h_T i.e., [CLS] and [SEP] tokens as they can never be a part of an entity(s) and are only required as a preprocessing step of BERT. Since BERT uses

WordPiece tokenization, we assigned the same label to the other tokenized input corresponding to their first sub-token. For e.g., the output of BERT’s Wordpiece tokenizer is ‘Jim Hen ##son’ for the input ‘Jim Henson’. We assigned the labels for the tokenized output as ‘B-Per I-Per I-per’, i.e., the second sub-word ‘##son’ was given the same label as the first sub-word ‘Hen’. The output of the softmax layer is:

$$y_{etype}^i = \text{softmax}(\mathbf{W}_{etype} \cdot h_i + b_{etype}) \quad (1)$$

where, h_i is the hidden state corresponding to the i^{th} token.

2. Entity Linking: The output of the *Entity Mention Detection Module* is a sequence of tokens along with its type (t_i) for a candidate entity. These mentioned entities still need to be linked to a KG node for traversal. In our work, we do not use any neural network for the linking process. Instead, we rely on an ensemble of string matching algorithms² and PageRank Page et al. (1999) to break the ties between candidate entities.

The Entity Mention Detection Module outputs as

²We used Levenshtein Distance and SequenceMatcher packages available in Python

many entities as provided in a query and their associated type (t_i). To link the mentioned entity in the NL query, we extract the candidates from $V(G)$ of type t_i . We then apply 3 string-matching algorithms, similar to (Mohammed et al., 2017), and take a majority voting to further break the ties. Finally, we apply the PageRank algorithm to link the mentioned entity with a KG entity. One way to understand the usability of the PageRank algorithm is to consider the notion of *popularity*. For e.g., if a user queries ‘Where was Obama born?’, the user here is more likely referring to the *famous Barack Obama*, compared to any other. A detailed description of the entity mention detection and the entity linking procedure is shown in Figure 4.

3. Path prediction Module: To generate the sequence of predicates for an input query, we augmented our architecture with a Transformer-based Vaswani et al. (2017) decoder which is often used in Neural Machine Translation (NMT) tasks. We define $y_{path} = \{p_1, p_2, \dots, p_N\}$ where each $p_i \in E(G)$. In our work, we do not constraint the number of predicates (multiple-hops) that are required to extract the final answer. Hence, an obvious choice was to use a decoder module which can stop generating the predicates once it has predicted the *end-of-sentence* ([EOS]) token (Figure 4).

4. Question Type and Answer Entity Type prediction module: In our work, we formulate the task of determining the question type and the AET as a classification task since we have a discrete set for both q_{type} and *Answer Entity Types*. Using the hidden states of the first special token from BERT, i.e., [CLS], we predict:

$$y_{q_{type}} = \text{softmax}(\mathbf{W}_{qt} \cdot h_1 + b_{qt}) \quad (2)$$

$$y_{\tau} = \text{softmax}(\mathbf{W}_{t\tau} \cdot h_1 + b_{t\tau}) \quad (3)$$

To jointly model all the task using a single architecture, we define our training objective as:

$$p(y|\mathbf{x}) = p(y_{e_{type}}, y_{path}, y_{q_{type}}, y_{\tau}|\mathbf{x}) \quad (4)$$

$$p(y|\mathbf{x}) = p(y_{q_{type}}|\mathbf{x}) \cdot p(y_{\tau}|\mathbf{x}) \cdot p(y_{e_{type}}|\mathbf{x}) \cdot p(y_{path}|\mathbf{x}) \quad (5)$$

The path and AET components of CQA-NMT are defined as,

$$p(y_{e_{type}}|\mathbf{x}) = \prod_{n=1}^N p(y_{e_{type}}^n|\mathbf{x}), \quad (6)$$

$$p(y_{path}|\mathbf{x}) = \prod_{t=1}^T p(p_t|p_1, p_2, \dots, p_{t-1}|\mathbf{x}). \quad (7)$$

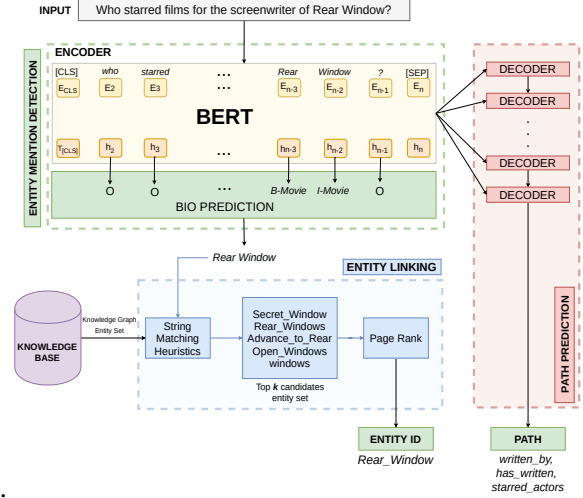


Figure 4: Entity Mention and Entity Linking Modules.

where,

$$y_{q_{type}} \in \{\text{factoid, count, boolean, simple}\} \quad (8)$$

$$y_{\tau} \in \{\text{entity types in KG}\} \quad (9)$$

$$y_{e_{type}}^i \in \{B, I\} \times \{\text{entity_types}\} \cup \{O\} \quad (10)$$

$$p_i \in E(G). \quad (11)$$

For training we maximize the conditional probability $p(y_{e_{type}}, y_{path}, y_{q_{type}}, y_{\tau}|\mathbf{x})$. The model is fine-tuned end-to-end via minimizing the cross-entropy loss.

5 Experiments and System details

In this section, we first introduce the datasets used for our experiments. We pre-process all NLQs (of all datasets) by downcasing and tokenizing.

5.1 Datasets, Metrics, and Baselines

LOCA Dataset: We introduce a new challenging dataset ‘LOCA’, which consists of 5010 entities, 42 unique predicates, and a total of 45,869 facts. The dataset has 3,275 one or multi-hop questions that have 0, 1, or more entities mentioned in the questions. It contains multiple question types like *count, factoid, and boolean*.

For the questions with multiple entities, we used an operator ‘;’ as a delimiter to separate paths corresponding to each entity (in Figure 1, query 5, 7, and 8). For the scope of this paper, we considered queries involving the only *intersection* which can be replaced with other operators like *union, set-difference, etc.* without loss of any generality. The operator ‘;’ help us detect and predict the different topologies involved in an NL

	Train	Dev	Test
MetaQA 1-hop	96,106	9,992	9,947
MetaQA 2-hop	118,980	14,872	14,872
MetaQA 3-hop	114,196	14,274	14,274
LOCA 1-hop	2,000	250	250
LOCA 2-hop	400	50	50
LOCA 3[or more]-hop	220	28	28

Table 1: Statistics of MetaQA and LOCA dataset

query.

MetaQA: The dataset proposed in Zhang et al. (2018) consists of 3 different datasets namely, *Vanilla*, *NTM*, and, *Audio Data*. All the datasets contain single and multi-hop (maximum 3-hop) questions from the *movie* domain. For our experiments, we used the Vanilla and the NTM version of the datasets and the KB as provided in Zhang et al. (2018). Since, both versions of MetaQA do not consider the AET and question type, we assigned a default label to both the tasks.

Metrics: We used different metrics for different subtasks. Since a query can contain partially mentioned entities, we used F-score to evaluate mention and its type detection module. For Inference Chain (or Path prediction), question type, and, answer entity type prediction we use the accuracy measure. In Table 2, similar to prior works, we have used the Hits@1 to evaluate the query-answer accuracy.

Baselines: We have used, KV-mem Bordes et al. (2015), GraftNet Sun et al. (2018), PullNet Sun et al. (2019), VRN Zhang et al. (2018), and EmbedKGQA Saxena et al. (2020) as the baselines.

5.2 Training Details

All the baselines and the proposed approach were trained on DGX 32GB NVIDIA GPU using TensorFlow Abadi et al. (2015) and Texar Hu et al. (2018) libraries. For CQA-NMT, we used the small uncased version of pre-trained BERT Devlin et al. (2018) model. Adam Kingma and Ba (2014) optimizer was employed with a learning rate of $2e-5$ for BERT and default for others. The training objective of each model was maximized using the cross-entropy loss and the best models were selected using the validation loss. Dropout values were set to .5 and were optimized as described in Srivastava et al. (2014). For BERT we used 10% of total training data for the warmup phase of BERT Vaswani et al. (2017). Finally, for the division of

dataset into train, test, and, dev, we used the same split as provided by Zhang et al. (2018) for the MetaQA dataset and a ratio of 80-10-10 for LOCA dataset.

5.3 Main Results

In this section, we report the results of the experiments on the MetaQA and the LOCA dataset. Next, we provide insights into the model outputs and results of error-analysis performed on LOCA dataset.

5.3.1 LOCA

The experimental results for LOCA dataset are shown in the last row of table 2. The results affirm that the proposed approach outperforms the baselines. We observed that the baselines’ inability to handle *Duplicate KG Entity* (Section 2 challenge 4) limits their performance. Additionally, the ability of the NMT Bahdanau et al. (2014) model to effectively handle complex and un-known topologies helped us retrieve answers with better accuracy for variable-hop (v-hop) queries.

5.3.2 MetaQA

The experimental results for MetaQA are shown in table 2. For *Vanilla* MetaQA, we achieved better answer accuracy on 1-hop and 3-hop settings. However, in a 2-hop setting, we were able to achieve comparable results to the state-of-the-art. An increment of about 2% and 4.9% Hits@1 can be seen in the 1-hop and 3-hop settings.

To obtain the performance of each baseline on v-hop (variable-hop) dataset, we re-use the existing models for 1-hop, 2-hop, and 3-hop and assume that there is an oracle which can redirect query to the correct model. Thus estimated accuracy of various approaches is shown in the 4th row of Table 2, while the actual results on v-hop dataset are shown in the 5th row. It is evident that CQA-NMT outperforms all the baselines on MetaQA dataset in variable hop setting.

To gauge the effectiveness and robustness of our model, we used the same models trained on vanilla MetaQA dataset and evaluated its performance on *NTM* MetaQA, i.e., in zero-shot setting. For this, we achieved better results on 1 and 3-hop. The worse performance of CQA-NMT on MetaQA-NMT(2-hop) can be because of zero shot setting. Because, as compared to VRN, we have not trained CQA-NMT on MetaQA-NTM dataset, we trained it on MetaQA vanilla dataset only.

	KV-Mem	GraftNet	PullNet	EmbedKGQA	VRN	CQA-NMT
MetaQA(1-hop)	96.2	97.0	97.0	97.5	97.5	99.96
MetaQA(2-hop)	82.7	94.8	99.9	98.8	89.9	99.45
MetaQA(3-hop)	48.9	77.7	91.4	94.8	62.5	99.78
MetaQA(V-hop) Estimated	73.79	89.11	96.05	-	-	99.69
MetaQA(V-hop)	-	83.67*	-	-	-	95.85
MetaQA-NTM(1-hop)	-	-	-	-	81.3	83.3
MetaQA-NTM(2-hop)	-	-	-	-	69.7	62.1
MetaQA-NTM(3-hop)	-	-	-	-	38.0	81.3
LOCA	51.07	72.27	-	-	-	78.29

Table 2: Results of the baselines and CQA-NMT on MetaQA (vanilla and NTM) and LOCA dataset. Results for MetaQA-NTM were obtained in zero-shot setting for CQA-NMT. All other baseline results are taken from Sun et al. (2019), Zhang et al. (2018), and Saxena et al. (2020), except for * marked numbers. Note: Source code for PullNet, VRN is not available. We were not able to replicate the same results of KV-Mem.

	Mention Detection (F1-score)	AET (Accuracy)	Inference Chains (Accuracy)	Question Type (Accuracy)	Answer (Accuracy)
Unsupervised	67.12	53.3	51.0	-	42.9
BERT (mention only)	83.3	53.22	51.9	-	47.22
BERT (AET only)	67.38	75.7	55.6	-	45.2
BERT (IC only)	67.8	50.9	80.1	-	40.1
BERT (mention and AET)	87.1	76.3	71.6	-	65.33
BERT (AET and IC)	66.3	74.1	77.8	-	60.39
BERT (Mention and IC)	87.3	53.1	79.6	-	51.3
CQA-NMT	93.66	79.89	81.95	97.65	71.01

Table 3: Effects for reducing the supervision from our approach. The numbers in *italics* are obtained without any supervision.

5.4 Further Results and Analysis

Advantage of Transformers: In the LSTM based implementation of mentioned entity detection, it could not detect different entity types for the same phrase “deep learning” in query 1 and 2 of Figure 1. However, in BERT-based approach it was able to. We therefore infer that such phenomenon could occur due to key features of BERT such as multi head attention, WordPiece embeddings, Positional embeddings, and/ or Segment embeddings. Moreover, in a different context, it was able to assign different types to the entities with the same mentions (Query 1 and 2 from Figure 1).

Effects of using less annotations: To study the importance of annotation in our approach, we removed several components from our proposed approach and studied the effects (Table 3). We first studied CQA-NMT after removing all the supervision and used heuristics-based-approaches for AET and Mention Detection (both the approaches were taken from Mohammed et al. (2017)). The shortest path, similar to Sun et al. (2018, 2019), between the linked KG entity and AET, was then taken to retrieve the answers. This setting (row 1) results in

the worst performance. In row 2, 3, and 4 of Table 3, we kept only one component of CQA-NMT as supervised and applied heuristics for others as mentioned above. As evident from these rows, mention detection plays a crucial role in extracting the correct answer (a jump in range of 2%-5% in answer accuracy). A similar analysis can be found in Dong and Lapata (2016); Guo et al. (2018); Srivastava et al. (2020), where authors found that entity linking error is one of the major errors leading to wrong predictions in KGQA.

In summary, while testing each components of CQA-NMT, we tried supervising different components at a time and used heuristics based approaches for the remaining components. The heuristics are:

- Shortest path algorithm, similar to Sun et al. (2018, 2019), for Path Prediction Task.
- Candidate and Relation Pairing Score Mohammed et al. (2017) for Entity Linking.
- LSTM based Classifier Mohammed et al. (2017) for Answer Entity Type Prediction. [however in Row 1 “Unsupervised”, we identify the AET if name of the entity-type is present in the NLQ].

Benefits of joint training: From row 5-7 of Table 3, we infer that joint training not only improves the scores of individual components (in range 15%-20%) but also, the overall answer accuracy. We observed that the challenges 5 and 6 from section 2 were handled significantly better after jointly training CQA-NMT for AET and mention detection (row 5). We found that the context used by a mentioned entity for AET was different in different queries. For e.g., in queries ‘Who heads Deep Learning?’ and ‘papers in Deep Learning’, *Deep Learning*, is a *research area* with AET ‘head’ in

the former, and in the later, is a *keyword* with AET ‘paper.title’. After jointly training the mention detection model with AET or IC, a jump of $\sim 4\%$ can be observed for the individual components and $\sim 11\text{-}20\%$ in answer accuracy.

Errors with Joint Training: From table 3, it seems that the joint-training helps. However, we also found that the error made by a single module produces a cascading of errors. For example, if for a **single** mention span ‘deep learning and ai’ with the gold label as ‘B-research_area I-research_area I-research_area I-research_area’, the entity detection predicts ‘B-research_area I-research_area O B-keyword’, the path prediction module generates two different paths, one for each entity, leading to wrong answer.

Similarly, if AET prediction module makes an ‘error’, it gets propagated to all the other modules simultaneously. The frequency of such error, however, was considerably small and in range of 1-2% of total training or test data.

Robustness against variable hops: We combined all the 3 vanilla MetaQA datasets into a single dataset. The results on this new dataset, *MetaQA v-hop (variable-hop)* is shown in table 2. Since the other approaches in the literature did not perform such analysis, MetaQA v-hop is a new challenge that we propose.

Motivation for PageRank: When we have more than one candidate entity for a mentioned entity, we want to choose the one with higher popularity (Sec 4). One of the most well established measure of popularity of nodes in graphs is PageRank. Therefore we have used it. Further, when more than one entity are mentioned in an NLQ, there can be more than one candidate entity for each of them. The graph-based approach also helps us choose the candidates that are well connected.

We also experimented using other measures such as in-degree and out-degree of nodes. However, for LOCA dataset, we achieved an increment of 22% using PageRank on Entity Linking task, as compared to the in-degree and out-degree measures. PageRank also helped in reducing the challenges 1-2 from Sec. 2.

5.4.1 Retrieval of answer(s) from KG

The final objective of a KG-QA system is to retrieve the correct answer from KG against a query q . To this end, we use the outputs of the different components of CQA-NMT and feed them to complete the pre-written SPARQL sketches. We defined

a bunch of rules for different question-types and used a simple-mapping rules to map the queries to the sketches. For e.g., consider the query, $q =$ “Who is working in automated regulatory compliance and has published a paper in NLP?”. The output of CQA-NMT contains all the information that is required to form a structured query such as SPARQL. The outputs of CQA-NMT are:

1. Linked Entities: {e5: automated regulatory compliance (sub-area), e6: NLP (keyword)}
2. Inference Chain: key_person; has_paper, author
3. Answer Entity Type (AET): researcher.name
4. Question Type (q_{type}): Factoid

After using the q_{type} information, we fill a sketch using other outputs. The generated SPARQL query is:

```
SELECT DISTINCT ?uri WHERE {<e5>
<key_person> <?uri> . <e6> <has_paper>
<?x> . <?x> <author> <?uri>}
```

Where, e5 and e6 are unique identities assigned to ‘automated regulatory compliance’ (of type *sub-area*) and NLP (of type *keyword*).

6 Conclusion

We presented a complex version of the KGQA problem, which involves mention of multiple entities in the question. Multiple sequence of relationships combined in complex topologies, are required to answer such questions. It is evident that such questions, while required to be answered in real world industrial setting, cannot be answered using prior approaches. We propose a novel CQA-NMT model to answer such questions and have performed a detailed comparison of our approach with prior art on MetaQA and Loca datasets. We have shown that CQA-NMT not only solves more complex task, but also performs better on MetaQA dataset as compared to baseline approaches.

Acknowledgements

We would like to thank the reviewers of this paper for their valuable comments and all the people involved in this work. We would also like to thank Vaibhav Varshney (varshney.v@tcs.com) for the diagrams and his valuable comments toward this paper.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado,

- Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. *TensorFlow: Large-scale machine learning on heterogeneous systems*. Software available from tensorflow.org.
- Puneet Agarwal, Maya Ramanath, and Gautam Shroff. 2019. Retrieving relationships from a knowledge graph for question answering. In *European Conference on Information Retrieval*, pages 35–50. Springer.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.
- Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425.
- Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and space-efficient entity linking for queries. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 179–188.
- 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 the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.
- Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.
- Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.
- Andreas Both, Dennis Diefenbach, Kuldeep Singh, Saedeh Shekarpour, Didier Cherix, and Christoph Lange. 2016. Qanary—a methodology for vocabulary-driven open question answering systems. In *European Semantic Web Conference*, pages 625–641. Springer.
- Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- 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*.
- Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. *arXiv preprint arXiv:1601.01280*.
- Mohnish Dubey, Debayan Banerjee, Debanjan Chaudhuri, and Jens Lehmann. 2018. Earl: joint entity and relation linking for question answering over knowledge graphs. In *International Semantic Web Conference*, pages 108–126. Springer.
- Mohnish Dubey, Sourish Dasgupta, Ankit Sharma, Konrad Höffner, and Jens Lehmann. 2016. Asknow: A framework for natural language query formalization in sparql. In *European Semantic Web Conference*, pages 300–316. Springer.
- Daya Guo, Duyu Tang, Nan Duan, Ming Zhou, and Jian Yin. 2018. Dialog-to-action: Conversational question answering over a large-scale knowledge base. In *Advances in Neural Information Processing Systems*, pages 2942–2951.
- Dilek Hakkani-Tür, Asli Celikyilmaz, Larry Heck, Gokhan Tur, and Geoff Zweig. 2014. Probabilistic enrichment of knowledge graph entities for relation detection in conversational understanding. In *Fifteenth Annual Conference of the International Speech Communication Association*.
- Zhiting Hu, Haoran Shi, Bowen Tan, Wentao Wang, Zichao Yang, Tiancheng Zhao, Junxian He, Lianhui Qin, Di Wang, Xuezhe Ma, et al. 2018. Texar: A modularized, versatile, and extensible toolkit for text generation. *arXiv preprint arXiv:1809.00794*.
- Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge graph embedding based question answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 105–113.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia—a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

- Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*.
- Denis Lukovnikov, Asja Fischer, and Jens Lehmann. 2019. Pretrained transformers for simple question answering over knowledge graphs. In *International Semantic Web Conference*, pages 470–486. Springer.
- Kangqi Luo, Fengli Lin, Xusheng Luo, and Kenny Zhu. 2018. Knowledge base question answering via encoding of complex query graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2185–2194.
- Salman Mohammed, Peng Shi, and Jimmy Lin. 2017. Strong baselines for simple question answering over knowledge graphs with and without neural networks. *arXiv preprint arXiv:1712.01969*.
- Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.
- Asish Pappu, Roi Blanco, Yashar Mehdad, Amanda Stent, and Kapil Thadani. 2017. Lightweight multilingual entity extraction and linking. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 365–374.
- Seonyeong Park, Soonchoul Kwon, Byungsoo Kim, and Gary Geunbae Lee. 2015. Isoft at qald-5: Hybrid question answering system over linked data and text data. In *CLEF (Working Notes)*.
- Siva Reddy, Mirella Lapata, and Mark Steedman. 2014. Large-scale semantic parsing without question-answer pairs. *Transactions of the Association for Computational Linguistics*, 2:377–392.
- Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. *Proceedings of the 2020 conference on Association for Computational Linguistics*.
- Tao Shen, Xiubo Geng, Tao Qin, Daya Guo, Duyu Tang, Nan Duan, Guodong Long, and Daxin Jiang. 2019. Multi-task learning for conversational question answering over a large-scale knowledge base. *arXiv preprint arXiv:1910.05069*.
- Kuldeep Singh, Arun Sethupat Radhakrishna, Andreas Both, Saeedeh Shekarpour, Ioanna Lytra, Ricardo Usbeck, Akhilesh Vyas, Akmal Khikmatullaev, Dharmen Punjani, Christoph Lange, et al. 2018. Why reinvent the wheel: Let’s build question answering systems together. In *Proceedings of the 2018 World Wide Web Conference*, pages 1247–1256.
- Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934.
- Himani Srivastava, Prerna Khurana, Saurabh Srivastava, Vaibhav Varshney, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2020. Improved question answering using domain prediction.
- 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.
- Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706.
- Haitian Sun, Tania Bedrax-Weiss, and William W Cohen. 2019. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. *arXiv preprint arXiv:1904.09537*.
- Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. *arXiv preprint arXiv:1809.00782*.
- 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.
- Amir Poursan Ben Veyseh. 2016. Cross-lingual question answering using common semantic space. In *Proceedings of TextGraphs-10: the workshop on graph-based methods for natural language processing*, pages 15–19.
- Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. 2013. Connecting language and knowledge bases with embedding models for relation extraction. *arXiv preprint arXiv:1307.7973*.
- Kun Xu, Sheng Zhang, Yansong Feng, and Dongyan Zhao. 2014. Answering natural language questions via phrasal semantic parsing. In *Natural Language Processing and Chinese Computing*, pages 333–344. Springer.
- Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- David Ziegler, Abdalghani Abujabal, Rishiraj Saha Roy, and Gerhard Weikum. 2017. Efficiency-aware answering of compositional questions using answer

type prediction. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 222–227.