Improved Neural Relation Detection for Knowledge Base Question Answering

Caleb Bryant[1] Jixin Feng[2]

EEL6935 T21

Department of
[1]Computer & Information Science & Engineering
[2]Electrical & Computer Engineering
University of Florida, Gainesville, FL

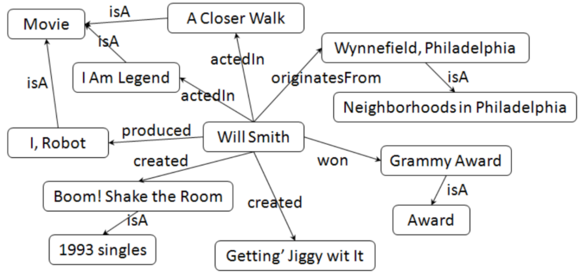
Paper Outline

Improved Neural Relation Detection for Knowledge Base Question Answering

Mo Yu[†] **Wenpeng Yin**[⋆] **Kazi Saidul Hasan**[‡] **Cicero dos Santos**[†]
Bing Xiang[‡] **Bowen Zhou**[†]
[†]AI Foundations, IBM Research, USA
[⋆]Center for Information and Language Processing, LMU Munich
[‡]IBM Watson, USA
{yum,kshasan,cicerons,bingxia,zhou}@us.ibm.com, wenpeng@cis.lmu.de

- Introduction
- Relation Extraction & Relation Detection
- Different Granularity in KB Relations
- Proposed Improved Relation Detection
- KBQA Enhanced by Relation Detection
- Experiment & Conclusion

What is Knowledge Base?



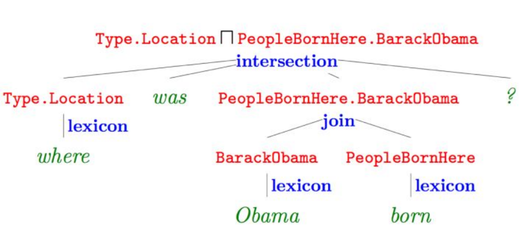
A knowledge base (KB) is a technology used to store complex **structured** and **unstructured** information used by a computer system.

–*Wikipedia*

It stores and manages the knowledge tuples: `<entity-relation-entity>`

A knowledge base can be represented as graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, if we treat entities as vertices and relations as edges.

What is a Question Answering System?



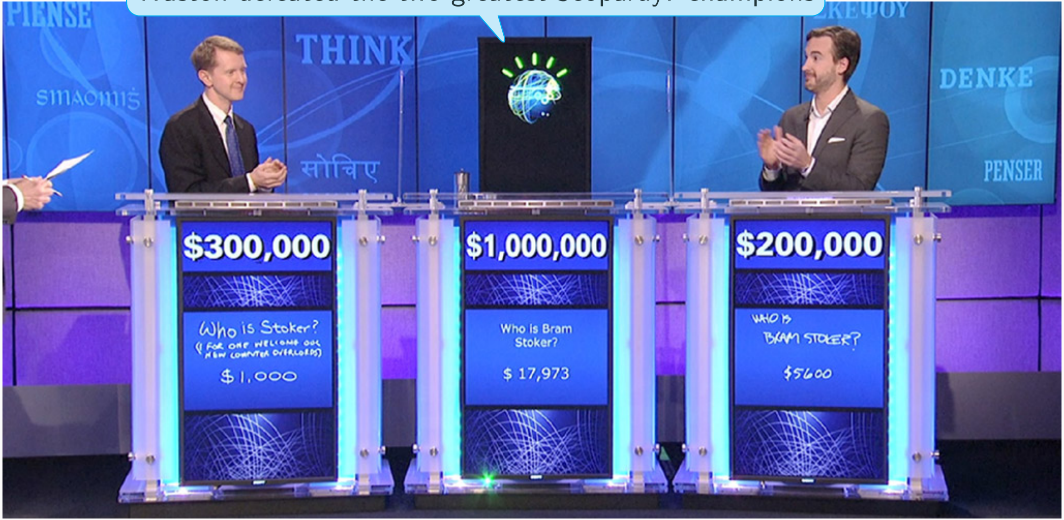
Question answering (QA) is a computer systems that can automatically answer questions posed by humans in natural language. QA systems often convert their input into a structured form which can be used to query a KB.

What is a Question Answering System?

IBM Watson is an automated QA system

What is a Question Answering System?

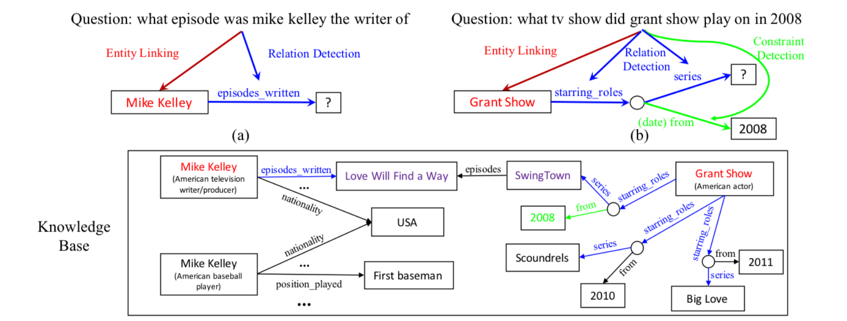
Waston defeated the two greatest Jeopardy! champions



What is a Question Answering System?

Won the match, outscoring both opponents combined

Introduction



- KBQA systems answer questions by obtaining information from KB tuples
- `input question` $\rightarrow$ `KB query` $\rightarrow$ `answer`
- Questions can be single-relation questions or multiple-relation questions

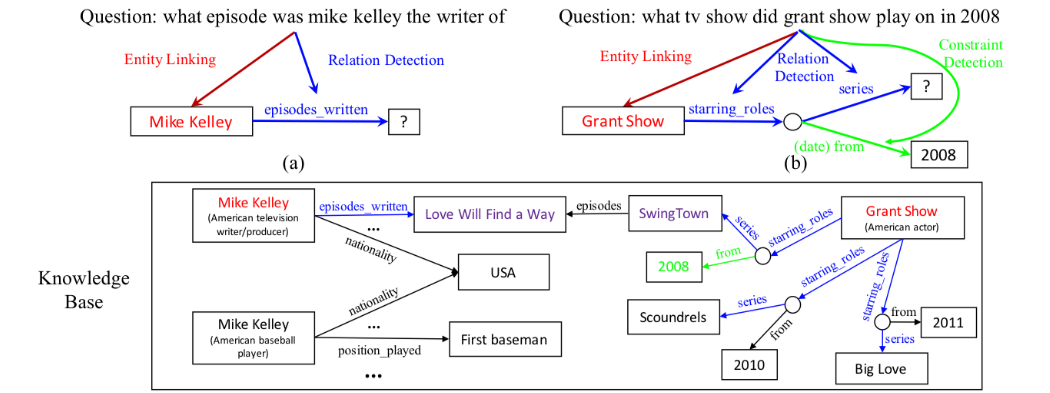


Figure 1: KBQA examples and its three key components. (a) A single relation example. We first identify the topic entity with ***entity linking*** and then detect the relation asked by the question with ***relation detection*** (from all relations connecting the topic entity). Based on the detected entity and relation, we form a query to search the KB for the correct answer "*Love Will Find a Way*". (b) A more complex question containing two entities. By using "*Grant Show*" as the topic entity, we could detect a chain of relations "*starring_roles-series*" pointing to the answer. An additional ***constraint detection*** takes the other entity "*2008*" as a constraint, to filter the correct answer "*SwingTown*" from all candidates found by the topic entity and relation.

Introduction

This paper focuses on improving relation detection.

- General relation detection has been well studied in NLP
  - Most research typically not KBQA driven
  - Significant gap between previous work and current needs
- Number of target relations is limited in most research papers
  - Often attempt detecting ≤ 100 relations
  - Even a small KB may contain ≥ 6000 relations
- Multiple relation questions require a chain of prediction
- Relation detection in KBQA often becomes a **zero-shot learning** task

Introduction

This paper focuses

- General relatio
  - Most resea
  - Significant
- Number of tar
  - Often atte
  - Even a sm
- Multiple relati
- Relation detection in KBQA often becomes a **zero-shot learning** task

> Zero-shot learning is being able to solve a task despite not having received any training examples of that task. For a concrete example, imagine recognizing a category of object in photos without ever having seen a photo of that kind of object before. If you've read a very detailed description of a cat, you might be able to tell what a cat is in a photograph the first time you see it.
>
> *–Ian Goodfellow*

Introduction

This paper focuses on improving relation detection.

- General relation detection has been well studied in NLP
  - Most research typically not KBQA driven
  - Significant gap between previous work and current needs
- Number of target relations is limited in most research papers
  - Often attempt detecting ≤ 100 relations
  - Even a small KB may contain ≥ 6000 relations
- Multiple relation questions require a chain of prediction
- Relation detection in KBQA often becomes a **zero-shot learning** task
- **Conclusion:** KB relation detection is much harder

Relation Extraction

- A related problem from the Information Extraction community
- **The problem:** given a natural language document and all entity occurrences inside it, find all of the relationships encoded within the document.
- RE usually formulated as classification task
  - Classify all possible `<entity1, paragraph, entity2>` vectors
  - Relationships with high enough confidence are collected
  - Traditional RE relies on large amount of hand-crafted features
  - Impovements from deep learning research: word embeddings, CNN, LSTM, etc.

Relation Extraction

- A related problem from the Information Extraction community
- **The problem:** given a natural language document and all entity occurrences inside it, find all of the relationships encoded within the document.
- RE usually formulated as classification task
  - Classify all possible `<entity1, paragraph, entity2>` vectors
  - Relationships with high enough confidence are collected
  - Traditional RE relies on large amount of hand-crafted features
  - Impovements from deep learning research: word embeddings, CNN, LSTM, etc.
- Traditionally assumes closed set of relation types to avoid zero-shot learning
  - Rarely goes beyond ≥ 100 features
  - Needs to be trained in supervised way
- Assumes two argument entities are both available
  - For QA, only single argument is available
- RE usually works on small, pre-defined relation sets

Relation Detection

- The focus of this paper
- **The problem:** given a question and a relationship type, output the probability of that relationship being encoded in the question
- Large relation vocabulary/open relation sets needed to support RD task
- Fits the goal of open-domain question answering
- Need to deal with zero-shot learning:
    - Treat questions and potential relationships as sequences and use **sequence matching and ranking**
- Matching and ranking works well because relation names usually form meaningful word sequences

Relation Detection

- The focus of this paper
- **The problem:** given a question and a relationship type, output the probability of that relationship being encoded in the question
- Large relation vocabulary/open relation sets needed to support RD task
- Fits the goal of open-domain question answering
- Need to deal with zero-shot learning:
    - Treat questions and potential relationships as sequences and use **sequence matching and ranking**
- Matching and ranking works well because relation names usually form meaningful word sequences
- This paper focuses on improving RD

Different Granularity in KB Relations

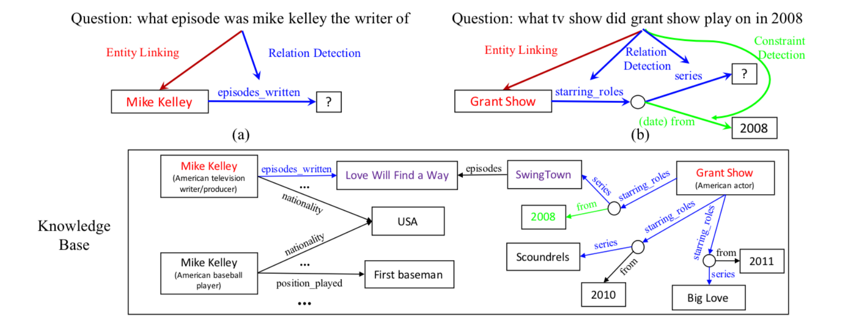
- **Goal:** Formulate KB relation detection as sequence matching problem
- Different types of relationship representation
  - Relation as a Single Token - use KB embeddings! (*relation-level*)
  - Relation as a Word Sequence - use word embeddings! (*word-level*)

	Relation Token	Question 1	Question 2
		what tv episodes were <e> the writer of	what episode was written by <e>
relation-level	episodes_written	*tv episodes were <e> the writer of*	*episode was written by <e>*
word-level	episodes	*tv episodes*	*episode*
	written	*the writer of*	*written*

Table 1: An example of KB relation (*episodes_written*) with two types of relation tokens (relation names and words), and two questions asking this relation. The topic entity is replaced with token <*e*> which could give the position information to the deep networks. The italics show the evidence phrase for each relation token in the question.

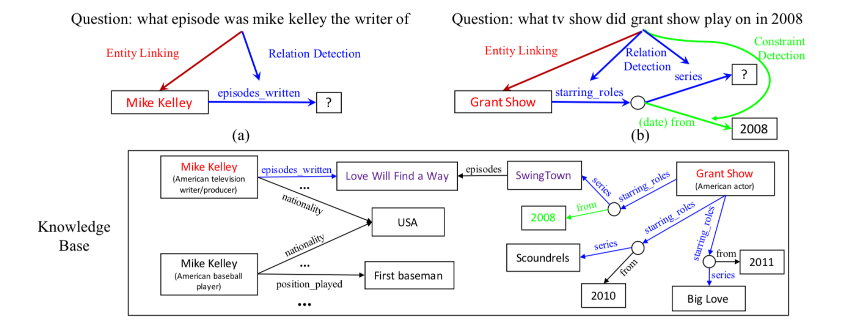
Relation as a Single Token

- Each relation is treated as a unique token
- This suffers from the low relation coverage due to limited training data
- For example, matching `episodes_written` and `starring_roles` will be very hard if no examples appear in the training data

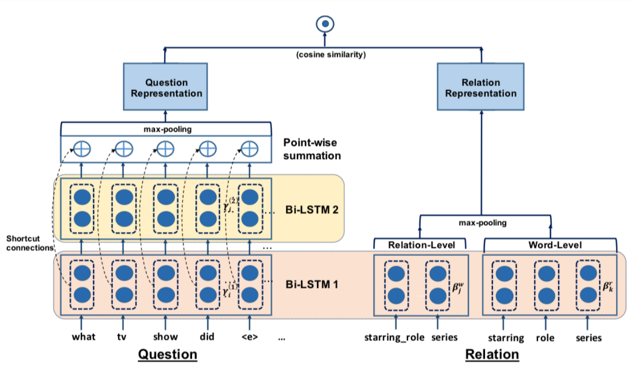


Relation as Word Sequence

- Relation is treated as a sequence of words from the tokenized relation name
- Better generalization but has difficulty learning higher-level concepts
- Very hard to rank `starring_roles` higher than `plays_produced`
    - Incorrect relation contains word "plays"
    - More similar to the question



Proposed Improved Relation Detection



- Word-level focuses more on local information but lacks of global information
- Relation-level focuses more on global information but suffers from data sparsity
- This paper propose a hierarchical matching approach that incorporates both information levels

Relation Representation from Different Granularities

- Tokenize input relation as

$$\mathbf{r} = \{r_1^{word}, r_2^{word}, \ldots, r_{M_1}^{word}\} \cup \{r_1^{rel}, r_2^{rel}, \ldots, r_{M_2}^{rel}\}$$

- first M_1 tokens are words and last M_2 tokens are relation names
- Transform each token to its word/relation embedding
- Then find hidden representation for the relation sequence

$$[\mathbf{B}_{1:M_1}^{word} : \mathbf{B}_{1:M_1}^{rel}]$$

  via two BiLSTMs
- Initialize relation sequence LSTMs with the final state representations of the word sequence
- Apply max-pooling to these two sets of vectors and get the final relation representation $\mathbf{h}^r$

Different Abstraction of Questions Representations

- Question representation vectors should summarize various length phrases to match relation representations of different granularity
- Achieved via applying deep BiLSTM on questions
- First layer works on word embeddings, converts question words $\mathbf{q} = \{q_1, \ldots, q_N\}$ to hidden representation $\mathbf{\Gamma}^{(1)}_{1:N} = [\gamma^{(1)}_1, \ldots, \gamma^{(1)}_N]$
- Second layer convert $\mathbf{\Gamma}^{(1)}_{1:N}$ to $\mathbf{\Gamma}^{(2)}_{1:N}$
- $\mathbf{\Gamma}^{(1)}_{1:N}$ and $\mathbf{\Gamma}^{(2)}_{1:N}$ could potentially match to either level of relation representation
- Need additional methods to reduce training difficulty

Hierachical Matching between Relation and Questions

- Two levels of question hidden representation are not guaranteed to be comparable
- Training deep BiLSTM is difficult - use residual connections between layers
- This paper proposed two ways of Hierarchical Residual Matching
  - Connecting $\gamma_i^{(1)}$ and $\gamma_i^{(1)}$ gives $\gamma_i' = \gamma_i^{(1)} + \gamma_i^{(2)}$, and the final question representation $\mathbf{h}^q$ becomes a max-pooling of all γ_i'
  - Applying max-pooling to $\mathbf{\Gamma}_{1:N}^{(1)}$, $\mathbf{\Gamma}_{1:N}^{(2)}$ we get

$$\mathbf{h}^q = \mathbf{h}_{max}^{(1)} + \mathbf{h}_{max}^{(1)}$$

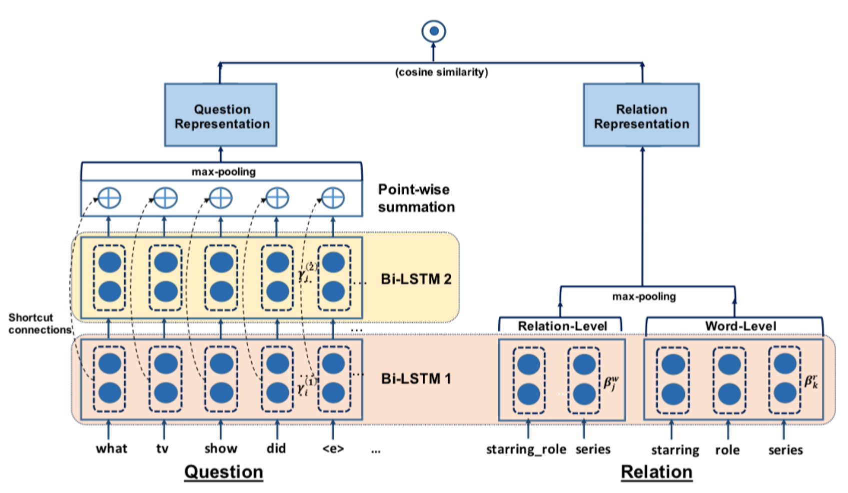
- Matching score between relation and question is

$$s_{rel}(\mathbf{r}; \mathbf{q}) = \cos(\mathbf{h}^r, \mathbf{h}^q)$$

- Ranking loss between golden relation $\mathbf{r}^+$ and other relations $\mathbf{r}^-$

$$l_{rel} = \max\{0, \gamma - s_{rel}(\mathbf{r}^+; \mathbf{q}) + s_{rel}(\mathbf{r}^-; \mathbf{q})\}$$

Hierarchical Residual BiLSTM Model



KBQA Enhanced by Relation Detection

- Take an existing entify linker to produce the top-K linked entities $EL_K(q)$ for question q
- Generate KB queries for q following a 4-step algorithm

Algorithm 1: KBQA with two-step relation detection

Input : Question q, Knowledge Base KB, the initial top-K entity candidates $EL_K(q)$

Output: Top query tuple $(\hat{e}, \hat{r}, \{(c, r_c)\})$

1 **Entity Re-Ranking** (*first-step relation detection*): Use the *raw question text* as input for a relation detector to score all relations in the KB that are associated to the entities in $EL_K(q)$; use the relation scores to re-rank $EL_K(q)$ and generate a shorter list $EL'_{K'}(q)$ containing the top-K' entity candidates (Section 5.1)

2 **Relation Detection**: Detect relation(s) using the *reformatted question text* in which the topic entity is replaced by a special token $<e>$ (Section 5.2)

3 **Query Generation**: Combine the scores from step 1 and 2, and select the top pair $(\hat{e}, \hat{r})$ (Section 5.3)

4 **Constraint Detection** (optional): Compute similarity between q and any neighbor entity c of the entities along $\hat{r}$ (connecting by a relation r_c) , add the high scoring c and r_c to the query (Section 5.4).

Experiment: Relation Detection

- Dataset: SimpleQuestion (Bordes et al., 2015) and WebQSP (Yih et al., 2016)
- Each question is labeled with the ground-truth semantic parse
- Direct performance evaluation is possible, as well as KBQA evaluation

Experiment:

Single-relation KBQA task.

- Dataset: SimpleQuestion (Bordes et al., 2015) and WebQSP (Yih et al., 2016)
- Each question is labeled with the ground-truth semantic parse
- Direct performance evaluation is possible, as well as KBQA evaluation

Experiment: Relation Detection

Multi-relation KBQA task.

- Dataset: SimpleQuestion (Bordes et al., 2015) and WebQSP (Yih et al., 2016)
- Each question is labeled with the ground-truth semantic parse
- Direct performance evaluation is possible, as well as KBQA evaluation

Experiment: Relation Detection

- Dataset: SimpleQuestion (Bordes et al., 2015) and WebQSP (Yih et al., 2016)
- Each question is labeled with the ground-truth semantic parse
- Direct performance evaluation is possible, as well as KBQA evaluation

Model	Relation Input Views	Accuracy SimpleQuestions	Accuracy WebQSP
AMPCNN (Yin et al., 2016)	words	91.3	-
BiCNN (Yih et al., 2015)	char-3-gram	90.0	77.74
BiLSTM w/ words	words	91.2	79.32
BiLSTM w/ relation names	rel_names	88.9	78.96
Hier-Res-BiLSTM (HR-BiLSTM)	words + rel_names	**93.3**	**82.53**
w/o rel_name	words	91.3	81.69
w/o rel_words	rel_names	88.8	79.68
w/o residual learning (weighted sum on two layers)	words + rel_names	92.5	80.65
replacing residual with attention (Parikh et al., 2016)	words + rel_names	92.6	81.38
single-layer BiLSTM question encoder	words + rel_names	92.8	78.41
replacing BiLSTM with CNN (HR-CNN)	words + rel_names	92.9	79.08

Experiment: KBQA End-Task

- Compared to baseline relation detector, proposed system improves KBQA end-task by 2%-3%

System	Accuracy SQ	Accuracy WQ
STAGG	72.8	**63.9**
AMPCNN (Yin et al., 2016)	*76.4*	-
Baseline: Our Method w/ baseline relation detector	75.1	60.0
Our Method	**77.0**	63.0
w/o entity re-ranking	74.9	60.6
w/o constraints	-	58.0
Our Method (multi-detectors)	**78.7**	**63.9**

Table 3: KBQA results on SimpleQuestions (SQ) and WebQSP (WQ) test sets. The numbers in *green* color are directly comparable to our results since we start with the same entity linking results.

Conclusions

- Proposed a novel KB relation detection Model, HR-BiLSTM, that performs hierarchical matching between questions and KB relations
- Proposed model outperforms the previous methods on KB relation detection
- Proposed proof-of-concept KBQA system achieved state-of-the-arts in both single relation and multiple relation tasks

Conclusions

Strong Points:

- Combined different granularities of KB relationship tokens to capture both local and global information
- Used shortcut connections between BiLSTMs to reduce training difficulty
- Hierarchical architecture learned different levels of abstraction to prevent over-fitting

Weak Points:

- Exact analysis of training difficulty is missing
- Result of ablation test is not strong
- Overall performance improvement is marginal

Thank You!

Questions?