Multi-Task Learning for Conversational Question Answering over a Large-Scale Knowledge Base

Tao Shen[1]*, Xiubo Geng[2], Tao Qin[2], Daya Guo[3], Duyu Tang[2], Nan Duan[2], Guodong Long[1] and Daxin Jiang[2]

[1]Centre for AI, School of Computer Science, FEIT, University of Technology Sydney
[2]Microsoft, Beijing, China
[3]The School of Data and Computer Science, Sun Yat-sen University

tao.shen@student.uts.edu.au, guodong.long@uts.edu.au
{xiubo.geng,taoqin,dutang,nanduan,djiang}@microsoft.com
guody5@mail2.sysu.edu.cn

Abstract

We consider the problem of conversational question answering over a *large-scale* knowledge base. To handle huge entity vocabulary of a large-scale knowledge base, recent neural semantic parsing based approaches usually decompose the task into several subtasks and then solve them sequentially, which leads to following issues: 1) errors in earlier subtasks will be propagated and negatively affect downstream ones; and 2) each subtask cannot naturally share supervision signals with others. To tackle these issues, we propose an innovative multi-task learning framework where a pointer-equipped semantic parsing model is designed to resolve coreference in conversations, and naturally empower joint learning with a novel type-aware entity detection model. The proposed framework thus enables shared supervisions and alleviates the effect of error propagation. Experiments on a large-scale conversational question answering dataset containing 1.6M question answering pairs over 12.8M entities show that the proposed framework improves overall F1 score from 67% to 79% compared with previous state-of-the-art work.

1 Introduction

Recent decades have seen the development of AI-driven personal assistants (e.g., Siri, Alexa, Cortana, and Google Now) that often need to answer factorial questions. Meanwhile, large-scale knowledge base (KB) like DBPedia (Auer et al., 2007) or Freebase (Bollacker et al., 2008) has been built to store world's facts in a structure database, which is used to support open-domain question answering (QA) in those assistants.

Neural semantic parsing based approach (Jia and Liang, 2016; Reddy et al., 2014; Dong and Lapata, 2016; Liang et al., 2016; Dong and Lapata, 2018; Guo et al., 2018) is gaining rising attention for knowledge-based question answer (KB-QA) in recent years since it does not rely on hand-crafted features and is easy to adapt across domains. Traditional approaches usually retrieve answers from a small KB (e.g., small table) (Jia and Liang, 2016; Xiao et al., 2016) and are difficult to handle large-scale KBs. Many recent neural semantic parsing based approaches for KB-QA take a stepwise framework to handle this issue. For example, Liang et al. (2016), Dong and Lapata (2016), and Guo et al. (2018) first use an entity linking system to find entities in a question, and then learn a model to map the question to logical form based on that. Dong and Lapata (2018) decompose the semantic parsing process into two stages. They first generate a rough sketch of logical form based on low-level features, and then fill in missing details by considering both the question and the sketch.

However, these stepwise approaches have two issues. First, errors in upstream subtasks (e.g., entity detection and linking, relation classification) are propagated to downstream ones (e.g., semantic parsing), resulting in accumulated errors. For example, case studies in previous works (Yih et al., 2015; Dong and Lapata, 2016; Xu et al., 2016; Guo et al., 2018) show that entity linking error is one of the major errors leading to wrong predictions in KB-QA. Second, since models for the subtasks are learned independently, the supervision signals cannot be shared among the models for mutual benefits.

To tackle issues mentioned above, we propose a novel multi-task semantic parsing framework for KB-QA. Specifically, an innovative pointer-equipped semantic parsing model is first designed for two purposes: 1) built-in pointer network toward positions of entity mentions in the question

* Work done while the author was an intern at Microsoft, Beijing, China.

Alias	Operator	Comments
A1/2/3	$start \rightarrow set/num/bool$	
A4	$set \rightarrow \text{find}(set, p)$	set of entities with a predicate p edge to entity e
A5	$num \rightarrow \text{count}(set)$	number of distinct elements in the input set
A6	$bool \rightarrow \text{in}(e, set)$	whether the entity e in set or not
A7	$set \rightarrow \text{union}(set_1, set_2)$	$set_1 \cup set_2$
A8	$set \rightarrow \text{inter}(set_1, set_2)$	$set_1 \cap set_2$
A9	$set \rightarrow \text{diff}(set_1, set_2)$	set_1 - set_2
A10	$set \rightarrow \text{large}(set, p, num)$	subset of set linking to more than num entities with predicate p
A11	$set \rightarrow \text{less}(set, p, num)$	subset of set linking to less than num entities with predicate p
A12	$set \rightarrow \text{equal}(set, p, num)$	subset of set linking to num entities with predicate p
A13	$set \rightarrow \text{argmax}(set, p)$	subset of set linking to most entities with predicate p
A14	$set \rightarrow \text{argmin}(set, p)$	subset of set linking to least entities with predicate p
A15	$set \rightarrow \text{filter}(tp, set)$	subset where entity e in set and belong to entity type tp
A16	$num \rightarrow u\_num$	transform number in utterance u_num to intermediate number num
A17	$set \rightarrow \text{set}(e)$	
A18/19/20/21	$e/p/tp/u\_num \rightarrow \text{constant}$	*instantiation for e, p, tp, u_num

Table 1: Brief grammar definitions for logical form generation. *instantiation of entity e, predicate p, type tp, number-in-question u_num, by corresponding constant parsed from the question.

can naturally empower multi-task learning with conjunction of upstream sequence labeling subtask, i.e., entity detection; and 2) it explicitly takes into account the context of entity mentions by using the supervision of the pointer network. Besides, a type-aware entity detection method is proposed to produce accurate entity linking results, in which, a joint prediction space combining entity detection and entity type is employed, and the predicted type is then used to filter entity linking results during inference phase.

The proposed framework has certain merits.

First, since the two subtasks, i.e., pointer-equipped semantic parsing and entity detection, are closely related, learning them within a single model simultaneously makes the best of supervisions and improves performance of KB-QA task.

Second, considering entity type prediction is crucial for entity linking, our joint learning framework combining entity mention detection with type prediction leverages contextual information, and thus further reduces errors in entity linking.

Third, our approach is naturally beneficial to coreference resolution for conversational QA due to rich contextual features captured for entity mention, compared to previous works directly employing low-level features (e.g., mean-pooling over word embeddings) as the representation of an entity. This is verified via our experiments in §4.2.

We evaluate the proposed framework on the CSQA (Saha et al., 2018) dataset, which is the largest public dataset for complex conversational question answering over a large-scale knowledge base. Experimental results show that the overall F1 score is improved by 12.56% compared with strong baselines, and the improvements are consistent for all question types in the dataset.

2 Task Definition

In this work, we target the problem of conversational question answering over a large-scale knowledge base. Formally, in training data, question U denotes an user utterance from a dialog, which is concatenated dialog history for handling ellipsis or coreference in conversations, and the question is labeled with its answer A. Besides, "IOB" (Insider-Outside-Beginning) tagging and entities linking to KB are also labeled for entity mentions in U to train an entity detection model.

We employ a neural semantic parsing based approach to tackle the problem. That is, given a question, a semantic parsing model is used to produce a logical form which is then executed on the KB to retrieve an answer. We decompose the approach into two subtasks, i.e., entity detection for entity linking and semantic parsing for logical form generation. The former employs IOB tagging and corresponding entities as supervision, while the latter uses a gold logical form as supervision, which may be obtained by conducting intensive BFS[1] over KB if only final answers (i.e., weak supervision) are provided.

3 Approach

This section begins with a description of grammars and logic forms used in this work. Then, the proposed model is presented, and finally, model's training and inference are introduced.

[1]Breadth-first search with limited buffer (Guo et al., 2018)

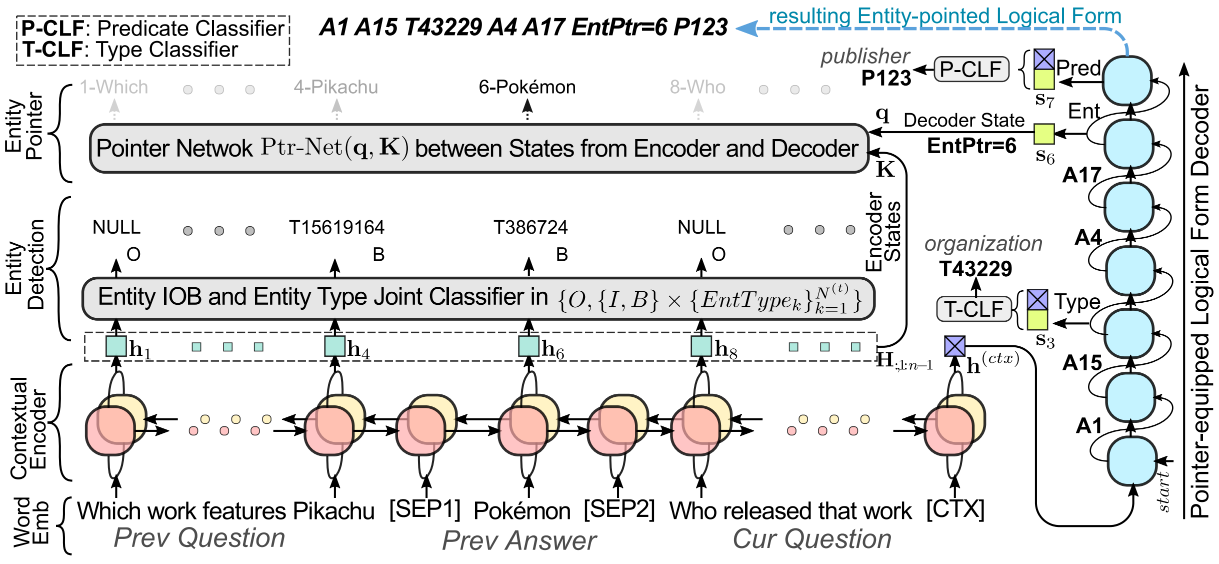
P-CLF: Predicate Classifier
T-CLF: Type Classifier
A1 A15 T43229 A4 A17 EntPtr=6 P123
resulting Entity-pointed Logical Form
publisher
P123
Pred
Ent
Decoder State
EntPtr=6
organization
T43229
Type
Entity Pointer
Pointer Netwok Ptr-Net(q, K) between States from Encoder and Decoder
1-Which
4-Pikachu
6-Pokémon
8-Who
Entity Detection
Entity IOB and Entity Type Joint Classifier in {O, {I, B} × {EntType_k}_{k=1}^{N^(t)}}
NULL
T15619164
T386724
Encoder States
Contextual Encoder
Word Emb
Which work features Pikachu [SEP1] Pokémon [SEP2] Who released that work [CTX]
Prev Question
Prev Answer
Cur Question
start
Pointer-equipped Logical Form Decoder

Figure 1: Proposed **M**ulti-t**a**sk **S**emantic **P**arsing (MaSP) model. Note that P* and T* are predicate and entity type ids in Wikidata where entity type id originally starts with Q but is replaced with T for clear demonstration.

3.1 Grammar and Logical Form

Grammar We use similar grammars and logical forms as defined in Guo et al. (2018), with minor modification for better adaptation to the CSQA dataset. The grammars are briefly summarized in Table 1, where each operator consists of three components: semantic category, a function name, and a list of arguments with specified semantic categories. Semantic categories can be classified into two groups here w.r.t. the ways for instantiation: one is referred to as *entry* semantic category (i.e., $\{e, p, tp, u\_num\}$ for entities, predicates, types, numbers) whose instantiations are constants parsed from a question, and another is referred to as *intermediate* semantic category (i.e., $\{start, set, num, bool\}$) whose instantiation is the output of an operator execution.

Logical Form A KB-executable logical form is intrinsically formatted as an ordered tree where the root is the semantic category $start$, each child node is constrained by the nonterminal (i.e., the un-instantiated semantic category in parenthesis) of its parent operator, and leaf nodes are instantiated entry semantic categories, i.e., constants.

To make the best of well-performed sequence to sequence (seq2seq) models (Vaswani et al., 2017; Bahdanau et al., 2015) as a base for semantic parsing, we represent a tree-structured logical form as a sequence of operators and constants via depth-first traversal over the tree. Note, given guidance of grammars, we can recover corresponding tree structure from a sequence-formatted logical form.

3.2 Proposed Model

The structure of our proposed **M**ulti-t**a**sk **S**mantic **P**arsing (MaSP) model is illustrated in Figure 1. The model consists of four components: i.e., word embedding, contextual encoder, entity detection and pointer-equipped logical form decoder.

3.2.1 Embedding and Contextual Encoder

To handle ellipsis or coreference in conversations, our model takes current user question combined with dialog history as the input question U. In particular, all those sentences are concatenated with a *[SEP]* separated, and then a special token *[CTX]* is appended. We apply wordpiece tokenizing (Wu et al., 2016) method, and then use a word embedding method (Mikolov et al., 2013) to transform the tokenized question to a sequence of low-dimension distributed embeddings, i.e., $\boldsymbol{X} = [\boldsymbol{x_1}, \cdots, \boldsymbol{x_n}] \in \mathbb{R}^{d_e \times n}$ where d_e denotes embedding size and n denotes question length.

Given word embeddings $\boldsymbol{X}$, we use stacked two-layer multi-head attention mechanism in the Transformer (Vaswani et al., 2017) with learnable positional encodings as an encoder to model contextual dependencies between tokens, which results in context-aware representations $\boldsymbol{H} = [\boldsymbol{h_1}, \cdots, \boldsymbol{h_n}] \in \mathbb{R}^{d_e \times n}$. And, contextual embedding for token *[CTX]* is used as the semantic representation for entire question, i.e., $\boldsymbol{h^{(ctx)}} \triangleq \boldsymbol{h_n}$

3.2.2 Pointer-Equipped Decoder

Given contextual embeddings $\boldsymbol{H}$ of a question, we employ stacked two-layer masked attention mechanism in (Vaswani et al., 2017) as the decoder to produce sequence-formatted logical forms.

In each decoding step, the model first predicts a token from a small decoding vocabulary $\mathbb{V}^{(dec)}$ = $\{start, end, e, p, tp, u\_num, \text{A1}, \cdots, \text{A21}\}$, where $start$ and end indicate the start and end of decoding, $A1, \cdots, A21$ are defined in Table 1, and e, p, tp and u_num denote entity, predicate, type and number entries respectively. A neural classifier is established to predict current decoding token, which is formally denoted as

$$\boldsymbol{p_j^{(tk)}}=\text{softmax}(\text{FFN}(\boldsymbol{s_j};\theta^{(tk)})), \qquad (1)$$

where $\boldsymbol{s_j}$ is decoding hidden state of current (i.e., j-th) step, $\text{FFN}(\cdot;\theta)$ denotes a θ-parameterized two-layer feed forward network with an activation function inside, and $\boldsymbol{p_j^{(tk)}}\in\mathbb{R}^{|\mathbb{V}^{(dec)}|}$ is a predicted distribution over $\mathbb{V}^{(dec)}$ to score candidates[2].

Then, a $\text{FFN}(\cdot)$ or a pointer network (Vinyals et al., 2015) is utilized to predict instantiation for entry semantic category (i.e., e, p, tp or u_num in $\mathbb{V}^{(vec)}$) if it is necessary.

- For predicate p and type tp, two parameter-untied $\text{FFN}(\cdot)$ are used as

$$\boldsymbol{p_j^{(p)}}=\text{softmax}(\text{FFN}([\boldsymbol{s_j};\boldsymbol{h^{(ctx)}}];\theta^{(p)})), \quad (2)$$

$$\boldsymbol{p_j^{(t)}}=\text{softmax}(\text{FFN}([\boldsymbol{s_j};\boldsymbol{h^{(ctx)}}];\theta^{(t)})), \quad (3)$$

  where $\boldsymbol{h^{(ctx)}}$ is semantic embedding of entire question, $\boldsymbol{s_j}$ is current hidden state, $\boldsymbol{p_j^{(p)}} \in \mathbb{R}^{N^{(p)}}$ and $\boldsymbol{p_j^{(t)}} \in \mathbb{R}^{N^{(t)}}$ are predicted distributions over the predicate and type instantiation candidates respectively, and $N^{(p)}$ and $N^{(t)}$ are the numbers of distinct predicates and types in the knowledge base.

- For entity e and number u_num, two parameter-untied pointer-networks (Vinyals et al., 2015) with learnable bilinear layer are employed to point toward the targeted entity[3] and number, which are defined as follows.

$$\boldsymbol{p_j^{(e)}} = \text{softmax}(\boldsymbol{s_j^T W^{(e)} H}_{:,1:n-1}), \quad (4)$$

$$\boldsymbol{p_j^{(n)}} = \text{softmax}(\boldsymbol{s_j^T W^{(n)} H}_{:,1:n-1}), \quad (5)$$

  where $\boldsymbol{H}_{:,1:n-1}$ is contextual embedding of tokens in the question except *[CTX]*, $\boldsymbol{W^{(e)}}$ and $\boldsymbol{W^{(n)}}$ are weights of pointer-network for entity and number, $\boldsymbol{p_j^{(e)}}, \boldsymbol{p_j^{(n)}} \in \mathbb{R}^{n-1}$ are the resulting distributions over positions of input question, and n is the length of the question.

[2]Superscript in bracket denotes the type instead of index.
[3]Toward the first one if entity consists of multiple words.

The pointer network is also used for semantic parsing in (Jia and Liang, 2016), where the pointer aims at copying out-of-vocabulary words from a question over small-scale KB. Different from that, the pointer used here aims at locating the targeted entity and number in a question, which has two advantages. First, it handles the coreference problem by considering the context of entity mentions in the question. Second, it solves the problem caused by huge entity vocabulary, which reduces the size of decoding vocabulary from several million (i.e., the number of entities in KB) to several dozen (i.e., the length of the question).

3.2.3 Entity Detection and Linking

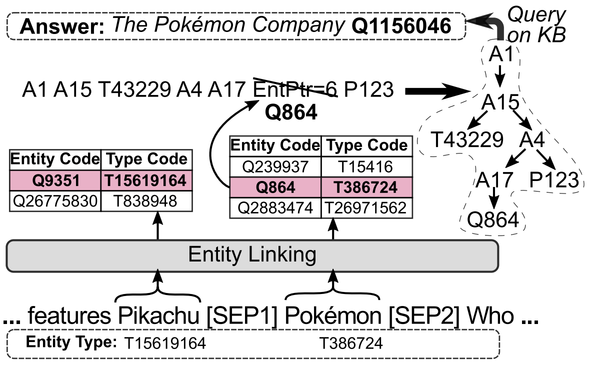

Figure 2: Transformation from entity-pointed logical form to KB-executable logical form for KB querying.

To map the pointed positions to entities in KB, our model also detects entity mentions for the input question, as shown as the "Entity Detection" part of Figure 1.

We observe that multiple entities in a large-scale KB usually have same entity text but different types, leading to named entity ambiguity. Therefore, we design a novel type-aware entity detection module in which the prediction is fulfilled in a joint space of IOB tagging and corresponding entity type for disambiguation. Particularly, the prediction space is defined as $\mathbb{E} = \{O, \{I, B\} \times \{ET_k\}_{k=1}^{N^{(t)}}\}$ where ET_k stands for the k-th entity type label, $N^{(t)}$ denotes number of distinct entity types in KB, and $|\mathbb{E}| = 2 \times N^{(t)} + 1$.

The prediction for both entity IOB tagging and entity type is formulated as

$$\boldsymbol{p}_{\boldsymbol{i}}^{\boldsymbol{(ed)}} = \mathrm{softmax}(\mathrm{FFN}(\boldsymbol{h}_{\boldsymbol{i}}; \theta^{(ed)})), \forall i \in [1, n-1] \quad (6)$$

where $\boldsymbol{h_i}$ is the contextual embedding of the i-th token in the question, and $\boldsymbol{p}_{\boldsymbol{i}}^{\boldsymbol{(ed)}} \in \mathbb{R}^{|\mathbb{E}|}$ is the predicted distribution over $\mathbb{E}$.

Given the predicted IOB labels and entity types, we take the following steps for entity linking. First, the predicted IOB labels are used to locate all entities in the question and return corresponding entity mentions. Second, an inverted index built on the KB is leveraged to find entity candidates in KB based on each entity mention. Third, the jointly predicted entity types are used to filter out the candidates with unwanted types, and the remaining entity with the highest inverted index score is selected to substitute the pointer. This process is shown as the bottom part of Figure 2.

During inference phase, the final logical form is derived by replacing entity pointers in entity-pointed logical form from §3.2.2 with entity linking results, and is then executed on the KB to retrieve an answer for the question, as shown as the top part of Figure 2.

3.3 Learning and Inference

Model Learning During the training phase, we first search gold logical forms for questions in training data over KB if only weak supervision is provided. Then we conduct multi-task learning for semantic parsing and entity detection. The final loss is defined as

$$L = \alpha L^{(sp)} + L^{(ed)}, \quad (7)$$

where $\alpha > 0$ is a hyperparameter for a tradeoff between semantic parsing and entity detection, and $L^{(sp)}$ and $L^{(ed)}$ are negative log-likelihood losses of semantic parsing and entity detection defined as follows.

$$L^{(sp)} = -\frac{1}{|\mathcal{D}|} \sum_{\mathcal{D}} \frac{1}{m} \sum_{j=1}^{m} \log \boldsymbol{p}_{\boldsymbol{j}}^{\boldsymbol{(tk)}}{}_{[tk'=y_j^{(tk)}]} \quad (8)$$
$$+ \sum_{c \in \{p,t,e,n\}} I_{(y_j^{(tk)}=c)} \log \boldsymbol{p}_{\boldsymbol{j}}^{\boldsymbol{(c)}}{}_{[c'=y_j^{(c)}]}$$

$$L^{(ed)} = -\frac{1}{|\mathcal{D}|} \sum_{\mathcal{D}} \frac{1}{n-1} \sum_{i=1}^{n-1} \log \boldsymbol{p}_{\boldsymbol{i}}^{\boldsymbol{(ed)}}{}_{[ed'=y_i^{(ed)}]} \quad (9)$$

In the two equations above, $y_j^{(tk)}$ is gold label for decoding token in $\mathbb{V}^{(dec)}$; $y_j^{(p)}, y_j^{(t)}, y_j^{(e)}$ and $y_j^{(n)}$ are gold labels for predicate, type, entity position and number position for instantiation; $\boldsymbol{p}_{\boldsymbol{j}}^{\boldsymbol{(tk)}}$, $[\boldsymbol{p}_{\boldsymbol{j}}^{\boldsymbol{(c)}}]_{c \in \{p,t,e,n\}}$, and $\boldsymbol{p}_{\boldsymbol{j}}^{\boldsymbol{(ed)}}$ are defined in Eq.(1-6) respectively; and m denotes the decoding length.

Here, we use a single model to handle two subtasks simultaneously, i.e., semantic parsing and entity detection. This multi-task learning framework enables each subtask to leverage supervision signals from the others, and thus improves the final performance for KB-QA.

Grammar-Guided Inference The grammars defined in Table 1 are utilized to filter illegal operators out in each decoding step. An operator is legitimate if its left-hand semantic category in the definition is identical to the leftmost nonterminal (i.e., un-instantiated semantic category) in the incomplete logical form parsed so far. In particular, the decoding of a logical form begins with the semantic category $start$. During decoding, the proposed semantic parsing model recursively rewrites the leftmost nonterminal in the logical form by 1) applying a legitimate operator for an intermediate semantic category, or 2) instantiation for one of entity, predicate, type or number for an entry semantic category. The decoding process for the parsing terminates until no nonterminals remain.

Furthermore, beam search is also incorporated to boost the performance of the proposed model during the decoding. And, the early stage execution is performed to filter out illegal logical forms that lead to empty intermediate result.

4 Experiments

4.1 Experimental Settings

Dataset We evaluated the proposed approach on Complex Sequential Question Answering (CSQA) dataset[4] (Saha et al., 2018), which is the largest dataset for conversational question answering over large-scale KB. It consists of about 1.6M question-answer pairs in ∼200K dialogs, where 152K/16K/28K dialogs are used for train/dev/test. Questions are classified as different types, e.g., simple, comparative reasoning, logical reasoning questions. Its KB is built on Wikidata[5] in a form of (subject, predicate, object), and consists of 21.2M triplets over 12.8M entities, 3,054 distinct entity types, and 567 distinct predicates.

[4]https://amritasaha1812.github.io/CSQA
[5]https://www.wikidata.org

Methods		HRED+KVmem	D2A (Baseline)	MaSP (Ours)	Δ
Question Type	**#Example**	F1 Score	F1 Score	F1 Score	
Overall	203k	9.39%	66.70%	79.26%	+12.56%
Clarification	9k	16.35%	35.53%	80.79%	+45.26%
Comparative Reasoning (All)	15k	2.96%	48.85%	68.90%	+20.05%
Logical Reasoning (All)	22k	8.33%	67.31%	69.04%	+1.73%
Quantitative Reasoning (All)	9k	0.96%	56.41%	73.75%	+17.34%
Simple Question (Coreferenced)	55k	7.26%	57.69%	76.47%	+18.78%
Simple Question (Direct)	82k	13.64%	78.42%	85.18%	+6.76%
Simple Question (Ellipsis)	10k	9.95%	81.14%	83.73%	+2.59%
Question Type	**#Example**	Accuracy	Accuracy	Accuracy	
Verification (Boolean)	27k	21.04%	45.05%	60.63%	+15.58%
Quantitative Reasoning (Count)	24k	12.13%	40.94%	43.39%	+2.45%
Comparative Reasoning (Count)	15k	8.67%	17.78%	22.26%	+4.48%

Table 2: Comparisons with baselines on CSQA. The last column consists of differences between MaSP and D2A.

Training Setups We leveraged a BFS method to search valid logical forms for questions in training data. The buffer size in BFS is set to 1000. Both embedding and hidden sizes in the model are set to $300D$, and no pretrained embeddings are loaded for initialization, and the positional encodings are randomly initialized and learnable. The head number of multi-head attention is 6 and activation function inside $\mathrm{FFN}(\cdot)$ is $\mathrm{Gelu}(\cdot)$ (Hendrycks and Gimpel, 2016). We used Adam (Kingma and Ba, 2015) to optimize the loss function defined in Eq.(7) where α is set to 1.5, and learning rate is set to 10^{-4}. The training batch size is 128 for 6 epochs. And we also employed learning rate warmup within the first 1% steps and linear decay within the rest. The source codes are available at `https://github.com/taoshen58/MaSP`.

Evaluation Metrics We used the same evaluation metrics as Saha et al. (2018) and Guo et al. (2018). F1 score (i.e., precision and recall) is used to evaluate the question whose answer is comprised of entities, and accuracy is used to measure the question whose answer type is boolean or number.

Baselines There are few works targeting conversational question answering over a large-scale knowledge base. HRED+KVmem (Saha et al., 2018) and D2A (Guo et al., 2018) are two typical approaches, and we compared them with our proposed approach. Particularly, HRED+KVmem is a memory network (Sukhbaatar et al., 2015; Li et al., 2017) based seq2seq model, which combines HRED model (Serban et al., 2016) with key-value memory network (Miller et al., 2016). D2A[6] is a memory augmented neural symbolic model for semantic parsing in KB-QA, which introduces dialog memory manager to handle ellipsis and coreference problems in conversations.

4.2 Model Comparisons

We compared our approach (denoted as MaSP) with HRED+KVmem and D2A in Table 2. As shown in the table, the semantic parsing based D2A significantly outperforms the memory network based text generation approach (HRED+KVmem), which thus poses a strong baseline. Further, our proposed approach (MaSP) achieves a new state-of-the-art performance, where the overall F1 score is improved by $\sim$12%. Besides, the improvement is consistent for all question types, which ranges from 2% to 25%.

There are two possible reasons for this significant improvement. First, our approach predicts entities more accurately, where the accuracy of entities in final logical forms increases from 55% to 72% compared with D2A. Second, the proposed pointer-equipped logical form decoder in the multi-task learning framework handles coreference better. For instance, given an user question with history, "*What is the parent organization of that one? // Did you mean Polydor Records ? // No, I meant Deram Records. Could you tell me the answer for that?*" with coreference, D2A produces "(*find* {*Polydor Records*}, *owned by*)" and in contrast our approach produces "(*find* {*Deram Records*}, *owned by*)". This also explains the substantial improvement for *Simple Question (Coreferenced)* and *Clarification*[7].

[6]Overall score of D2A reported in this paper is superior to that in the original paper since our re-implemented grammars for CSQA achieve a better balance between the simple and non-simple question types. For rational and fair comparisons, we report re-run results for D2A in this paper.

[7]In CSQA, the performance of *Clarification* closely depends on F1 score for next question, 88% of which belong to

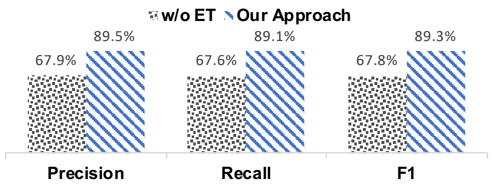

Figure 3: Performance of entity linking. "w/o ET" denotes removing entity type filtering.

We also observed that the improvement of MaSP over D2A for some question types is relatively small, e.g., 1.73% for logical reasoning questions. A possible reason is that there are usually more than one entities are needed to compose the correct logical form for logical reasoning questions, and our current model is too shallow to parse the multiple entities. Hence, we adopted deeper model and employed BERT (Devlin et al., 2018) as the encoder (latter in §4.4), and found that the performance of logical reasoning questions is improved by 10% compared to D2A.

4.3 Ablation Study

Methods	Ours	w/o ET	w/o Multi	w/o Both
Question Type	F1	F1	F1	F1
Overall	79.26%	70.42%	76.73%	68.22%
Clarification	80.79%	68.01%	66.30%	54.64%
Comparative	68.90%	66.35%	61.12%	58.04%
Logical	69.04%	62.63%	67.81%	62.51%
Quantitative	73.75%	73.75%	64.56%	64.55%
Simple (Co-ref)	76.47%	64.94%	74.35%	63.15%
Simple (Direct)	85.18%	75.24%	84.93%	75.19%
Simple (Ellipsis)	83.73%	78.45%	82.66%	77.44%
Question Type	Accu	Accu	Accu	Accu
Verification	60.63%	45.40%	60.43%	45.02%
Quantitative	43.39%	39.70%	37.84%	43.39%
Comparative	22.26%	19.08%	18.24%	22.26%

Table 3: Ablation study. "w/o ET" stands for removing entity type prediction in Entity Detection of §3.2.3; "w/o Multi" stands for learning two subtasks separately in our framework; and "w/o Both" stands for a combination of "w/o ET" and "w/o Multi".

There are two aspects leading to performance improvement, i.e., predicting entity type in entity detection to filter candidates, and multi-task learning framework. We conducted an ablation study in Table 3 for in-depth understanding of their effects.

Effect of Entity Type Prediction (w/o ET) First, the entity type prediction was removed from the entity detection task, which results in 9% drop of overall F1 score. We argue that the performance of the KB-QA task is in line with that of entity linking. Hence, we separately evaluated the entity linking task on the test set. As illustrated in Figure 3, both precision and recall of entity linking drop significantly without filtering the entity linking results w.r.t. the predicted entity type, which verifies our hypothesis above.

Simple Question(Coreference) .

Accuracy	Ours	w/o Multi
Entity pointer	79.8%	79.3%
Predicate	96.9%	96.3%
Type	86.8%	84.1%
Number	89.1%	88.3%
Operators	79.4%	78.7%

Table 4: Prediction accuracy on each component composing the pointer-equipped logical form.

Effect of Multi-Task Learning (w/o Multi) Second, to measure the effect of multi-task learning, we evaluated the KB-QA task when the two subtasks, i.e., pointer-equipped semantic parsing and entity detection, are learned separately. As shown in Table 3, the F1 score for every question type consistently drops in the range of 3% to 14% compared with that with multi-task learning. We further evaluated the effect of multi-task learning on each subtask. As shown in Table 4, the accuracy for each component of the pointer-equipped logical form drops with separate learning. Meanwhile, we found 0.1% F1 score reduction (99.4% vs. 99.5%) for entity detection subtask compared to the model without multi-task learning, which only poses a negligible effect on the downstream task. To sum up, the multi-task learning framework increases the accuracy of the pointer-based logical form generation while keeping a satisfactory performance of entity detection, and consequently improves the final question answering performance.

Note that, considering a combination of removing the entity type filter and learning two subtasks separately (i.e., w/o Both in Table 3), the proposed framework will degenerate to a model that is similar to Coarse-to-Fine semantic parsing model, another state-of-the-art KB-QA model over small-scale KB (Dong and Lapata, 2018). Therefore, an improvement of 11% of F1 score also verifies the advantage of our proposed framework.

4.4 Model Setting Analysis

As introduced in §4.1 and evaluated in §4.2, the proposed framework is built on a relatively shal-

Methods	Vanilla	w/ BERT	w/ Large Beam
Question Type	F1	F1	F1
Overall	79.26%	80.60%	81.55%
Clarification	80.79%	79.46%	83.37%
Comparative	68.90%	65.99%	69.34%
Logical	69.04%	77.53%	69.41%
Quantitative	73.75%	70.43%	73.75%
Simple (Co-ref)	76.47%	77.95%	79.03%
Simple (Direct)	85.18%	86.40%	88.28%
Simple (Ellipsis)	83.73%	84.82%	86.96%
Question Type	Accuracy	Accuracy	Accuracy
Verification	60.63%	63.85%	61.96%
Quantitative	43.39%	47.14%	44.22%
Comparative	22.26%	25.28%	22.70%

Table 5: Comparisons with different experimental settings. "Vanilla" stands for standard settings of our framework, i.e, MaSP. "w/ BERT" stands for incorporating BERT. And "w/ Large Beam" stands for increasing beam search size from 4 to 8.

low neural network, i.e., stacked two-layer multi-head attention, which might limit its representative ability. Hence, in this section, we further exploited the performance of the proposed framework by applying more sophisticated strategies.

As shown in Table 5, we first replaced the encoder with pre-trained BERT base model (Devlin et al., 2018) and fine-tuned parameters during the training phase, which results in 1.3% F1 score improvement over the vanilla one. Second, we increased beam search size from 4 to 8 during the decoding in the inference phase for the standard settings, which leads to 2.3% F1 score increase.

4.5 Error Analysis

We randomly sampled 100 examples with wrong logical forms or incorrect answers to conduct an error analysis, and found that the errors mainly fall into the following categories.

Entity Ambiguity Leveraging entity type as a filter in entity linking significantly reduces errors caused by entity ambiguity, but it is still possible that different entities with same text belong to the same type, due to coarse granularity of the entity type, which results in filtering invalidity. For example, it is difficult to distinguish between two persons whose names are both *Bill Woods*.

Wrong Predicted Logical Form The predicted components (e.g., operators, predicates and types) composing the logical form would be inaccurate, leading to a wrong answer to the question or an un-executable logical form.

Spurious Logical Form We took a BFS method to search gold logical forms for questions in training set, which inevitably generates spurious (incorrect but leading to correct answers coincidentally) logical forms as training signals. Take the question "*Which sexes do King Harold, Queen Lillian and Arthur Pendragon possess*" as an example, a spurious logical form only retrieves the genders of "*King Harold*" and "*Queen Lillian*", while it gets correct answers for the question. Spurious logical forms accidentally introduce noises into training data and thus negatively affect the performance of KB-QA.

5 Related Work

Our work is aligned with semantic parsing based approach for KB-QA. Traditional semantic parsing systems typically learn a lexicon-based parser and a scoring model to construct a logical form given a natural language question (Zettlemoyer and Collins, 2007; Wong and Mooney, 2007; Zettlemoyer and Collins, 2009; Kwiatkowski et al., 2011; Andreas et al., 2013; Artzi and Zettlemoyer, 2013; Zhao and Huang, 2014; Long et al., 2016). For example, Zettlemoyer and Collins (2009) and Artzi and Zettlemoyer (2013) learn a CCG parser, and Long et al. (2016) develop a shift-reduce parser to construct logical forms.

Neural semantic parsing approaches have been gaining rising attention in recent years, eschewing the need for extensive feature engineering (Jia and Liang, 2016; Ling et al., 2016; Xiao et al., 2016). Some efforts have been made to utilize the syntax of logical forms (Rabinovich et al., 2017; Krishnamurthy et al., 2017; Cheng et al., 2017; Yin and Neubig, 2017). For example, Dong and Lapata (2016) and Alvarez-Melis and Jaakkola (2017) leverage an attention-based encoder-decoder framework to translate a natural language question to tree-structured logical form.

Recently, to handle huge entity vocabulary existing in a large-scale knowledge base, many works take a stepwise approach. For example, Liang et al. (2016), Dong and Lapata (2016), and Guo et al. (2018) first process questions using a name entity linking system to find entity candidates, and then learn a model to map a question to a logical form based on the candidates. Dong and Lapata (2018) decompose the task into two stages: first, a sketch of the logical form is predicted, and then a full logical form is generated with considering both the question and the predicted sketch.

Our proposed framework also decomposes the task into multiple subtasks but is different from existing works in several aspects. First, inspired by pointer network (Vinyals et al., 2015), we replace entities in a logical form with the starting positions of their mentions in the question, which can be naturally used to handle coreference problem in conversations. Second, the proposed pointer-based semantic parsing model can be intrinsically extended to jointly learn with entity detection for fully leveraging all supervision signals. Third, we alleviate entity ambiguity problem in entity detection & linking subtask, by incorporating entity type prediction into entity mention IOB labeling to filter out the entities with unwanted types.

6 Conclusion

We studied the problem of conversational question answering over a *large-scale* knowledge base, and proposed a multi-task learning framework which learns for type-aware entity detection and pointer-equipped logical form generation simultaneously. The multi-task learning framework takes full advantage of the supervisions from all subtasks, and consequently increases the performance of final KB-QA problem. Experimental results on a large-scale dataset verify the effectiveness of the proposed framework. In the future, we will test our proposed framework on more datasets and investigate potential approaches to handle spurious logical forms for weakly-supervised KB-QA.

Acknowledgments

We acknowledge the support of NVIDIA Corporation and MakeMagic Australia with the donation of GPUs for our research group at University of Technology Sydney. And we also thank anonymous reviewers for their insightful and constructive suggestions.

References

David Alvarez-Melis and Tommi S Jaakkola. 2017. Tree-structured decoding with doubly-recurrent neural networks. In *ICLR*.

Jacob Andreas, Andreas Vlachos, and Stephen Clark. 2013. Semantic parsing as machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 47–52.

Yoav Artzi and Luke Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, 1:49–62.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

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. AcM.

Jianpeng Cheng, Siva Reddy, Vijay Saraswat, and Mirella Lapata. 2017. Learning structured natural language representations for semantic parsing. *arXiv preprint arXiv:1704.08387*.

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

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 731–742. Association for Computational Linguistics.

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 2946–2955.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

R. Jia and P. Liang. 2016. Data recombination for neural semantic parsing. In *Association for Computational Linguistics (ACL)*.

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

Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. 2017. Neural semantic parsing with type constraints for semi-structured tables. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1516–1526.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2011. Lexical generalization in ccg grammar induction for semantic parsing. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1512–1523. Association for Computational Linguistics.

Zheng Li, Yun Zhang, Ying Wei, Yuxiang Wu, and Qiang Yang. 2017. End-to-end adversarial memory network for cross-domain sentiment classification. In *IJCAI*, pages 2237–2243.

Chen Liang, Jonathan Berant, Quoc Le, Kenneth D Forbus, and Ni Lao. 2016. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. *arXiv preprint arXiv:1611.00020*.

Wang Ling, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, Andrew Senior, Fumin Wang, and Phil Blunsom. 2016. Latent predictor networks for code generation. *arXiv preprint arXiv:1603.06744*.

Reginald Long, Panupong Pasupat, and Percy Liang. 2016. Simpler context-dependent logical forms via model projections. *arXiv preprint arXiv:1606.05378*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*.

Maxim Rabinovich, Mitchell Stern, and Dan Klein. 2017. Abstract syntax networks for code generation and semantic parsing. *arXiv preprint arXiv:1704.07535*.

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.

Amrita Saha, Vardaan Pahuja, Mitesh M Khapra, Karthik Sankaranarayanan, and Sarath Chandar. 2018. Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *NIPS*.

Ashish Vaswani, Shazeer, Noam, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Yuk Wah Wong and Raymond Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 960–967.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Chunyang Xiao, Marc Dymetman, and Claire Gardent. 2016. Sequence-based structured prediction for semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1341–1350.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question answering on freebase via relation extraction and textual evidence. *arXiv preprint arXiv:1603.00957*.

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *ACL-IJCNLP*.

Pengcheng Yin and Graham Neubig. 2017. A syntactic neural model for general-purpose code generation. *arXiv preprint arXiv:1704.01696*.

Luke Zettlemoyer and Michael Collins. 2007. Online learning of relaxed ccg grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Luke S Zettlemoyer and Michael Collins. 2009. Learning context-dependent mappings from sentences to logical form. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 976–984. Association for Computational Linguistics.

Kai Zhao and Liang Huang. 2014. Type-driven incremental semantic parsing with polymorphism. *arXiv preprint arXiv:1411.5379*.