Learning to Align Question and Answer Utterances in Customer Service Conversation with Recurrent Pointer Networks

Shizhu He,[1] **Kang Liu,**[1,2] **Weiting An**[3]
[1]National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
[3]Alibaba Group, Hangzhou, China
{shizhu.he, kliu}@nlpr.ia.ac.cn weiting.awt@alibaba-inc.com

Abstract

Customers ask questions, and customer service staffs answer those questions. It is the basic service manner of customer service (CS). The progress of CS is a typical multi-round conversation. However, there are no explicit corresponding relations among conversational utterances. This paper focuses on obtaining explicit alignments of question and answer utterances in CS. It not only is an important task of dialogue analysis, but also able to obtain lots of valuable train data for learning dialogue systems. In this work, we propose end-to-end models for aligning question (Q) and answer (A) utterances in CS conversation with recurrent pointer networks (RPN). On the one hand, RPN-based alignment models are able to model the conversational contexts and the mutual influence of different Q-A alignments. On the other hand, they are able to address the issue of empty and multiple alignments for some utterances in a unified manner. We construct a dataset from an in-house online CS. The experimental results demonstrate that the proposed models are effective to learn the alignments of question and answer utterances.

Introduction

Customer service (**CS**) provides an irreplaceable platform for sellers to answer the questions and solve the problems of customers. In general, in the practical CS (e.g., CS of e-commerce website), **Customer**s pose questions that will be answered by customer service staffs (hereinafter called **Server**), and the communication between a customer and a server is a typical multi-round conversation. In fact, as shown in Figure 1, there are certain corresponding relations among conversation utterances raised by different participants (e.g., customer and server). It is because each server utterance is replying to some of previously customer utterances with consulting intention. Actually, there are no explicit relations among those conversational utterances.

In this paper, we focus on the utterance alignment (**UA**), which is to align the question (**Q**) and answer (**A**) utterances in the multi-round conversation between customer and server. Finding the alignments among utterances with different participants is very important in conversation analysis. Based on the alignments, the questions posed by a customer are connected with the corresponding responses from the server. It could benefit the server to analyze the consulting hot issues and perform quality control (e.g., check whether the customer is competent or not). Moreover, most intelligent dialogue systems such as deep learning methods (Ji, Lu, and Li 2014; Vinyals and Le 2015; He et al. 2017; Cui et al. 2017) are starving of the high-quality "aligned" question-answer (**Q-A**) pairs. Obviously, such Q-A pairs could be automatically acquired through UA. In fact, there are already some related work on utterance alignment in the multi-round conversation. Discourse parsing aims at finding the discourse structure in multi-party chat dialogues (Elsner and Charniak 2011; Afantenos et al. 2015; Jiang et al. 2018) or technical web forums (Wang et al. 2011). (Du, Poupart, and Xu 2017) devotes to discovering the dependencies of conversational messages. And (Jiang et al. 2018) learns to disentangle interleaved conversational threads and clusters utterances with the same chatting topic. Compared with previous works which concentrate more on alignment chatty messages of multi-party conversation, we focus on the alignments of question and answer utterances in two-party CS which mainly communicate with objective information (e.g., the color and style of some specific clothes).

In fact, the most simple and direct way to obtain the Q-A alignments is learning a content matching function between question and answer utterances. And the alignment question for each server's answer is the most matching one

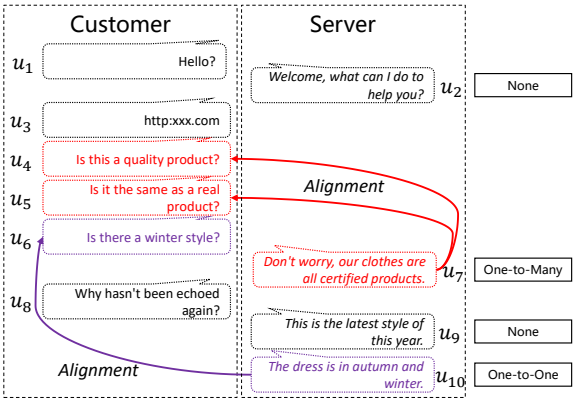

Figure 1: Question and answer alignments of multi-round conversation utterances in customer service.

in all customer's utterances (vice versa). However, this simple strategy will ignore several important issues. First, the meaning of an utterance is context dependent. It makes the individual classification (Du, Poupart, and Xu 2017; Jiang et al. 2018) for each utterance pair is not enough to align the whole conversation utterances. Second, the alignments of different utterances are correlating and interactional with each other. For example, if question Q_1 (e.g., u_4 in Figure 1) is very *similar* to Q_2 (u_5) in content, and the question Q_2 (u_5) aligns with answer A_1 (u_7), the question Q_1 (u_4) should also *align* with answer A_1 (u_7). And if question Q_1 (u_5) is very *dissimilar* to Q_2 (u_6), and the question Q_2 (u_6) aligns with answer A_1 (u_{10}), the question Q_1 (u_5) should *not align* with answer A_1 (u_{10}). Finally, there are various alignments of Q-A utterances in a CS conversation. For one thing, chatting messages may be interspersed in the conversation utterances, which raises the **empty** alignment (***None***) issue (no alignment for a given utterance, e.g., u_2 and u_9). For another, customers may consult one thing with multiple similar questions, which raises the **multiple** alignment (***One-to-Many***) issue (one answer may align with multiple questions, e.g, u_7 should align with two utterances: u_4 and u_5). The *None* and *One-to-Many* alignments obviously cannot be addressed by the content matching based models. Therefore, the most challenge of learning to align question and answer utterances in a CS conversation is: ***how to build a unified model, which not only is able to model the conversational contexts and the mutual influence of different utterances, but also address the issue of empty and multiple alignments.***

To this end, this paper proposes end-to-end models to obtain Q-A alignments in a CS conversation. We first utilize *recurrent networks* to encode the conversational utterances, and we also adopt two separate encoders for intensive modeling the customer's and server's information. Then, *pointer networks* are adopted to decode all alignments for server's utterances. Specifically, the proposed models are able to deal with *None* alignment by appending a shared placeholder representation in each conversation representation memory. We adopt a *classification loss* to learn the basic model. Moreover, we adopt a *regression loss* for conveniently and elegantly addressing the case of *One-to-Many* alignments.

Besides, we construct a dataset from an in-house online CS. The results demonstrate the efficiency of the proposed models. ***Compared with the state-of-the-arts, the proposed models are able to promote the F1 from 79.5% to 84.86% on the overall alignments. Moreover, for None alignments, we improve the F1 from 91.1% to 94.07%. And for One-to-Many alignments, the F1 is improved from 76.27% to 80.39%.***

In brief, the main contributions are as follows:

- We focus on the task of obtaining the alignments of question and answer utterances in a CS conversation, it is very useful for CS conversation analysis and building intelligent dialogue systems.
- We propose end-to-end models to learning the alignments of question and answer utterances in conversations with recurrent pointer networks. Our models not only are able to model the conversational contexts and consider the mutual influence of different Q-A alignments, but also address the issue of *None* and *One-to-Many* alignments in a unified manner.
- We construct a dataset from an in-house online CS[1]. The experimental results demonstrate that the proposed models are effective to obtain Q-A alignments, even for the utterances with *None* and *One-to-Many* alignments.

Task Description

The Task

There are two participants in a customer service (CS): a customer and a server. In most case, the customer raises the questions which will be answered by the server. In fact, the server also may ask questions to the customer for obtaining more detailed intention. In this work, we mainly focus on the former, and the proposed methods could be easily extended to deal with the latter when we simply reverse the alignment direction. Therefore, we devote to finding the alignments between each server's utterance (answer) and the corresponding customer's utterances (question). We formulate the task as a sequence multi-classification multi-label classification task, and we can learn supervised models with annotation training data.

Let $CS = [(u_1, t_1), (u_2, t_2), ..., (u_n, t_n)]$ denotes a CS conversation, $u_i = [w_1, ..., w_{|u_i|}]$ indicates the natural language sentence (word sequence) and t_i indicates the role of speaker. In this scenario, only two roles (Customer (c) and Server (s)) are considered. For each server's utterance $(u_i, t_i), t_i = s$, we should find the alignment utterances from all customer's raised ones before i ($\{j|j < i, t_j = c\}$) which could be answered by u_i.

Data

We construct a dataset from an in-house online CS to facilitate the research. We first sample 10,000 conversations from a human-to-human customer service system, which own about 6-20 utterances for each conversational episode. To construct the explicit alignments between customer's utterances (question) and server's utterances (answer), we built a labeling website and invite five annotators. They need to annotate the aligned customer's utterances when given a server's utterance. It is independent in labeling different answering utterances. If the server's utterance is a meaningless utterance and cannot answer any questions, it will be labeled as "None" alignment. The coincidence rate of the five annotators is about 85% (Fleiss' Kappa was 0.523). Another annotator is employed to review all labeled alignments and remove some inconsistency cases. In the end, we obtain 5,741 labeled conversations. There are 6.0 utterances of customers with 22.7 words and 4.5 utterances of servers with 6.2 words on average per conversation. The answer is aligned with 1.38 questions on average when neglecting the *None* alignment (1.15 questions when considering the *None* alignment). And the alignments of *None* and *One-to-Many* account for 57% and 12%, respectively.

[1]The data and codes will be shared in the academic community.

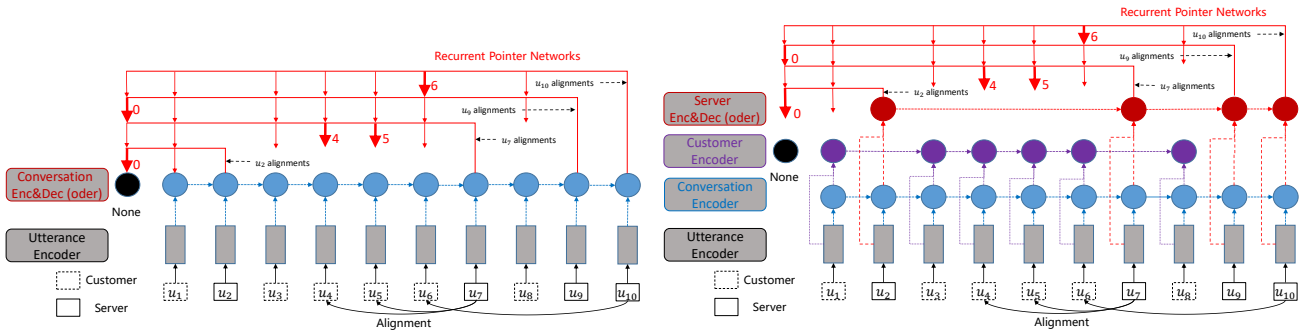

Figure 2: Main architectures of Recurrent Pointer Networks (RPN) for aligning question and answer utterances in a multi-turn conversation. The red arrows and the bold ones indicate candidate pointers (all previous customer's utterances) and right pointers (alignment utterances), respectively. **`Model-I`** (Left) utilizes a sharing encoder for customer and server. **`Model-II`** (Right) utilizes independent encoders for customer and server, which are built upon a sharing conversation encoder.

Challenges

A conversation may contain more than one consultation (e.g., two consults in the Figure 1). Considering the sequence of the dialogue process, we can align utterances by their posing orders (e.g., u_7 align u_6, u_9 align u_8). However, in a real scenario, this simple alignment rule may not work. At first, customer may raise a number of questions at a time (e.g., u_3, u_4, u_5, u_6), and the server may answer the questions with different orders. In addition, each server may need to communicate with multiple customers at the same time, so each customer's question may not be answered immediately. The situation will aggravate the above situation. On the other hand, people may express the same intention in a number of short and simple utterances in the spoken communicating environment (e.g., u_4 and u_5 express the same meaning). Therefore, some answers should align with more than one questions (e.g., u_7 should align with both u_4 and u_5). At last, there may have some chat messages (e.g., u_9) interspersed in the consulting process, and they should not be aligned with any utterance. The different alignment types mixed together in a CS conversation is another challenge in learning the alignment models. Besides the challenges in the language understanding and dialogue analysis, the data from the real applications are very noisy and contains typos, non-standard and other informal expressions.

Methodology

Inspired by the work on the encoder-decoder framework, especially the Pointer Networks (Vinyals, Fortunato, and Jaitly 2015), we propose end-to-end neural network models for aligning question and answer utterances in multi-round conversation (as illustrated in Figure 2). Compared with typical dialogue modeling, there are two different roles in customer service, therefore, besides utilizing one model to encode all utterance (**`Model-I`**, as shown in Figure 2(a)), we also utilize two extra models to independently encode customer's and server's utterances (**`Model-II`**, as shown in Figure 2(b)).

Our end-to-end neural network models consist of three components, including **Utterance Encoder**, **Conversation Encoder** and **Alignment Decoder**. Besides,the **`Model-II`** adopts two additional encoders for independent model customer and server conversations. Specifically, **Utterance Encoder** transforms the natural language sentence u_i into a numerical representation $\mathbf{v}_{u_i}$ which is able to feed into neural models. **Conversation Encoder** considers the relevance of conversation utterances. In `Model-II`, **Customer Encoder** and **Server Encoder** are similar to the Conversation Encoder, but only deal with customer's and server's utterances, respectively. **Alignment Decoder** takes the representation of server's utterance into account and predicts the alignments with the previous customer's utterances (or do not align with anyone). Considering the empty alignment with a joint manner, we set a shared representation for the "None" placeholder (as shown in Figure 2).

Utterance Encoder

Each utterance ($u_i = [w_1, ..., w_{|u_i|}]$) is a word sequence. By adopting word embeddings and semantic composition models such as convolutional neural network (CNN) and Recurrent Neural Network (RNN), we can learn the utterance representation. We adopt the CNN model to learning the sentence representation. To distinguish utterances raised by customer and server, we utilize low-dimensional vectors $\boldsymbol{ro}_{u_i}$ to represent different roles. That is, two vectors are adopted to represent customer and server, respectively. The final utterance representation $\boldsymbol{v}_{u_i}$ is concatenated of the CNN result $\boldsymbol{v}'_{u_i}$ and the role embedding $\boldsymbol{ro}_{u_i}$.

Conversation Encoder

Utterance encoder considers each word sequence separately, which neglects the context of other utterances in the conversation. In fact, the meaning of an utterance usually depends on its contexts. Therefore, we need to consider the conversation environments for obtaining the representations of each utterance. Different from the previous and later units are selected as the contexts in other tasks (e.g., word representation learning (Mikolov et al. 2013)), the meaning of utterances in the conversation only consider its previous utterances. Therefore, RNN is suitable to encode the conver-

sation in CS. Moreover, consider the customer and server owning different knowledge (consulting questions/answers) in a conversation, we utilize two individual RNN models in Model-II. RNN models such as LSTM (Hochreiter and Schmidhuber 1997) and GRU (Chung et al. 2014) are able to transform the original utterance representations (sequential vectors) $\{\mathbf{v}_{u_1}, \mathbf{v}_{u_2}, ..., \mathbf{v}_{u_n}\}$ into context dependent representations $\{\mathbf{h}_{u_1}, \mathbf{h}_{u_2}, ..., \mathbf{h}_{u_n}\}$ (another sequential vectors) in a CS conversation.

Customer and Server Encoders In model-II, we employ two RNN models to encode customer and server's utterances, respectively. Here, we take the customer encoder to illustrate its implement progress, the server encoder owns the similar progress. Each input of customer encoder $\mathbf{x}_{c_t}$ is the concatenation of the original utterance representation $\mathbf{v}_{c_t}$ and the output of conversation encoder $\mathbf{h}_{c_t}$ (c_t indicates the t-th utterance of **c**ustomer.). Customer encoder is used to transform the original customer's utterance representations $\{\mathbf{x}_{c_1}, \mathbf{x}_{c_2}, ..., \mathbf{x}_{c_n}\}$ into customer context dependent representations $\{\mathbf{h}_{c_1}, \mathbf{h}_{c_2}, ..., \mathbf{h}_{c_n}\}$. The server encoder can also be used to obtain the utterance representations in the server context $\{\mathbf{h}_{s_1}, \mathbf{h}_{s_2}, ..., \mathbf{h}_{s_n}\}$, s_t indicates the t-th utterance of **s**erver.

Alignment Decoder

This component is the core step in the proposed models, which is to find the alignments between each server's utterance and previous customer's utterances. The empty and multiple alignments should be considered in a unified manner. In practice, the progress of aligning utterance is to find the "matching utterance ids" for each utterance. Therefore, we adopt a pointer network as the alignment decoder. We define the alignment score $a_{i,j}$ as the probability or weight of i-th utterance (server's) to answer the j-th utterance (customer's). There are some explicit rules which can be simply implanted in the model with mask vectors:

$$a_{i,j} = 0, if: \ j >= i. \tag{1}$$

$$a_{i,j} = 0, if: \ t_i = c. \tag{2}$$

$$a_{i,j} = 0, if: \ t_j = s. \tag{3}$$

To learn a better align model, these rules are merely utilized in the testing stage.

The alignment score in model-I and model-II can be computed by the following functions:

$$\begin{aligned} &\texttt{Model-I:} \quad a_{i,j} = f(\mathbf{h}_{u_i}, \mathbf{h}_{u_j}) \\ &\texttt{Model-II:} \quad a_{s_i,c_j} = f(\mathbf{h}_{s_i}, \mathbf{h}_{c_j}) \end{aligned} \tag{4}$$

Function f can be modeled by neural networks such as bilinear function $f(\mathbf{h}_{u_i}, \mathbf{h}_{u_j}) = \mathbf{h}_{u_i}^T \cdot \mathbf{W} \cdot \mathbf{h}_{u_j}$ ($\mathbf{W}$ is the model parameter and $\mathbf{h}_{u_j}^T$ indicates the transpose of the column vector $\mathbf{h}_{u_j}$) . We also utilize $f(\mathbf{h}_{u_i}, \mathbf{h}_{None})$ and $f(\mathbf{h}_{s_i}, \mathbf{h}_{None})$ to compute the score of empty alignment. $\mathbf{h}_{None}$ is the representation of the "*None*" placeholder which owns the same shape of utterance's representation.

Based on the alignment scores, we can obtain the original alignment results. However, the numerical values are not a probability value, which hampers the testing and training. To address this problem, we utilize two different methods to transform the score vector $[a_{i,0}, a_{i,1}, ..., a_{i,n}]$ into an easily compared probability vector.

Classification Alike to the typical recurrent prediction tasks, such as text generation and sequence labeling, we utilize the `softmax` function to obtain the prediction probability distribution. In each time step (for each server's utterance), the score vector is converted to the distribution vector $[a'_{i,0}, a'_{i,1}, ..., a'_{i,n}]$ with $\texttt{softmax}([a_{i,0}, a_{i,1}, ..., a_{i,n}])_j = [\frac{exp(a_{i,j})}{\sum_{k=0}^{n} exp(a_{i,k})}]$. And we can obtain the final alignments through the $\arg\max$ operation:

$$\hat{a}_i = \arg\max_j a'_{i,j}. \tag{5}$$

However, the $\arg\max$ operation merely obtain one alignment, and it cannot deal with the multiple alignments. To deal with such issue, we simply return the utterances when the difference between their alignment scores and the maximum alignment score are less than a certain threshold:

$$\{\hat{a}_i\} = \{j | a'_{i,j} + \lambda \geq max_{k=0}^{n}\{a'_{i,k}\}\}. \tag{6}$$

We call this type of prediction method as classification.

Regression The benefit of the above classification method is that it can coordinate the alignments among different utterances by through `softmax` function. However, its obvious weakness is that it cannot elegantly handle the case of multiple alignments. To avoid the setting of threshold λ in the classification method, we convert each align score $a_{i,j}$ into a probability value $p_{i,j}$ which indicate the possibility of the answer utterance i aligns the question utterance j. Specifically, we utilize the `sigmoid` function to compute the probability value: $p_{i,j} = sigmoid(a_{i,j}) = \frac{1}{1+e^{-a_{i,j}}}$. And we can return the utterance as the predicted alignments when the probability value larger than 0.5 (called `regression` method):

$$\{\hat{a}_i\} = \{j | p_{i,j} \geq 0.5\}. \tag{7}$$

Even the probability value $p_{i,j}$ does not consider the relevance of i-th answer utterance with other utterances, the representations of i-th and j-th utterances in computing the alignment score could consider the context of conversation environment. In this way, the computation of probability $p_{i,j}$ is also able to model the relevance of different utterances and their alignments in a multi-round conversation.

Training

The proposed models jointly predict all alignments of server's utterances. The inputs are utterance sequences, and the outputs are indexes sequences of aligned utterances. Therefore, we adopt a sequence to sequence (Seq2Seq) learning to train our models. The goal of Seq2Seq learning is to maximize the probability of observing an output target sequence by given an input source sequence. Given the batches of the source utterances and target indexes of alignment utterances, the objective function is to minimize the negative log-likelihood:

$$\mathcal{L} = -\frac{1}{M} \sum_{k=1}^{M} \sum_{i=1}^{N} L(\bar{\mathbf{a}}_i, \hat{\mathbf{a}}_i). \tag{8}$$

, where the superscript (k) indicates the index of one conversation episode which contains utterances with variable number (N). $\bar{\mathbf{a}}_i$ and $\hat{\mathbf{a}}_i$ indicate the golden and predicted alignments for i-th utterance, which are represented by one-hot vectors for the convenience of calculation, especially the backpropagation of gradients. L is the loss function in fitting the standard result $\bar{\mathbf{a}}_i$ and the predict result $\hat{\mathbf{a}}_i$. There are some minor differences in computing the loss between the methods of `classification` and `regression`.

Classification Loss: We use cross-entropy to compute the loss of two probability distributions:

$$L_C(\bar{\mathbf{a}}_i, \hat{\mathbf{a}}_i) = -\sum_{k=0}^{n} \hat{\mathbf{a}}_{i,k} log(\bar{\mathbf{a}}_{i,k}). \quad (9)$$

, $\hat{\mathbf{a}}_{i,k}$ is the k-th value of the alignment distribution vector computed by the `softmax` function for the i-th utterance. And $\bar{\mathbf{a}}_i$ is a golden probabilistic alignment vector (e.g., the original alignment vector $[0, 0, 1, 1, 0]$ should convert to $[0, 0, 0.5, 0.5, 0]$).

Regression Loss: We use L_2 norm to compute the compatibility of two probability values' vectors.

$$L_R(\bar{\mathbf{a}}_i, \hat{\mathbf{a}}_i) = \frac{1}{2}||\bar{\mathbf{a}}_i - \hat{\mathbf{a}}_i||_{L_2} + \alpha||\hat{\mathbf{a}}_i||_{L_1}. \quad (10)$$

,$||\hat{\mathbf{a}}_i||_{L_1}$ is the L_1 norm of predicting vector, which can polarize probability values (the alignment probability tent to 0 or 1). α is the coordinate coefficient of L_1 norm which is a very small value, usually.

Experiments

In this section, we present our main experimental results, which devote to answering the following questions: 1) *Are the proposed models able to effectively obtain all utterance alignments in a unified manner?* 2) *Are the proposed models able to deal with the empty and multiple alignments?*

Comparison Methods

The multi-round conversation in CS has some specific characteristics: 1) it has only two participants (a customer and a server); 2) the customer mainly poses the questions; and 3) the server mainly poses the answers. Therefore, we can simply obtain the utterance alignments using some heuristic rules based on the posing order of utterances. For example, the rule `Greedy-1` selects the nearest customer's utterance as the aligned question for the given server's answer. The rule `Greedy-N` selects the multiple adjacent customer's utterances until meeting a server's utterance. And the rules with `Jump` indicate that we can jump the adjacent servers' utterances to select aligned customer's utterances. Four rules are adopted and their sample results for the conversation in Figure 1 are given in Table 1.

The above rule-based methods is very limited, for example, they are not applied to multi-party communication (Ouchi and Tsuboi 2016). Therefore, we are still committed to finding the alignments of utterances through conversational analysis and natural language processing. We learn different Q-A alignment models based on utterance content matching (marked as `Match`). They all utilize deep neural networks to score the matching degree between customer's and server's utterances. And we adopt the CNN, the basic RNN and the LSTM models to encode the customer's and server's utterances. Take the CNN for example, the matching score $s_{Q,A}$ of a customer's question Q and a server's answer A is calculated as: $s_{Q,A} = sigmoid(CNN(Q)^T \cdot \mathbf{M} \cdot CNN(A))$, $CNN(S)$ indicates the vector representation of utterance S (word sequences) by CNN, and the matrix $\mathbf{M}$ is the parameter of the matching model. We utilize the following margin-based ranking loss to train aforementioned matching models: $L = max(0, s_{Q,A'} + \lambda_m - s_{Q,A})$ (or $L = max(0, s_{Q',A} + \lambda - s_{Q,A})$), Q' and A' indicate the random selected non aligned utterances. For each server's utterance, we choose the customer's utterance with the maximum matching score as the aligned question (marked as greedy (`G`)). In addition, a threshold strategy is adopted to deal with some answers which may own multiple aligned questions (marked as threshold (`T`)).

Rules	ID	Golden Alignments	Predicted Alignments
Greedy-1	U_6	[U_3,U_4]	[U_5]
Greedy-N	U_6	[U_3,U_4]	[U_2,U_3,U_4,U_5]
Greedy-1+Jump	U_9	[U_5]	[U_7]
Greedy-N+Jump	U_9	[U_5]	[U_2,U_3,U_4,U_5,U_7]

Table 1: Sample utterance alignments obtained by rules.

Besides the rule-based and match-based methods, we also re-implement two state-of-the-arts approaches. The first one is discriminative models (we construct a Chinese question word set and run a LDA model on our conversation corpus), including the *Discriminative* (marked as `Disc`) and *Discriminative + LDA* (marked as `Disc+LDA`), which have been proposed in (Du, Poupart, and Xu 2017). The second one is a two-stage method which has been proposed in (Jiang et al. 2018). The first stage selects message pairs and trains a pairwise similarity model with a siamese hierarchical CNN (`SHCNN`). And the second stage first constructs a graph with similarity ranking (`CISIR`) and then finds the connected components as the final linked utterances.

Configurations

The dataset is split into training (80%), validation (10%) and testing set (10%). The utterances in Chinese are segmented into word sequences with Jieba[2] tool after some basic preprocessing such as convert all URLs to a special label "*URL*". And we use the words with the frequency more than 3, which covering 96.8% of the word in the corpus. All the out of vocabulary words are replaced by a special token "*UNK*".

For a fair comparison, we set word embedding dimension is 100 for all deep learning based models. For CNN based models, we utilize three filter sizes (2,3,4) and 50 filter numbers with each size. The margin λ_m are choose from $[0.3, 0.5, 0.7, 0.9]$. The threshold value (the λ in Formula 6) is set as 0.2 when using classification (We tried 0.1, 0.2 and

[2]https://github.com/fxsjy/jieba

0.3. The results of 0.1 and 0.2 are better, but there is little difference between them.). We choose the best parameters with grid search in validation data. We use LSTM for all encoders and decoder in all proposed models with hidden layer size = 300. We use the Adam optimizier to update gradients in all experimental configures.

Evaluation metrics: Based on the human-labeled alignments for each server's utterances, we calculate the precision (`P`), recall (`R`) and `F1` for utterance alignments. Considering that there are multiple alignments, we utilize the micro averaging to obtain the overall metrics for equally treating all utterance pairs.

Overall Performance

The overall experimental results are shown in Table 2. The first four rows are the results of rule-based methods. The following six rows come from the Q-A alignment models based on utterance content matching. And the next three rows are two alignment results of methods proposed in (Du, Poupart, and Xu 2017) and one result of (Jiang et al. 2018). The last four rows are the results of our proposed models with recurrent pointer networks (RPN). *C* and *R* indicate the classification and regression loss adopted in the training process, respectively.

Models	P	R	F1
Greedy-1	57.95	53.99	55.9
Greedy-1+Jump	45.17	42.08	43.57
Greedy-N	55.39	58.88	57.08
Greedy-N+Jump	12.66	51.26	20.3
Match-G (CNN)	51.08	47.6	49.28
Match-G (RNN)	37.72	35.15	36.39
Match-G (LSTM)	40.84	38.06	39.4
Match-T (CNN)	33.17	77.68	46.49
Match-T (RNN)	32.44	77.95	45.81
Match-T (LSTM)	33.2	80.02	46.93
Disc (Du, Poupart, and Xu 2017)	**98.55**	49.01	65.46
Disc+LDA (Du, Poupart, and Xu 2017)	77.32	51.83	62.06
SHCNN+CISIR (Jiang et al. 2018)	80.1	78.92	79.5
Model-I (C)	82.44	83.2	82.82
Model-I (R)	81.03	**84.55**	82.75
Model-II (C)	83.12	83.86	**83.49**
Model-II (R)	**84.22**	**85.51**	**84.86**

Table 2: The **P**recision, **R**ecall and **F1** (%) for all test data.

From the four upper parts of Table 2, we can observe that: 1) The effects of rule-based methods are not very bad. The overall F1 even exceed the results of matching methods. The possible reason is that the annotators' vision is difficult to cover a longer context (more previous utterances), it is a challenge and should be addressed in future work. 2) The rules without "Jump" own a better precision and a better recall. The main reason is that there are lots of empty alignments, and the "Jump" rules tent to obtain more invalid alignments. 3) The proposed methods in (Du, Poupart, and Xu 2017) obtain a very high precision, but do not have a good recall. The main reason is that the human-designed one-hot features and the simple discriminative models have not a good generalization ability, while comparing with deep learning methods with symbol embeddings and deep semantic composition models. 4) The proposed method in (Jiang et al. 2018) obtain greatly improvements, it illustrates the importance of modeling the interaction effect among multiple utterance alignments. 5) It is very hard to obtain satisfactory results only based on matching utterance content, even the threshold-based matching methods are only able to improve some recalls to a certain extent.

And the last part of Table 2 shows that: 1) Two proposed models are able to promote the performances of utterance alignments on both precision and recall. The results demonstrate that the alignments of conversational utterances not only rely on the language expressions but also depend on relevant utterances. 2) Two separate encoders for customer and server are better on modeling the conversational knowledge which models on the top of the whole conversation. We have tried to remove the outputs of conversation encoder to the inputs of customer and server encoder, but the results are far from satisfied. Therefore, the alignment of one $Q - A$ pairs not only rely on the relevance of other Q' with Q and other A' with A, but also rely on the other $Q' - A'$ pairs. 3) For precision, the classification and regression loss obtain roughly the same results, but for recall, the regression loss has certain advantages.

None and *One-to-Many* Alignments

To validate the performance of our proposed methods on *None* and *One-to-Many* alignments. We compute the P, R, and F1 with different alignment types. And the experimental results are shown in Table 3 and Table 4. The cases of empty alignment are classified as "One-to-One" in Table 4.

From Table 3 and Table 4, we can observe that: 1) The performances of different methods are very quite different from the effect of different alignment types. 2) The rule-based methods are hard to obtain a higher precision in the case of a relatively better recall. Some rules (e.g., "Greedy-1+Jump") favor precision and some other rules (e.g., "Greedy-N+Jump") are beneficial to recall. 3) The Q-A alignment models based on utterance content matching have no advantages both on *Empty* and *One-to-Many* alignments. 4) The methods in (Du, Poupart, and Xu 2017) are failed in *Non-Empty* and *One-to-Many* alignments, despite they obtain a very high precision on empty alignments. 5) The proposed method in (Jiang et al. 2018) is very good at dealing with the *Non-Empty* alignments and not good at the *Empty* alignments. 6) RPN-based alignment models are able to obtain appropriate results on *Empty* and *One-to-Many* alignments, which illustrate that the unified models are effective for all alignment types. 7) Compared with classification loss, the regression loss has more advantages for *One-to-Many* alignments.The main reason is that the classification loss heavily rely on the threshold λ, which is hard to turn and may own different values in different conversational episodes, and the regression loss avoids using this hyper-parameter.

Related Work

There are many tasks in NLP and AI involve the alignment. Word alignment (Zhang et al. 2017) is an important task

Models	Empty			Non-Empty		
	P	R	F1	P	R	F1
Greedy-1	54.99	54.99	54.99	62.65	52.66	57.22
Greedy-1+Jump	14.65	14.65	14.65	**93.51**	78.6	**85.41**
Greedy-N	51.47	54.99	53.17	60.66	64.06	62.31
Greedy-N+Jump	3.75	14.65	5.97	23.58	**100**	38.16
Match-G (CNN)	71.48	71.48	71.48	18.79	15.8	17.17
Match-G (RNN)	26.94	26.94	26.94	54.78	46.08	50.06
Match-G (LSTM)	30.46	30.46	30.46	57.27	48.18	52.34
Match-T (CNN)	30.44	81.36	44.3	38.29	72.8	50.18
Match-T (RNN)	30.11	81.2	43.93	36.59	73.64	48.89
Match-T (LSTM)	30.05	79.52	43.62	38.48	**80.7**	52.11
Disc (2017)	**98.89**	84.45	91.1	81.25	1.82	3.56
Disc+LDA (2017)	72.36	61.87	66.7	**90.61**	38.46	54
SHCNN+CISIR (2018)	82.32	84.35	83.32	74.30	74.41	**74.35**
Model-I (C)	91.72	89.65	90.68	70.24	71.96	71.09
Model-I (R)	91.15	91.91	91.53	68.57	74.76	71.53
Model-II (C)	91.4	**94.91**	**93.12**	72.02	71.82	71.92
Model-II (R)	**94.02**	**94.12**	**94.07**	71.6	74.05	72.81

Table 3: The **P**recision, **R**ecall and **F1** (%) for Empty and Non-Empty alignments.

Models	One-to-One			One-to-Many		
	P	R	F1	P	R	F1
Greedy-1	57.41	57.41	57.41	66.85	29.31	40.75
Greedy-1+Jump	42.08	42.08	42.08	**96.07**	42.12	58.56
Greedy-N	53.07	58.47	55.64	78.93	61.82	69.34
Greedy-N+Jump	10.46	44.5	16.94	38.78	**100**	55.88
Match-G (CNN)	52.73	52.73	52.73	24.02	10.59	14.7
Match-G (RNN)	35.59	35.59	35.59	72.63	32.02	44.44
Match-G (LSTM)	38.7	38.7	38.7	75.98	33.5	46.5
Match-T (CNN)	31.74	79.78	45.42	56.57	62.56	59.42
Match-T (RNN)	31.03	79.95	44.71	55.25	63.55	59.11
Match-T (LSTM)	31.47	80.98	45.33	59.05	73.15	65.35
Disc (2017)	**98.61**	55.67	71.16	80	0.99	1.95
Disc+LDA (2017)	76.2	54.88	63.81	**96.03**	29.8	45.49
SHCNN+CISIR (2018)	82.24	82.24	82.24	93.24	64.53	76.27
Model-I (C)	81.72	85.52	83.58	88.55	57.14	69.46
Model-I (R)	80.35	86.44	83.28	87.54	70.94	**78.37**
Model-II (C)	82.66	**87.57**	**85.04**	89.7	66.5	76.38
Model-II (R)	**83.47**	**87.43**	**85.4**	91.51	**71.67**	**80.39**

Table 4: The **P**recision, **R**ecall and **F1** (%) for One-to-One and One-to-Many alignments.

which has been widely embedded in many tasks such as machine translation (Fraser and Marcu 2007). Coreference resolution needs to find all expressions that align with the same entity in a text (Lee et al. 2017). Entity and ontology alignment are important modules of knowledge integration (Zhu et al. 2017; Noy, Musen, and others 2000). Moreover, there are also many tasks devote to obtaining sentence-level alignments, such as sentence simplification (Hwang et al. 2015), paraphrase (Barzilay and Lee 2003), and cross-language sentences alignments (Chen 1993; Moore 2002). The tasks mentioned above focus on aligning different expressions with the same meaning, but the utterance alignment devotes to aligning the question sentences for each answer sentence (vice versa).

The most related task to this paper is discourse parsing, which aims at finding the entire discourse structure of discourse and dialogue. (Qin, Wang, and Kim 2017) was only classifying the discourse relations (pre-defined limited set) by two utterances given. By contrast, our utterance alignment is to find which utterance pairs are needed to link. (Elsner and Charniak 2011; Wang et al. 2011; Afantenos et al. 2015; Jiang et al. 2018) devote to finding which discourse units are attached to which ones in multi-party chat dialogues or technical web forums. (Du, Poupart, and Xu 2017) is the most related work, which devotes to discovering dependencies between chatty messages in a dialog. And this paper mainly focuses on the Q-A alignments of two-party dialogue in customer service. Compared with (Du, Poupart, and Xu 2017), we not only propose two end-to-end models which are able to deal with the *Empty* and *One-to-Many* alignments with a unified manner, but also construct a larger dataset and perform a more detailed experiment.

Pointer network is a kind of sequences to sequences neural network models, which initial to be used in learning approximate solutions for addressing geometric problems such as traveling salesman problem (Vinyals, Fortunato, and Jaitly 2015). It is also has been applied to many NLP tasks, such as question answering (He et al. 2017) and dialogue system (Gu et al. 2016). The core idea of pointer network is that the decoder could directly obtain the sub-sequences of the source in generating the target. Compared with the methods above which mainly own a fixed-size source sequence, the pointer customer's utterances will be changed for different server's utterances in the alignment of a CS conversational episode.

Conclusion

In this paper, we focus on a practical task in CS which devotes to obtaining the alignments of customer's questions and server's answers in CS conversation. We propose two end-to-end models with recurrent pointer networks (RPN). RPN-based alignment models are not only able to model the conversational contexts and consider the mutual influence of different utterances, but also address the issue of empty and multiple alignments for some answer utterances. And the future work includes: a) incorporating the goods' knowledge information with memory networks (Sukhbaatar et al. 2015) in learning the utterance alignments; b) developing unsupervised or semi-supervised methods which may alleviate the labor-intensive manual labeling in current methods.

Acknowledgments

This research work is supported by the Natural Key R&D Program of China (No.2018YFC0830101), the National Science Foundation of China (No. 61533018, 61702512) and the independent research project of National Laboratory of Pattern Recognition. This work was also supported by Alibaba Group through Alibaba Innovative Research (AIR) Program, CCF-DiDi BigData Joint Lab and CCF-Tencent

Open Research Fund.

References

Afantenos, S.; Kow, E.; Asher, N.; and Perret, J. 2015. Discourse parsing for multi-party chat dialogues. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 928–937.

Barzilay, R., and Lee, L. 2003. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, 16–23. Association for Computational Linguistics.

Chen, S. F. 1993. Aligning sentences in bilingual corpora using lexical information. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, 9–16. Association for Computational Linguistics.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Cui, L.; Huang, S.; Wei, F.; Tan, C.; Duan, C.; and Zhou, M. 2017. Superagent: A customer service chatbot for e-commerce websites. *Proceedings of ACL 2017, System Demonstrations* 97–102.

Du, W.; Poupart, P.; and Xu, W. 2017. Discovering conversational dependencies between messages in dialogs. In *AAAI*, 4917–4918.

Elsner, M., and Charniak, E. 2011. Disentangling chat with local coherence models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, 1179–1189. Association for Computational Linguistics.

Fraser, A., and Marcu, D. 2007. Measuring word alignment quality for statistical machine translation. *Computational Linguistics* 33(3):293–303.

Gu, J.; Lu, Z.; Li, H.; and Li, V. O. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL*. Association for Computational Linguistics.

He, S.; Liu, C.; Liu, K.; and Zhao, J. 2017. Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 199–208.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Hwang, W.; Hajishirzi, H.; Ostendorf, M.; and Wu, W. 2015. Aligning sentences from standard wikipedia to simple wikipedia. In *HLT-NAACL*, 211–217.

Ji, Z.; Lu, Z.; and Li, H. 2014. An information retrieval approach to short text conversation. *Computer Science*.

Jiang, J.-Y.; Chen, F.; Chen, Y.-Y.; and Wang, W. 2018. Learning to disentangle interleaved conversational threads with a siamese hierarchical network and similarity ranking. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, 1812–1822.

Lee, K.; He, L.; Lewis, M.; and Zettlemoyer, L. 2017. End-to-end neural coreference resolution. *arXiv preprint arXiv:1707.07045*.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Moore, R. 2002. Fast and accurate sentence alignment of bilingual corpora. *Machine Translation: From Research to Real Users* 135–144.

Noy, N. F.; Musen, M. A.; et al. 2000. Algorithm and tool for automated ontology merging and alignment. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00). Available as SMI technical report SMI-2000-0831*.

Ouchi, H., and Tsuboi, Y. 2016. Addressee and response selection for multi-party conversation. In *EMNLP*, 2133–2143.

Qin, K.; Wang, L.; and Kim, J. 2017. Joint modeling of content and discourse relations in dialogues. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 974–984.

Sukhbaatar, S.; Weston, J.; Fergus, R.; et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, 2440–2448.

Vinyals, O., and Le, Q. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, 2692–2700.

Wang, L.; Lui, M.; Kim, S. N.; Nivre, J.; and Baldwin, T. 2011. Predicting thread discourse structure over technical web forums. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 13–25. Association for Computational Linguistics.

Zhang, M.; Liu, Y.; Luan, H.; and Sun, M. 2017. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 1959–1970.

Zhu, H.; Xie, R.; Liu, Z.; and Sun, M. 2017. Iterative entity alignment via knowledge embeddings. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.