Noname manuscript No.
(will be inserted by the editor)

A Survey on Construction and Enhancement Methods in Service Chatbots Design

Zhenhui Peng · Xiaojuan Ma

Received: date / Accepted: date

Abstract Chatbots are being widely applied in many service industries to help schedule meetings, online shopping, restaurant reservations, and customer care and so on. The key to the success of the service chatbots design is to provide satisfying responses to the given user's requests. This survey aims to provide a comprehensive review of chatbots construction and enhancement methods. We first introduce major techniques for the three core design philosophies, which are rule-based, retrieval-based and generation-based methods, followed by a brief summary of the evaluation metrics. Then we present methods to enhance service chatbot's capabilities with either an ensemble of multiple chatbots, collaborating with human workers or learning from users. Finally, in future directions we discuss the promising response generation models for chatbots using the recent progress in the Transformer and contextual embeddings, as well as potential ways to construct a chatbot with personality to achieve a better user experience.

Keywords Chatbots · Chatbot design methods · Human-chatbot collaboration · Chatbot survey

1 Introduction

Chatbots, also known as dialogue/conversational systems/agents, are computer programs which conduct a natural conversation with users via speech or text [37,53]. Initially, chatbots were mainly designed for passing Turing test [59] or for fun, such as ELIZA conducting psychology interviews [64], and

Zhenhui Peng
Hong Kong University of Science and Technology
Tel.: +852-62280396
E-mail: zpengab@connect.ust.hk

Xiaojuan Ma
Hong Kong University of Science and Technology

Microsoft Xiaoice acting as a poet, a singer, a kid audio-books reciter or a journalist [65]. With the advances in machine learning and the rapid development of social media over the past decade, service chatbots which are designed to provide 24/7, easy-to-access services to the users in specific domains, have been a trend recently [36]. They can either be task-oriented chatbots which get information from the user to help complete some structured tasks like scheduling meetings [12], online shopping [23] and restaurant reservations [34], or chatbots that accomplish unstructured tasks like customer care in social media [66,21] and answering human resource questions [30].

Despite the popularity of chatbots, designing service chatbots to provide satisfying responses to the given users' requests still remains a crucial challenge. Given any user's speech or text as an input request, a service chatbot is required to understand the input and give appropriate (e.g., the same topic, make sense), helpful (e.g., contains useful and concrete information) and even tone-aware (e.g., conveys feelings like empathy and passion) responses [66, 21]. One popular approach to construct such a chatbot is the frame-based method [69,38], which predefines the structure of a dialog state as a set of slots to be filled during a conversation and gives responses based on some hand-crafted rules. For example, in a restaurant booking chatbot, the slots can be the reserved date, the cuisine, or the location of the restaurant, while the rules can be to ask questions until all the slots are filled and to generate responses based on the template. However, such kinds of rule-based chatbots are limited to a narrow domain as manually constructing and updating rules for complex systems are usually expensive.

Different from the rule-based chatbots, service chatbots that are based on data-oriented approaches can handle more types of user's requests [55]. They can either retrieve an existing response from a pre-compiled dataset (i.e., retrieval-based) [67], or generate a new response word by word based on the input sequence (i.e., generation-based) [51]. Retrieval-based chatbots can generate literally correct responses to the user, but they are limited by the size of the corpus as they can not generate new responses. While generation-based chatbots could address this problem, they are prone to give responses that are not grammatically correct or contain no useful information [55].

To build a more powerful chatbot that can handle a broader scope of service requests, chatbot designers should also consider how to enhance a chatbot's capabilities over time. One straightforward way is to make use of multiple chatbots to cover cross-domain requests. The multi-chatbot framework learns to select one response from the responses proposed by each chatbot to address user's requests [44]. However, such a framework can not learn new responses or actions outside the capabilities of the chatbot ensemble. Moreover, in complex real-world scenarios, fully automatic chatbots are potentially problematic, as they do not share the same experience as human on how to avoid serious mistakes that might negatively affect users. Therefore, some chatbot designers design chatbots with humans in the loop [17]. For example, chatbots can be designed to work with human workers collaboratively, and gradually learn to deal with unknown requests [34]. Chatbots can

also be designed to learn from a user's demonstration, where the user teaches the chatbot new skills to provide more personalized services [28].

1.1 Contribution of This Work

In this work, we try to systematically present and analyze notable works for chatbots design from both the perspectives of construction and enhancement methods. We aim to inspire more flexible ways to design service chatbots that can meet different user needs. The main contributions of this work are outlined below:

- A comprehensive analysis of different chatbot construction techniques along with their comparison and suitable usage scenarios.
- A brief summary of automatic and human-based metrics for chatbot design evaluations.
- Various chatbot enhancement techniques ranging from multi-chatbots to human-in-the-loop are presented along with their comparison.
- The advanced methods (e.g., Transformer) that could help chatbot design is presented.
- Data-driven techniques for chatbot personality design is discussed.

1.2 Link and Distinction to Related Surveys

This survey is inspired by several related papers [46,55,9] that classify the chatbots into rule-based, retrieval-based and generative-based. Apart from the high-level comparison like those papers, our survey uses equations and example models to illustrate the detailed mechanism behind each technique for better understanding of its strengths and drawbacks. Furthermore, to the best of our knowledge, this is the first survey that categorizes the chatbot enhancement methods. In addition, this work provides guidelines to developers for selecting the appropriate techniques that fit their requirements in terms of performance, usage scenarios, or cost.

1.3 Taxonomy and Organization of This Article

The taxonomy of this survey is showed in Fig. 1:

Design Philosophy: The core principle of the chatbot architecture that is designed to give a response based on the request. According to whether it is data-driven and whether it can generate new responses, we classify the chatbot architecture into three categories: 1) Rule-based chatbot, which identifies the characteristic variables of the utterance and gives a pre-defined response based on the variables and hand-crafted rules; 2) Retrieval-based chatbot, which uses the input utterance as the query to search for candidate responses from the dataset by some matching metrics; 3) Generation-based

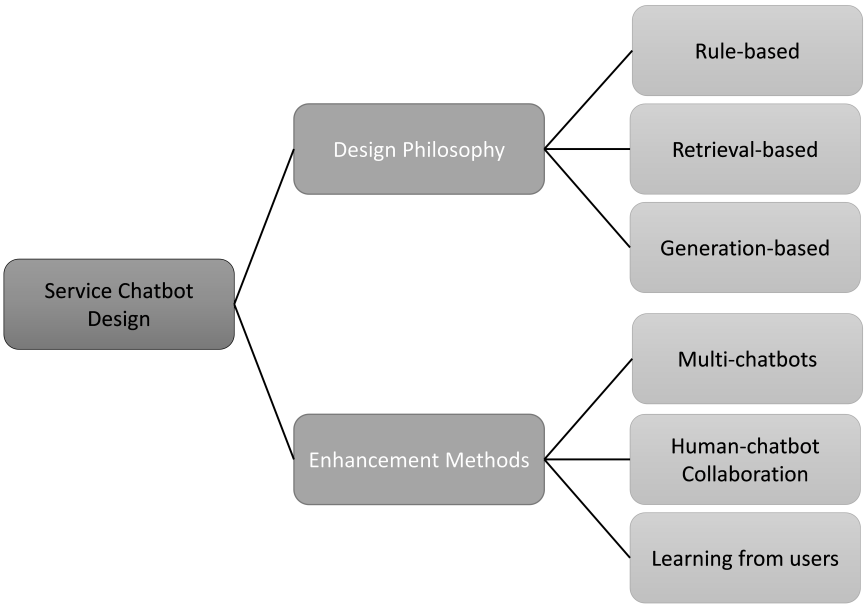


Fig. 1 The taxonomy of this survey

chatbot, which treats the conversation as the input-output mapping problem and learns to generate responses after training on a large amount of data.

Enhancement Methods: The design methods that can enable the chatbots to handle more kinds of requests or extend their functions. According to whether the service chatbots are mainly powered by the ensemble of multiple chatbots, human workers, or users, we classify these methods as: 1) Multi-chatbots methods, which combine rule-based, retrieval-based or generation-based chatbots together into a framework; 2) Human-chatbot collaboration methods, which enable the chatbot to work with and learn from human workers; 3) Learning from users, which improves the chatbot via user's direct feedback or instructions.

The organization is as below:

Section 2 focuses on three kinds of core design philosophies in building a chatbot, i.e., rule-based, retrieval-based, and generation-based methods. We showcase the principle of the representative methods in each design philosophy, as well as compare their advantages and disadvantages in terms of complexity and user experience in this section. Then in section 3, we focus on the methods that incorporate different bots, human workers or users into the chatbot design to enhance its capability. Finally in section 4, we discuss the opportunities of using advanced techniques like Transformer in chatbot design and how to incorporate personality into chatbot using data-driven methods.

Table 1 Chatbot construction techniques overview.

Philosophy	Techniques	Representative Papers
Rule-based	Pattern matching	*(Weizenbaum et al., 1966 [64]); (Colby et al., 1972 [11]); (Wallace, 2009 [63])*
	Modular task-oriented system	*(Chen et al., SIGKDD 2017 [9])*
Retrieval-based	TF-IDF	*(Lowe et al., SIGDIAL 2015 [32])*
	DNN-based	*(Lu et al., NIPS 2013 [33]);(Hu et al., NIPS 2014 [19])*
	RNN-based	*(Lowe et al., SIGDIAL 2015 [32]); (Zhou et al., EMNLP 2016 [70])*
Generation-based	Statistical Machine Translation	*(Ritter et al., EMNLP 2011 [47])*
	Seq2Seq	*(Sutskever et al., NIPS 2014 [57]); (Xu et al., CHI 2017 [66])*
	Seq2Seq + attention mechanism	*(Shang et al., ACL 2015 [52])*
	Seq2Seq + hierarchical structure	*(Serban et al., AAAI 2016 [51])*
	Seq2Seq + memory network	*(Ghazvininejad et al., AAAI 2018 [16])*

2 Core Design Philosophy

Given user's speech or text as the input requests, the chatbots usually preprocess the requests to be a text sequence and fit into the response model to get the responses [69,50]. Therefore, in the rest of the survey, we consider both the requests and responses as textual utterances. There are a number of ways to construct the model that generate responses to the input requests. In this section, we introduce three kinds of core design philosophies (i.e., rule-based, retrieval-based and generation-based) and several popular construction methods under each philosophy (Table 1).

2.1 Rule-based Methods

Rule-based chatbots are mainly built on manually constructed rules. These manually constructed rules can be the patterns in the input request, the "if-then" logic that triggers the action or response, or the template that is about to be filled in the response. Dating back to 1966, the development of chatbots started from analyzing the input sentence based on decomposition rules which are triggered by key words in a sentence [64]. A typical example provided by a prominent chatbot "ELIZA" is that the user asks "It seems that you like me". ELIZA can only recognize the words "you" and "me", but does not know what "It seems that" and "like" mean. Based on its decomposition rule "0 YOU 0 ME" (where 0 stands for indefinite number of words) and the reassembly rule "WHAT MAKES YOU THINK I 3 YOU" (where 3 stands for the third component of the the subject decomposition, here is "like"), the ELIZA will reply "What makes you think I like you". Following the same pattern-response rules as ELIZA, another chatbot "Parry" passed the Turing test [59] in 1972 [11]. Parry adds some affect variables like "fear", "anger" and "mistrust" to the more complex rules. For example, when a user mentions Parry, Parry decreases fear if mistrust is low and increases anger if mistrust is high. Such rules keep the conversation going and make people feel that they are really chatting to another person.

Extended from ELIZA, Richard Wallace developed the Artificial Linguistic Internet Computer Entity (ALICE) in 1995 [63]. ALICE applies the Arti-

ficial Intelligence Markup Language (AIML) and is primarily designed for historical and philosophical ruminations on human consciousness. The key part in AIML is the category that forms the unit of knowledge. As the example shown below, a category combines a pattern (e.g., question or stimulus), a template (e.g., answer or response), and optional context (e.g., a previous utterance).

< category >
< pattern > YES < /pattern >
< that > DO YOU LIKE MOVIES < /that >
< template > What is your favorite movie? < /template >
< /category >

Since the 1990s, a lot of research has been done on designing similar rule-based chatbots to provide services in specific domains [62,8]. These chatbots are also known as modular task-oriented dialog systems, which guide a user to finish some well-structured tasks such as restaurant reservations and movie bookings. As shown in figure 2, modular task-oriented dialog system mainly consists of three components [54]:

1) Spoken Language Understanding (SLU), which turns user utterance into user intention and slot-values, and outputs structured user action;

2) Dialogue Manager (DM), which tracks the dialog state based on past state, action and current user action, and outputs system action based on some policies;

3) Natural Language Generation (NLG), which turns a system action into natural language and outputs it to the user.

There are multiple available platforms such as Microsoft LUIS [39], IBM Watson Assistant [2] and Dialogflow [1] that provide easy-to-use SLU, DM and NLG services to help chatbot designers build service chatbots. For instance in designing an airline travel planning chatbot (Fig. 2), the designer needs to define the intents (i.e., user's goal or purpose) and entities (i.e., terms or objects that provide clarification or specific context for a particular intent) that might occur in the user's input. The designer also needs to define the rules to ask for the missing slots in the DM module and define the templates to generate the task-completion information in the NLG module. Inside each of the SLU, DM and NLG services provided by these platforms, there are techniques like Recurrent Neural Network [49], Long Short-Term Memory [18] and Reinforcement Learning [25] to handle each module. A survey of task-oriented chatbots can be found in [9].

Building a modular task-oriented chatbot could be easy in the domains that have expert knowledge and a well-defined structure. For example, in an online shoe shopping chatbot [23], the shoes have limited features such as price, color, material, style and brand, which can easily be used to filter the shoes interactively. Also in the restaurant reservation, food ordering, and movie ticket booking domains, the users' goals are clear and the designer can

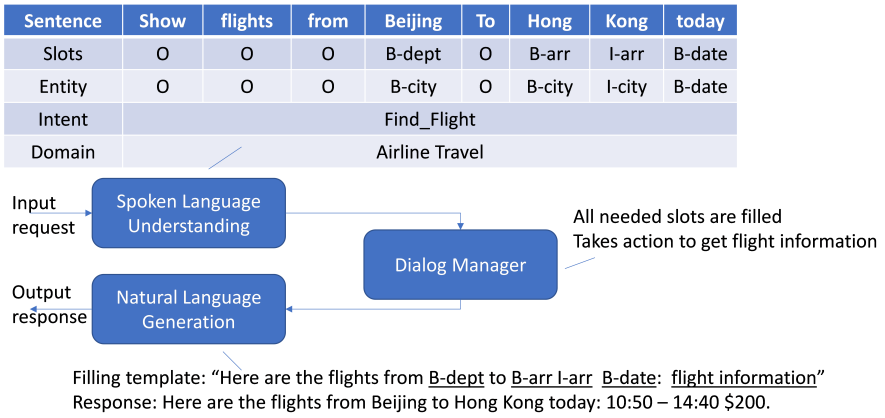


Fig. 2 Main components and example of a modular task-oriented chatbot.

easily design the logic to get the needed information (e.g., which restaurant / food / movie, when, and how many people) for task completion. However, as the knowledge space gets larger and users' expectations get higher (e.g., wanting some playful interaction like chitchat [30]), it becomes difficult and expensive to anticipate users' intentions and design the rules to handle them. Therefore, researchers seek to use data-driven methods to automatically build up the service chatbots, which are discussed in next two sections.

2.2 Retrieval-based Methods

The retrieval-based chatbots select the response that best matches the users' requests by searching a pre-constructed conversational repository [26,24,67]. The key to retrieval-based chatbots is request-response matching. Given a request q and a repository of request/response pairs, there are two types of strategies to retrieve the response r' [47]:

1) Request-based strategy $[r_{\text{argmax}_i sim(q,q_i)}]$, which retrieves the response r_i whose associated request q_i is most similar to the user's input q;

2) Response-based strategy $[r_{\text{argmax}_i sim(q,r_i)}]$, which retrieves the r_i that is most similar to the user's input q.

To compute the similarity between the q and r_i, it is prescribed to transform them into some numeric or vector representation. As early as in 2000, Isbell et al. developed a retrieval-based chatbot Cobot which interacted with the users in a game LambdaMOO [26]. Cobot used a request-response matching method based on word occurrence, in which each potential response is weighted by the number of words that match to the words in the request. A more popular occurrence-based method used in chatbots is TF-IDF [32]. TF-IDF stands for "term frequency - inverse document frequency" and it measures the importance of a word in a document (i.e., the request in chatbot

case) in relation to the whole repository. The "term frequency" is the number of times the word appears in a given request, while the "inverse document frequency" puts a penalty on how often this word appears elsewhere in the repository. The final score is calculated as:

$$\text{tf-idf}(w, q, D) = f(w, q) \times \log \frac{N}{|q \in D : w \in q|} \tag{1}$$

where $f(w, q)$ is the number of times word w appears in the request q, N is the total number of requests or responses in the repository based on a different strategy, D represents the collection of the requests or responses, and $|q \in D : w \in q|$ is the number of requests or responses in which w appears. Take the response-based strategy as an example, for the request and each candidate responses, their TF-IDF vectors are calculated by concatenating the all tf-idf scores together. The candidate response with the highest cosine similarity to the request vector will be selected as the final response.

Although TF-IDF is simple to use without the need of training on dataset, it is not accurate enough since it does not consider the location of the word in the sentence and can not efficiently represent the importance of the words. With the success of deep neural networks (DNN) in computer vision, a couple of works have sought to use DNN-based methods to handle request-response matching in natural language processing [33,19]. In 2013, Lu et al. proposed a DNN matching architecture DEEPMATCH to attack the problem of matching short texts [33]. As shown in Fig. 3 (a), DEEPMATCH models the similarity between two utterances by first constructing a two-dimensional interaction space of their bag-of-words vectors, then going through two patch-induced layers and one fully connected layer, finally summarizing the decisions with the logistic regression unit. Their experiments on a traveling-related (Question, Answer) pairs dataset and a (Weibo, comment) dataset showed that DEEPMATCH's retrieval performance is better than three inner-product based models.

Later in 2014, Hu et al. proposed a convolutional DNN matching model ARC-II Fig. 3 (b), which first models all the possible combinations of the word embedding vectors (trained with the Word2Vec [40]) of two utterances via one-dimensional (1D) convolutions, then performs a 2D max-pooling to shrink the size of the representation by half, and uses more 2D convolution and pooling layers to obtain higher levels of representation. The final level of representation fits into a multi-layer perception (MLP) [6] to get the matching degree of two utterances. Compared with DEEPMATCH, ARC-II can better represent the patterns shared in two utterances and thus perform better in Weibo (tweet, response) retrieval task.

Besides CNN, the recurrent neural network (RNN) [49] is also widely used in the request-response matching deep architectures [32,70,71]. In their released Ubuntu dialogue corpus which is about technical support for Ubuntu-related problems [32], Lowe et al. proposed an RNN model (Fig. 4 (a)) to capture the representation for the request (c in their case) and response (r), and then use the dot product to measure the similarity. The RNN can model the

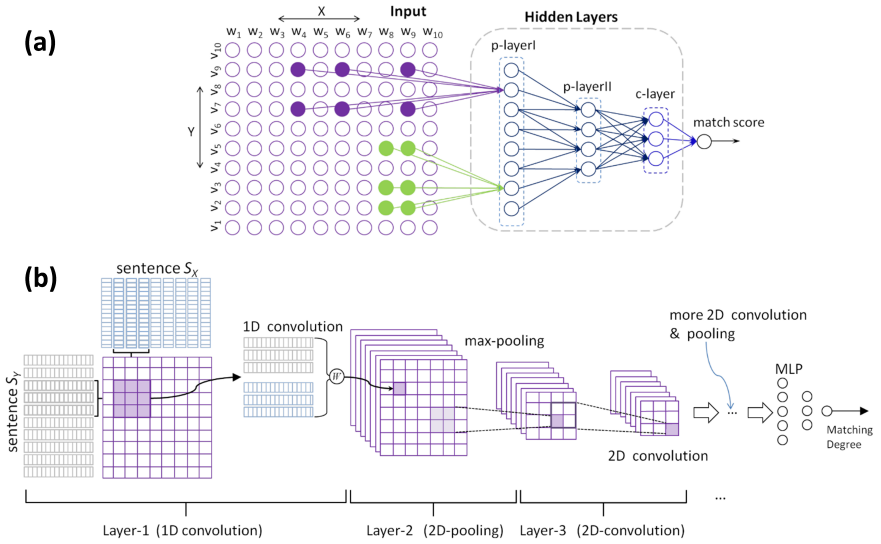


Fig. 3 Two DNN-based matching models: (a) DEEPMATCH [33], which represents the interaction of two utterances via an overlap in their bag-of-words vectors; (b) ARC-II [20], which represents the interactions via a convolution of their word embedding vectors.

time-dependent sequence data like the sentence, and can be formally represented by:

$$h_t = f(h_{t-1}, x_t) = f(W_h h_{t-1} + W_x x_t) \tag{2}$$

where h_t is the hidden state at time step t (e.g., index of word), x_t is the observed variable (e.g., word) at the same time step, and W_h, W_x are the weights learned from the training data. However, a conventional RNN can easily fail to capture the long-term dependence due to the gradient vanishing (or exploding) problem. To relieve this problem, Long Short-Term Memory (LSTM) [18] or Gated Recurrent Unit (GRU) [10] is commonly used to replace the hidden units in the RNN models [32,70]. In Lowe's work [32], the LSTM model outperforms the RNN model and the TF-IDF method in terms of the Recall@k metric in the response selection task.

Nevertheless, the models above only consider single-turn information to retrieve a response from the repository, while in the conversation the chatbot's response at the current turn usually needs the information from previous turns. To this end, Zhou et al. proposed a multi-view response selection model (Fig. 4 (b)) which uses GRU to learn the word-level semantics and dependencies in the connected utterances (i.e., context, u_1, u_2, u_3 in this case), as well as the utterance-level semantic and discourse information in a convolutional manner [70]. In the same response selection task, this model performs better than the LSTM model in Lowe's work, and better than the GRU model that only learns the word-level or utterance-level information. Recently, Zhou

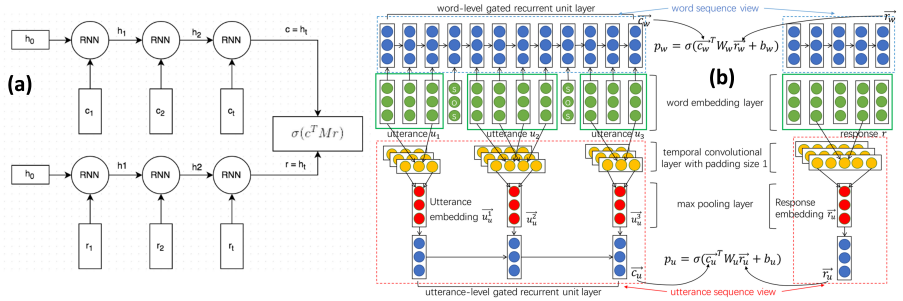


Fig. 4 Two RNN-based matching models: (a) The RNN model that only considers single-turn information [32]; (b) The Multi-view model that considers both word-level and utterance-level dependencies in multi-turn dialogues [70].

et al. proposed a deep attention matching (DAM) network which extends the attention mechanism of Transformer [60] in the self-attention (for intra word-level dependencies) and in the cross-attention (for dependence between a latently matched pair) [71]. The DAM achieves the best result in response selection task in the Ubuntu corpus [32] compared to RNN models.

Although not all of these request-response matching methods are originally designed for conversation, they are effective in the conversational studies [47,66]. The advantages of retrieval-based chatbots are that they need much fewer hand-crafted features than the rule-based chatbots and promise that the response is grammatically correct and diverse. However, they can easily provide an inappropriate response which is not really in the context of the request. It is also hard to persist with the same tone since the repository is diverse. Therefore, retrieval-based methods are more commonly used in chit-chat chatbots like Xiaoice [65] in open domain and question-answering chatbots which answer questions in specific domains such as travel [33] or give technical support [66,32].

2.3 Generation-based Methods

The generation-based chatbots synthesize a new sentence word by word as the response to the user's requests [57]. Originally, research on generation-based chatbots was mainly in chit-chat-style, open domains [47,51,52]. However, they have huge potential to be deployed directly for the tasks (e.g., customer care service [66,21]) which do not have a measurable goal if the task-related data is available. Therefore, in this section, we summarize the generation-based construction methods of general chatbots (either domain-specific or open domain).

The development of generation-based chatbots is inspired by the work in machine translation. In 2011, Ritter et al. used a phrase-based statistical machine translation (SMT) method in the response generation for the first

time, and showed that their SMT techniques are better-suited than retrieval-based methods on the Twitter dataset [47]. More specifically, the phrase-based SMT model considers the strong structural relation between many request-response pairs (e.g., "the soup smells delicious" - "I will bet it looks gorgeous too"), and extracts phrases like "smell-look" and "delicious-gorgeous" from the dataset to translate the request to the response. However, this model could work badly since the responses are often not semantically matched to the requests as in a translation. For example, it is likely that for a request the responses "having my fruit salad now", "but it is 2 am now" and "which restaurant" are both appropriate.

Sequence-to-sequence (seq2seq) [57], another technique derived from machine translation, is shown to work better in response generation tasks. The seq2seq model contains an encoder which encodes a request word by word and represents it as a vector, and a decoder which decodes the vector and generates the response word by word (Fig. 5 (a)). To better capture the dependencies in the utterance, LSTM [18], GRU [10], and techniques like bidirection and reverse order are commonly used to design the seq2seq models [51,66]. Formally, a standard seq2seq model uses RNN to encode the request into a vector $\mathbf{c}$ (i.e., the hidden state of the last word) using the equation 2 (shown in the previous section). Then at the time step t in the decoder, the probability distribution p_t of candidate words for the response is:

$$s_t = f(y_{t-1}, s_{t-1}, c) \tag{3}$$

$$p_t = \text{softmax}(s_t, y_{t-1}) \tag{4}$$

where s_t is the hidden state of the decoder RNN and y_t is the selected word for the response sequence at time step t. The objective function is:

$$p(y_1, ..., y_{T'}|x_1, ..., x_T) = p(y_1|\mathbf{c}) \prod_{t=2}^{T'} p(y_t|\mathbf{c}, y_1, ..., y_{t-1}) \tag{5}$$

where T and T' are the length of the request and response sequence respectively. Vinyals et al. showed that such a seq2seq model can generate more relatively preferable responses than the rule-based CleverBot, in both a domain-specific IT Helpdesk Troubleshooting dataset and a open-domain OpenSubtitles dataset [61]. Xu et al. further suggested that it can generate more appropriate, more helpful and more empathetic responses than the retrieval-based method in branded customer care service [66]. However, it could be problematic to simply use the hidden state of the last word as the context vector c, since each word in the response may strongly relate to different parts of the words in the request. The attention mechanism [5] is then introduced to address this problem. As shown in Fig. 5 (b), for the word y_t at time step t in the decoder, there is a corresponding context vector c_t specifically contributing to it, where c_t is calculated by the sum of all the hidden states in the request

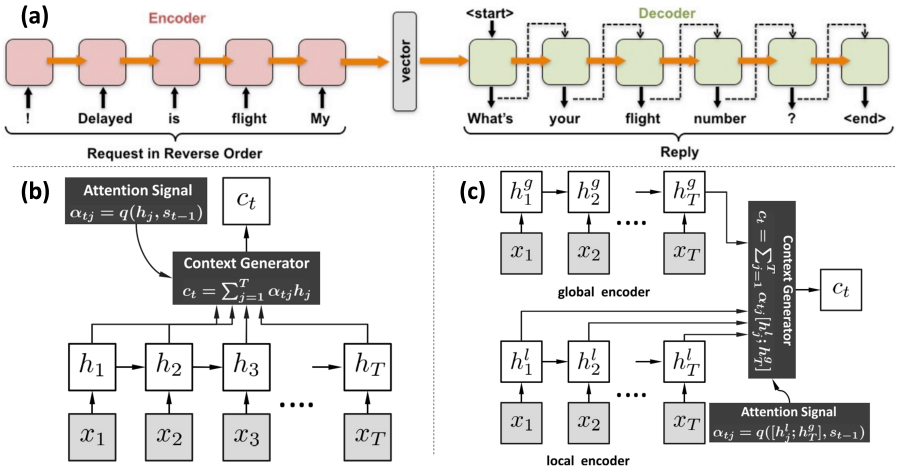


Fig. 5 The seq2seq models: (a) With LSTM neural networks, used in customer care services [66]; (b) With attention mechanism; (c) Combine (b) and (a) for a new attention signal [52].

sequence weighted by the attention signal α. The attention mechanism can be formulated as the following two equations:

$$c_t = \sum_{j=1}^{T} \alpha_{tj} h_j \tag{6}$$

$$\alpha_{tj} = q(h_j, s_{t-1}) \tag{7}$$

where q is a score function (e.g., $h_j^T W s_{t-1}$) that calculates how much the hidden state h_j relates to the previous hidden state s_{t-1} in the decoder at time step t. Shang et al. proposed a neural responding machine that combines both the global encoder that only considers the last hidden state and the local encoder that considers all the hidden states in the request by concatenation (Fig. 5 (c)) [52]. The responses generated by this hybrid model in the Twitter-like Weibo dataset are perceived more suitable than those generated by the models that only consider global / local encoders, or by the models that are retrieval-based or SMT-based.

However, the works discussed above only consider generating a response based on one previous request, while the huge amount of information derived from previous turns of the dialogue are ignored. To incorporate dialogue history into response generation, Serban et al. adopted the hierarchical recurrent encoder-decoder (HRED) and showed that their model is competitive with other models in the *MovieTriples* in terms of word perplexity [51]. The HRED uses RNN to not only encode the tokens appearing within the utterance, but also encode the temporal structure so far in the dialogue. It treats the dialogue as a sequence of utterances $\{U_1, ..., U_M\}$ and each utterance U_m

with N_m tokens is represented as $U_m = \{w_{m,1}, ..., w_{m,N_m}\}$, where $w_{m,1}$ is a random variable taking values in the vocabulary V and represents the token at position n in utterance m. The probability distribution P over the set of all possible dialogues is:

$$P_\theta(U_1, ..., U_M) = \prod_{m=1}^{M} P_\theta(U_m|U_{<m}) = \prod_{m=1}^{M}\prod_{n=1}^{N_m} P_\theta(w_{m,n}|w_{m,<n}, U_{<m}) \tag{8}$$

where θ represents all parameters, $U_{<m} = \{U_1, ..., U_{m-1}\}$ is the input utterances so far, $w_{m,<n} = \{w_{m,1}, ..., w_{m,n-1}\}$ is the tokens preceding n in the utterance U_m. The response generation can be performed as in standard language modeling: sampling one word at a time from $P_\theta(w_{m,n}|w_{m,<n}, U_{<m})$ conditioned on previous sampled words. As shown in Fig. 6 (a), the context hidden state c takes in the information from both the current utterance representation and the historical utterance information represented in the previous context hidden state.

To provide more appropriate and more informative responses to the requests, it usually needs to make use of external knowledge outside the dialogue history. For this purpose, Ghazvininejad et al. took advantage of the Memory Network [56] and proposed to condition responses on both the dialogue history and external "facts" [16]. As shown in Fig. 6 (b), they used the request (e.g., conversation history) to retrieve all contextually relevant facts $F = \{f_1, ..., f_k\}$, where each f_i is represented by bag of words as r_i to input to the facts encoder (i.e. Memory Network). Together with the summary of the request u (i.e., h_T) from the encoder RNN, the memory network calculates the initialized hidden state s_1 of the decoder as:

$$m_i = Ar_i; c_i = Cr_i; p_i = softmax(u^T m_i); o = \sum_{i=1}^{k} p_i c_i; s_1 = o + u \tag{9}$$

where A, C are the parameters of the memory network. Specifically, they used multi-task learning [35] to train their model on the tasks with / without the fact encoder, and the task with the fact encoder but replace the response with each of the facts in the training data. They demonstrated that their model is informative and applicable in the Twitter dataset grounded by Foursquare tips (e.g., comments about restaurant and other commercial establishments).

Despite the progress in seq2seq style models, the generation-based chatbots are still prone to generate universal sentences (e.g. "I'm OK") that lack helpful information. Also, a good generation-based service chatbot needs a well-collected dataset that is full of domain-specific conversation and less biased [48]. Nevertheless, the generation-based method is still the research hot spot on building customized and context-aware chatbots.

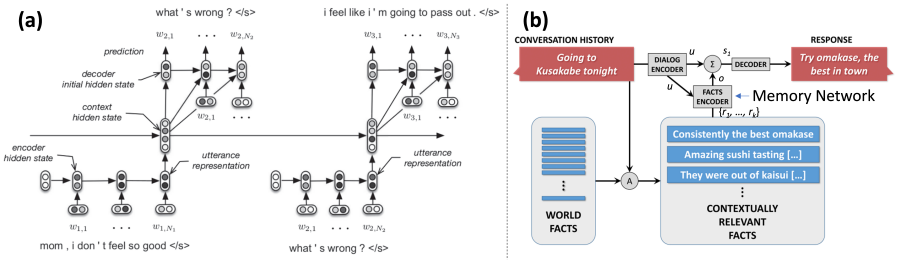


Fig. 6 The modified seq2seq models: (a) Hierarchical Recurrent Encoder-Decoder [51] that adds historical utterances in; (b) Knowledge-Grounded Neural Conversation Model [16] that adds external knowledge in.

2.4 Evaluation Methods

In this section, we look into the common methods which evaluate the quality of chatbot's response given the request, rather than the methods which evaluate the chatbot's objective performance like the task-success rate in spoken language understanding tasks and recall rate in response retrieval tasks. We summarize these methods as automatic metrics and human-based metrics.

(1) Automatic Metrics

Although it is still an open question to have a well-established method for automatic evaluation of the response quality, there are two common automatic metrics for reference. The first one is word perplexity originally for probabilistic language models [7] and later adapted to evaluate some generative dialogue models [51,16]. For example, for a model with parameters θ, dataset with N triples $\{U_1^n, U_2^n, U_3^n\}_{n=1}^N$, the word perplexity can be defined as [51]:

$$\exp\left(-\frac{1}{N_W}\sum_{n=1}^{N}\log P_\theta(U_1^n, U_2^n, U_3^n)\right) \tag{10}$$

where N_W is the number of tokens in the entire dataset and $P_\theta(U_1^n, U_2^n, U_3^n)$ is the probability of regenerating the exact (U_1^n, U_2^n, U_3^n). A better model will have lower perplexity. The word perplexity is appropriate because it measure the model's ability to account for the syntactic structure of the dialogue (e.g., turn taking) and the syntactic structure of each utterance (e.g., punctuation marks).

The second commonly used automatic metric is the BLEU (bilingual evaluation understudy) [41] originally for machine translation and later adapted to the evaluation of some dialogue models [66,16]. BLEU grades an output response according to n-gram matches to the reference, i.e., the real response from a human worker. Basically, it can be defined as:

Table 2 Summary of three core design philosophies for chatbot construction.

Philosophy	Techniques	Pros	Cons	Scenarios
Rule-based	Pattern matching; Modular task-oriented system	Easy to start; Robust, safe in narrow domains; Context-aware	Hard to extend; Needs structured domain knowledge; Needs a lot of hand -crafted features	Restaurant reservations; Movie bookings; Ordering food; Online shopping
Retrieval-based	TF-IDF; DNN-based; RNN-based	Can handle more requests; Literal human utterances; Various expression	Easily out of context; Inconsistent personality; Limited by size of repository	Domain-specific (e.g., travel) question answering ; Technical support
Generation-based	Phrase-based SMT; Seq2Seq-based: + attention; + hierarchical structure; + memory network	Can generate new responses; Can add in external knowledge; Highly coherent	Prone to generate universal sentences; Easily not informative; Needs a huge training dataset	Online customer care; Technical support; Entertainment

$$\mathrm{BLEU} = \mathrm{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{11}$$

where BP is the brevity penalty on the length of the utterance, p_n is probability that the n-grams in a generated response occur in the real response, N is the max number of gram (normally 4) and w_n is the weight for each n-gram (normally $\frac{1}{4}$). The higher BLEU score is indicative of a better model as the generated response is closer to the real one.

(2) Human-based Metrics

The automatic metrics can save time and money compared to human judgments, but can not really examine whether the generated response is appropriate, helpful and tone-aware. Liu et al. [31] showed that BLEU could correlate poorly with human judgment. Currently human evaluation is still the more convincing method for judging the response quality and is widely applied in chatbot evaluation. One common human-based metric is conducting pair-wise comparison to let humans choose which of the two responses is more suitable, more appropriate, and more helpful, etc [47,52]. Another way is to ask people to rate the appropriateness, helpfulness, passion, etc, of each response given the same request on a 5-point or 7-point Likert Scale [66,21]. The comparison among their proposed method and other methods are conducted on the average score of randomly sampled responses. Besides that, case studies with response example are often needed to analyze the quality of the response in depth [16,21].

2.5 Summary

Based on the discussion above, we summarize the rule-based, retrieval-based and generation based chatbot construction methods in terms of major techniques, pros, cons and suitable application scenarios in Table 2. However, ser-

Table 3 Chatbot enhancement techniques.

Enhancement Methods	Techniques	Representative Papers
Multi-chatbots Design	Reinforcement Learning for re-ranker policy	*(Serban et al., CoRR 2017 [50])*
	Data-driven re-ranker models	*(Qiu et al., ACL 2017 [45]); (Song et al., IJCAI 2018 [55])*
Human-chatbot Collaboration	CoChat: external memory + HRNN	*(Luo et al., AAAI 2018 [34])*
	Evorus: crowd-powered, automates itself over time	*(Huang et al., CHI 2018 [22])*
Learning from Users	Programming by demonstration	*(Li et al., CHI 2017 [29])*
	Verbal instruction	*(Azaria et al., AAAI 2016 [4])*

vice chatbots built merely on these techniques are usually not enough to meet user's various needs. To handle more types of requests (e.g., cross-domain) and expand its functionality, chatbots need to be designed to enhance themselves over time. In the next section, we discuss some of these enhancement methods (Table 3).

3 Enhancement of Chatbot

3.1 Multi-chatbot Design

Most available service chatbots are designed to perform tasks in highly specific domains like flight and hotel bookings. To satisfy the user's multiple-domain needs, there is a need to combine available chatbots that have different expertise [15]. Besides, even in the same domain, chatbots with different design philosophies have their own strengths and drawbacks (Table 2). Some researchers then seek to assemble chatbots built on different methods (e.g., retrieval-based and generation-based) to absorb their merits [55].

In a typical multi-chatbot framework, the responses from the different chatbots form the candidate responses pool for the same given request, and the key is to design a proper re-ranker (or response selection policy) which scores all candidates to pick the highest-scored response [55,45,50]. In 2016 Amazon Alexa Prize competition, Serban et al. proposed MILABOT, which combines 22 response models including neural retrieval-based models, neural network based generative models and template-based systems [50]. The MILABOT was trained on crowdsourced data (from Amazon Mechanical Turk) and real-world user interactions via reinforcement learning, to select an appropriate response from the models in its ensemble. In the reinforcement learning framework [58], they consider selecting the appropriate response as a sequential decision making problem. They then treat the re-ranker as the agent which observes the dialogue history h_t at each time step $t = 1, 2, ..., T$ and selects one of the K actions (i.e. candidate responses $a_t^1, ..., a_t^K$). The goal of the agent is to maximize the accumulated reward R:

$$R = \sum_{t=1}^{T} \gamma^t r_t \tag{12}$$

where $\gamma \in (0, 1]$ is a discount factor, and r_t is the reward after taking the action at time step t, which is the labeled 1-5 points of appropriateness in the MILABOT case. The MILABOT trained by the off-policy REINFORCE approach or the Q-learning approach achieved a great performance in terms of user average score, dialogue length and positive utterances.

However, the reinforcement learning method for the re-ranker design in MILABOT needs a lot of labeled data. For a multi-chatbot framework that only incorporates the retrieval-based and generation-based chatbots and only targets the one-round response generation to the request, there are some data-driven models for the re-ranker design [55,45]. For example, in *AliMe Chat*, Qiu et al. used an attentive Seq2Seq model to score the candidate answers (i.e. responses) with an input question [45] (Fig. 7 (a)). For each of the retrieved k candidates, the score function $s^{\text{Mean-Prob}}$ is defined as:

$$s^{\text{Mean-Prob}} = \frac{1}{n} \sum_{i=1}^{n} p(y_i = w_i | \theta_i) \tag{13}$$

where $w_1, ..., w_n$ are the word sequences of the candidate answer, and θ_i is the parameters for generating w_i at time step i. If the maximum candidate score is not less than a threshold T (determined by cross validation on top-1 accuracy), *AliMe Chat* selects that candidate as the final answer. Otherwise, it selects the answer generated by the same attentive Seq2Seq model. With such a re-ranker, *AliMe Chat* performs better than merely a retrieval-based chatbot in the online A/B test.

To further make use of the retrieved real responses, Song et al. proposed to feed the retrieved candidates to the generative model in addition to the original request, and re-rank both the retrieved candidates and generated candidate [55] (Fig. 7 (b)). They deployed a Gradient Boosting Decision Tree [68] in the re-ranker, and utilized several high-level features for training and scoring, such as term similarity, entity similarity, topic similarity, statistical machine translation, length and fluency. Trained on 1,606,741 query-reply pairs from Baidu Tieba, their chatbots ensemble framework outperformed other methods in terms of BLEU and human score.

Although a hybrid of different chatbots can help to handle cross-domain requests and generate more appropriate responses, it can not learn new skills outside the capability of the chatbot ensemble and will probably fail when facing out-of-domain requests. To deal with such circumstances, designers may need to add human intervention or user customization mechanisms into the chatbot, which will be introduced in the following two sections.

3.2 Human-chatbot Collaboration

As the fully automatic service chatbots are still potentially problematic by generating the wrong responses or taking unexpected actions, it is necessary to involve human workers in some failure-sensitive real-world domains. One

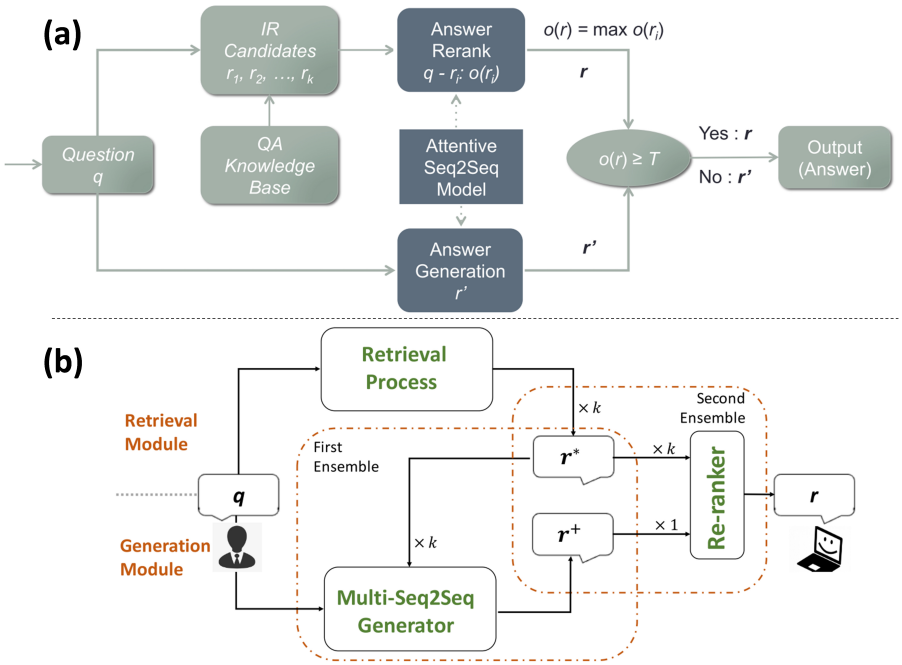


Fig. 7 Two multi-chatbot frameworks: (a) *AliMe Chat* that uses an attentive Seq2Seq model as the re-ranker [45]; (b) An ensemble of retrieval-based and generation-based chatbots that use a decision tree as the re-ranker [55].

of the main challenges in such human-chatbot collaboration scenarios is to enable the chatbots to learn a new action from a human worker for handling similar cases later. To handle this challenge, Luo et al. introduced CoChat, a dialogue manager framework that utilizes the MemHRNN (i.e., an external memory + hierarchical recurrent neural network) to enable human intervention and chatbot improvement at any time [34] (Fig. 8 (a)). Similar to the HRED introduced in Section 2.3, MemHRNN concatenates the utterance u^{m}, entity form v^{m} and taken action w^{m} features into a vector x^{m} in each user's turn, and then feeds this vector into a higher level recurrent network for dialogue encoding $z^{\mathrm{m}+1}$ in the next turn. The latest dialogue encoding $z^{\mathrm{m}+1}$, together with the current $\mathrm{u}^{\mathrm{m}+1}$ and $v^{\mathrm{m}+1}$, is then used to select action for the current turn through a two-layer fully-connected network. The external memory is introduced to address the one-shot learning challenges of the new actions by increasing their possibilities if their recorded dialogue states are similar to the current dialogue state. For example, when a human worker refuses all existing candidate actions and inputs a new action, the new action will be added into the action pool, and the corresponding dialogue state (i.e., entity form + context vector) will be stored in the memory. When a similar user utterance comes later, the possibility of selecting the newly added action

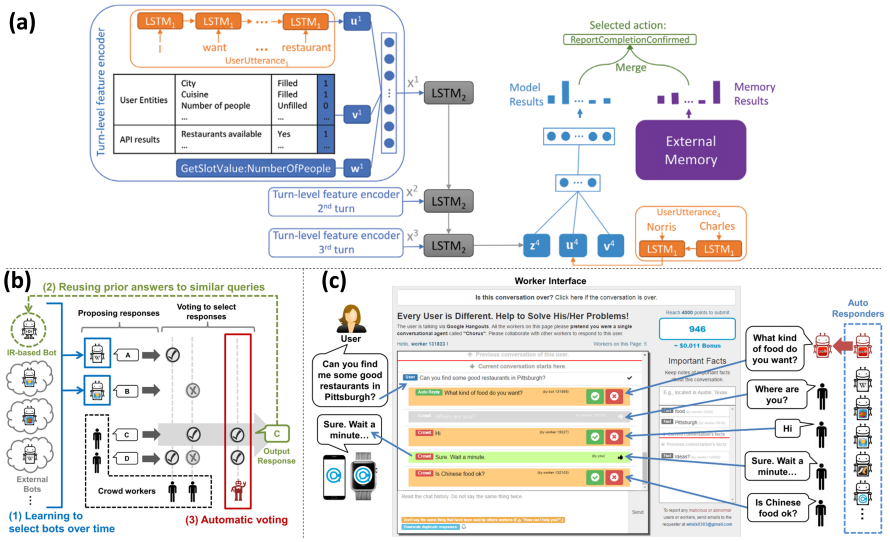


Fig. 8 Two human-chatbot collaboration frameworks: (a) CoChat that utilizes external memory to handle one-shot learning challenges for new action [34]; (b)&(c) Evorus that learns to automate itself from crowd workers over time [22].

from the HRNN is still low because of the limited samples, but inside the external memory the possibility is high and it is merged with the model results to increase its chances.

However, the MemHRNN above is only suitable in narrow domains with structured knowledge. To incorporate human workers with service chatbots in relatively unstructured domains, Huang et al. introduced Evorus, a crowd-powered chatbot built to automate itself over time [22] (Fig. 8 (b)(c)). Evorus selects the response to the user's request via a voting mechanism, in which each human worker votes for the candidate responses proposed by different chatbots and other human workers. With these upvotes and downvotes labels, the re-ranker of the multiple chatbots is trained to select candidate responses to the given request. Also, these labels are used to train a LibLinear [14] classifier for automatic voting, which aims to reduce human workload in the future. Another way to save human effort in Evorus is that it uses an information-retrieval-based (IR-based) method to find proper answers to similar queries in prior conversations. With such techniques, Evorus is able to improve itself over time. However, the cost paid for the crowd is expensive, and the answers will not be consistent.

The drawback of the human-chatbot collaboration method is that it always consumes human workers effort. In the cases that human intervention is the must, the future research should focus on providing a better experience for the human workers and improving the chatbots more quickly.

3.3 Learning from Users

As another way to enhance service chatbots' abilities, enabling the chatbots to directly learn from users' instructions can also help in handling more type of requests. For example, when chatbots serving as a smart phone personal agent to handle a request "order a cup of greentea", it is possible that the chatbot can not fulfill the user's request due to the undefined intent or lack of corresponding built-in APIs. In this case, a general response like "here is what I found from google" could be frustrating, while a mechanism for the user to teach it how to handle similar requests could be more helpful.

To this end, Li et al. designed a programming-by-demonstration system SUGLITE which enables the user to teach the chatbot how to execute unknown commands [28] (Fig. 9 (a)). Take the unknown "green tea ordering" command as an example, SUGILITE will answer "Sorry but I don't understand. Would you like to teach me?" The user can then say "yes" and start to demonstrate the procedure of ordering a cup of green tea via the drink ordering app (e.g., Starbucks) step by step. SUGLITE will guide and confirm with the user to finish the demonstration, and after that it parameterizes the script based on the command and the UI hierarchy of that app. The next time the user says "order a DRINK please", SUGLITE is able to understand and order that DRINK directly using that app if available. Alternatively, the user can view / edit the recording and script to deal with ambiguities and failure issues. The authors also showed that they can successfully teach SUGLITE in many other scenarios like "send *[AMOUNT]* dollars to *[NAME]*" and "show me the CHI paper by *[NAME]*". However, SUGLITE is limited to the structured tasks through structured apps. SUGLITE can not understand the semantics in the demonstration and be applied in web-based applications.

Apart from the graphical programming-by-demonstration techniques, using verbal instruction to teach the chatbots opens another opportunity to generate new commands. For example, in an email environment, Azaria et al. proposed a Learning by Instruction Agent (LIA), which allows users to teach it some primitive commands such as sending emails via solely natural language interaction [4] (Fig. 9 (b)). LIA uses a semantic parser to assign executable semantics to each natural language command (i.e., text command paired with logical form). During the verbal teaching, the user can teach LIA new concepts along with with their fields and concepts (e.g., "a contact has an email address" and "bob is a contact"). The user can also teach procedural knowledge such as how to forward an email by updating the semantic parser using known natural language instruction. The authors showed that people with little or no programming knowledge could easily teach LIA with very little training. Yet the rules behind each task are complicated and it is thus inconvenient to generalize LIA in other domains.

The chatbot learning from users through demonstration or verbal instruction is a promising way to enhance and customize the service chatbots. However, existing methods can only achieve it in narrow domains and need a lot of hand-crafted features. It will be an interesting research direction to lever-

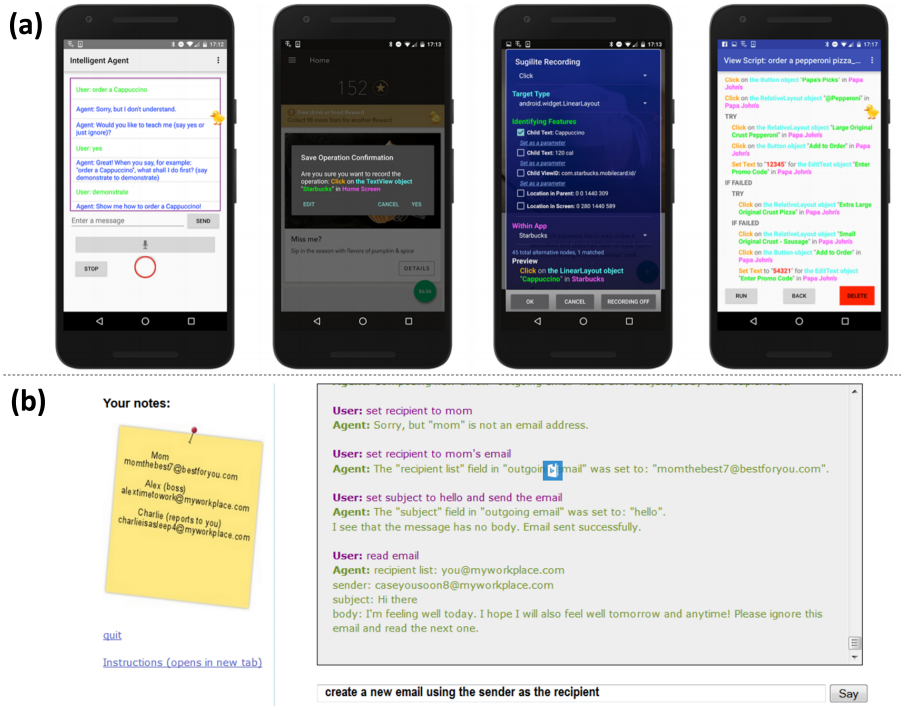


Fig. 9 Two chatbots learning from users examples: (a) SUGILITE, which learns how to execute new commands from user's demonstration [28]; (b) LIA, which learns new concepts or new procedural knowledge from user's verbal instructions [4].

Table 4 Summary of three kinds of chatbot enhancement methods.

Enhancement Methods	Techniques	Pro	Cons
Multi-chatbots Design	Reinforcement Learning for re-ranker policy	Can handle cross-domain requests; Make use of existing chatbots	Unable to learn new skills outside the capability of ensemble; Fails in out-of-domain requests
	Data-driven re-ranker models		
Human-chatbot Collaboration	CoChat: external memory + HRNN	Learn new skills from human workers; More robust and able to handle complexrequests	Expensive; Not consistent; Long delays
	Evorus: crowd-powered, automates itself over time		
Learning from Users	Programming by demonstration	Learn unknown commands from users; Friendly to novice	Limited to narrow domains; Need to design hand-crafted rules
	Verbal instruction		

age big data and machine learning to automatically learn from users and provide a better user experience.

3.4 Summary

According to the example and discussion above, we summarize the advantages and disadvantages of current techniques with three kinds of enhancement methods in Table 4. These techniques show a promising future for building a stronger and a more user-friendly service chatbot. The construction of these chatbots not only needs the steady development of natural language processing, but also the human-computer interaction design considerations of human workers as well as ended users. In the next section, we discuss two potential future directions on service chatbot design.

4 Future Directions

Previous construction methods of service chatbots normally suffer from out-of-context, universal response generation as well as inconsistent personality. In this section, we discuss how the recent Transformer and contextual embedding techniques can potentially help to generate more context-related responses. We also discuss the need for including personality into service chatbots and building a tone-aware chatbot.

4.1 Response Generation with Transformer and Contextual Embedding

Over the past decade, the success of many natural language processing (NLP) tasks (including the response generation given a request in this survey) is mainly built on the improvement of deep learning models and language models. Up to 2017, most state-of-the-art deep learning models on response generation were limited to the ensemble of sequential RNN and other components like LSTM, GRU, attention mechanism and memory network. These sequential models can capture the time information of the input sentences (e.g., how next word / sentence is conditioned on previous words / sentences), but largely ignore the global information of the whole sentence (e.g., a word could have different relations with all the words in the sentence). This could be the main reasons for the out-of-context and universal responses generated by these response generation models. Moreover, these models usually use pre-trained language models like word2vec [40] and Glove [42] to represent the input, while these language models are context-free, i.e. the word (e.g., blue) is always represented by the same vector regardless of different contexts (e.g., color or mood). These context-free language models may not properly represent the words and sentences in a certain context, which also limits the performance of the response generation models in previous service chatbots. Therefore, to generate a context-related and informative response, the service chatbots should better capture the global information of the requests. The recent developments in Transformer [60] and contextual embedding [43,13] point out a potentially better way for this purpose.

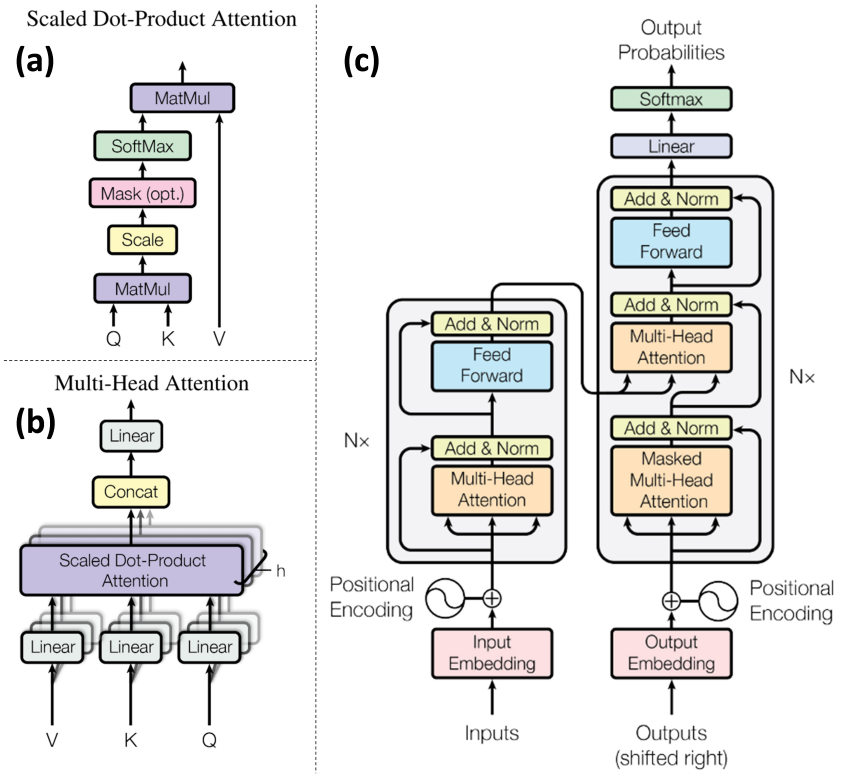


Fig. 10 The Transformer model [60]: (a) Scaled Dot-Product Attention; (b) Multi-Head Attention; (c) Model architecture.

Unlike RNN-based models which condition the next word on previous words, the Transformer enables words to connect with each other in the sentence. It uses Scaled Dot-Product Attention (Fig. 10 (a)), which can be defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{14}$$

where Q and K are a set of queries and keys of dimension d_k, and V is the values of dimension d_v paired with the keys. In the cases of machine translation or response generation, the Q, K, V can be set to the same set of words X in the sentence, i.e., self-attention. Then through the dot-production operation, each word (i.e., $q_t = x_t$) builds connections with other words (i.e., $K = V = X$) in the sentence. Furthermore the Transformer uses Multi-Head Attention (Fig. 10 (b)) to jointly attend to information from different representation subspaces at different positions, which is somehow similar to the idea of convolutional neural network. Multi-head attention can be formulated as:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \tag{15}$$

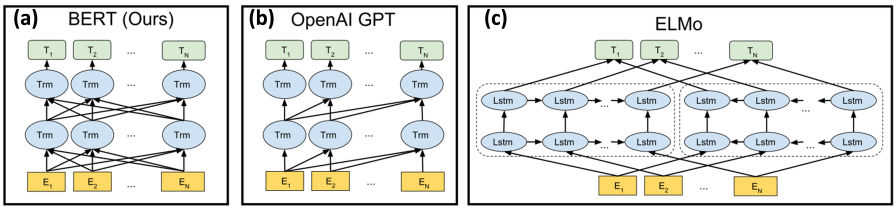


Fig. 11 The different pre-training models of contextual word embeddings [13]: (a) BERT [13] that uses a bidirectional Transformer; (b) OpenAI GPT [3] that uses a left-to-right Transformer; (c) ELMo [43] that uses the concatenation of independently trained left-to-right and right-to-left LSTM.

where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$, and W_i^Q, W_i^K, W_i^V, W^O are parameter matrices. The final Transformer model (Fig. 10 (c)) also takes positional encoding as another input to make use of the order of the sequence. The Transformer model has an encoder-decoder structure, where the encoder maps an put sequence $(x_1, ..., x_n)$ to a sequence of continuous representations $z = (z_1, ..., z_n)$, and the decoder generates an output sequence $(y_1, ..., y_m)$ of symbols one element at a time given z and previously generated symbols. The Transformer is shown to outperforms the best previously reported models on the WMT English-to-German translation task in terms of BLEU score. Since the task of response generation in service chatbots is similar to machine translation (MT) in some aspects, and previously it has successfully used MT models (e.g. SMT, Seq2Seq) for response generation, it will be a promising future direction to apply the Transformer to the service chatbots design (especially in retrieval-based and generation-based methods).

Besides the Transformer, the rapid development of contextual word embedding in 2018 is also an exciting direction for better response generation. Unlike context-free word embedding like word2vec and Glove, the contextual word embedding represents each word in the context of the sentences through some pre-training models like ELMo [43], OpenAI GPT [3] and BERT [13]. As shown in Fig. 11, these models take some context-free word embedding as the input, generate the features of the words and sentences inside the model, and output the contextual word embedding. ELMo (Fig. 11 (c)) uses the concatenation of independently trained left-to-right and right-to-left LSTM to generate features, and it improved the state-of-the-art results on some question answering and language inference tasks at that time. Later, the OpenAI GPT (Fig. 11 (b))uses a left-to-right Transformer to replace LSTM, which could potentially improve the model but is still unidirectional. The latest BERT (i.e., Bidirectional Encoder Representation from Transformers) model proposed by Devlin et al. [13] instead jointly conditions the representations on both the left and right context in all layers (Fig. 11 (a)). It takes the sum of the token embeddings, the segmentation embeddings and the positional embeddings as the input embeddings, and is trained on a masked LM (MLM) task and a next sentence prediction task. In the MLM task, 15% of

the tokens are randomly masked and the model is trained to predict these masked tokens. While the next sentence prediction task aims to capture the relationship between two text sentences. The upper models built on the pre-trained BERT representations achieve the current state-of-the-art results on eleven NLP tasks, including the sentence pair classification tasks and SQuAD v1.1 question answering. It is possible that applying these models to construct contextual word embeddings of the original request can also improve the model performance in response generation tasks for chatbots.

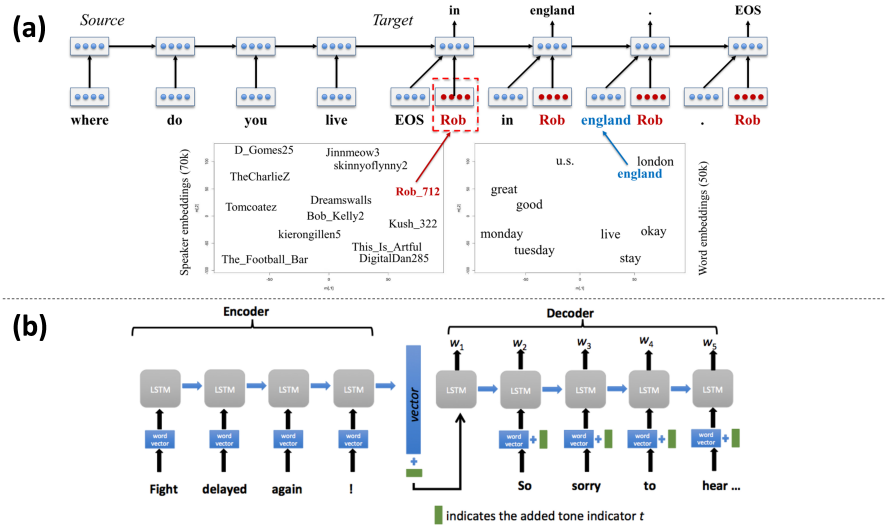
It has become a trend to apply the Transformer and contextual word embeddings to many NLP tasks. Similarly in the future, we can use the Transformer to replace the RNN, LSTM and GRU in previous retrieval-based and generation-based models, and use the contextual word embeddings as the new input features. With these techniques, the service chatbots are supposed to provide more appropriate and informative responses given the request.

4.2 Building Chatbot with Personality

Although recent progress in machine learning has enabled chatbots to generate more grammatically correct responses, it is still hard to design a chatbot that has consistent personality, which will have a significant impact on user experience [21]. A chatbot with a consistent personality should have a corresponding persona (e.g., background facts or profile [27]) and have a certain speaking style (e.g., toned responses [21]). The designers can easily design the personality for simple rule-based chatbots, but it becomes impossible when chatbots get more and more complicated. For retrieval-based and generation-based chatbots, a straightforward thought of designing a consistent personality is to build a highly consistent dataset, but it would be unrealistic as the dataset is usually diverse. Merging personality information into the response generation models, instead, is a promising way to design chatbots with a consistent personality in the wild.

In Li et al.'s work [27], additional speaker embeddings are concatenated with the word embeddings in the decoder of the standard Seq2Seq model (Fig. 12 (a)). The speaker model cluster users along some of the traits (e.g., age, country of residence) based on the responses alone, and the embeddings are jointly learned with word embeddings via backpropagation. Specifically during the training, each source-target pair has already been clustered into one corresponding speaker, and the randomly initiated speaker embeddings will be trained with all the pairs inside the same speaker cluster. Then given the same request as the input, the model can give a different answer based on the speakers' personality.

Inspired by this model in the open-domain, Hu et al. explored how to generate toned responses for customer care chatbots on social medial [21]. They first conducted a formative study to annotate the initial 53 tones and summarize them into eight major tones, i.e., *anxious, frustrated, impolite, passionate, polite, sad, satisfied* and *empathetic*. Then they established eight linear



Request	Your website is down and I really need to buy chair! ! ! No joke.
Passionate	Hi there! We are experiencing technical issues with our website. We are working to resolve this.
Empathetic	Hi I apologize for the inconvenience. Can you please dm your account or phone number?
Neutral	Hi there. Can you dm us your number so we can look into this for you?
Human	Hi Inga we were aware that the website was down and we apologize for this. It is now up and running again.

Fig. 12 Two example of building chatbots with personality: (a) Persona-based neural conversation model that uses speaker embeddings [27]; (b) tone-aware seq2seq model and examples of toned responses [21].

regression analyses for the eight major tones, and found that *empathetic* and *passionate* tones are beneficial for customer care. A keywords extraction was then conducted on these two selected tones. Using these keywords as indicators of different tones, their tone-aware seq2seq model concatenates the indicator and the word embedding in the decoder to train itself on the preprocessed conversations data with tone information (Fig. 12 (b)). Their user studied shows that the toned responses are perceived to be significantly more appropriate and helpful than the neural responses.

However, the two works above do not evaluate how different chatbot personalities could affect the end user engagement and how to automatically adjust the speaking style. In the future, we can investigate the user satisfaction relationship between the tones in the requests and the tones in the responses via the annotated data. We can then train a service chatbot that not only has a consistent personality, but also senses a user's emotional status and adjust its speaking style.

5 Conclusion

Using chatbots to assist or replace human workers is a trend in service industries, owing to chatbots being cheaper, more easy-to-access and more ob-

jective. However, it is still a huge challenge to design service chatbots which provide as satisfying responses to users' requests as human workers can. This survey presents the common construction methods of service chatbots, which can be classified into rule-based, retrieval-based and generation-based methods. We further present some enhancement methods, which can power the chatbots either by an ensemble of multiple chatbots, collaboration with human workers, or by instruction from the users. With the progress seen in both the Transformer and contextual embedding over the past two years, we discuss promising future directions in the design of chatbots which could better understand users' requests and generate more helpful responses. Moreover, we show a potential way to build service chatbots with the persona and speaking style of a real human, which can provide a better user experience.

References

1. Dialog flow (2018). Retrieved: 2018-12-10. https://dialogflow.com/
2. Ibm watson assistant (2018). Retrieved: 2018-12-10. https://assistant-us-south.watsonplatform.net/us-south/b3a5bd9b-9ea9-4be8-9ec7-145f04f69453/home
3. Alec Radford Karthik Narasimhan, T.S., Sutskever, I.: Improving language under- standing with unsupervised learning. In: Technical report, OpenAI (2018)
4. Azaria, A., Krishnamurthy, J., Mitchell, T.M.: Instructable intelligent personal agent. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI'16, pp. 2681–2689. AAAI Press (2016). URL http://dl.acm.org/citation.cfm?id=3016100.3016277
5. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. CoRR **abs/1409.0473** (2014)
6. Bengio, Y.: Learning deep architectures for ai. Found. Trends Mach. Learn. **2**(1), 1–127 (2009). DOI 10.1561/2200000006. URL http://dx.doi.org/10.1561/2200000006
7. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.: A neural probabilistic language model. J. Mach. Learn. Res. **3**, 1137–1155 (2003). URL http://dl.acm.org/citation.cfm?id=944919.944966
8. Chai, J., Lin, J., Zadrozny, W., Ye, Y., Stys-Budzikowska, M., Horvath, V., Kambhatla, N., Wolf, C.: The role of a natural language conversational interface in online sales: A case study. International Journal of Speech Technology **4**(3), 285–295 (2001). DOI 10.1023/A:1011316909641. URL https://doi.org/10.1023/A:1011316909641
9. Chen, H., Liu, X., Yin, D., Tang, J.: A survey on dialogue systems: Recent advances and new frontiers. SIGKDD Explor. Newsl. **19**(2), 25–35 (2017). DOI 10.1145/3166054.3166058. URL http://doi.acm.org/10.1145/3166054.3166058
10. Chung, J., ÄĞaglar GülÄğehre, Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. CoRR **abs/1412.3555** (2014)
11. Colby, K.M.: Artificial Paranoia: A Computer Simulation of Paranoid Processes. Elsevier Science Inc., New York, NY, USA (1975)
12. Cranshaw, J., Elwany, E., Newman, T., Kocielnik, R., Yu, B., Soni, S., Teevan, J., Monroy-Hernández, A.: Calendar.help: Designing a workflow-based scheduling agent with humans in the loop. In: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, CHI '17, pp. 2382–2393. ACM, New York, NY, USA (2017). DOI 10.1145/3025453.3025780. URL http://doi.acm.org/10.1145/3025453.3025780
13. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. CoRR **abs/1810.04805** (2018)
14. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: Liblinear: A library for large linear classification. J. Mach. Learn. Res. **9**, 1871–1874 (2008). URL http://dl.acm.org/citation.cfm?id=1390681.1442794

15. Ferrucci, D.A., Brown, E.W., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A., Lally, A., Murdock, J.W., Nyberg, E., Prager, J.M., Schlaefer, N., Welty, C.A.: Building watson: An overview of the deepqa project. AI Magazine **31**(3), 59–79 (2010). URL http://dblp.uni-trier.de/db/journals/aim/aim31.html#FerrucciBCFGKLMNPSW10
16. Ghazvininejad, M., Brockett, C., Chang, M.W., Dolan, W.B., Gao, J., tau Yih, W., Galley, M.: A knowledge-grounded neural conversation model. In: AAAI (2018)
17. Grudin, J., Jacques, R.: Chatbots, humbots, and the quest for artificial general intelligence. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI '19, pp. 209:1–209:11. ACM, New York, NY, USA (2019). DOI 10.1145/3290605.3300439. URL http://doi.acm.org/10.1145/3290605.3300439
18. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). DOI 10.1162/neco.1997.9.8.1735. URL http://dx.doi.org/10.1162/neco.1997.9.8.1735
19. Hu, B., Lu, Z., Li, H., Chen, Q.: Convolutional neural network architectures for matching natural language sentences. In: Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (eds.) Advances in Neural Information Processing Systems 27, pp. 2042–2050. Curran Associates, Inc. (2014). URL http://papers.nips.cc/paper/5550-convolutional-neural-network-architectures-for-matching-natural-language-sentences.pdf
20. Hu, B., Lu, Z., Li, H., Chen, Q.: Convolutional neural network architectures for matching natural language sentences. In: Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (eds.) Advances in Neural Information Processing Systems 27, pp. 2042–2050. Curran Associates, Inc. (2014). URL http://papers.nips.cc/paper/5550-convolutional-neural-network-architectures-for-matching-natural-language-sentences.pdf
21. Hu, T., Xu, A., Liu, Z., You, Q., Guo, Y., Sinha, V., Luo, J., Akkiraju, R.: Touch your heart: A tone-aware chatbot for customer care on social media. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI '18, pp. 415:1–415:12. ACM, New York, NY, USA (2018). DOI 10.1145/3173574.3173989. URL http://doi.acm.org/10.1145/3173574.3173989
22. Huang, T.H.K., Chang, J.C., Bigham, J.P.: Evorus: A crowd-powered conversational assistant built to automate itself over time. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI '18, pp. 295:1–295:13. ACM, New York, NY, USA (2018). DOI 10.1145/3173574.3173869. URL http://doi.acm.org/10.1145/3173574.3173869
23. Jain, M., Kota, R., Kumar, P., Patel, S.N.: Convey: Exploring the use of a context view for chatbots. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI '18, pp. 468:1–468:6. ACM, New York, NY, USA (2018). DOI 10.1145/3173574.3174042. URL http://doi.acm.org/10.1145/3173574.3174042
24. Ji, Z., Lu, Z., Li, H.: An information retrieval approach to short text conversation. CoRR **abs/1408.6988** (2014)
25. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. J. Artif. Int. Res. **4**(1), 237–285 (1996). URL http://dl.acm.org/citation.cfm?id=1622737.1622748
26. Lee Isbell Jr, C., J. Kearns, M., P. Kormann, D., Singh, S., Stone, P.: Cobot in lambdamoo: A social statistics agent. pp. 36–41 (2000)
27. Li, J., Galley, M., Brockett, C., Spithourakis, G., Gao, J., Dolan, B.: A persona-based neural conversation model. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 994–1003. Association for Computational Linguistics (2016). DOI 10.18653/v1/P16-1094. URL http://aclweb.org/anthology/P16-1094
28. Li, T.J.J., Azaria, A., Myers, B.A.: Sugilite: Creating multimodal smartphone automation by demonstration. In: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, CHI '17, pp. 6038–6049. ACM, New York, NY, USA (2017). DOI 10.1145/3025453.3025483. URL http://doi.acm.org/10.1145/3025453.3025483
29. Li, Y., Luo, X., Zheng, Y., Xu, P., Fu, H.: Sweepcanvas: Sketch-based 3d prototyping on an rgb-d image. In: Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology, pp. 387–399. ACM (2017)

30. Liao, Q.V., Mas-ud Hussain, M., Chandar, P., Davis, M., Khazaeni, Y., Crasso, M.P., Wang, D., Muller, M., Shami, N.S., Geyer, W.: All work and no play? In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI '18, pp. 3:1–3:13. ACM, New York, NY, USA (2018). DOI 10.1145/3173574.3173577. URL http://doi.acm.org/10.1145/3173574.3173577
31. Liu, C.W., Lowe, R., Serban, I., Noseworthy, M., Charlin, L., Pineau, J.: How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2122–2132. Association for Computational Linguistics (2016). DOI 10.18653/v1/D16-1230. URL http://aclweb.org/anthology/D16-1230
32. Lowe, R., Pow, N., Serban, I., Pineau, J.: The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In: SIGDIAL Conference (2015)
33. Lu, Z., Li, H.: A deep architecture for matching short texts. In: C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger (eds.) Advances in Neural Information Processing Systems 26, pp. 1367–1375. Curran Associates, Inc. (2013). URL http://papers.nips.cc/paper/5019-a-deep-architecture-for-matching-short-texts.pdf
34. Luo, X., Lin, Z., Wang, Y., Nie, Z.: Cochat: Enabling bot and human collaboration for task completion. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (2018)
35. Luong, M.T., V. Le, Q., Sutskever, I., Vinyals, O., Kaiser, L.: Multi-task sequence to sequence learning. Proceedings of ICLR, San Juan, Puerto Rico (2015)
36. Magazine, C.: Chatbot report 2018: Global trends and analysis (2018). Retrieved: 2018-12-10. https://chatbotsmagazine.com/chatbot-report-2018-global-trends-and-analysis-4d8bbe4d924b
37. Mauldin, M.L.: Chatterbots, tinymuds, and the turing test: Entering the loebner prize competition. In: Proceedings of the Twelfth National Conference on Artificial Intelligence (Vol. 1), AAAI '94, pp. 16–21. American Association for Artificial Intelligence, Menlo Park, CA, USA (1994). URL http://dl.acm.org/citation.cfm?id=199288.199285
38. Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tur, D., He, X., Heck, L., Tur, G., Yu, D., Zweig, G.: Using recurrent neural networks for slot filling in spoken language understanding. Trans. Audio, Speech and Lang. Proc. **23**(3), 530–539 (2015). URL http://dl.acm.org/citation.cfm?id=2817174.2817185
39. Microsoft: Cognitive services: Language understanding (luis) (2018). Retrieved: 2018-12-10. https://www.luis.ai/home
40. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS'13, pp. 3111–3119. Curran Associates Inc., USA (2013). URL http://dl.acm.org/citation.cfm?id=2999792.2999959
41. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (2002). URL http://aclweb.org/anthology/P02-1040
42. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: EMNLP, vol. 14, pp. 1532–1543 (2014)
43. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. In: Proc. of NAACL (2018)
44. Qiu, M., Li, F.L., Wang, S., Gao, X., Chen, Y., Zhao, W., Chen, H., Huang, J., Chu, W.: Alime chat: A sequence to sequence and rerank based chatbot engine. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 498–503. Association for Computational Linguistics (2017). DOI 10.18653/v1/P17-2079. URL http://aclweb.org/anthology/P17-2079
45. Qiu, M., Li, F.L., Wang, S., Gao, X., Chen, Y., Zhao, W., Chen, H., Huang, J., Chu, W.: Alime chat: A sequence to sequence and rerank based chatbot engine. In: ACL (2017)
46. Ramesh, K., Ravishankaran, S., Joshi, A., Chandrasekaran, K.: A survey of design techniques for conversational agents. In: S. Kaushik, D. Gupta, L. Kharb, D. Chahal (eds.) Information, Communication and Computing Technology, pp. 336–350. Springer Singapore, Singapore (2017)

47. Ritter, A., Cherry, C., Dolan, W.B.: Data-driven response generation in social media. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11, pp. 583–593. Association for Computational Linguistics, Stroudsburg, PA, USA (2011). URL http://dl.acm.org/citation.cfm?id=2145432.2145500
48. Schlesinger, A., O'Hara, K.P., Taylor, A.S.: Let's talk about race: Identity, chatbots, and ai. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI '18, pp. 315:1–315:14. ACM, New York, NY, USA (2018). DOI 10.1145/3173574.3173889. URL http://doi.acm.org/10.1145/3173574.3173889
49. Schuster, M., Paliwal, K.: Bidirectional recurrent neural networks. Trans. Sig. Proc. **45**(11), 2673–2681 (1997). DOI 10.1109/78.650093. URL http://dx.doi.org/10.1109/78.650093
50. Serban, I., Sankar, C., Germain, M., Zhang, S., Lin, Z., Subramanian, S., Kim, T., Pieper, M., Chandar, A.P.S., Ke, N.R., Mudumba, S., de Brébisson, A., Sotelo, J., Suhubdy, D., Michalski, V., Nguyen, A., Pineau, J., Bengio, Y.: A deep reinforcement learning chatbot. CoRR **abs/1709.02349** (2017)
51. Serban, I.V., Sordoni, A., Bengio, Y., Courville, A., Pineau, J.: Building end-to-end dialogue systems using generative hierarchical neural network models. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI'16, pp. 3776–3783. AAAI Press (2016). URL http://dl.acm.org/citation.cfm?id=3016387.3016435
52. Shang, L., Lu, Z., Li, H.: Neural responding machine for short-text conversation. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 1577–1586. Association for Computational Linguistics (2015). DOI 10.3115/v1/P15-1152. URL http://aclweb.org/anthology/P15-1152
53. Shawar, B.A., Atwell, E.: Chatbots: Are they really useful? LDV Forum **22**, 29–49 (2007)
54. Shum, H.y., He, X.d., Li, D.: From eliza to xiaoice: challenges and opportunities with social chatbots. Frontiers of Information Technology & Electronic Engineering **19**(1), 10–26 (2018). DOI 10.1631/FITEE.1700826. URL https://doi.org/10.1631/FITEE.1700826
55. Song, Y., Li, C.T., Nie, J.Y., Zhang, M., Zhao, D., Yan, R.: An ensemble of retrieval-based and generation-based human-computer conversation systems. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, pp. 4382–4388. International Joint Conferences on Artificial Intelligence Organization (2018). DOI 10.24963/ijcai.2018/609. URL https://doi.org/10.24963/ijcai.2018/609
56. Sukhbaatar, S., szlam, a., Weston, J., Fergus, R.: End-to-end memory networks. In: C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, R. Garnett (eds.) Advances in Neural Information Processing Systems 28, pp. 2440–2448. Curran Associates, Inc. (2015). URL http://papers.nips.cc/paper/5846-end-to-end-memory-networks.pdf
57. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14, pp. 3104–3112. MIT Press, Cambridge, MA, USA (2014). URL http://dl.acm.org/citation.cfm?id=2969033.2969173
58. Sutton, R.S., Barto, A.G.: Introduction to Reinforcement Learning, 1st edn. MIT Press, Cambridge, MA, USA (1998)
59. TURING, A.M.: Computing machinery and intelligence. Mind **LIX**(236), 433–460 (1950). DOI 10.1093/mind/LIX.236.433. URL http://dx.doi.org/10.1093/mind/LIX.236.433
60. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (eds.) Advances in Neural Information Processing Systems 30, pp. 5998–6008. Curran Associates, Inc. (2017). URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf
61. Vinyals, O., Le, Q.V.: A neural conversational model. CoRR **abs/1506.05869** (2015)
62. Walker, M., S. Aberdeen, J., Boland, J., Bratt, E., S. Garofolo, J., Hirschman, L., N. Le, A., Lee, S., Narayanan, S., Papineni, K., L. Pellom, B., Polifroni, J., Potamianos, A., Prabhu, P., Rudnicky, A., Sanders, G., Seneff, S., Stallard, D., Whittaker, S.: Darpa communicator dialog travel planning systems: the june 2000 data collection. pp. 1371–1374 (2001)
63. Wallace, R.S.: The Anatomy of A.L.I.C.E., pp. 181–210. Springer Netherlands, Dordrecht (2009). DOI 10.1007/978-1-4020-6710-5_13. URL https://doi.org/10.1007/978-1-4020-6710-5_13

64. Weizenbaum, J.: Eliza—a computer program for the study of natural language communication between man and machine. Commun. ACM **9**(1), 36–45 (1966). DOI 10.1145/365153.365168. URL http://doi.acm.org/10.1145/365153.365168
65. WIKIPEDIA: Xiaoice (2018). Retrieved: 2018-12-10. https://en.wikipedia.org/wiki/Xiaoice
66. Xu, A., Liu, Z., Guo, Y., Sinha, V., Akkiraju, R.: A new chatbot for customer service on social media. In: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, CHI '17, pp. 3506–3510. ACM, New York, NY, USA (2017). DOI 10.1145/3025453.3025496. URL http://doi.acm.org/10.1145/3025453.3025496
67. Yan, R., Song, Y., Wu, H.: Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '16, pp. 55–64. ACM, New York, NY, USA (2016). DOI 10.1145/2911451.2911542. URL http://doi.acm.org/10.1145/2911451.2911542
68. Ye, J., Chow, J.H., Chen, J., Zheng, Z.: Stochastic gradient boosted distributed decision trees. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM '09, pp. 2061–2064. ACM, New York, NY, USA (2009). DOI 10.1145/1645953.1646301. URL http://doi.acm.org/10.1145/1645953.1646301
69. Young, S., GaÅąiÄĞ, M., Thomson, B., Williams, J.D.: Pomdp-based statistical spoken dialog systems: A review. Proceedings of the IEEE **101**(5), 1160–1179 (2013). DOI 10.1109/JPROC.2012.2225812
70. Zhou, X., Dong, D., Wu, H., Zhao, S., Yu, D., Tian, H., Liu, X., Yan, R.: Multi-view response selection for human-computer conversation. In: EMNLP (2016)
71. Zhou, X., Li, L., Dong, D., Liu, Y., Chen, Y., Zhao, W.X., Yu, D., Wu, H.: Multi-turn response selection for chatbots with deep attention matching network. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1118–1127. Association for Computational Linguistics (2018). URL http://aclweb.org/anthology/P18-1103