JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 39, 671-689 (2023)
DOI: 10.6688/JISE.202305_39(3).0014

A Dialogue Model for Customer Support Services

TING-YI KUO AND ANTHONY J. T. LEE+
Department of Information Management
National Taiwan University
Taipei, 106 Taiwan
E-mail: tinye1021@gmail.com; jtlee@ntu.edu.tw

Many dialogue models have been proposed to learn the language model from the input queries for answering user requests. However, most models are not proposed for customer support services. Some shed light on answering user queries in a customer support system; however, they do not consider domain or emotion features implicitly hidden in user queries. In this study, we propose a deep learning framework to automatically answer user queries of customer support services. The proposed framework extracts domain and emotion features from user queries and then incorporates the extracted features into a generative adversarial networks model to generate the response to an input query. The extracted domain features may reveal user needs while the extracted emotion features may show the emotions implicitly hidden in the input queries. Therefore, the proposed model can better understand user requests and generate better responses. The experimental results show that our proposed framework outperforms the comparing methods and can generate better responses for user queries. Our framework may help companies provide 24/7/365 customer support services with less effort.

Keywords: generative adversarial networks, deep learning, attention mechanism, latent Dirichlet allocation model, customer support services

1. INTRODUCTION

Customer support services are essential for businesses to maintain high satisfaction and gain customer loyalty. With good customer support services, companies may be able to better understand customer needs and build long-term relationships with customers. However, it is time-consuming and inefficient to gather user feedback with human resources. A report from Forbes shows that the average cost of a Twitter response could be reduced to 1/6 when a company switches the service from traditional calls to call channels on online platforms.[1] In addition, a study of Marketing Land indicates that 72% of consumers expect a response within an hour while complaining to a brand on Twitter.[2]

As the form of customer support services has gradually changed from offline phone calls to online platforms, customers' expectations for online customer support, such as quick responses and all-day services, have been steadily growing.[3] To reduce the cost of customer support services while maintaining service quality, many companies have developed chatbots for providing round-the-clock services to fulfill customer needs. However,

Received February 5, 2022; revised April 29, 2022; accepted May 30, 2022.
Communicated by Mi-Yen Yeh.
[1] How to Use Twitter for Customer Service, https://www.forbes.com/sites/shephyken/2016/04/30/how-to-use-twitter-for-customer-service/#18afd1eb49f2.
[2] 72% of People Who Complain on Twitter Expect a Response Within an Hour, https://marketingland.com/study-72-of-consumers-expect-brands-to-respond-within-an-hour-to-complaints-posted-on-twitter-63496.
[3] Why Customer Service Needs to Be Faster than Ever, https://www.shopify.com/enterprise/94678726-the-need-for-speed-why-customer-service-needs-to-be-faster-than-ever.
+ Corresponding author.

many chatbots are hand-coding, with predefined template rules of responses for some queries [16, 17, 19, 29]. Thus, these chatbots cannot cover all the queries and learn domain knowledge from user queries.

On the other hand, many deep learning dialogue models [2, 13-15, 18, 20-25, 28, 31, 33, 34] have been proposed to learn the language model from the input queries for answering user questions. However, most models do not consider emotion and domain features. Some studies added an additional feature to their models. Prior studies [11, 22, 34] embedded emotions into their models to capture users' feelings. Ghazvininejad *et al.* [6] extracted the external facts relevant to user queries from Foursquare and embedded them into their model for answering the queries on Twitter. Jiang *et al.* [12] extracted the world knowledge from Wikipedia and then proposed a model to consider the utterance subject drifts. However, these models are not proposed for customer support services and do not consider the domain and emotion features. The domain features may reveal user needs while the emotion features may indicate user emotions hidden in user queries. Therefore, these models may be not good for customer support services.

For customer support services, Xu *et al.* [27] applied the seq2seq model to generate the response to a user query. Hu *et al.* [9] presented a model by incorporating three manually-labelled tones into their model for generating a tone-aware response. However, it may be time-consuming and require considerable human effort to annotate the training data. Therefore, we propose a framework to automatically extract emotion and domain features from user queries and incorporate both features into the proposed model to generate a response for better meeting user needs.

Generative adversarial networks (GAN) have demonstrated good performance in modeling real data distributions and generating synthetic data that are highly similar to real data [8]. They have been well applied in many applications, such as dialogue models [15], image generation [32], text generation [10, 29], and age progression [26]. In this study, based on GAN, we propose a deep learning framework to automatically answer user queries in customer support services. The proposed framework contains three phases. First, we employ the latent Dirichlet allocation (LDA) model [3] to extract a sequence of domain feature vectors for each query. Next, we employ the SenticNet 5 sentiment dictionary [5] to derive a sequence of emotion feature vectors for each query. Finally, based on the extracted domain and emotion feature vectors, we propose a GAN-based model to automatically generate the response to an input query.

The contributions of this study are summarized as follows. First, based on GAN, we propose a dialogue model for customer support services by considering domain knowledge and emotions in user queries. Our model can automatically extract domain and emotion features from user queries. The extracted domain features may reveal user needs while the extracted emotion features may show user feelings hidden in the query. Incorporating these two features into our model can generate better responses. Next, our study has advanced the literature by incorporating state, domain and emotion attention mechanisms into a dialogue model. We believe that this extension can broaden the applications as well as enrich the research on understanding user requests in dialogue systems. Last, our proposed framework can help companies automatically answer users' requests and provide 24/365 customer support services with less effort.

The rest of this paper is organized as follows. We survey the related literature in Section 2 and then present our proposed framework in Section 3. Next, we evaluate the performance of the proposed framework in Section 4. Finally, concluding remarks and future work are made in Section 5.

2. RELATED WORK

Many dialogue generation methods [16, 17, 19, 29] have been proposed. These methods can be classified into the following two categories: rule-based and retrieval-based [27]. Ritter *et al.* [19] developed a statistical machine translation model, a rule-based model, for generating a response to reduce the restrictions of word-based translation. Misu *et al.* [16] proposed a rule-based model by using pre-existing query-response pairs to generate the most relevant response. Yu *et al.* [30] used a retrieval-based model to calculate the score of each response in the pre-existing dialogue database and retrieve the response with the highest score to answer the input query. Nakano and Komatani [17] used an expert selection module to select experts for managing various types of dialogues, where each expert had different dialogue knowledge. These models require human experts to manually define rules, keywords and matching methods of generating the responses. Therefore, they can only deal with predefined cases and be difficult to extend for specific tasks.

On the other hand, many deep learning dialogue models have been proposed. Sutskver *et al.* [24] built a sequence to sequence (seq2seq) model by using an encoder to transform a user query into a representation and a decoder to decode the transformed representation for generating a response. Shang *et al.* [21] used a recurrent neural network (RNN) to model dialogue generation in short conversations of a microblogging style such as Twitter. Serban *et al.* [20] presented a hierarchical encoder-decoder framework for dialogue generation and demonstrated that pretraining word embeddings can improve the performance of response generation. To avoid generating dull, irrelevant, and repetitive sentences, Li *et al.* [13] used the maximum mutual information [1] between the input query and the response to reduce the generic responses. Su *et al.* [23] proposed a restaurant dialogue system based on the reinforcement learning approach. Nuruzzaman and Hussain [18] presented a dialogue model for the insurance industry. Zhang *et al.* [31] used user attributes to collect relevant responses for building a personalized task-oriented dialogue system. Li *et al.* [14] solved the misalignment problem in the seq2seq model by using the dialogs simulated by two virtual agents, which rewarded the generated sequences with a predefined reward function. Based on GAN, Li *et al.* [15] built a dialogue model. Belainine *et al.* [2] presented an encoder-decoder framework with a multidimension attention mechanism for dialogue generation. Wang *et al.* [25] proposed an information-enhanced hierarchical self-attention network to answer user queries. Yan *et al.* [28] developed a framework to incorporate the unsupervised translation alignment to learn the shared information between different languages for training multilingual question-answering mapping and generation. Zhao *et al.* [33] presented a weighted heterogeneous graph-based dialogue system for disease diagnosis. However, these models do not consider domain knowledge and emotions implicitly hidden in user queries.

Some studies incorporate an additional feature into their models to enrich the diversity of generated responses. Zhou *et al.* [34] added an emotion embedding to generate a response. Song *et al.* [22] extended the seq2seq model with a lexicon-based attention mechanism to increase the probability of emotion words being generated. Huang *et al.* [11]

proposed a dialogue model to express prespecified emotions in the responses and showed that their proposed model, called Enc-att, outperformed the comparing models. Although those models consider emotions in user queries, they do not consider domain knowledge, which can be used to generate responses for better meeting user needs. In addition, Ghazvininejad *et al.* [6] presented a model to answer user queries on Twitter by embedding the external facts into their model, where the external facts were relevant to the domain features of user queries and extracted from another platform such as Foursquare. Jiang *et al.* [12] proposed a model to consider the utterance subject drifts by facilitating knowledge selection and incorporation in a dialogue system, where the knowledge was retrieved from Wikipedia. However, the external facts extracted from Foursquare and the knowledge retrieved from Wikipedia may be not good for customer support services.

For customer support services, Xu *et al.* [27] applied the seq2seq model to generate the response to a user query. Hu *et al.* [9] presented a model by incorporating three tones (*i.e.*, passionate, empathetic and neutral tones) into their model for generating a tone-aware response. Since the tones are manually labeled, it may be time-consuming and require considerable human effort to annotate the data. Table 1 shows the differences between our framework and prior models.

Table 1. Differences between our framework and prior models.

	CSS	EF	DF	GAN
[27]	√			
[6, 12]			√	
[9]	√	√		
[11, 22, 34]		√		
[15]				√
Ours	√	√	√	√

Note: CSS stands for customer support services, EF for emotion features, DF for domain features, and GAN for generative adversarial networks.

3. THE PROPOSED FRAMEWORK

In this section, we propose a framework to automatically answer user requests in customer support services, where each request may contain multiple turns of queries and responses, and can be represented as a dialogue thread (thread hereafter). Fig. 1 shows an example thread, where a human agent suggests a solution to resolve the user request, and "direct messages" are sent by the human agent and can only be seen by the intended recipient. We call the responses generated by the human agent real responses.

The proposed framework contains three phases, as shown in Fig. 2. First, we employ the LDA model [3] to cluster user queries into various topics, derive a domain feature vector for each word, and generate a sequence of domain feature vectors for each query. Next, we employ the SenticNet 5 sentiment dictionary [5] to generate a sequence of emotion feature vectors for each query. Finally, based on the generated domain and emotion feature sequences, we propose a GAN-based model to automatically generate the response to an input query.

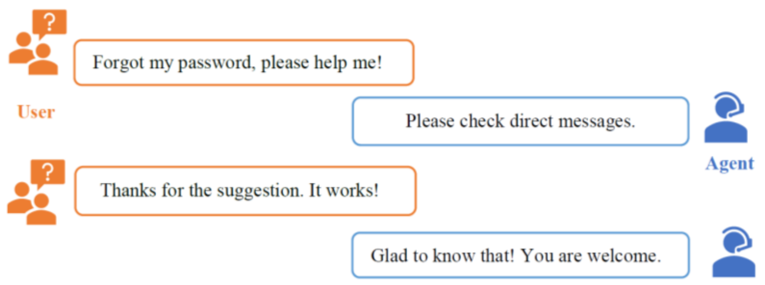
Forgot my password, please help me!
User
Please check direct messages.
Agent
Thanks for the suggestion. It works!
Glad to know that! You are welcome.

Fig. 1. An example thread.

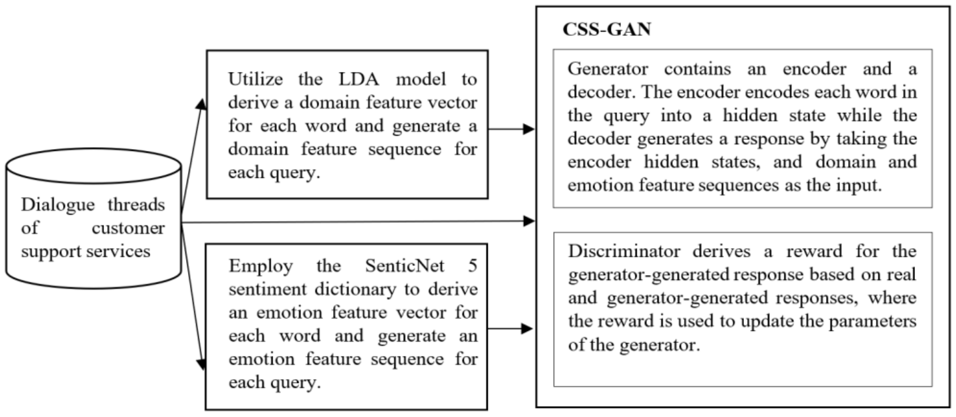
Dialogue threads of customer support services
Utilize the LDA model to derive a domain feature vector for each word and generate a domain feature sequence for each query.
Employ the SenticNet 5 sentiment dictionary to derive an emotion feature vector for each word and generate an emotion feature sequence for each query.
CSS-GAN
Generator contains an encoder and a decoder. The encoder encodes each word in the query into a hidden state while the decoder generates a response by taking the encoder hidden states, and domain and emotion feature sequences as the input.
Discriminator derives a reward for the generator-generated response based on real and generator-generated responses, where the reward is used to update the parameters of the generator.

Fig. 2. The proposed framework.

3.1 Domain Features

For each query, we remove stop words, lemmatize the inflected forms of words and save the remaining words as a document. Then, we employ the LDA model to build a topic model for these documents. After the topic model is built, each word has a probability belonging to each topic and is represented as a domain feature vector, where the *i*th element of the vector denotes the probability of the word belonging to the *i*th topic.

Next, we represent each query as a domain feature sequence, $(d_1, d_2, \ldots, d_M)$, where M is the number of words in the query. d_j is the domain feature vector of the *j*th word of the query if the *j*th word is not a stop word, and is a null vector otherwise, $1 \leq j \leq M$. Note that we do not remove any stop words while generating the domain feature sequence so that each domain word in the query can be aligned with its domain feature vector, where the domain feature vector of each stop word is represented as a null vector.

Fig. 3 shows an example of generating a domain feature sequence for a query, "Forgot my password." The query is lemmatized into "forget my password" and then converted into a domain feature sequence by the LDA model, where the number of topics is 5. "forget" is represented as a domain feature vector, (0, 0, 0.873, 0.073, 0.054), where its probability belonging to each topic is 0 to the first and second topics, 0.873 to the third, 0.073 to the fourth, and 0.054 to the fifth. Since "my" is a stop word, it is represented as a null vector. "password" is represented as (0, 0.043, 0.91, 0, 0.047). The domain feature sequence of the query is the sequence formed by these three domain feature vectors.

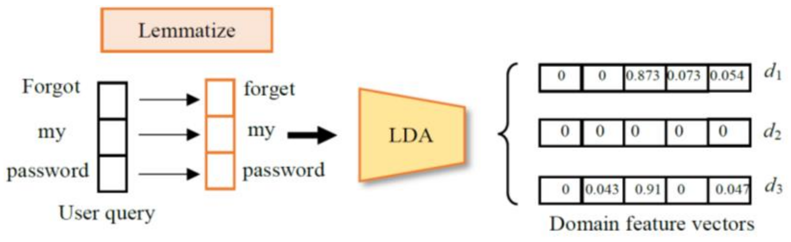

Fig. 3. Example of deriving a domain feature vector sequence.

3.2 Emotion Features

Next, we employ the SenticNet 5 sentiment dictionary [5] to derive an emotion feature sequence for each query, where each emotion word is denoted by its primary mood and five sentiment attribute values, namely, "polarity," "pleasantness," "attention," "sensitivity," and "aptitude." "polarity" stands for positive or negative favor of the word, "pleasantness" for the user's happiness towards the service provided, "attention" for the user's interest in the provided information, "sensitivity" for the user's comfort with the interface, and "aptitude" for the user's willingness to use the application [4].

SenticNet 5 classifies moods into 8 categories, namely, "admiration," "surprise," "fear," "disgust," "sadness," "joy," "anger," and "interest." Thus, each emotion word can be represented by a 13-dimensional emotion feature vector, where the first eight elements in the vector represent the one-hot encoding of eight moods, which denotes the presence of the corresponding mood, and the remaining five elements represent the five sentiment attribute values.

We represent each query as an emotion feature sequence, $(e_1, e_2, \ldots, e_M)$, where e_j is the emotion feature vector of the jth word of the query if the jth word is an emotion word, and is a null vector otherwise, $1 \leq j \leq M$. Fig. 4 shows an example of deriving an emotion feature sequence for a query, "please help me." "please" and "help" are emotion words; however, "me" is not. The primary moods of "please" and "help" are "interest" and "joy," respectively. The one-hot encoding of "interest" is (0, 0, 0, 0, 0, 0, 0, 1). The sentiment attribute values of "please" are (0.92, 0, 0.96, 0, 0.89). Thus, "please" is represented by a 13-dimensional emotion feature vector (0, 0, 0, 0, 0, 0, 0, 1, 0.92, 0, 0.96, 0, 0.89). Similarly, "help" is represented by (0, 0, 0, 0, 0, 1, 0, 0, 0.82, 0.97, 0.8, 0, 0.7). Since "me" is not an emotion word, it is represented by a null vector (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0). Therefore, the emotion feature sequence of the query is the sequence formed by these three emotion feature vectors.

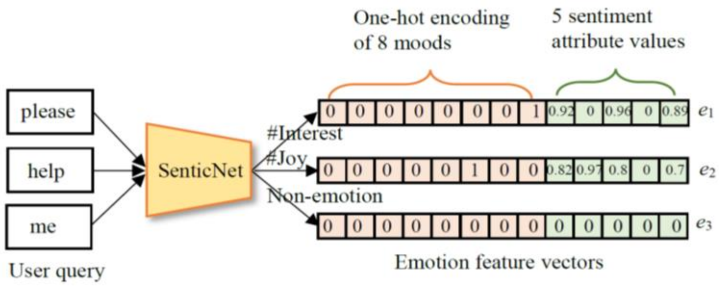
One-hot encoding of 8 moods
5 sentiment attribute values
please
help
me
User query
SenticNet
#Interest
#Joy
Non-emotion
0 0 0 0 0 0 0 1 0.92 0 0.96 0 0.89 e_1
0 0 0 0 0 1 0 0 0.82 0.97 0.8 0 0.7 e_2
0 0 0 0 0 0 0 0 0 0 0 0 0 e_3
Emotion feature vectors

Fig. 4. Example of deriving an emotion feature sequence.

3.3 Customer Support Service Model

In this section, we propose a customer support service GAN model, called CSS-GAN, to automatically answer user queries. CSS-GAN contains a generator and a discriminator. The generator attempts to generate a response as close to real responses as possible to fool the discriminator, while the discriminator aims to distinguish between generator-generated responses and real ones.

3.3.1 Generator

Given an input query $X = \{x_1, x_2, \ldots, x_M\}$, the generator will generate a response, $Y = \{y_1, y_2, \ldots, y_N\}$, to the query, where x_m denotes the mth word of the query, y_n denotes the nth word of the response, M is the number of words in the query, N is the number of words in the response, $\{x_1, x_2, \ldots, x_M\}$ denotes the concatenation of all the words in X, $1 \leq m \leq M$, $1 \leq n \leq N$. We extend Li *et al.*'s model [15] by incorporating three attention mechanisms, namely, state, domain, and emotion, into our model for implementing the generator.

Fig. 5 shows the architecture of the generator, where each hexahedron denotes an attention mechanism, and the encoder and decoder are implemented by *Gated Recurrent Units* (GRU). The encoder encodes the user query into a sequence of hidden states. The decoder employs three attention mechanisms to generate the response by using the encoder hidden states as well as domain and emotion feature vectors. The attention mechanisms help the decoder pay various attention to some hidden states (the state attention mechanism), certain domain feature vectors (the domain attention mechanism) and some emotion feature vectors (the emotion attention mechanism) when generating each word of the response.

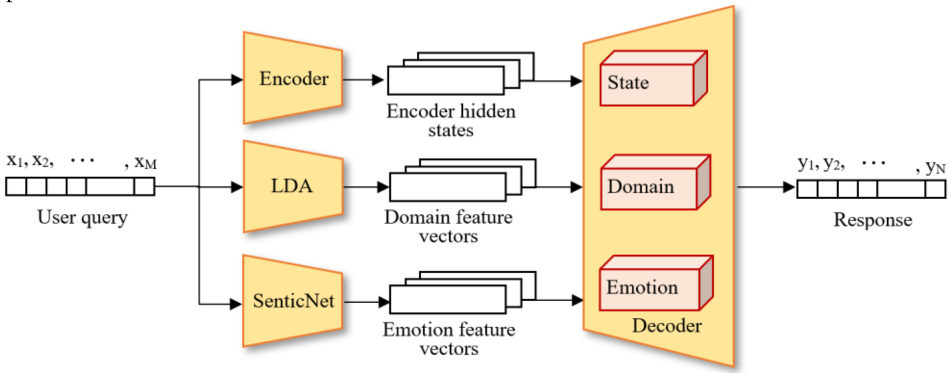

Fig. 5. Generator architecture.

(1) Encoder

The encoder aims at extracting the contextual information of the input query by converting the input query into a sequence of encoder hidden states. Specifically, the encoder is implemented by Eq. (1), where h_m denotes the encoder hidden state at time step m, representing the contextual information of the sequence $\{x_1, x_2, \ldots, x_m\}$, $1 \leq m \leq M$. That is, the encoder derives a hidden state h_m at time step m based on the previous hidden state h_{m-1} and the input word x_m. The encoder derives the hidden state for each time step, $m = 1, 2, \ldots,$

M. Thus, there are M encoder hidden states.

$$h_m = GRU(x_m, h_{m-1}) \tag{1}$$

(2) Decoder

The decoder employs three attention mechanisms, namely, state, domain, and emotion, to generate the response to the input query. First, the decoder uses the state attention mechanism to derive an attentive state context vector by assigning different attention weights to each encoder hidden state, where a higher weight enables the decoder to pay more attention to the corresponding encoder hidden state in the decoding process. Thus, the encoder hidden state with a higher weight will contribute more to the derivation of the attentive state context vector. Specifically, the state attention mechanism is implemented by Eqs. (2)-(4), where s_{n-1} denotes the decoder hidden state at time step $n - 1$, a_m denotes the attention weight of h_m with respect to s_{n-1}, $\{s_{n-1}, h_m\}$ denotes the concatenation of s_{n-1} and h_m, $c_{h,n}$ denotes the attentive state context vector at time step n, and W_1 and W_2 denote learnable parameter matrices. That is, at time step n (of the decoder), based on its previous state s_{n-1}, the decoder uses the state attention mechanism to assign different weights to each encoder hidden state and then utilizes these weights to derive the attentive state context vector $c_{h,n}$. Note that the encoder and decoder have their own time steps since they both are implemented by GRU.

$$a_m = \frac{\exp(score(s_{n-1}, h_m))}{\sum_{i=1}^{M} \exp(score(s_{n-1}, h_i))} \tag{2}$$

$$score(s_{n-1}, h_m) = W_1 tanh(W_2\{s_{n-1}, h_m\}) \tag{3}$$

$$c_{h,n} = \sum_{i=1}^{M} a_m h_m \tag{4}$$

Next, the decoder uses the domain attention mechanism to derive an attentive domain context vector. The domain attention mechanism enables the decoder to pay more attention to some domain feature vectors in the decoding process. The decoder derives the attentive domain context vector $c_{d,n}$ at time step n by replacing h_m with d_m and the learnable parameter matrices (W_1, W_2) with new ones (W_3, W_4) in Eqs. (2)-(4), where d_m denotes the mth element of the domain feature sequence. That is, at time step n, based on its previous state s_{n-1}, the decoder uses the domain attention mechanism to derive the attentive domain context vector $c_{d,n}$.

Similarly, the decoder uses the emotion attention mechanism to derive an attentive emotion context vector. It derives the attentive emotion context vector $c_{e,n}$ at time step n by replacing h_m with e_m and the learnable parameter matrices (W_1, W_2) with new ones (W_5, W_6) in Eqs. (2)-(4), where e_m denotes the mth element of the emotion feature sequence. That is, the decoder uses the emotion attention mechanism to derive the attentive emotion context vector $c_{e,n}$.

Last, the decoder exploits the three derived attentive context vectors to derive its hidden state and the output word at time step n as defined in Eqs. (5) and (6), where y denotes a candidate word, each vocabulary in the input queries is a candidate, $y_{<n} = \{y_1, y_2, \ldots, y_{n-1}\}$, *softmax* denotes the softmax function, W_i denotes a learnable parameter matrix, $i = 7 - 12$. That is, to derive its hidden state at time step $n(s_n)$, the decoder consolidates its previous hidden state s_{n-1}, the previous output word y_{n-1}, and these three attentive context vectors

together by Eq. (5). Then, it uses s_n to derive the probability of each candidate word by Eq. (6). Based on the derived probabilities, the decoder samples a word as the output word at time step $n(y_n)$. That is, the higher the probability of a candidate word, the more likely that the candidate word is selected as y_n.

$$s_n = tanh(W_7 y_{n-1} + W_8 s_{n-1} + W_9 c_{h,n} + W_{10} c_{d,n} + W_{11} c_{e,n}) \tag{5}$$

$$p(y|X, y_{<n}) = softmax(W_{12} s_n) \tag{6}$$

After y_n is selected, $y_{<n+1}$ is rewarded by the discriminator to reflect how close $y_{<n+1}$ is to real responses. Based on the reward given by the discriminator, the generator guides the decoder to generate the output word one by one for generating a response close to real ones. To do so, once a new output word y_k is generated, $y_{<k+1}$ is rewarded by the discriminator. Then, the generator updates its learnable parameters, including the learnable parameters in the encoder and decoder, to maximize the reward given by the discriminator. The procedure is performed repeatedly until the whole response Y is generated.

The loss function of the generator is defined in Eq. (7), where $D_\phi(X, Y)$, calculated by the discriminator D_ϕ, denotes the probability of the generator-generated response being a real one, G_θ denotes the generator, θ denotes the learnable parameters of G_θ and ϕ denotes the learnable parameters of D_ϕ. The gradient of the loss function is defined in Eq. (8), which is approximated by the likelihood ratios [7]. Therefore, the generator updates its parameters to maximize the reward by Eq. (9), where r denotes a learning rate.

$$L(G_\theta) = -E_{Y\sim P(Y|X)}[\log(D_\phi(X, Y)|\theta)] \tag{7}$$

$$\nabla_\theta L(G_\theta) = -\sum_{n=1}^{N} E_{y_n \sim p(y_n|X, y_{<n})}[\nabla_\theta \log(p(y_n \mid X, y_{<n}))\cdot D_\phi(X, y_{<n})] \tag{8}$$

$$\theta \leftarrow \theta - r\nabla_\theta L(G_\theta) \tag{9}$$

3.3.2 Discriminator

The discriminator, implemented by hierarchical recurrent neural networks, aims to distinguish between generator-generated responses and real responses. It serves as a binary classifier by taking real responses as positive samples and generator-generated responses as negative ones.

The generator and discriminator are trained by adversarial learning [8], where they update their learnable parameters iteration by iteration. For each iteration, the generator is trained to update its learnable parameters for generating a response as close as to real ones, *i.e.*, for maximizing the reward given by the discriminator, where the reward reflects how close the generator-generated response is to real ones. On the other hand, based on the generator-generated and real responses, the discriminator is trained to update its learnable parameters for identifying that the generator-generated responses are not real, *i.e.*, for minimizing the rewards of generator-generated responses. That is, in each iteration, the generator is trained to maximize the reward given by the discriminator while the discriminator is trained to minimize the rewards of generator-generated responses. By updating the learnable parameters in the generator and discriminator iteration by iteration, the generator keeps improving for generating a better response close to real ones while the discriminator keeps enhancing its identification capability. The training process will be performed repeatedly until the prespecified number of iterations is reached. After the training is finished, the generator is ready to answer user queries.

The loss function of the discriminator is defined in Eq. (10), where $E_{Y\sim P_{data}}[\log(D_\phi(X, Y))]$ denotes the expected reward for real responses and $E_{Y\sim P(Y|X)}[\log(D_\phi(X, Y))]$ denotes the expected reward for generator-generated responses. By minimizing the loss function, the goal of the discriminator is to maximize the rewards of real responses while minimizing the rewards of generator-generated ones.

$$L(D_\phi) = -E_{Y\sim P_{data}}[\log(D_\phi(X, Y))] - E_{Y\sim P(Y|X)}[1 - \log(D_\phi(X, Y))] \tag{10}$$

4. EXPERIMENTAL RESULTS

In this section, we evaluate the performance of our proposed model. We first provide an overview of the datasets, the data preprocessing and the experimental setup in Section 4.1. Then, we evaluate our proposed model in Section 4.2.

4.1 Data Preprocessing and Experiment Setup

We collect the customer support dialogue data from Twitter, a website of widely adopted for customer support services. Sony PlayStation's customer support service on Twitter has more than 1.53 million followers.[4] Also, Xbox's customer support service on Twitter has more than 1.58 million followers.[5] Therefore, we extract the dialogue threads of PlayStation and Xbox from the pre-collected dataset – "The Customer Support on Twitter" from Kaggle,[6] which contains queries and responses from the biggest brands on Twitter.

Next, we extract multiturn threads and remove non-English threads. For each thread, we remove mentions (@), hashtags (#) and emojis, and replace URLs with "_url_." Then, we evaluate the performance of our proposed model on single-turn and multiturn threads. For the multiturn threads, we focus on the two-turn threads, which are the majority of multiturn threads in our dataset. For the single-turn (multiturn) threads, we randomly sample 90% threads for training and the rest of them for testing. The statistics of our datasets are shown in Table 2. There are 9,290 single-turn threads, 8,361 threads for training and 929 threads for testing.

Table 2. Statistics of PlayStation and Xbox datasets.

	PlayStation		Xbox	
	Single-turn	Multi-turn	Single-turn	Multi-turn
Training	8,361	3,242	15,239	4,685
Testing	929	360	1,693	521

Next, we use the perplexity, a widely used metric for the LDA model, to decide the number of topics. The perplexity is a measurement of how well a probability model predicts a sample and may be used to compare various probability models. The lower the

[4] AskPlaystation, https://twitter.com/askplaystation
[5] XboxSupport, https://twitter.com/xboxsupport
[6] Customer Support on Twitter, https://www.kaggle.com/thoughtvector/customer-support-on-twitter

perplexity is, the better the probability model. Fig. 6 shows the perplexity of the LDA model versus the number of topics for the PlayStation dataset. Since the perplexity is lowest when the number of topics is 28, we set the number of topics to 28 for the PlayStation dataset in the following experiments. Similarly, we set the number of topics to 33 for the Xbox dataset.

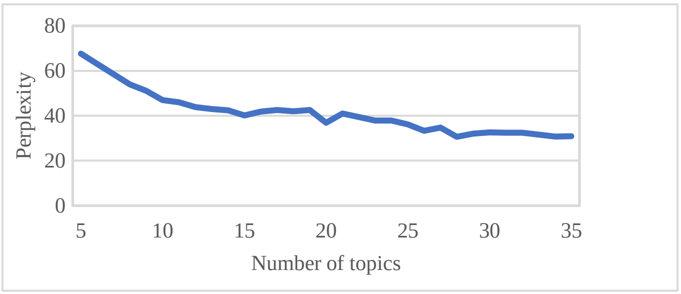

Fig. 6. Perplexities of the LDA models.

4.2 Performance Evaluation

We compare our proposed framework with three models, namely, the model of Xu *et al.* [27], the model of Li *et al.* [15], and the Enc-att model [11]. Xu *et al.* [27] utilized the seq2seq model to develop a dialogue model for customer support services. Based on the seq2seq model, Li *et al.* [15] proposed the first GAN-based model for dialogue generation. The Enc-att model generates a response with a prespecified emotion. Thus, it is required to specify an emotion in the Enc-att model. We choose the emotion "joy" for the Enc-att model to generate the responses since "joy" is the most frequent emotion in our dataset.

Table 3 shows the response time of answering a query for each model. The GAN-based models spend about 0.27 and 0.4 second in answering a query in the PlayStation and Xbox datasets, respectively. They require more response time than the other two models since they contain more learnable parameters. It appears that each model can respond a query in a timely manner.

Table 3. Response time.

Model	PlayStation (sec)	Xbox (sec)
Xu *et al.*'s	0.198	0.122
Enc-att	0.137	0.243
Li *et al.*'s	0.269	0.398
CSS-GAN w/o D	0.274	0.393
CSS-GAN w/o E	0.267	0.493
CSS-GAN	0.260	0.437

4.2.1 Automatic evaluation

The BLEU and ROUGH scores are widely-used metrics for dialogue generation. The BLEU score measures how many n-grams in the generated response appear in the real response while the ROUGE score measures how many *n*-grams in the real response appear in the generated response. It is often that a dialogue model with a high BLEU score has a low ROUGE score, and vice versa. A dialogue model with a high BLEU score tends to generate a generic response to direct user requests to human agents. To reduce the cost of human agents in customer support services, following the previous study [27], we use the BLEU score as the metrics to evaluate the performance of our proposed model.

(1) Single-turn

Tables 4 and 5 respectively show the BLEU score of each method for single-turn responses on PlayStation and Xbox, where CSS-GAN w/o D is a variant of CSS-GAN without using domain feature vectors, CSS-GAN w/o E is a variant of CSS-GAN without using emotion feature vectors, and Li *et al.*'s is also a variant of CSS-GAN without using both domain and emotion feature vectors. Xu *et al.*'s performs worst. This is because all the models are based on the seq2seq model. Moreover, all the models except Xu *et al.*'s integrate additional features or GAN-based framework to capture more dialogue patterns. The variants of our proposed model perform better than Enc-att since the GAN-based framework may help our variants generate responses with more words overlapping with the ones generated by human agents. That is, the reward given by the discriminator can effectively guide the generator to generate better responses. In comparison with Li *et al.*'s model, a significant performance increase of BLEU score can be observed in the other variants of our model namely CSS-GAN w/o D, CSS-GAN w/o E and CSS-GAN, which indicates that the model with domain or emotion feature vectors can effectively learn how to generate the responses close to the ones generated by human agents. The BLEU scores in Table 5 are lower than those in Table 4. It is because the responses in PlayStation are with typical patterns whereas those of Xbox are more diverse.

Table 4. BLEU score of each method for single-turn responses on PlayStation.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Xu *et al.*'s	11.13%	7.81%	7.30%	7.02%
Enc-att	13.91%	8.86%	8.02%	7.49%
Li *et al.*'s	14.83%	11.25%	10.68%	10.36%
CSS-GAN w/o D	18.71%	12.61%	11.70%	11.34%
CSS-GAN w/o E	21.50%	14.43%	13.22%	12.57%
CSS-GAN	**22.91%**	**15.14%**	**13.84%**	**13.21%**

Table 5. BLEU score of each method for single-turn responses on Xbox.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Xu *et al.*'s	9.08%	6.09%	5.61%	5.26%
Enc-att	10.22%	6.86%	6.3%	5.87%
Li *et al.*'s	10.91%	7.38%	6.79%	6.38%
CSS-GAN w/o D	13.85%	8.87%	8.04%	7.55%
CSS-GAN w/o E	14.65%	10.15%	9.49%	9.1%
CSS-GAN	**16.22%**	**11.04%**	**10.11%**	**9.5%**

(2) Multi-turn

Tables 6 and 7 respectively show the BLEU scores of each method for multi-turn responses on PlayStation and Xbox. Similar to the results in single-turn responses, all the variants of our models perform better than the other comparing methods in all BLEU scores, and CSS-GAN outperforms all the other models. Li *et al.*'s and Enc-att outperform Xu *et al.*'s model in both datasets, which shows utilizing either an attention mechanism on hidden states or an attention mechanism on emotions can help a model learn a better language model from dialogue threads. The patterns can also be seen in the comparison between CSS-GAN w/o D and Li *et al.*'s, which indicates adding an attention mechanism on emotion feature vectors in CSS-GAN w/o D is as well beneficial for learning the language model in customer support service dialogs.

CSS-GAN w/o E outperforms Li *et al.*'s on every BLEU score with at least 4%, higher than CSS-GAN w/o D, which shows adding an attention mechanism on domain feature vectors can help the model capture domain features in the input query and thus generate responses that are more correlated to the query. Similar to the results of single-turn, the BLEU scores in Table 7 are lower than those in Table 6.

Table 6. BLEU score of each method for multi-turn responses on PlayStation.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Xu *et al.*'s	12.04%	8.17%	7.61%	7.36%
Enc-att	15.28%	9.24%	8.18%	7.55%
Li *et al.*'s	15.84%	11.0%	10.3%	10.02%
CSS-GAN w/o D	19.56%	14.31%	13.71%	13.69%
CSS-GAN w/o E	23.24%	16.79%	15.86%	15.58%
CSS-GAN	**24.13%**	**18.15%**	**17.44%**	**17.36%**

Table 7. BLEU score of each method for multi-turn responses on Xbox.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Xu *et al.*'s	10.3%	6.52%	5.88%	5.45%
Enc-att	11.03%	7.56%	6.99%	6.55%
Li *et al.*'s	11.52%	7.77%	7.17%	6.75%
CSS-GAN w/o D	15.59%	10.66%	9.99%	9.7%
CSS-GAN w/o E	16.68%	11.92%	11.37%	11.19%
CSS-GAN	**18.33%**	**12.81%**	**12.09%**	**11.84%**

In multi-turn responses, CSS-GAN w/o D (CSS-GAN w/o E) outperforms Li *et al.*'s on every BLEU score with at least 3.3% (5.5%) on PlayStation and 2.8% (4.1%) on Xbox, respectively. CSS-GAN even yields a 7% improvement in BLEU score over Li *et al.*'s on PlayStation. It indicates that the domain and emotion attention mechanisms can catch more 2-grams, 3-grams or 4-grams from multi-turn dialogue threads for response generations since they can help GAN-based models to better attend on domain and emotion information hidden in a longer dialogue thread. In addition, most of the BLEU scores in multi-turn are higher than those in single-turn as shown in Tables 4-7. For the BLEU-2, BLEU-3 and BLEU-4 scores, the increment of each score from single-turn to multi-turn for CSS-GAN w/o D, CSS-GAN w/o E, and CSS-GAN models is larger than those of the other models. This is because multi-turn dialogue threads contain more information than single-turn threads, which is beneficial for a dialogue model to capture the latent semantics, emotion and domain features.

4.2.2 Human evaluation

To evaluate the responses generated by each model, we employ 10 human experts to evaluate the responses. All experts are proficient English users and have graduated from universities. Next, we prepare two evaluation sets of dialogue threads namely ST and MT. ST is used to evaluate the single-turn responses while MT is used to evaluate the multi-turn responses. For ST, we sample 10 threads from each single-turn dataset, and use the first query of each sampled thread as the input and the first response as the ground-truth. For MT, we sample 5 threads for MT from each multi-turn dataset. For each sampled thread, we merge the first query, the first response and the second query together, and then use the merged result as the input and the second response as the ground-truth.

Following the previous studies [19, 27], we employ three evaluation metrics to evaluate the response quality (1) Appropriateness: Whether a response is on the same topic as the user query, natural to the input query and with proper grammar [19]; (2) Helpfulness: Whether a response contains useful and specific advice that can help users to address the query [27]; (3) Emotion: Whether the emotional expression of a response can be captured by human, that is, whether human judges consider the response make them feel valued [27]. The ratings of metrics are on a 5-point scale from strongly disagree (1) to strongly agree (5) for evaluating how the responses meet with the given rating criterion of each metric.

To evaluate the generated responses, experts are asked to learn the 3 metrics with definitions and examples. Next, they are given a user query with the response generated by each model, and asked to rate these responses on the 3 metrics for each evaluation set. The experts' ratings on each metric of each model is averaged by the number of queries and the number of experts. That is, the response quality is measured by the average ratings.

Tables 8 and 9 show the average human evaluation scores of each method for single-turn responses on PlayStation and Xbox while Tables 10 and 11 present the average human evaluation scores of each method for multi-turn responses. CSS-GAN significantly outperforms all the comparing models in all three metrics (Wilcoxon signed-rank test, $p < 0.05$ for all three metrics). The Enc-att model performs better than Xu *et al.*'s and Li *et al.*'s models on all the metrics in single-turn responses but worse than Li *et al.*'s in multi-turn responses. It shows that utilizing emotion vectors to generate responses might help the Enc-att model better capture the emotions of short input queries, whereas GAN-based models are good at capturing the latent information for multi-turn dialogue.

Table 8. Average human evaluation score of each method for single-turn responses on PlayStation.

Metrics / Model	Appropriateness	Helpfulness	Emotion
Xu *et al.*'s	2.56*	2.52*	2.68*
Enc-att	3.04*	2.98*	3.18*
Li *et al.*'s	2.74*	2.64*	2.69*
CSS-GAN	**3.9**	**3.92**	**3.62**

Note: For the significant level of CSS-GAN outperforming the comparing model, '**' denotes extremely significant ($p < 0.001$), and '*' denotes significant ($p < 0.05$).

Table 9. Averge human evaluation score of each method for single-turn responses on Xbox.

Metrics / Model	Appropriateness	Helpfulness	Emotion
Xu *et al.*'s	2.36**	2.32**	2.47**
Enc-att	2.86**	2.8**	3.0*
Li *et al.*'s	2.66**	2.58**	2.63*
CSS-GAN	**3.89**	**3.88**	**3.58**

Table 10. Average human evaluation score of each method for multi-turn responses on PlayStation.

Metrics / Model	Appropriateness	Helpfulness	Emotion
Xu *et al.*'s	2.84*	2.7*	3.01*
Enc-att	2.96*	2.82*	3.15*
Li *et al.*'s	3.26*	3.13*	3.22*
CSS-GAN	**4.02**	**3.9**	**4.12**

On both PlayStation and Xbox, the scores of Li *et al.*'s model are higher than those of Xu *et al.*'s in both single-turn and multi-turn, which shows that Li *et al.*'s model can generate responses with more words overlapping with the human-generated ones and thus results in higher automatic evaluation scores as well as higher human evaluation scores. Li *et al.*'s model performs worse than Enc-att in single-turn; however, it outperforms Enc-att in multi-turn. This is because experts might be influenced by the length or emotion of sentences due to the nature of customer support services which not only aim to solve specific problems but also require comforting user feelings as well [27]. Our model uses the domain and emotion attention mechanisms to pay various attention to user emotions and needs when answering the queries. Thus, it outperforms the comparing methods in terms of appropriateness, helpfulness, and emotion. In addition, the scores in Xbox are slightly lower than those of PlayStation. This is because the responses in Xbox are rather lively, whereas those of PlayStation are politer, and thus they might attract human judges more in customer support services.

Table 11. Average human evaluation score of each method for multi-turn responses on Xbox.

Metrics / Model	Appropriateness	Helpfulness	Emotion
Xu *et al.*'s	2.48*	2.38*	2.8*
Enc-att	2.75*	2.6*	2.93*
Li *et al.*'s	3.17*	3.04*	3.12*
CSS-GAN	**3.92**	**3.7**	**3.92**

5. CONCLUSIONS AND FUTURE WORK

In this study, we propose a framework to automatically answer user requests for customer support services. The proposed framework first employs the LDA model to generate a domain feature sequence and the SenticNet 5 sentiment dictionary to derive an emotion feature sequence for each input query. Next, it employs a GAN-based model to generate the response to an input query by using three attention mechanisms, namely, state, domain, and emotion. The state attention mechanism can learn the contextual information from dialogue threads. The domain attention mechanism can learn user needs from the queries while the emotion attention mechanism can learn the emotions implicitly hidden in dialogue threads. Therefore, our model can better understand user requests and generate better responses.

The experimental results show that our proposed model significantly outperforms the comparing models in terms of BLEU scores. For automatic evaluation, Xu *et al.*'s performs worst. The variants of our model perform better than Enc-att since the GAN-based framework helps our variants generate responses with more words overlapping with the ones generated by human agents. In comparison with Li *et al.*'s model, each variant of our model has a significant performance increase of BLEU score, which shows that the domain and emotion attention mechanisms can catch more n-grams from multi-turn dialogue threads for response generations. Thus, our model significantly outperforms the comparing models. For human evaluation, the scores of Li *et al.*'s model are higher than those of Xu *et al.*'s in both single-turn and multi-turn. Li *et al.*'s model performs worse than Enc-att in single-turn; however, it outperforms Enc-att in multi-turn. Our model uses the domain and emotion attention mechanisms to pay various attention to user emotions and needs when answering the queries. Therefore, it outperforms the comparing methods in terms of appropriateness, helpfulness, and emotion.

In the future, we may extend our model in the following directions. First, we evaluate our model by using the PlayStation and Xbox datasets. We may further evaluate our model by using different customer support service datasets and examine how our model is affected by different datasets to further improve the proposed model. Second, we focus on two-turn dialogue threads for the multiturn experiment. We may further evaluate our model with longer multiturn dialogue threads to evaluate the performance of our model. Third, the emojis and emoticons can sometimes provide certain emotional hints. We may incorporate them into our model in the future. Last, we may take time factors into consideration, incorporate the time factors into our model, and observe how customer support services are affected by the time factors.

ACKNOWLEDGEMENTS

The authors are grateful to the anonymous referees for their helpful comments and suggestions. This research was supported in part by the Ministry of Science and Technology, Taiwan under Grant No. MOST 109-2410-H-002-073-MY3 and National Taiwan University under Grant No. NTU-CC-109L891401.

REFERENCES

1. L. Bahl, P. Brown, P. Souza, and R. Mercer, "Maximum mutual information estimation of hidden Markov model parameters for speech recognition," in *Proceeding of IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1986,

pp. 49-52.

2. B. Belainine, F. Sadat, and M. Boukadoum, “End-to-end dialogue generation using a single encoder and a decoder cascade with a multidimension attention mechanism,” *IEEE Transactions on Neural Networks and Learning Systems*, 2022.
3. D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent Dirichlet allocation,” *Journal of Machine Learning Research*, Vol. 3, 2003, pp. 993-1022.
4. E. Cambria, A. Livingstone, and A. Hussain, “The hourglass of emotions,” in *Proceedings of International Conference on Cognitive Behavioural Systems*, 2012, pp. 144-157.
5. E. Cambria, S. Poria, D. Hazarika, and K. Kwok, “SenticNet 5: Discovering conceptual primitives for sentiment analysis by means of context embeddings,” in *Proceedings of AAAI International Conference on Artificial Intelligence*, 2018, pp. 1795-1802.
6. M. Ghazvininejad, C. Brockett, M. W. Chang, B. Dolan, J. Gao, W. T. Yih, and M. Galley, “A knowledge-grounded neural conversation model,” in *Proceedings of AAAI International Conference on Artificial Intelligence*, 2018, pp. 5110-5117.
7. P. Glynn, “Likelihood ratio gradient estimation for stochastic systems,” *Communications of the ACM*, Vol. 33, 1990, pp. 75-84.
8. I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, “Generative adversarial nets,” in *Proceedings of International Conference on Neural Information Processing Systems*, Vol. 2, 2014, pp. 2672-2680.
9. T. Hu, X. Anbang, Z. Liu, *et al.*, “Touch your heart: A tone-aware chatbot for customer care on social media,” in *Proceedings of CHI International Conference on Human Factors in Computing Systems*, 2018, pp. 1-12.
10. Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing, “Toward controlled generation of text,” in *Proceedings of International Conference on Machine Learning*, Vol. 70, 2017, pp. 1587-1596.
11. C. Huang and O. R. Zaiane, “Generating responses expressing emotion in an open-domain dialogue system,” in *Proceedings of International Workshop on Chatbot Research, in Conjunction with the International Conference on Internet Science*, 2019, pp. 100-112.
12. B. Jiang, J. Yang, C. Yang, *et al.*, “Knowledge augmented dialogue generation with divergent facts selection,” *Knowledge-Based Systems*, Vol. 210, 2020, p. 106479.
13. J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan, “A diversity-promoting objective function for neural conversation models,” in *Proceedings of Internatonal Conference of North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 110-119.
14. J. Li, W. Monroe, A. Ritter, *et al.*, “Deep reinforcement learning for dialogue generation,” in *Proceedings of International Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics*, 2016, pp. 1192-1202.
15. J. Li, W. Monroe, T. Shi, *et al.*, “Adversarial learning for neural dialogue generation,” in *Proceedings of International Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2157-2169.
16. T. Misu, K. Georgila, A. Leuski, and D. Traum, “Reinforcement learning of question-answering dialogue policies for virtual museum guides,” in *Proceedings of Annual*

Meeting of the Special Interest Group on Discourse and Dialogue, 2012, pp. 84-93.
17. M. Nakano and K. Komatani, "A framework for building closed-domain chat dialogue systems," *Knowledge-Based Systems*, Vol. 204, 2020, p. 106212.
18. M. Nuruzzaman and O. K. Hussain, "IntelliBot: A dialogue-based chatbot for the insurance industry," *Knowledge-Based Systems*, Vol. 196, 2020, p. 105810.
19. A. Ritter, C. Cherry, and W. B. Dolan, "Data-driven response generation in social media," in *Proceedings of International Conference on Empirical Methods in Natural Language Processing*, 2011, pp. 583-593.
20. I. V. Serban, A. Sordoni, Y. Bengio, A. C. Courville, and J. Pineau, "Building end-to-end dialogue systems using generative hierarchical neural network models," in *Proceedings of Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence*, 2016, pp. 3376-3784.
21. L. Shang, Z. Lu, and H. Li, "Neural responding machine for short-text conversation," in *Proceedings of Association for Computational Linguistics-International Joint Conference on Natural Language Processing*, 2015, pp. 1577-1586.
22. Z. Song, X. Zheng, L. Liu, M. Xu, and X. Huang, "Generating responses with a specific emotion in dialog," in *Proceedings of Annual Meeting of Association for Computational Linguistics*, 2019, pp. 3685-3695.
23. P. H. Su, M. Gasšíc, N. Mrkšíc, *et al.*, "On-line active reward learning for policy optimisation in spoken dialogue systems," in *Proceedings of Annual Meeting of Association for Computational Linguistics*, 2016, pp. 2431-2441.
24. I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of International Conference on Neural Information Processing Systems*, 2014, pp. 3104-3112.
25. J. Wang, X. Sun, Q. Chen and M. Wang, "Information-enhanced hierarchical self-attention network for multiturn dialog generation," *IEEE Transactions on Computational Social Systems*, 2022.
26. R. C. Xie, Z. T. Chen, and G. S. Hsu, "Successive multitask GAN for age progression and regression," *Journal of Information Science and Engineering*, Vol. 37, 2021, pp. 779-792.
27. A. Xu, Z. Liu, Y. Guo, V. Sinha, and R. Akkiraju, "A new chatbot for customer service on social media," in *Proceedings of CHI Conference on Human Factors in Computing Systems*, 2017, pp. 3506-3510.
28. R. Yan, W. Liao, J. Cui, H. Zhang, Y. Hu, and D. Zhao, "Multilingual COVID-QA: Learning towards global information sharing via web question answering in multiple languages," in *Proceedings of Web Conference*, 2021, pp. 2590-2600.
29. L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in *Proceedings of AAAI International Conference on Artificial Intelligence*, 2017, pp. 2852-2858.
30. Z. Yu, A. Papangelis, and A. Rudnicky, "TickTock: A non-goal-oriented multimodal dialog system with engagement awareness," in *Proceedings of AAAI Spring Symposium Series: Turn-Taking and Coordination in Human-Machine Interaction*, 2015, pp. 108-111.
31. B. Zhang, X. Xu, X. Li, *et al.*, "A memory network based end-to-end personalized task-oriented dialogue generation," *Knowledge-Based Systems*, Vol. 207, 2020, p. 106398.

32. H. Zhang, T. Xu, H. Li, *et al.*, "StackGAN: Text to photo-realistic imagesynthesis with stacked generative adversarial networks," in *Proceedings of the International Conference on Computer Vision*, 2017, pp. 5908-5916.
33. X. Zhao, L. Chen, and H. Chen, "A weighted heterogeneous graph-based dialog system," *IEEE Transaction on Neural Networks and Learning Systems*, 2022.
34. H. Zhou, M. Huang, T. Zhang, X. Zhu, and B. Liu, "Emotional chatting machine: Emotional conversation generation with internal and external memory," in *Proceedings of AAAI International Conference on Artificial Intelligence*, 2018, pp. 730-738.

Ting-Yi Kuo (郭亭儀) received BBA and MBA degrees in Information Management from National Taiwan University, Taiwan. She is now a DevOps Engineer in Taiwan Semiconductor Manufacturing Co. Her research interests include data mining, social media analysis, knowledge management and business intelligence.

Anthony J. T. Lee (李瑞庭) received a BS in Information Engineering and Computer Science from National Taiwan University, Taiwan, and a Ph.D. degree in Computer Science from University of Illinois at Urbana-Champaign, USA, respectively. He joined the Department of Information Management, College of Management, National Taiwan University and he is now a Professor. His papers have appeared in Pattern Recognition, Pattern Recognition Letters, Information Sciences, Information Systems, Journal of Systems and Software, Data and Knowledge Engineering, Expert Systems with Applications, Journal of Information Management, ACM Transactions on Management Information Systems, Decision Support Systems, *etc.* His research interests include data mining, knowledge management, decision support systems, business intelligence, information economics, and business modeling. He is a member of ACM and INFORMS.