robotics

MDPI

Article

Developing Emotion-Aware Human–Robot Dialogues for Domain-Specific and Goal-Oriented Tasks †

Jhih-Yuan Huang, Wei-Po Lee *, Chen-Chia Chen and Bu-Wei Dong

Department of Information Management, National Sun Yat-sen University, Kaohsiung 80424, Taiwan; abner0908@gmail.com (J.-Y.H.); 0224037@nkust.edu.tw (C.-C.C.); sky2730zxcv@gmail.com (B.-W.D.)

* Correspondence: wplee@mail.nsysu.edu.tw

† This paper is an extended version of our paper Huang, J.-Y.; Lee, W.-P.; Dong, B.-W. Learning Emotion Recognition and Response Generation for a Service Robot. In Proceedings of the 6th IFToMM International Symposium on Robotics and Mechatronics, Taipei, Taiwan, 28–30 October 2019; pp. 286–297.

Received: 30 March 2020; Accepted: 3 May 2020; Published: 7 May 2020

Abstract: Developing dialogue services for robots has been promoted nowadays for providing natural human–robot interactions to enhance user experiences. In this study, we adopted a service-oriented framework to develop emotion-aware dialogues for service robots. Considering the importance of the contexts and contents of dialogues in delivering robot services, our framework employed deep learning methods to develop emotion classifiers and two types of dialogue models of dialogue services. In the first type of dialogue service, the robot works as a consultant, able to provide domain-specific knowledge to users. We trained different neural models for mapping questions and answering sentences, tracking the human emotion during the human–robot dialogue, and using the emotion information to decide the responses. In the second type of dialogue service, the robot continuously asks the user questions related to a task with a specific goal, tracks the user's intention through the interactions and provides suggestions accordingly. A series of experiments and performance comparisons were conducted to evaluate the major components of the presented framework and the results showed the promise of our approach.

Keywords: human–machine interaction; service robot; emotion recognition; dialogue modeling; deep learning

1. Introduction

Researchers and engineers have been building service robots that can interact with people and achieve given tasks. To deploy practical service robots, two major concerns need to be seriously considered, including the system architecture for launching the services and the creation of the service functions. At present, the services are mostly laboring services, in which robots take actions in the physical environment to assist people. However, robots are now expected to play more important roles in providing domain-specific knowledge services and task-oriented services. To deliver these services, robots communicate with users through a natural way of spoken language because conversation is a key instrument for developing and maintaining mutual relationships. Following our previous studies that adopted a service-oriented architecture to develop action-oriented robot services, in this work we presented a trainable framework for modeling emotion-aware human–robot dialogues to provide the aforementioned services.

Regarding the many choices of supportive software architecture, some researchers have proposed to adopt cloud-based service-oriented architecture (SOA). SOA is an architectural style based on interacting software components, providing services as fundamental units to design, build and compose the service-oriented software systems [1]. A service is a function made available by a service provider

in order to deliver results to a consumer. Moreover, services are autonomous platform-independent entities that can be described, published, discovered and loosely coupled. To effectively and efficiently deploy different kinds of services, researchers have proposed to link SOA to a cloud computing environment. With this way, the robots are no longer limited by onboard computation, memory and programming, leading to a more intelligent robotic network. Our former work implemented a cloud-based system to support a variety of user-created services [2,3]. To ensure its expandability and shareability, we constructed a service configuration mechanism and deployed the system on the ROS (robot operating system, [4]) computing nodes in practice.

The most common way for achieving natural language-based human–robot interaction is to build a dialogue system to be a vocal interactive interface. Essentially, the dialogue system includes a knowledge base (i.e., dataset) with organized domain questions and their corresponding answers and the dialogue service is to design an accurate mapping mechanism that can correctly retrieve answers in response to the users' questions. The system is performed in a question-answering manner, and most traditional approaches are based on hand-crafted rules or templates. Recently, the deep learning-based methods have been successfully employed to infer neural models for question and answer sentences. These neural systems mainly use a sequence to sequence (seq2seq) model as a backbone to perform mappings from entire sequences of words or characters to other sequences, for example [5,6]. In addition to the dialoguing content, emotion plays a significant role in determining the relevance of the answer to a specific question. By integrating emotion information into the applications, a service system can enable its services to automatically adapt to changes in the operational environment, leading to enhanced user experience.

To enhance the service performance and equip the robot with social competences, in this work, we developed an emotion-aware human–robot dialogue framework extended from our previous research presented in [7], with a series of additional experiments and newly developed dialogue services. To this end, this extended framework included two types of dialogue services. One was to enable the robot to work as a consultant to provide domain-specific knowledge services. The main focus was on constructing a deep learning model for mapping questions and answer sentences, tracking the human emotion during the process of the human–robot dialoguing and using this additional information to determine the relevance of the sentences obtained by the model. The other was to provide task-oriented dialogue services which raised considerable interests due to its broad applicability for assisting users in achieving specific goals (e.g., for booking flight tickets or scheduling meetings). To verify the presented approach, we conducted a series of experiments as described below to evaluate the major system components. The results showed the effectiveness and efficiency of the presented approach.

The remaining part of this paper is arranged as follows. Section 2 provides the research background and reviews the dialogue-related research work. Section 3 describes the framework, including the functional modules of emotion classification and dialogue response selection, and the deep learning techniques used for modeling. Section 4 presents the experimental outcomes and the performance comparisons of the different methods. Finally, Section 5 concludes the paper.

2. Related Works

As mentioned previously, at present most of the service robot frameworks have been connected to various cloud-computing environments to exploit their large amounts of resources. Among others, the most representative work is RoboEarth [8], driven by an open-source cloud robotics platform [9]. With this platform, the robots can distribute highly loaded computation to the cloud and access the RoboEarth knowledge repository to download required resources. There are also other platforms developed for cloud robotic systems. For example, Pereira et al. proposed the ROSRemote framework [10], which enabled users to work with ROS remotely to create several applications. More extensive surveys were found in [11,12]. More recently, due to the rapid advances of the Internet of Things (IoT), researchers proposed the concept of the Internet of Robot Things (IoRT) to describe a new approach to robotics [13,14]. In this way, smart devices can monitor events, fuse sensor data from a variety of

sources and use local and distributed intelligence to determine a best course of action. This expands the ability of service robots, improves a robot's understanding during the human–machine interaction and leads to a more intelligent robotic network. Moreover, to deal with the scalability problem, researchers have started to extend the cloud computing concept for service robots to edge or fog computing to utilize the resources in a more efficient way [15,16].

Instead of investigating issues related to resource allocation and utilization, this work aimed to develop emotion-aware dialogues for a service robot, in which the most important issues were to recognize the emotions from the user utterances and to generate appropriate machine responses. Many methods have been proposed to solve these problems from different perspectives. Because this work adopted deep learning models to address the above two issues, in the following we discuss the most relevant studies with similar computational methods.

In general, using a deep learning-based approach to develop dialogues, responses are generated based on sequence-to-sequence (seq2seq) neural network models, with an objective function of the maximum-likelihood estimation [17]. This model is to take dialogue modeling as learning a mapping between human utterances and machine responses. The focus is on how to generate a suitable response from a corpus to a human utterance. For the training of dialogue models, generative and retrieval-based methods are often used. Although generative methods have the potential to generate sentences of rich content, current generative models often have the disadvantages of lacking coherence and producing unnatural responses. In contrast, though retrieval-based methods are more restricted, they have the advantage of producing informative and fluent responses. Thus, the retrieval-based methods are more practical. As can be observed, retrieval-based methods rely on the exploitation of a large and varied corpus (human–human or human–machine interactions) [18] and deep learning models have been employed to derive mappings (that is, a selection mechanism) between questions and answers (e.g., [5,19]).

The basic seq2seq model consists of two recurrent neural networks (RNNs): one works as an encoder to process the input; the other, a decoder to generate the output. With the characteristic of making predictions based on running texts of varying lengths, the long short-term memory networks (LSTMs) are often adopted to train the answer selection mechanism. This model has now been widely applied to conversation generation and most existing works have mainly focused on developing more advanced techniques (such as decoding strategies or network models) to improve the content quality of the responses. Many neural dialogue systems have been constructed based on this design principle. For example, Serban et al. used a hierarchical LSTM network for a conversation application [20], and Wen et al. proposed a task-oriented model to generate the correct answers in response to the needs of the given dialogue [21]. To overcome the problem of overly general (i.e., safe) responses, Wu et al. proposed a hybrid-level encoder–decoder model, which utilized both word-level and character-level features [22]. Although these models, in theory, are better at maintaining the dialogue state using memory components, they require longer training time and excessive searching for hyper-parameters.

In contrast to the above domain-specific dialogue systems that aim to generate fluent and engaging responses, the other type of neural dialogue systems that has attracted a lot of attention is task-oriented [23,24]. Task-oriented dialogue systems need to complete a specific task (to achieve a goal), for example, restaurant reservation, by interacting with users (i.e., a response generation process). Existing task-oriented systems can be divided into two categories: the modularized pipeline and the end-to-end single-module systems. The former decomposes the task-oriented dialogue task into modularized pipelines to be solved separately, while the latter proposes to use an end-to-end model to produce a sequence of output tokens directly to solve the overall task. End-to-end systems are often more superior than pipeline systems, due to their unique characteristics, such as global optimization and easier adaptation to new domains. In the task-oriented dialogue systems, the most critical component is the goal tracker [25]. The system must update the state of the dialogue according to each user's query and their intent. Given the current dialogue state, the system can then decide how to respond best to the user to accomplish the desired task.

In addition to employing more sophisticated models and advanced tuning mechanisms towards proper response generation, some recent works attempted to augment the emotional information of the neural dialoguing models to generate more meaningful and humanized machine responses. For example, Zhou et al. presented a model that assumed the emotion category of human utterance was known and taken as an additional input to train a model of responses [26]. Sun et al. adopted a LSTM neural network for conversation modeling [27] in which an emotional category label was added to the encoder, which regarded emotional information as an additional source to the conversational model. Moreover, Asghar et al. discussed the feasibility of employing emotion information to help generate diverse responses [28]. They proposed a model of affective response generation to generate sentences conditioned on emotional word embeddings, affective objective functions and diverse beam search. However, these methods only focused on emotional factors while ignoring content relevance, possibly resulting in a decline in the quality and diversity of a response. The integration of emotion and content is still a challenging task for several reasons. The first is that high-quality emotion-labeled data are difficult to obtain in a large-scale corpus because emotions are subjective and difficult to annotate. Moreover, it is difficult to deal with emotions coherently because balancing grammaticality and the expressions of emotions is needed [29].

3. Developing Human–Robot Dialogues

3.1. The Framework

In this work, we adopted a service-oriented robotic framework that could provide various services and resources and develop emotion-aware dialoguing services. This computing platform included two parts: the on-board processors mounted on the robot side (to handle robot functions requiring fast responses, such as those related to perception and actuation) and the computing nodes located on the cloud side to perform highly loaded computing services (such as service planning and deep learning). To realize the proposed design in practice, we configured the framework with ROS to deliver different types of services. Figure 1 illustrates our robotic system architecture and its ROS configuration. As shown, the Graphics Processing Unit (GPU) acceleration virtual machine (VM) and the cloud parallel computing virtual machine are used to support computation. To provide different services on the cloud, we defined different types of computing nodes in the framework. Through the ROS frame protocol, where the management of data interchange is between nodes, the framework could easily combine different services to launch new functions. The module of service planning was described in our previous work [2,3]. Here, we focused on the dialogue module, in which the major functional components were indicated.

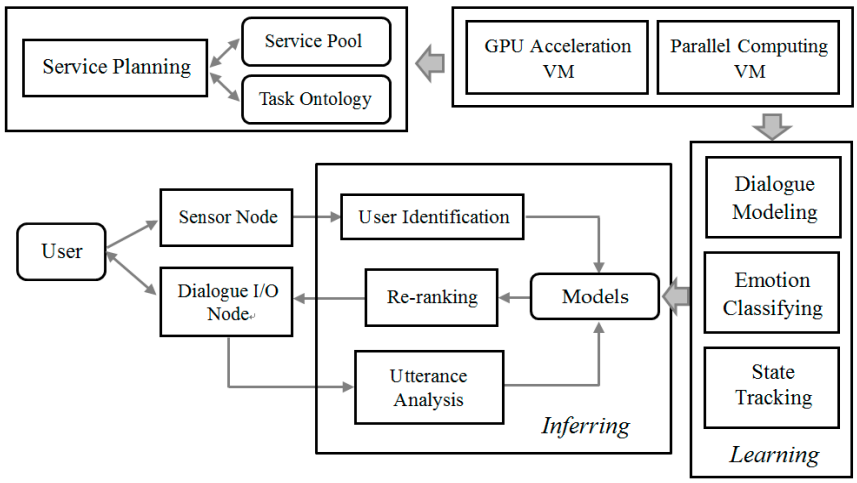


Figure 1. Overview of the proposed framework for the human–robot dialogues.

Our framework included two types of dialogue services, one for domain-specific dialogues and the other for task-specific (task-oriented) dialogues. In contrast to the open-domain conversation performed by the general purpose chatbots, the domain-specific dialogue presented here aims to provide knowledge services of a certain domain (e.g., finance or insurance) through a question-answering manner between the user and the robot. In contrast, the task-oriented dialogue service was to achieve the specific goal for a certain task (e.g., restaurant recommendation) by conducting the iterative human–robot dialogue to adapt to the user's intention or preference related to the task goal. This type of service is especially important in the coming conversational commerce.

At present, the functions of user identification and emotion recognition are constructed independently from the dialogue model, mainly because of the lack of a dataset containing complete information of a human face, utterance emotion and dialoguing content. The current strategy was that the identified user was assigned to a certain type of user group and the corresponding model was retrieved to perform dialoguing. Then, the candidate sentences produced by the model were re-ranked (based on the recognized emotion) following a set of hand-crafted rules and the sentence with the highest rank was selected as the robot's response. In this work, we only applied the emotion mechanism to the first type of dialogue (i.e., domain-specific) as a representative example of emotion-aware services. The same approach can also be applied to the task-oriented dialogue service. The major components of our framework are described in the following subsections.

3.2. Learning Emotion Recognition

3.2.1. Text Processing

In addition to the traditional text processing steps to clean and purify texts, we apply semantic rules to perform sentence segmentation. For example, if there is a disjunctive such as "but" or "although" in the sentence, the emotion of the entire sentence is usually biased toward the former or the latter clause. To tackle such a problem, this study adopted a set of five semantic rules (selected from those proposed in [30]) to perform more precise sentence segmentation. For example, using one of the rules: "If a sentence contains *but*, disregard all previous sentiment and only take the sentiment of the part after *but*," the sentence "I really, really, really wanna go, but I can't." is simplified to be "I can't". The details of the rules refer to [30].

After the above sentence segmentation, we employed the Natural Language Processing Toolkit (NLTK, [31]) to build a dictionary, consisting of more than 7000 words, of which the most frequent stop words were removed. However, because the dialogue dataset used for building classifiers contained some short responses (such as "He?" and "You?"), the list of stop words was thus not fully applied to filter them out. In addition, adverbs such as "more", "most" and "very" are tone aggravation in conversation, therefore, they were retained. Then, a procedure of stemming was performed to strip off word endings, reducing them to a common core or stem.

As indicated above, our framework adopted deep learning for model training and an encoding (embedding) scheme was needed to transfer the natural language sentences into vector representations. Therefore, once the word processing procedure was completed, the GloVe method (Global Vectors for Word Representation) was employed to map the words into vectors, due to its high training efficiency [32]. The training process was performed on aggregated global word–word co-occurrence statistics from a corpus. Through the mapping, the words were represented by real numbers and words with similar meanings which could have similar representations. In this study, the words were mapped into vectors of 300 dimensions. GloVe provides pre-training word vectors, which contain 400 k vocabularies trained from a corpus of 6 billion token words. The vectors were used as the input of the training algorithm to build the model.

3.2.2. Learning Emotion Classifiers

In this work, we trained a deep learning network to recognize emotions from the utterances in dialogues. Figure 2 illustrates the model that includes a convolutional neural network (CNN) followed by a long short-term memory network (LSTM). As shown, the inputs are the dialoguing sentences processed and converted to the vectors by the procedure described above. In this network, three convolutional layers with lengths of three, four and five were arranged to extract the local features of the sentences. Then, the features were combined and served as the input of the next learning layer (i.e., LSTM).

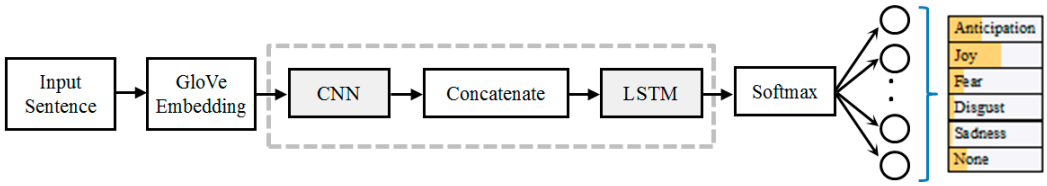


Figure 2. The deep learning model used for the emotion recognition.

It has been well known that LSTM can overcome the vanishing gradient problem in gradient-based machine learning methods. However, this situation still occurs when the sentence length is too long and the network needs to be deepened. In this work, we adopted LSTM with ReLU (Rectified Linear Unit [33]) to train a better model, as ReLU was proved to be effective in overcoming the vanishing gradient problem. Moreover, ReLU has the property of sparse activation, making the neural network sparse to alleviate the problem of over-fitting. In the above learning process, the widely adopted gradient descent optimization algorithm Adam [34] was used as an optimizer.

As shown in Figure 2, we used the activation function widely used in deep learning model, "Softmax", to map the outputs of the neurons into the interval of (0–1). In this way, a probability distribution over the possible classes could be obtained and the node with the highest probability was selected as our prediction emotion class. To calculate the error between the prediction class and the actual class, a loss function was used and the weight update of the deep neural network was performed accordingly. Here, the function "LabelEncoder" of the machine learning tool sciki-learn (https://scikit-learn.org/) and the loss function "categorical_crossentropy" of the deep learning framework Keras (https://keras.io/) were employed to normalize the class label and convert it into a one-hot code of the binary matrix to perform the numerical calculation.

3.3. Domain-Specific Dialogue Modeling

3.3.1. Learning Dialogue Models

To develop dialogues for the robot, we adopted the neural language model from our previous works [3,35] for training the answer selection mechanism. Figure 3 illustrates our model that included a LSTM network with a CNN network. The LSTM contained memory blocks in the recurrent hidden layer that could store the temporal state of the network. With this characteristic, this model could better capture information over longer time steps to meet our goal.

For training the deep learning model, the sentences were organized as the question-answering pairs. The question sentence Q was the input question encoded into an internal vector form Q_V by the word-embedding procedure described above. To enhance the performance, we established the word2vec [36] weights for the entire corpus and used them as the pretrained model of the embedding layer. The output then flows to the LSTM and CNN layers. In this procedure, for each question Q there was a corresponding positive answer $A+$ with a very high probability to be the correct answer among all the answers in the dataset (i.e., the confirmed correct answer). As shown in Figure 3, after

the embedding layer, an output vector E was obtained and then calculated through the LSTM function to derive a hidden vector L as the following:

$$E = \text{EMBED}(x_1, \ldots, x_n; W_e) \tag{1}$$

$$L = \text{LSTM}(E; W_L) \tag{2}$$

In the above equations, E can be represented as $\{e_1, e_2, \ldots, e_n\}$, $E \in \mathbb{R}^{n \times d}$ in which n is the maximal sentence length and d is the dimension of embedding. W_e is the weight matrix $W \in \mathbb{R}^{v \times d}$ (v is the number of words in the dictionary), e is the vector embedded for word x and W_L is the LSTM weight matrix.

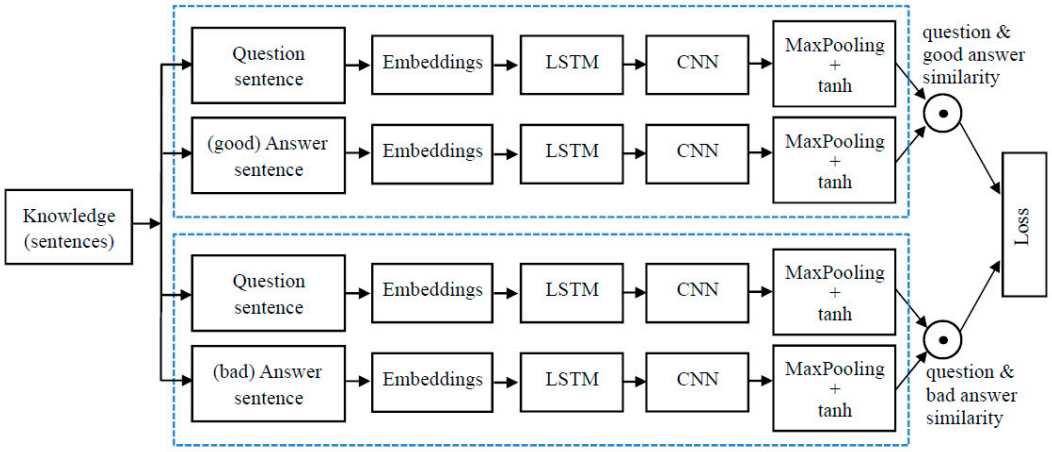


Figure 3. The deep learning model used for the domain-specific human–robot dialogues.

For performance enhancement, we used the genism package [37] to establish the weights for the entire corpus and used them as the pretrained model of the embedding layer. Though the LSTM layer described above, one can extract the features of word sequences in the sentences of our network. Furthermore, we connected the tensor L (Equation (2)) to a convolutional layer to extract more complicated features for performance enhancement. As indicated in Figure 3, the "MaxPooling" function was performed and the "tanh" function was used to transfer and output the decoding result. The above two functions have been widely used in deep learning models for language processing [38].

In the model training procedure, the question Q, the correct answer $A+$ and the wrong answer $A-$(sampled from the answer space) are encoded into vector representations V_Q, V_{A+} and V_{A-}, respectively, and the similarities between the question and the two answers are calculated separately. Here, the similarity of the two vectors is defined as

$$Similarity(V_Q, V_A) = \frac{1}{1 + \|V_Q - V_A\|} \times \frac{1}{1 + \exp\left(-\gamma\left(dot\left(V_Q, V_A\right) + c\right)\right)} \tag{3}$$

This equation was adopted from [38], and it has been shown to offer good performance. In the above equation, the parameter γ is 1.0 and c is 1. V_A is a positive or negative answer (i.e., V_{A+} or V_{A-}). Then, the distance between the two similarities is compared (meaning the difference between an answer and the ground truth) to a pre-defined margin m (a maximum number of steps often used to reduce the running time). If the distance is less than m, the network parameters are updated; otherwise another negative example is sampled until the distance is less than m. The above operations were to ensure that the similarity distance (to be minimized) could reach a certain level. As defined in [38], the loss function corresponding to the above similarity is:

$$Loss = \max\{0, m - Similarity(V_Q, V_{A+}) + Similarity(V_Q, V_{A-})\} \tag{4}$$

During the human–robot dialoguing period (i.e., the test phase), this dialogue service calculates the similarity between a question sentence (asked by the user) and each answer sentence (in the knowledge base). A set of answers with the highest similarity scores is selected and they are re-ranked by the pre-defined rules. The first-ranking sentence is then used as the robot's response.

3.3.2. Knowledge Enrichment

In addition to the learning model and method, the dataset with the domain questions and the corresponding answers also played a critical role in dialogue modeling, because a rich dataset represents abundant knowledge for a system to interact with human users. It was thus important to include more knowledge resources to enrich the dataset (meaning better conversation comprehension) for a dialogue system equipped with a service robot. Many strategies can be developed to include more knowledge resources (e.g., external knowledge resources) for dialogue modeling. In this work, we used a language translation system to translate a dataset to achieve knowledge sharing between different languages. This method was especially important for developing human–machine dialogues with a resource-restricted language (very few data are available for model training). Here, we translated a dataset from English to Chinese as an example to investigate the corresponding effect.

Word segmentation was a very important sentence preprocessing step in the dialogue modeling with the dataset in Chinese. This step was to determine word boundaries for a Chinese sentence. That is, a sentence can be segmented into different combinations of words and therefore the ambiguity exists for Chinese word segmentation. Several segmentation systems have been proposed for Chinese text. Among others, the most often used segmentation systems are the CKIP (http://ckipsvr.iis.sinica.edu.tw/), Stanford Pars (http://nlp.stanford.edu/software/lex-parser.shtml) and the JIEBA (https://github.com/ldkrsi/jieba-zh_TW) system. Following a preliminary evaluation, we chose to use the JIEBA system to perform the word segmentation. Then, the word embedding procedure was performed in which the Wiki Chinese text documents were used to pre-train the corpus for performance enhancement, and the same type of dialogue model can be trained by the deep learning method as in the above section.

3.4. Developing Task-Oriented Dialogues

In addition to the above domain-specific dialogue modeling, this section presents the task-oriented subsystem we developed for the service robot to achieve practical applications with specific goals. As mentioned previously, existing task-oriented dialogue methods can be divided into two categories: modularized pipeline and end-to-end single-module systems. Among others, the hybrid code network (HCN, [24]) is a popular and useful end-to-end framework for developing practical task-oriented dialogue applications. It allows a developer to hybrid the data-driven learning method and knowledge-based hand-coded rules. This approach can learn an RNN with considerably less training data and express domain knowledge via software and action templates. Therefore, in this work, we adopted a simplified HCN framework with some enhanced functions to develop task-oriented dialogue services for the robot. Figure 4 presents our revised framework for the task-oriented dialogues. We also implemented a restaurant recommendation application as an illustrative example. The goal was to request the robot to make a restaurant reservation for a user, given all his constraints on the location, cuisine, price range, atmosphere and party size, which were derived iteratively from the human–robot dialogue.

The operational flow of our revised HCN included four major phases as illustrated in Figure 4. The first phase, which was mainly for text processing, included three steps to extract different types of features from a user utterance. The first step was to extract the context features (entities to be traced). Since we used the DSTC dataset (Dialog State Tracking Challenge dataset [39]) for network training, the context features here were the same as the original dataset: atmosphere, cuisine, location, party size and price, each with a value of 0 or 1 (as a placeholder in the entity tracking slot). The second step was to extract the words (bag of words) to be representatives and the one-hot encoding scheme was used to form a word vector. The third step was to perform word embedding and here the word2vec

was employed. As shown in the figure, in the second phase the text and entities mentioned were then passed to a module of dialogue state tracking, which grounds and maintains entities. In contrast to the original HCN work, we adopted a deep CNN network and defined label ontology to further improve the state tracking performance (described in Section 3.4.1).

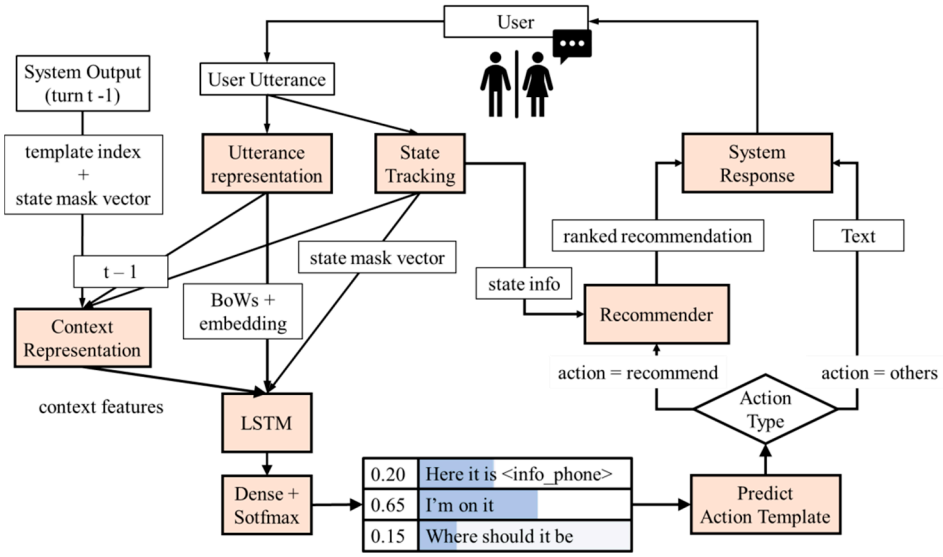


Figure 4. The framework used for the task-oriented dialogues.

In the third phase, the results obtained from the above phases were then concatenated to be a feature vector and a traditional LSTM was adopted for training. As shown in Figure 4, the output of the LSTM model was passed to a dense layer with a Softmax activation, in which the output dimension was equal to the number of distinct action templates. The output was a distribution over the action templates. In the fourth phase, the action mask was applied and an action was selected accordingly. Then, the selected action was used to produce a fully formed action. Following the above phases, a recommendation module was developed to revise some entities according to the user's preferences. The details are described in Section 3.4.2.

3.4.1. Belief Tracker

The belief tracker (i.e., the dialogue state tracking) is an important component in a dialogue system in which a dialogue state is a full and temporal representation of each participant's intention. A belief tracker can track what has happened with the system outputs, user utterances and context from previous turns. It provides a direct way to validate the system's understanding of the user's goal at each dialogue step through the intention estimation.

Traditionally, the rule-based systems were built for state tracking, but they hardly model uncertainty. Recently, researchers have turned to develop neural models to overcome the uncertainty in tracking dialogue states. In task-oriented dialogue systems, the end-to-end neural networks have been successfully employed for state tracking via interacting with an external knowledge base. However, in task-oriented dialogues, a state tracker is usually trained from a large amount of manually annotated corpora. Considering the huge efforts required for human annotation, we used the available dataset for model training and focused on the model performance.

As indicated above, we adopted a simplified HCN model with a dialogue state tracker. However, in some situations the original state tracker could misjudge the ambiguous user utterances or wrongly spell words and produce incorrect answers. For example, using the original HCN tracker to analyze the user utterance "I'm asking my friend if she wants to do Rome", the word "Rome" (entity value) is wrongly taken as the final location, but in fact the decision has not yet been made. This was because the

original tracker uses a string-matching method for entity identification so the mismatches cannot be corrected. As the neural belief tracker was able to deliver a better performance [40], we thus adopted this method and used a deep CNN model to solve this problem. In addition, a small ontology was established to ensure the semantic correctness of the mentioned entities (i.e., slot values).

3.4.2. Autoencoder

Following the above dialogue flow, we developed a recommender (as shown in Figure 4) to enhance the system performance and user experience. This module was to refine some entities from the selected response according to the user preferences. In this work, we used a deep learning-based method and adopted the deep neural network and autoencoder [41,42] to realize collaborative recommendation.

Autoencoder is a superior tool for dimensionality reduction and it can be regarded as a strict generalization of principle component analysis. It is a network with implementations of two transformations (encoder and decoder), aiming to reconstruct inputs in the output layer via a low-dimensional latent space to predict the missing ratings. Then, the learning goal is to minimize the error between the original vector (input) and the transformed vector (output). One of the popular autoencoder-based recommendation models is AutoRec [42]. In this model, denoising techniques are used to discover more robust representations and to avoid learning an identity function. These techniques mean to learn the latent representations of the corrupted user-item preferences and they can be used to reconstruct the users' full preferences and reduce the overfitting situations. In this work, our recommender was developed based on AutoRec. The overall architecture is illustrated in Figure 5, in which the encoder, code-layer and decoder are the major parts of the model (included in the dotted line rectangle). Both the encoder and the decoder consist of feed-forward neural networks with fully connected layers and the depth of the model was increased (marked as the deep stack) to enhance the corresponding performance.

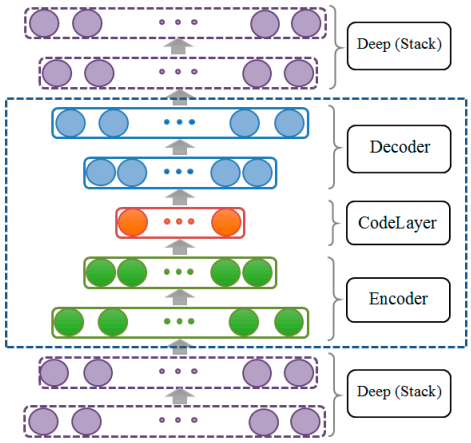


Figure 5. The neural model used for the recommendation.

4. Experiments and Results

To evaluate the presented emotion-aware dialoguing service for human–robot interaction, several sets of experimental trials were conducted. As mentioned previously, due to the lack of a dataset with full information on the human face, utterance emotion and dialoguing content, in the experiments we used four datasets to evaluate these modules separately. The evaluations are described in the following subsections.

4.1. Performance Metrics

In the experiments, we employed the criteria often used in data classification for performance evaluation and a five-fold cross-validation strategy was also used. We first measured the numbers of

true positives (TP), false positives (FP), true negatives (TN), false negatives (FN) and then used them to calculate the metrics of accuracy (proportion of correctly predicted instances relative to all predicted instances), precision (proportion of retrieved instances that were relevant), recall (proportion of relevant instances that were retrieved) and F-measure (the combined effect of precision and recall that often conflict in nature) [43]. The metrics are defined as follows:

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{5}$$

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{6}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7}$$

$$\text{F-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{8}$$

In addition to accuracy, to evaluate the performance of the answer selection in the dialogue modeling, we adopted a statistical measure MRR (mean reciprocal rank, the average of the reciprocal ranks of results for a sample of n queries). It is defined as

$$\text{MRR} = \frac{1}{n} \cdot \sum_{i=1}^{n} \frac{1}{rank_i} \tag{9}$$

where $rank_i$ refers to the rank position of the first relevant document for the i-th query.

4.2. User Identification

In this work, a cloud-based system was built for a service robot and we configured a ROS framework on top of a Linux OS to connect the sensing camera nodes. Often, a system built with ROS consists of a number of processes on a set of hosts, which are connected at runtime in a peer-to-peer topology. Here, the ROS master was a PC running the roscore and serving as the resource center for all the other ROS nodes connected to the network. The cloud parallel computing virtual machine had eight CPUs and eight GB memory, and the GPU acceleration virtual machine had eight CPUs, 32 GB memory and a NVIDIA Tesla K80 GPU.

For user identification, the experiments were conducted to evaluate the performance of face recognition. The goal was to train the robot to recognize human faces in a static manner and we adopted OpenCV (https://opcv.org, an open source computer vision library) to train the classifiers. An online face dataset [44] was used. It included 90 image sets of different persons, in which each set included face images taken from different viewpoints, from 90 to −90 degrees (stepping by 5). The results showed that the trained classifiers performed the best in the recognition of the front face images. The faces in the images could be detected correctly with a reasonable rate of accuracy when the variation of the rotating angle was less than 30 degrees, and the faces could be recognized with a good accuracy if the view angle was within the range of 10 to −10 degrees.

4.3. Performance of Emotion Recognition

4.3.1. Performance Evaluation

To assess the performance of the emotion recognition module, we adopted the dataset used in [45], which was derived from the Movie Dialog Corpus. The sentences in this dataset were categorized into six classes of emotions: fear, disgust, joy, sadness, anticipation and none (neutral). The deep learning approach described in Section 3.2.2 was employed to train a model for multi-class emotion recognition. In addition, two popular learning methods, the random forest (RF) and the support vector machine (SVM) methods, were used for performance comparison.

For RF and SVM, we used the *n*-gram method to extract more text features from the original data for building classifiers to enhance their performance, in addition to the word features extracted from the text-processing procedure. *N*-gram can express the sequence relationships between the words, and the unigram, bigram and trigram (*n* is 1, 2 and 3, respectively) models are often used. After a preliminary test, in this work we used the above three models to extract more text features, and the combined feature vectors were used as the input of the above two machine learning methods (RF and SVM) to enhance their performance.

Figure 6a illustrates the accuracy, precision, recall and F-score for each of the three methods. As can be seen, RF performed the best in all the metrics. The main reason could be that RF is a type of ensemble machine learning algorithm and the way it handled (samples) data for the grouped multiple classifiers made it perform better than the others for the imbalanced dataset here.

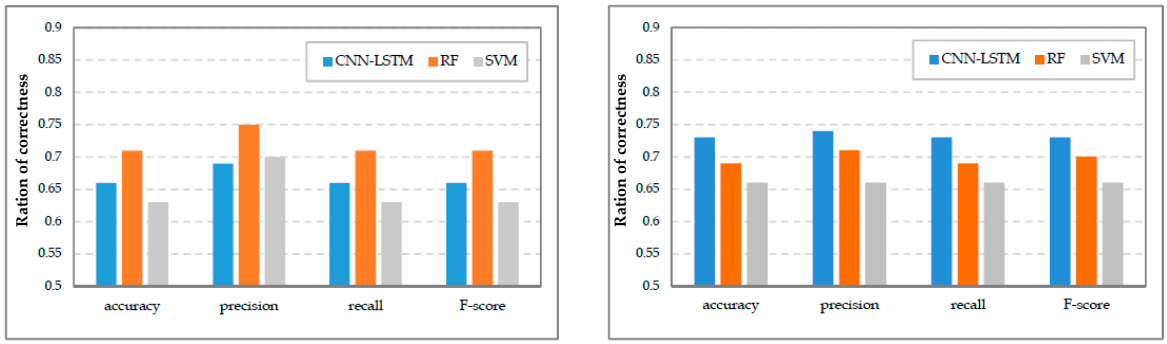


Figure 6. Results of the three machine learning methods; (**a**) without and (**b**) with the enhanced techniques of the semantic rules and data balance.

After comparing the three aforementioned methods, we applied two data processing techniques to the dataset, including semantic rules and data balance, with the above learning methods to investigate their effects in performance. For the semantic rules, the five rules mentioned in Section 3.2.1 were used to perform more precise sentence segmentation; for data balance, we adopted the sciki-learn tool to produce a set of specific class weights for different types of emotions. The results for accuracy, precision, recall and F-score are illustrated in Figure 6b. As can be seen, in general our CNN-LSTM method obtained the best results on all performance metrics. In addition to the data balance effect, the reason for the performance improvement could be that the semantic rules removed the irrelevant words and filtered out their effects on the sentence emotions. Thus, the learning methods were able to focus on the emotions delivered by the most related parts of the sentences to be predicted.

4.3.2. Comparisons with IBM Tone Analyzer

In addition to comparing the different machine learning methods, we evaluated a well known emotion detection system, the IBM Watson's Tone Analyzer (https://natural-language-understanding-demo.ng.bluemix.net/), for further comparison. Interestingly, the emotions the Tone Analyzer considered were slightly different from what we defined in this work, and it gave degrees (values) of multiple emotions for an input sentence (also different from our work). To conduct the performance comparison, we projected the two sets of emotions (one for our work and one for the Tone Analyzer) into the well known emotional valence and arousal space (i.e., V-A space, [46]). In this space, valence indicates the hedonic value (positive or negative), ranging from inactive to active; and arousal indicates the emotional intensity, ranging from unpleasant to pleasant. The valence and arousal dimensions can be projected onto Euclidean space, where emotions are represented as point-vectors. In this way, the user's emotion can be located in this space to be a tuple of valence and arousal values.

In the experiments, we first projected the five classes (annotated in the dataset) into the V-A space to retrieve the corresponding valence and arousal values (based on the emotion positions defined in [46]). For the data of each (sentence), we took the positions of the actual (correct) class and the predicted classes in the space and obtained the set of V-A values. Then, the emotion values produced by the trained model were taken as class weights and the weighted sum was derived for these specific data. Consequently, our method and the Tone Analyzer were compared.

For the data of each, we chose the two closest classes (with the largest weights) and calculated their weighted distance to represent the distance between the predicted and the actual classes. To compare the performances, we divided the distance into eight intervals and counted the numbers of data within each interval. Table 1 presents the results, in which x is the weighted distance. As can be seen, in general the results obtained by the presented method were better than those obtained by the Tone Analyzer for the dataset used.

Table 1. Performance comparison for the two methods.

Distance	CNN-LSTM	Tone Analyzer
$x = 0$	861	692
$0 < x <= 0.1$	160	156
$0.1 < x <= 0.3$	68	101
$0.3 < x <= 0.5$	73	95
$0.5 < x <= 0.7$	61	107
$0.7 < x <= 0.9$	125	169
$0.0 < x <= 1.0$	51	68
$1.0 < x$	104	115

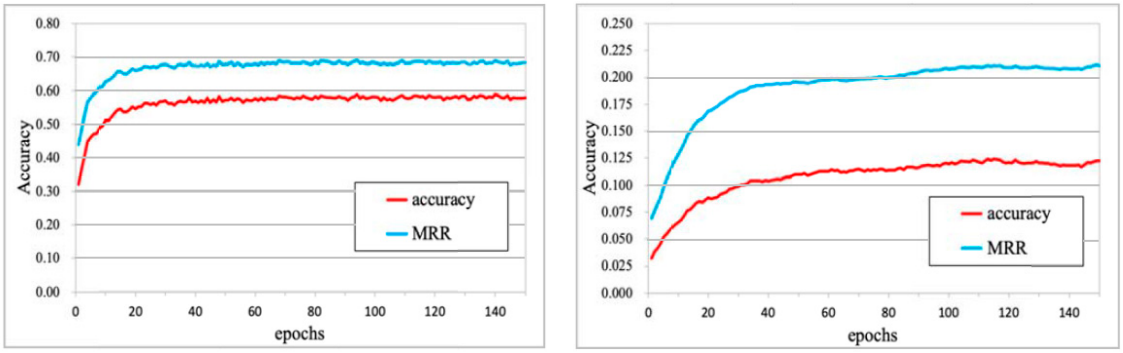
4.4. Performance of Training a Dialogue Model

The next set of experiments was to examine the system performance of model training in retrieving (selecting) answers. In this series of experiments, a large dataset was adopted [38]. It was collected from the Insurance Library website that included 12,889 questions and 21,325 answers, after a data preprocessing procedure was performed. This procedure was to remove unsuitable data that could not form the proper input question–answer pairs, to clean the irrelevant terms (such as html tags) and to transfer the text context into internal identifiers (to form the vectors). In the experiments, the above dataset was divided into two parts, in which a part of 2000 questions and a part of 3308 answers were used for testing. The complete experiments of dialoguing were described in our previous work [35], and here we focused on reporting the results most related to the model training for human–robot interaction.

As described in Section 3.3, in the model training phase, for each question sentence a positive and a negative answer were needed to constitute a training instance. However, in a real-world application, the correct answer $A+$ for a question Q can be determined easily (by the confirmation of the person asking the question), while the wrong answers are often not explicitly specified. Therefore, in the experiments here, all other answers in the dataset were considered candidates of wrong answers to Q. To find the most suitable wrong answer $A-$ for each question in the dataset, we used the above model training procedure to perform the preprocessing procedure of the wrong answer selection. Due to the large amount of answers, in this work we randomly chose ten (instead of all) answers for each question to perform training to reduce computational time.

In the learning process, the random shuffling strategy was used to combine the correct and wrong answers for each question to work as the training data. The model and method presented in Section 3.3 were used for training. Figure 7a illustrates the results of the two performance metrics often used in retrieval-based dialogue modeling, accuracy and MRR. Here, the accuracy was in fact the top-one precision mentioned in the other relevant studies. It means that the model's predictive result (i.e., the top score answer) must be exactly the expected one as recorded in the dataset. As shown, the LSTM-CNN model could achieve the best performance with a correct prediction rate of 0.61 and the MRR was 0.70. The results were similar to those presented in the related study [38], whereas the

presented method involved a smaller set of parameters and was more efficient in learning. In addition to the LSTM-CNN model, a traditional embedding model (using only word embedding technique) was also implemented for performance comparison. The results are shown in Figure 7b. As presented, the accuracy of the embedding model is 0.12 and for the MRR is 0.21. These results indicated that the LSTM-CNN model was more efficient; it obtained a better result within less iterations.

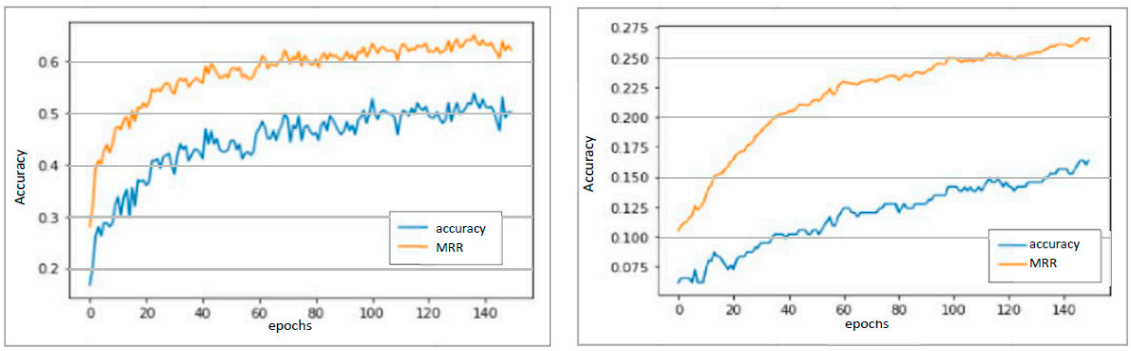


(**a**) (**b**)

Figure 7. Performance comparison of the two methods for the original dataset: (**a**) LSTM-CNN model; (**b**) embedding model.

In addition to the performance evaluation of the dialogue modeling, we performed another set of trials to examine the performance of the shared knowledge translated by a dataset from a different language. In the experiments, the dataset used in the above set of experiments was translated according to the steps described in Section 3.3, and the same deep learning model and method were used for the training. As mentioned previously, in contrast to English sentences, a Chinese sentence could be segmented into various combinations of words by different segmentation methods and this often led to different modeling results. Therefore, before evaluating the performance of the model training, we conducted a set of trails to investigate the effect of two popular segmentation methods: the Jiaba and the HanLP, and the results showed that the Jiaba performed better than the HanLP. We thus chose Jiaba segmentation to continue the experiments of model training.

The results (i.e., accuracy and MRR) are presented in Figure 8a. As shown in the figure, the LSTM-CNN model can achieve a best performance (accuracy) of 0.54 and an MRR of 0.64. Moreover, the traditional embedding model was implemented for comparison and the results are shown in Figure 8b. Similar to the experiments conducted for the original (untranslated) dataset, the results here indicated that the LSTM-CNN model was more efficient than the traditional embedding method.



(**a**) (**b**)

Figure 8. Performance comparison of the two methods for the translated dataset: (**a**) LSTM-CNN model; (**b**) embedding model.

Compared to the results obtained from the original dataset, the accuracy declined from 0.61 to 0.54. This indicated that the model built from the translated knowledge (i.e., dataset) could not keep the modeling performance at the same level; nevertheless, the results showed that the translated knowledge was learnable with an acceptable performance and was thus useful in building models for the resource-restricted language. The performance could be further improved when more advanced text translation techniques are applied.

4.5. Evaluation of Training Task-Oriented Dialogues

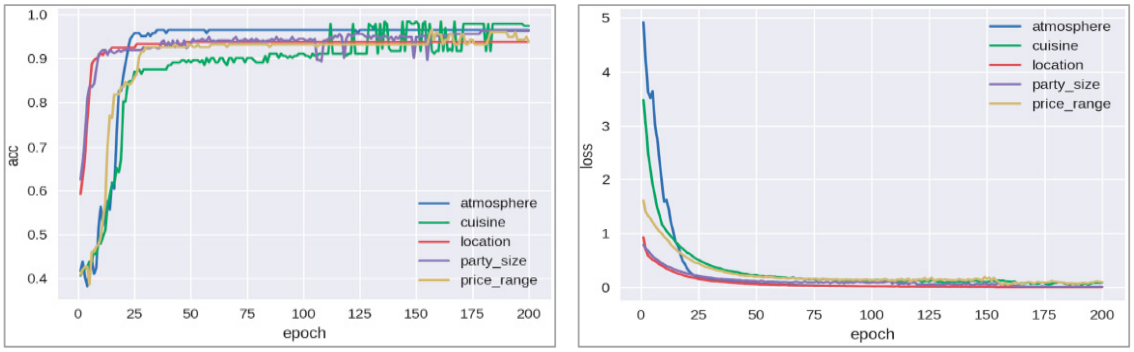
4.5.1. Performance Evaluation of Neural Belief Tracker

As mentioned previously, the belief tracker plays an important role in a goal-oriented dialogue system; it can be used to track each participant's intention from the continuous dialoguing utterances between the participant and the robot. In this work, we implemented a deep CNN model to work as a neural belief tracker. The application task was to perform restaurant recommendation through the user−robot dialogue. A set of entities were pre-defined and the robot iteratively interacted with the user to derive all the missing entity values. The DSTC6 dataset was used for training the tracker. In this task (dataset), five entities were tracked, namely cuisine, location, price range, atmosphere and party size, and the system had to infer their corresponding slot values for making an appropriate recommendation. Table 2 lists the values defined for the entities.

Table 2. Values for all the tracked entities.

Task Entity	Values
cuisine	Italian, British, Indian, French, Spanish
location	Rome, London, Bombay, Paris, Madrid
price range	cheap, moderate, expensive
atmosphere	casual, business, romantic
seat number	two, four, six, eight

To achieve the task, a state tracker was trained for each entity. In the experiments, the performance metrics were the accuracy (the number of correct responses divided by the number of turns) and the loss (here, root mean square error). The training performance for all the entities is presented in Figure 9, in which (a) shows how the accuracy was improved during the training process, and (b) illustrates the reduced loss. As is shown in Figure 9, an accuracy of 0.9 can be obtained after 200 epochs and the loss converged to a small value after 75 epochs and approximated toward zero in the end of the training (200 epochs).



(**a**) (**b**)

Figure 9. Performance of learning the neural belief tracker: (**a**) accuracy; (**b**) loss.

4.5.2. Performance Evaluation of Autoencoder

The DSTC6 dataset used in the above section for training the neural belief tracker contained only dialogue information which cannot be used for making a recommendation. Therefore, in this section we adopted another public dataset (i.e., Yelp [47]) to evaluate the recommendation performance of the presented model. The original dataset contained a large amount of users and their ratings of a set of shops. This dataset had a very high sparsity. To achieve our task of restaurant recommendation and to connect the recommendation module to the dialogue system, we chose the relevant data (69,634 users, 41,019 restaurants and 1,817,955 ratings) to evaluate our approach.

As mentioned above, we revised the autoencoder model through a set of experimental investigations to enhance the corresponding performance. The first phase was to investigate the effect of the code size (the number of nodes in the code layer). A set of code sizes (32, 64, 128 and 256) were evaluated and the results showed that with a size of 32, the model could obtain its best performance. After the preliminary test for code size, in the second phase we evaluated the performance of the different activation functions, including ELU, SeLU, ReLU, Sigmoid and tanh, which were often used in deep learning models. The loss (root mean squared error) was employed to measure the prediction performance and the results are shown in Figure 10, in which (a) is the training process and (b) is the corresponding test process. Figure 10 indicates that the overfitting situation occurred in all cases and the case of ELU obtained the best result. We then performed an additional set of trials on the dropout (the dropping out unit in a neural network) and chose a dropout value of 0.8 to alleviate the overfitting. The third phase was to investigate the effect of the number of hidden layers arranged in the deep network. In this set of experiments, we evaluated five different numbers of layers: 2, 4, 6, 8 and 10, and the results are shown in Figure 11. As shown in Figure 11, though with more hidden layers the model can obtain better training performance, it caused overfitting. We thus chose to use six hidden layers in the final experiments for performance comparison.

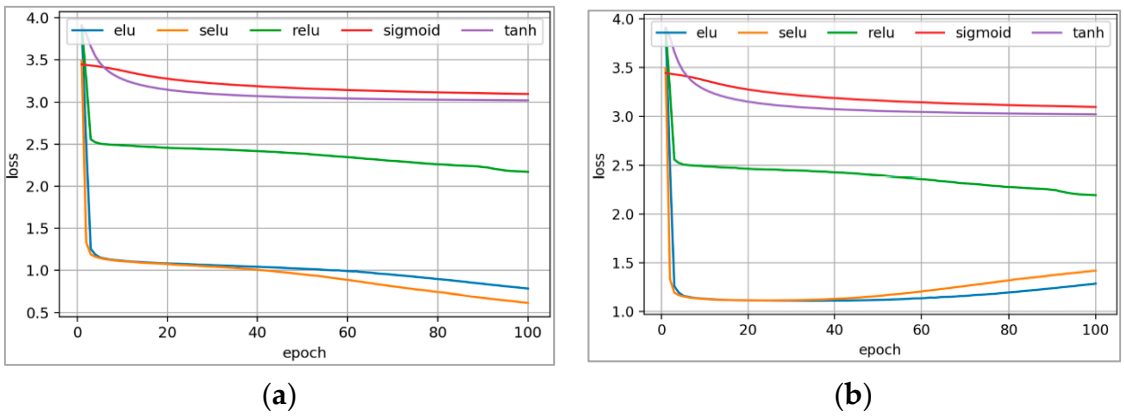


Figure 10. Comparison of the different activation functions in: (**a**) training phase; (**b**) test phase.

After conducting the above evaluation steps for the determination of the network parameters, we then compared our enhanced approach to other popular collaborative filtering methods, including the well known autoencoder AutoRec, and the latent factor model NNMF (non-negative matrix factorization [48]) which is one of the best models in the relevant studies. In the experiments, for all three methods, the number of epochs was 100, the code size (for our model and AutoRec) and the latent factor (for NNMF) was 32, and the learning rate (for our model and AutoRec) was 0.005. As a result, the loss (error) for the proposed model, the AutoRec model and the NNMF method were 1.0868, 1.4758 and 1.1293, respectively. Such results showed that the proposed method outperformed other methods and can provide better recommendation performance.

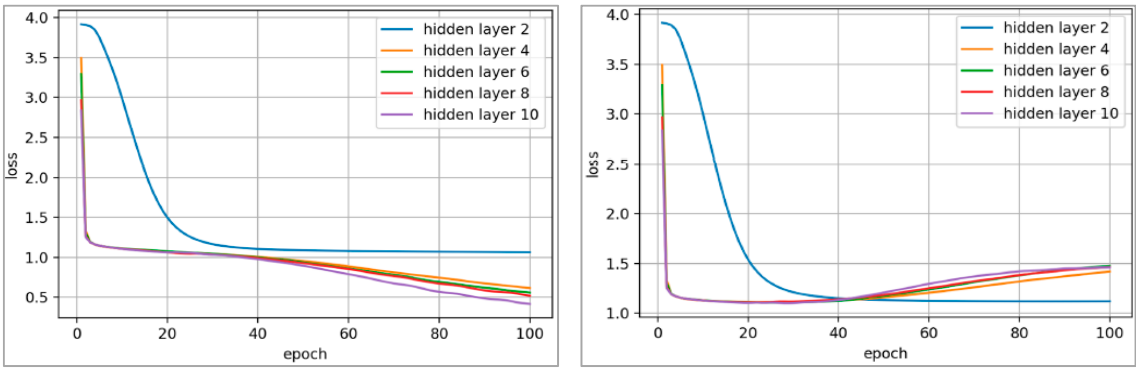


Figure 11. Comparison of the different numbers of hidden layers in: (**a**) training phase; (**b**) test phase.

4.6. Discussion

The above experiments evaluated our approach for a service robot to provide knowledge services. As presented, in our current design, the emotion recognition was constructed separately from the dialogue modeling. The model was trained by a data-driven process with a static dataset. The emotion classifier was then used to re-rank the sentences selected by the model. The separation of emotion recognition and dialogue modeling has several advantages. The first is that the modules of emotion recognition and response generation can be constructed by any effective methods if available; the system thus operates more flexibly. Meanwhile, the reasons why the system generated these responses can be interpretable to users for further analysis. The two subsystems can be integrated into one model to optimize the corresponding structure and performance, for example, to adopt a monolithic model with an attention mechanism to capture emotion as a special context. However, the integrated system may thus become relatively difficult to understand and computationally expensive.

Considering the dialogue modeling, this work trained models by a data-driven process with a static dataset. Therefore, in addition to the learning method, the quality and quantity of the dataset also had influences on the overall performance. It was thus important to strengthen the role of knowledge (i.e., dataset) to infer an enriched domain-specific model. Different strategies can be developed to exploit more knowledge resources, ranging from directly linking the dataset to the up-to-date external knowledge bases, reorganizing the dataset to obtain an optimized data use and to a complicated procedure of transferring knowledge between different domains. For a resource-restricted language, a straightforward way is to take the translated datasets as shared knowledge for modeling. We showed the effect of using translated knowledge. Our application case revealed that the translated knowledge was learnable and the modeling performance could be kept at a similar level as when using the original data. More advanced language translation techniques can be developed to further improve the performance.

In contrast to the non-task-oriented dialogues, a task-oriented dialogue has a specific goal to achieve. The dataset is more focused and thus relatively smaller. In such a system, the most critical component is the goal tracker that is used to track the user's intent during the dialogue to infer the dialogue state. The system can then decide the best response accordingly to achieve the goal (e.g., the recommendations in our experiments). Through the application presented in this work, we have demonstrated that task-oriented dialogues can be practically launched for a manageable task with a clear goal and a constrained dataset. During such dialogues, a fine-tuning procedure for the model parameters needs to be carefully performed to find the best results. When the task becomes complicated or has a high-level (or abstract) goal to achieve, more advanced state tracking and an inferring mechanism is needed to better understand the users' intentions.

5. Conclusions

In this work, we presented an emotion-aware dialogue framework for a service robot to achieve natural human–robot communication. To deploy this framework, we adopted a cloud-based service-oriented architecture and developed the emotion recognition and two types of dialogue modeling modules on it. In the first type of service, the robot worked as a consultant to deliver domain-specific knowledge to users. We employed a deep learning method to train different neural models for mapping questions and answer sentences, tracking the human emotion during the process of the human–robot dialoguing and using this additional information to determine the relevance of the sentences obtained by the model. In the second type of dialogue service, task-oriented dialogues were provided for assisting users to achieve specific goals. The robot continuously asked the user questions related to the task, tracked the user's intention through the interactions and provided suggestions accordingly. To verify our framework, we conducted a series of experiments to evaluate the major system components. The results confirmed the effectiveness and efficiency of the presented approach. Currently, we are developing techniques of knowledge transfer that can extract pairs of questions and answers from the text documents of different domains, in order to automatically enrich the dataset for retrieval-based model training. Moreover, we plan to investigate the use of Kansei engineering with hedge algebras to improve the granularity of the semantic and linguistic analysis in the dialogue sentences. We also plan to integrate the characteristics and preferences of the users into the learning model to achieve personalized dialogues.

Author Contributions: All authors discussed and commented on the manuscript at all stages. More specifically: investigation, methodology and software, J.-Y.H., W.-P.L. and B.-W.D.; supervision, W.-P.L., data analysis and processing, J.-Y.H., C.-C.C. and B.-W.D.; writing—original draft preparation, J.-Y.H. and W.-P.L.; writing—review and editing, J.-Y.H., W.-P.L., C.-C.C.; funding acquisition, W.-P.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the Ministry of Science and Technology of Taiwan, under Contract MOST-108-2221-E-110-054.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Erl, T. *Service-Oriented Architecture*; Prentice Hall: New York, NY, USA, 2005; Volume 8.
2. Yang, T.-H.; Lee, W.-P. A service-oriented framework for developing home robots. *Int. J. Adv. Robot. Syst.* **2013**, *10*, 122. [CrossRef]
3. Huang, J.-Y.; Lee, W.-P.; Lin, T.-A. Developing context-aware dialogue services for a cloud-based robotic system. *IEEE Access* **2019**, *7*, 44293–44306. [CrossRef]
4. Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Ng, A.Y. ROS: An open-source robot operating system. In Proceedings of the IEEE International Conference on Robotics and Automation, Workshop on Open-Source Robotics, Kobe, Japan, 12–17 May 2009.
5. Gao, J.; Galley, M.; Li, L. Neural approaches to conversational AI. In Proceedings of the 41st ACM SIGIR International Conference on Research and Development in Information Retrieval, Ann Arbor, MI, USA, 8–12 July 2018; pp. 1371–1374.
6. 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, Beijing, China, 26–31 July 2015; Volume 1, pp. 1577–1586.
7. Huang, J.-Y.; Lee, W.-P.; Dong, B.-W. Learning emotion recognition and response generation for a service robot. In Proceedings of the 6th IFToMM International Symposium on Robotics and Mechatronics, Taipei, Taiwan, 28–31 October 2019; pp. 286–297.
8. Waibel, M.; Beetz, M.; Civera, J.; D'Andrea, R.; Elfring, J.; Galvez-Lopez, D.; van de Molengraft, M.J.; Schiesle, B. RoboEarth-A world wide web for robots. *IEEE Robot. Autom. Mag.* **2011**, *18*, 69–82. [CrossRef]
9. Mohanarajah, G.; Hunziker, D.; D'Andrea, R.; Waibel, M. Rapyuta: A cloud robotics plat-form. *IEEE Trans. Autom. Sci. Eng.* **2015**, *12*, 481–493. [CrossRef]

10. Pereira, A.B.M.; Bastos, G.S. ROSRemote, using ROS on cloud to access robots remotely. In Proceedings of the 18th IEEE International Conference on Advanced Robotics, Hong Kong, China, 10–12 July 2017; pp. 284–289.
11. Kehoe, B.; Patil, S.; Abbeel, P.; Goldberg, K. A survey of research on cloud robotics and automation. *IEEE Trans. Autom. Sci. Eng.* **2015**, *12*, 398–409. [CrossRef]
12. Saha, O.; Dasgupta, P. A comprehensive survey of recent trends in cloud robotics architectures and applications. *Robotics* **2018**, *7*, 47. [CrossRef]
13. Simoens, P.; Dragone, M.; Saffiotti, A. The internet of robotic things: A review of concept, added value and applications. *J. Adv. Robot. Syst.* **2018**, *15*. [CrossRef]
14. Ray, P.P. Internet of robotic things: Concept, technologies, and challenges. *IEEE Access* **2016**, *4*, 9489–9500. [CrossRef]
15. Tian, N.; Chen, J.; Zhang, R.; Huang, B.; Goldberg, K.; Sojoudi, S. A fog robotic system for dynamic visual servoing. In Proceedings of the IEEE International Conference on Robotics and Automation, Montreal, QC, Canada, 20–24 May 2019; pp. 1982–1988.
16. Galambos, P. Cloud, fog, and mist computing: Advanced robot applications. *IEEE Syst. Man Cybern. Mag.* **2020**, *6*, 41–45. [CrossRef]
17. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2014; Volume 27, pp. 3104–3112.
18. Serban, I.V.; Lowe, R.; Charlin, L.; Pineau, J. A survey of available corpora for building data-driven dialogue systems. *arXiv* **2017**, arXiv:1512.05742v3.
19. Hu, B.; Lu, Z.; Li, H.; Chen, Q. Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2014; Volume 27, pp. 2042–2050.
20. 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 30th AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 3776–3783.
21. Wen, T.H.; Gasic, M.; Mrksic, N.; Su, P.H.; Vandyke, D.; Young, S. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In Proceedings of the International Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 1711–1721.
22. Wu, S.; Zhang, D.; Li, Y.; Xie, X.; Wu, Z. HL-EncDec: A hybrid-level encoder-decoder for neural response generation. In Proceedings of the International Conference on Computational Linguistics, Santa Fe, NW, USA, 20–26 August 2018; pp. 845–856.
23. Li, X.; Chen, Y.-N.; Li, L.; Gao, J.; Celikyilmaz, A. End-to-end task-completion neural dialogue systems. In Proceedings of the 8th International Joint Conference on Natural Language Processing, Taipei, Taiwan, 3 March 2017; pp. 733–743.
24. Williams, J.D.; Asadi, K.; Zweig, G. Hybrid code networks: Practical and efficient end-to-end dialog control with supervised and reinforcement learning. *arXiv* **2017**, arXiv:1702.03274v2.
25. Kuchaiev, O.; Ginsburg, B. Training deep autoencoders for collaborative filtering. *arXiv* **2017**, arXiv:1708.01715v3.
26. Zhou, H.; Huang, M.; Zhang, T.; Zhu, X.; Liu, B. Emotional chatting machine: Emotional conversation generation with internal and external memory. In Proceedings of the 32th AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 Frbruary 2018; pp. 730–738.
27. Sun, X.; Peng, X.; Ding, S. Emotional human-machine conversation generation based on long short-term memory. *Cogn. Comput.* **2018**, *10*, 389–397. [CrossRef]
28. Asghar, N.; Poupart, P.; Hoey, J.; Jiang, X.; Mou, L. Affective neural response generation. In *40th European Conference on Information Retrieval Research*; Springer: Cham, Switzerland, 2018; pp. 154–166.
29. Ghosh, S.; Vinyals, O.; Strope, B.; Roy, S.; Dean, T.; Heck, L. Contextual LSTM (CLSTM) models for large scale NLP tasks. *arXiv* **2016**, arXiv:1602.06291.
30. Appel, O.; Chiclana, F.; Carter, J.; Fujita, H. A hybrid approach to the sentiment analysis problem at the sentence level. *Knowl. Based Syst.* **2016**, *108*, 110–124. [CrossRef]
31. Bird, S.; Klein, E.; Loper, E. *Natural Language Processing with Python*; O'reilly Media: Reading, MA, USA, 2009.
32. Pennington, J.; Socher, R.; Manning, C.D. GloVe: Global vectors for word representation. In Proceedings of the International Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1532–1543.

33. Ramachandran, P.; Barret, Z.; Le Quoc, V. Searching for activation functions. In Proceedings of the Sixth International Conference on Learning Representations, Workshop Track, Vancouver, BC, Canada, 30 April–3 May 2018.
34. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the Third International Conference for Learning Representations, San Diego, CA, USA, 22 December 2015.
35. Huang, J.-Y.; Lin, T.-A.; Lee, W.-P. Using deep learning and an external knowledge base to develop human-robot dialogues. In Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, Miyazaki, Japan, 7–10 October 2018; pp. 3699–3704.
36. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*; Curran Associates, Inc.: New York, NY, USA, 2013; pp. 3111–3119.
37. Řehůřek, R.; Sojka, P. Software Framework for Topic Modelling with Large Corpora. In Proceedings of the LREC 2010 Workshop New Challenges for NLP Frameworks, Valletta, Malta, 22 May 2010; pp. 46–50.
38. Feng, M.; Xiang, B.; Glass, M.R.; Wang, L.; Zhou, B. Applying deep learning to answer selection: A study and an open task. In Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, Scottsdale, AZ, USA, 13–17 December 2015; pp. 813–820.
39. Hori, C.; Perez, J.; Higashinaka, R.; Hori, T.; Boureau, Y.L.; Inaba, M.; Tsunomori, Y.; Takahashi, T.; Yoshino, K.; Kim, S. Overview of the sixth dialog system technology challenge: DSTC6. *Comput. Speech Lang.* **2019**, *55*, 1–25. [CrossRef]
40. Mrkšić, N.; Séaghdha, D.O.; Wen, T.H.; Thomson, B.; Young, S. Neural belief tracker: Data-driven dialogue state tracking. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; Volume 1, pp. 1777–1788.
41. Baldi, P. Autoencoders, unsupervised learning, and deep architectures. In Proceedings of the International Conference on Machine Learning, Workshop on Unsupervised and Transfer Learning, Edinburgh, Scotland, 26 June–1 July 2012; pp. 37–49.
42. Sedhain, S.; Menon, A.K.; Sanner, S.; Xie, S. AutoRec: Autoencoders meet collaborative filtering. In Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, 18–22 May 2015; pp. 111–112.
43. Tan, P.; Steinbach, M.; Kumar, V. *Introduction to Data Mining*; Addison-Wesley: Reading, MA, USA, 2005.
44. The Face Dataset. Available online: http://robotics.csie.ncku.edu.tw/Databases/FaceDetect_Pose_Estimate.htm#Our_Database (accessed on 10 January 2018).
45. Phan, D.A.; Shindo, H.; Matsumoto, Y. Multiple emotions detection in conversation transcripts. In Proceedings of the 30th Pacific Asia Conference on Language, Information and Computation, Seoul, Korea, 28–30 October 2016; pp. 85–94.
46. Russell, J.A. A circumplex model of affect. *J. Personal. Soc. Psychol.* **1980**, *39*, 1161. [CrossRef]
47. The Yelp Dataset. Available online: https://www.yelp.com/dataset/ (accessed on 20 May 2019).
48. Févotte, C.; Idier, J. Algorithms for nonnegative matrix factorization with the β-divergence. *Neural Comput.* **2011**, *23*, 2421–2456. [CrossRef]