A New Chat Solution for Shared Services using Natural Language Processing Models

M. Saravanan
Ericsson Research
Chennai, India
m.saravanan@ericsson.com

Satheesh K. Perepu
Ericsson Research
Chennai, India
perepu.satheesh.kumar@ericsson.com

Sudipta Bose
Dept. Computer Science & Engg
IIT Dhanbad, India
sudiptabose94@gmail.com

Abstract

In the service industry, the shared services denote an accountable entity which started taking the lead over managed services for the past three decades. In shared services, resources are shared across different engagements or projects thereby making them cost-efficient as they centralize back-office operations that are used by multiple divisions of the same company and eliminate redundancy. Quite often in shared services, people face issues with addressing multiple queries. In this paper, we have designed a new chat solution named as **ECHAT** which can answer the complex queries raised by the service engineers related to the product by employing Knowledge Graph and Deep Learning Algorithm. EChat would engage with the service engineers in natural language and would be available as any messaging platform to induce automation in the process. Related to this, we have introduced a new framework which explores different Natural Language Processing (NLP) methods to consider the position of the word, its relatedness and to generate random vector representation to avoid the false positives in responses. Moreover, the significance of our framework is that it can be implemented in any new environment with the available dataset to endure fast training and give more relevant and accurate answer to user queries. EChat has been evaluated and found that the accuracy in providing correct responses for the complex user queries will reach 91% on an average.

1 Introduction

Shared services represent business operations that are handled by multiple parties in an organization. Mainly there are two rationales for shared services set up in an industrial environment i) Less of a common resource ii) Improve Efficiency. Nowadays the industries are very cautious in handling the high operational cost, by allotting the only minimum required number of managers to run the departments with shared resources and also to maintain the efficiency through industrialization which is based on specialization and standardization. Shared services are more than just central organization or consolidation of similar activities in a location. It involves running service activities like a business deliverable for internal customers at a cost, quality and timeliness that is competitive with alternatives.

Recently by introducing automation through digitalization, most of the companies prefer to use chat application to handle the customer conversations with the help of a machine which are called ChatBots. Chatbots are expected to understand a user's query and deliver prompt answers to solve the customer's issues on a real-time basis. Regarding this, a chat interface is suitably designed to allow a bot to converse with multiples users in a simple, fastest and easiest way as possible. It clearly necessitates the need for chatbot to establish automatic conversation in the industry for shared service operations to speed up the present communication channel.

A chatbot is essentially a digital employee that can answer simple customer service questions autonomously in the present digital world. It is generally categorized into two types: Command-based and AI-based. Command-based chatbot relies on a newly constructed database of replies and the relevant heuristics. The bots reply in a way by selecting an answer that matches the context of a query. It is not trained to extend for the creation of new texts and hence it can answer only a limited set of questions. In general, command-based chatbot can perform on its own limitations. On the other hand, AI-based or Machine Learning (ML)-based chatbots gives replies based on training and applying NLP techniques for understanding and interpreting the texts. These chatbots become smarter with time, learning from past questions and answers. Chatbots used for different purposes are typically limited to conversations regarding a specialized purpose and not for the entire range of human communication.

Based on the specific requirements in shared services, we have designed a new chat framework named as EChat which follows AI-based

Proc. of ICON-2018, Patiala, India. December 2018 ©*2018 NLPAI*, pages 87–91

implementations for service engineer's conversation with developers relate to the product issues. It explores different NLP methods and ML techniques in introducing efficient custom-built chatbot to answer most of the customer queries and only very few are forwarded to human agent to exhibit details with the profundity of the issue based on the user requirement. We have evaluated the performances of EChat by implementing in a specific shared service of our company for managing a single product query.

2 Related Works

Shared services in any organization relate to the idea of sharing resources and communication within an organization or group. The goal of a shared services delivery model is to allow each business division to focus its limited resources on activities that support the division's business goals. In this paper, we have introduced a chat solution (EChat) for establishing 24*7 communication with service engineers. The chat interface design is the process that offers credible interaction between the user and the machine.

For texting dialects and SMS writing, Choudhury et al (2007) proposed a Hidden Markov Model for normalizing the text. A statistical classifier based normalization for text messages was introduced by Pennell and Liu (2010). These methods were used to study which character to delete and when to normalize the text by reversing the representations. Xi et al (2004) utilize a positioning function to choose the most applicable messages to client questions in newsgroup searches, but, in which the author feature demonstration is not feasible. An Enhanced language-independent approach of conversational agent for question answering using semantic web knowledge has been developed by Alexandru Dobrila (2010). Cui et al (2015) proposed to constrain morphologically similar words to have similar representations. Soricut and Och (2015) described a method to learn vector representations of morphological transformations, allowing to obtain representations for untrained words by applying pre-determined rules. Nishimura et al (2005) build up a learning base for a question-answering framework that answers compose 'how' questions. From these studies, we understand the feasibility and impact of applying the different statistical technique for processing the text in our proposed framework.

A popular language for creating chatbot is the Artificial Intelligence Markup Language (AIML). With the use of AIML, one can program a computer to give a specified set of answers to a specified set of questions. Thomas and Amrita (2016) build up an AIML and LSA based chatbot to solve the client requirements in managing E-business sites. Shahriare and Shamim (2015) demonstrated the audit of utilization of the chatbot which are created utilizing the AIML contents. It is very clear that AIML based chatbots are lightweight and productive to any work environment. We have used AIML in our proposed chat framework.

Vinyals and Le (2015) proposed a recurrent neural network sequence generation based chatbot to generate the optimal response. For text to speech systems, Sproat and Jaitly, (2016) proposed different Recurrent Neural Network architectures to normalize texts to correctly spoken form. Duplessis et al (2016) built a new chat application named as ChatterBot. The chatterbot focuses on the local coherence of dialogue. Indeed, it only takes into account the last user utterance to select its response. Since this application is very close to our approach, we have tried to compare our proposed chatbot with the chatterbot.

After considering the advantages and disadvantages, we have constructed a new AI-based chatbot using AIML by demystifying Deep Learning techniques in a new framework under the conversation-oriented platform for efficient handling of extracted information.

3 EChat Framework and Models

We have built a new chat application and named it as **EChat.** It is entrusted in an intelligent way of question answering in shared service tasks. Users can enter their questions in the text area of user interface and our system processes the question and responds to the user's queries with suitable resolutions/solutions. Some cases, it rationalizes the selection with a few additional questions.

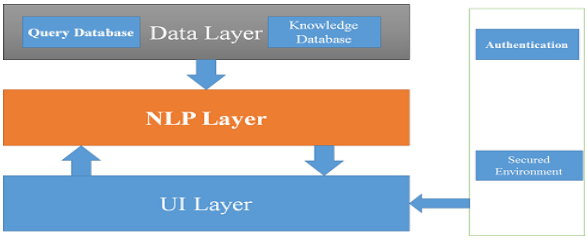

Figure 1. EChat Framework

The proposed framework of EChat is shown in Fig 1. It has three layers (i) User Interface (UI) layer (ii) Natural Language Processing (NLP) layer which performs all the needed text computation and (iii) data layer, where the historical data is available in the form of database.

The UI layer is significant to establish meaningful communication with users. It acts as the user interface between humans who are interacting to get the details using UI and NLP layer where the backend processing of texts happens. The UI layer is written in angular java script (angular JS), which sends the data to NLP layer written in Python. It can also acquire the data generated by the backend to display in UI. To interface the angular JS with python, we have used flask server (Miguel 2014), as it is an efficient way of doing the integration.

The NLP layer forms the backend of the EChat. This layer utilizes the data already available in the form of queries and solutions from the database to answer the questions entered by the service Engineers. To answer the queries not present in the database, we utilize the text in other resources of the product.

To identify the context of the query, we have applied three different models (i) Support Vector Machines (SVM), (ii) Convolutional Neural Network (CNN) and (iii) Conditional Random Field (CRF). The above models were trained through the available dataset. Further, we used the above mentioned trained models to classify the context. As the size of the training dataset is small, the models built resulted in poor accuracy generation. Hence, we prefer to go with a model free approach to solve the problem.

Our approach is explained through two phases (i) training phase and (ii) testing phase. In training, we use the dataset available to generate features. In testing, we use the trained features and user input query to give the required solution.

For training, we performed the following operations on the dataset. First, we extract all the unique words from all the sentences and find out the position of the words in each sentence. Using the position of the words in each sentence as row and unique words as column we form a matrix that provides a useful information on the number of times a particular word present in a specific position. Next step is to assign a random number of 10 dimensions for each unique word in the data. Here, we have to multiply the obtained random number of each word with the corresponding position of the word and further sum up the values to obtain 10-dimension vector for each sentence in the dataset. Finally, we normalize the value obtained to standardize the values between to 0 to 1. In the end, we obtain a ten-dimensional vector for each query available in the database.

To test the chatbot designed, we first follow the steps by considering the importance of processing query sentences.

a. Understand the importance of conjunctions present in the query sentence.

b. To handle the case of the user query similar to the one in the dataset but with different vocabulary, we used a thresholding mechanism to transform the query to that one available in the dataset. For that, we apply the Euclidean distance measure to calculate query vector with possible queries available in the training phase. Here, if the distance between vectors is low, then it means that the query gets the exact match in the database. In this case, we can give the user a solution for the matching query in the database without any additional process. If the distance obtained is lower than a specific threshold, then we will ask the user to provide more information relevant to query words by questioning them with categorical questions. This can help in establishing a conversation-oriented platform in our chat application.

c. To handle large distance values, we use trained CNN model from the knowledge graph to identify the context and pass on the solution to the user. Here we have used the other documentation available related to the product to pick the relevant answers. It the end, if both the cases failed to provide an answer then the specific question will be redirected to concerned service engineer to answer the query.

As discussed in the introduction, the proposed EChat built with intelligence system which helps users to get answers for some queries which are not available in the dataset used to train the model. For this, we consider the documentation of the product available in the knowledge database in the data layer and any other supporting documents belong to the product. First, we preprocess all the data by removing the stop words, punctuation etc. as with any traditional NLP exercise. Further, we extract the terms which are categorized as nouns and verbs by the inbuilt gensim [Radim and Petr, 2010] categorizer. In addition, we also consider the words which are important for the product, as there are chances that these words are not categorized as verbs or nouns in the general dictionary. These

words are manually labeled as nouns or verbs to use them in the exercise. As the next step, we collect instances where these words are present in the system and create knowledge graph by connecting the nouns in the system. The knowledge graph relates the relation between the keywords in the form of the entire sentence.

Hence, to provide an accurate answer we need a model to identify the context of the query. It should be noted that the knowledge graph, in general, can have many nodes (as many as the nouns in the text) and also many edges (as many as the verbs in the text).

Finally, for the initial testing, we have been provided with a text database for one of the product as a sample set. As discussed, we have generated two databases relevant to text processing. (i) Query database and (ii) Extracted Knowledge database. As part of product usage, customers come up with their queries relevant for day-to-day functioning and the engineers should provide the solutions to persuade them. These conversations happened in the form of e-mail conversations. As part of a single query, it is possible to have two or more mail chains associated with this. This is because the engineers need more details from the customers related to their problem for giving an efficient solution. To create a database of the queries and solutions, we collect the first query mailed by the customer and the relevant solution proposed by the engineer. Also, we collect some other details as a category of the query, the time is taken to provide the solution etc. To tackle the non-availability of queries in the query database, we have generated a knowledge database by collecting the details from documentation and frequently answered questions of the product. In addition, we also collected documentation of the open source tools used for the development of the product for the efficient answering of the queries which are not present in the query database.

4 Evaluation and Discussion

There are many evaluation metrics available to measure the chatbot performances. One such important metric is accuracy. Accuracy is measured in terms of how many times the chatbot provides the correct answer for the user queries. In addition to the accuracy, we also considered precision and recall measures, since we have used the classification model to understand the context present in the query. Here, precision is a percentage of relevant documents retrieved and recall is the percentage of retrieved relevant documents. We have calculated the discussed measures for both EChat and chatterbot and the details are provided in two different tables.

Table I: Accuracy Measurement

Query/Statement	Chatterbot Accuracy	EChat
Direct statement	100%	100%
Changing the position Of the statement	40%	93.33%
Combined two statement to form a single statement	45.83%	87.5%
Partial sentence as statement	66.6%	83.33%

The performance of chatbots is evaluated based on the type of questions user queried to understand the effectiveness of the proposed EChat. We have evaluated EChat performances for three different types of queries (i) direct query, (ii) partial query and (iii) combined query (two queries in the database combined). The same exercise will be done with chatterbot, a chatbot considered for comparison. The accuracies obtained for three different cases for both chatbots are reported in Table I. First case i.e. when the query is direct statement i.e. user's query exactly matches the question stored in dataset then the proposed EChat and also the chatterbot delivers 100% accuracy. Further to the second case i.e. when the user enters a partial query, the chatterbot fails to provide better results as chatterbot searches for the closest match known statement to the query available in the database. However, the EChat generates better results due to the consideration of the position of the words in a query. Moreover, in the third case when the user combines two questions or statements from the dataset to form one single statement and pass it as a query. In this case, the system should return the answer to the question which forms the larger part of the combined statement. In the third case also our approach gives the better accuracy compared to chatterbot due to an intelligent implementation of the knowledge graph.

Table II: Average Precision and Recall Score

Evaluation	EChat bot	Chatterbot
Avg. precision score	76.92%	61%
Avg. recall score	89.42%	78.52%

As already discussed, precision and recall values are calculated for three different queries as discussed. For each type, we calculate the precision,

recall and the values are averaged to obtain average precision and average recall for all the different types and the values are given in Table II. From the results, it is evident that the proposed chatbot performs exceedingly well compare to chatterbot in terms of precision and recall. In addition to the above measures, we have also tested our EChat which can deliver answers even if the queries are not similar to the queries present in the query dataset. In the domain testing, we found that 90% of times EChat returned correct answer using the words extracted from the documentation of the product. The accuracy can be improved if we use additional documents in addition to the documentation of the product. For this case, we skip the comparison as the chatterbot is not designed for the same.

5 Conclusion and Future work

We have developed EChat for shared services in industrial set up to address the queries raised by the service Engineers. We trained our dataset using query importance, sentence selection and keyword based probabilistic and knowledge-based models. Compared with the conventional chatbots like chatterbot, the proposed EChat has the strong ability to give more accurate and relevant answer to the user's questions and also work with the human agent to handle inquiries that cannot be resolved programmatically. It also delivered good performance and delivered answers with higher accuracy. Further, to answer queries which are not present in the query database, we used the knowledge graph constructed from additional documents to supply answers. Future directions include automatic regeneration of answers in our conversational chatbot which provides the best experience to the users.

References

http://www.atkearney.in/operations/ideas-insights/article/-/asset_publisher/LCcgOeS4t85g/content/rewriting-india-s-shared-services-

Reshmi, S., &Balakrishnan, K. 2016. Implementation of an inquisitive ChatBot for database supported knowledge bases. Sādhanā, 41(10), 1173- 1178.

Vinyals, O. and Le, Q. 2015. A neural conversational model. ICML Deep Learning Workshop.

Thomas N. T., Amrita Vishwa. 2016. "An E-business ChatBot using AIML and LSA", Intl. Conference on Advances in Computing, Communications and Informatics (ICACCI), Sept. 21-24, Jaipur, India

ALICE,(2011).A.L.I.C.E. 2011. Artificial Intelligence Foundation [online].Available from:http:/ /alicepandorabots. com/ [Accessed 05/25/2011].

Shawar, Bayan Abu, and Eric Atwell. 2007. "ChatBots: are they really useful?." LDV Forum. Vol. 22. No. 1.

Md. ShahriareSatu, Md. HasnatParvez, 2015. "Review of integrated applications with AIML based ChatBot", 1st International Conference on Computer & Information Engineering, Dept. of CSE, Rajshahi University of Engineering & Technology, Rajshahi, Bangladesh.

M. J. Pereira, and L. Coheur, 2013. "Just. Chat-a platform for processing information to be used in ChatBots", Master Thesis, Lisboa

W. Xi, J. Lind and E. Brill. 2004. Learning Effective Ranking Functions for Newsgroup Search. InProceedings of SIGIR 2004, pp.394-401.

R. Nishimura, Y. Watanabe and Y. Okada. 2005. A Question Answer System Based on Confirmed Knowledge Developed by Using Mails Posted to a Mailing List. In Proceedings of the IJCNLP 2005, pp.31-36.

Alexandru Dobrila, 2010. From Semantic Web Knowledge to a Functional Conversational Agent: A Practical Approach.

Miguel Grinberg. 2014. Flask Web Development: Developing Web Applications with Python (1st ed.). O'Reilly Media, Inc.

Radim Rehurek and Petr Sojka, 2010. ˇ Software Framework for Topic Modelling with Large Corpora, Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks (Valletta, Malta), ELRA, pp. 45–50 (English).

Duplessis, Dubuisson, Vincent Letard, Anne-Laure Ligozat, and Sophie Rosset. 2016. "Joker chatterbot re-wochat 2016-shared task ChatBot description report." In RE-WOCHAT: Workshop on Collecting and Generating Resources for ChatBots and Conversational Agents-Development and Evaluation Workshop Programme.