Automated Assistance in E-commerce: An Approach based on Category-Sensitive Retrieval

Anirban Majumder[1], Abhay Pande[1], Kondalarao Vonteru[2], Abhishek Gangwar[2], Subhadeep Maji[1], Pankaj Bhatia[1] and Pawan Goyal[2]

[1] Flipkart Internet Pvt. Ltd.
{majumder.a, abhay.pande,subhadeep.m,pankaj.bhatia}@flipkart.com
[2] IIT Kharagpur
sunnysai12345@iitkgp.ac.in,abhishek.g307@gmail.com,pawang@cse.iitkgp.ac.in

Abstract. This paper aims towards building an automated conversational assistant to help customers in an e-commerce scenario. Our dataset consists of live chat messages between human agents and buyers. These chats belong to many different issue types and we build a multi-instance SVM classifier to automatically classify these chats into the corresponding issue types. We further use this insight to append the category information obtained from the classifier to an LSTM based architecture to be able to provide appropriate responses given an utterance by a human agent. We find that using class information along with the base dual encoder model helps in improving the quality of the retrieved responses in terms of BLEU scores. Human judgement experiments validate that using class information is able to bring out relevant messages in top-3 and top-5 responses much more number of times than the base model that does not use the class information.

1 Introduction

To build a conversational agent and/or chatbot with sufficient artificial intelligence has always been long cherished goal for researchers and practitioners. It is very challenging for computers to do a coherent, continuous and meaningful conversation with humans. These automatic conversation models are of great importance to a wide variety of applications, starting from open-domain entertaining chatbots which can naturally and meaningfully converse with humans on open-domain topics, to goal-oriented technical support systems which can assist users towards completion of a task. In an open domain setting, user can take conversation in any direction, there is not a well defined intention or goal. In a closed domain setting, the domain of inputs and outputs is somewhat limited as the user is trying to achieve very specific goal. These systems need to fulfill their specific task as efficiently as possible.

Customer support in the e-commerce scenario sees customers reaching out with a wide variety of inquiries like the status of the order, the return process, delays in processing of refund and offer inquiries. As the business scales, the number of contacts from the customers about such inquiries also increase at the same rate. An automated conversational agent that can help address customer queries, goes a long way in providing cost-effectiveness as well as scalability.

While the traditional systems required a lot of domain expertise to be crafted manually [1], in the recent years, an increasing amount of research has happened to build purely data-driven conversational systems. These systems mainly use two types of approaches. *Retrieval-based models* use a repository of either predefined responses or context-response pairs along with a retrieval/ranking mechanism to pick an appropriate response with the help of the input and context. For instance, Lowe et al. [2] introduced the Ubuntu Dialogue corpus, and also presented an LSTM based framework to provide a score to any candidate response given an input message. Yan et al. [3] proposed a retrieval based approach that can also leverage on unstructured documents in addition to the context-response pairs. Yan et al. [4] introduced a chat companion system, which given the human utterances as queries, responds with corresponding replies retrieved and highly ranked from a massive conversational data repository. They perform ranking with and without using the context for multi-turn and single-turn conversations, respectively. *Generative models*, on the other hand, do not rely on predefined responses. Instead, they generate new responses. There have been a number of related attempts to address the problem using generative models with the help of neural networks. Sequence to Sequence Learning [5] uses a multi layered Long Short-Term Memory (LSTM) [6] to encode the input sequence to a vector of a fixed dimensionality, which is used to generate (decode) the response. Sordoni et al. [7] proposed to encode the message along with context in a recurrent language model based architecture. Yao et al. [8] modeled both attention and intention processes for generating natural responses.

E-commerce is a closed domain, where typical conversation is a mix of standard responses and context based dynamic responses to customer queries. Our work, as one of the very first experiments with conversational agents in e-commerce domain, demonstrates experiments with retrieval approaches, where we evaluate the hypothesis that in e-commerce domain, leveraging the issue type classes with existing approaches improves the quality of responses.

2 Dataset Description

Our dataset consists of real time chats that took place between human agents and the buyers during issue resolution by customer care for Flipkart[3] between the months of July to December, 2016. One such example chat is shown below:

Customer:Hi
Auto:Hi , I'm ⟨ consultant name⟩. We had spoken earlier and I'll be happy to help again.
Customer: I will go for replacement.
Customer: For my product.
CX: Sure, I will escalate to cancel the refund request so within next 48 hours it will be canceled and we will intimate you about this over email on your registered email id.

Messages starting with "Customer" indicate what the customer said, while those starting with "CX" indicate what the customer support consultant said. Messages with "Auto" are automated messages sent by the system. Every chat interaction between the customer and the human agent is categorized into a

[3] https://www.flipkart.com/

group of issue types and sub-issue types by human agents. These issue types and sub-issue types identify the nature of the problem being faced by the buyer. We sampled the data for the month of July and found that while there were 24 issue types, only 13 issue types had significant number of chat sessions, including an issue type to classify spams, appreciations or incomplete requests. The remaining 12 issue types covered 93% of the non spam / incomplete chats in the month of July. A distribution among these 13 issue types is shown in Figure 1.

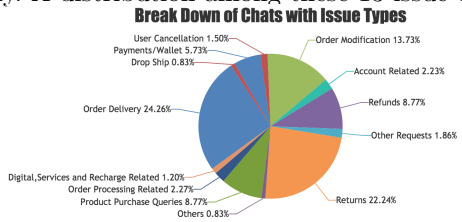

Fig. 1: The distribution of Issue Types

# Chat Sessions	39,000
# Context-Response pairs	448,335
# Issue Types	13
Average # words per context	13.65
Average # words per response	20.62

Table 1: Properties of the e-commerce chat corpus used in this study

The dataset we used for our experiments consists of **448,335** context-response pairs, taken from chats during the months of July to December. Table 1 shows the details of our dataset. We convert all consecutive occurrences of "Customer" into context and those of "CX" into utterance to get a context-response pair. We have contextual information such as dates, product names, consumer names and customer agent names. In pre-processing steps, we replace the contextual information above by placeholders. The data also contains generic responses and greetings which are noise to our training purposes and we had them removed in our pre-processing steps.

Issue-type Classifier: In the e-commerce domain, a conversation can be classified to various issue types. This class information can help the agents / bot to narrow down to the problem faster and get to the appropriate response. For our experiments, we use a classifier trained on a small dataset annotated with issue type labels. However, these annotations are at a chat level rather than individual context-response pair. Further, different sequence pairs in a chat can belong to different issue types. The classifier that we used is based on the multi-instance SVM proposed by Andrews et al. [9]. The multi-instance SVM is a variation of SVM which specializes on training labels for a bag of instances with one or more labels and extrapolates that to a bag of single instance.

The SVM classifier was trained on a dataset of size 80K, completely disjoint from the dataset for the LSTM model. The training of the model was done on the annotated dataset. We used word embeddings weighed with the tf-idf scores of the word as features for classification. We use cross validation to evaluate the results of these classifiers which gave an accuracy of 80% for the above-mentioned 13 classes. We also performed a manual evaluation on a small held-out dataset of 1K instances and the accuracy was found to be close to 70%.

3 Automating Retrieval of Conversational Responses

We used the retrieval model architecture used in [2], an LSTM based recurrent neural network, which tries to learn representations of the context and response and scores how appropriate a response is for a given context. During the training

phase, the model encodes the context and response using an LSTM. The authors of [2] first obtain **context embeddings** from the given input context (LSTM hidden state corresponding to the last input), and then learn a layered perceptron network to obtain the **model response embeddings** from this context embedding. Now, to rank the candidate responses, the model [2] first embeds them using LSTM using the same approach as above, and then gives a probability score for each context-response pair based on how well these response embeddings match the model response embeddings. The loss is computed using cross entropy of the targets and scores computed.

Using Chat Category Information: Our hypothesis is that using the class information together with the context can help in providing a better response. We aim to use our issue-type classifier to classify these conversations into the particular issue type. Below are some example contexts.

Context 1: i have an order but i cancelled because of changing my address but now i didnt get my refund

Context 2: i have an order of ¡prod name¿ but i have to change my address can u help

For context 1 above, the customer talks about cancelling an order due to the change in delivery address, and for the next context, he just wants to change the address. From the text, it seems that both of these contexts come into the **Order Modification** class, but the first context actually belongs to class **Refunds**. Here, the class information can help in proper response retrieval.

To incorporate this class information, we use exactly the same retrieval-based architecture as mentioned above, except that the class information, as obtained from the automated classifier, is provided as input. One standard method is to provide it as input to the initial states of the recurrent neural network. Introducing the classifier as the initial states, we ensure that the class information is used along with the context to encode the response as well as the context embeddings. Formally,

$$c_0 = g(class_i) \mid class_i \in \mathbb{I} \tag{1}$$

$$h_0^{'} = \langle h_0, c_0 \rangle \tag{2}$$

where the $class_i$ denotes the classifier output as the most appropriate issue type, which belongs to the set of the issue types $\mathbb{I}$ and $g(.)$ provides the vector representation of the class information using one-hot encoding. The initial states of the LSTM contain the initial hidden and cell state. We replace the initial cell state with the class information. We will explain the results of this variation of the model and compare with the base model in Section 4.

4 Experiments

We used 222,209 context-response pairs as obtained after some preprocessing steps for training. The basic preprocessing steps involved greetings or chit-chats removals, removal of very short context-response pairs[4] as well as replacing various entities such as product names, dates, customer names with generic entity

[4] We fixed the minimum length of these context-response pair to be 4. These context-response pairs correspond to almost 39K chats.

tags like 'PRODUCT', 'DATE' etc. We trained all our models for 22K epochs in a standard GPU machine which took less than 1 hour.

We also created a test dataset of 10K context-response pairs from the actual chats, completely disjoint from the training set. This set was created after performing the same pre-processing steps. Since the retrieval model only ranks the candidate responses, Okapi BM25 was used to retrieve the top 10 responses from the training set of context-response pairs corresponding to the given context. The LSTM used for the model was build on the Tensorflow library.

Evaluation Metrics: As per the earlier works, we also evaluate the performance of our models using BLEU scores [10] for top-k responses ranked by our model with reference to the ground truth. We use T1-BLEU, T3-BLEU and T5-BLEU to denote the BLEU scores based on top 1, 3 and 5 responses.

Experimental Results: Table 2 shows the results of our evaluation. We see that adding class information to the base retrieval model helps in improving the performance for all the three cases (top 1, 3 and 5). The improvements were highest for the top retrieved sentence.

Models	**T1 BLEU**	**T3 BLEU**	**T5 BLEU**
BM-25+LSTM ranking[2]	22.62	28.48	31.72
[2]+Class Info	**24.21**	**29.59**	**32.22**

Table 2: Comparison between different competing models based on Bleu scores

Models	**Rel. top 3**	**Rel. top 5**
Base Retrieval Model [2]	26	32
With Class Info	**34**	**43**

Table 3: Comparison between different retrieval models as per human judgement experiments

Manual Evaluation: We also perform a human judgment experiment to verify the importance of the class information on LSTM ranking with 100 randomly chosen context-response pairs. 2 participants, different from the authors, were given 50 contexts each, where every context had 5 responses as ranked by the two retrieval models: LSTM and LSTM+class info. The participants were asked to answer whether a relevant response appears within top 3 (and top 5) of these ranked lists, without disclosing the identities of the two models. Table 3 shows the results of this experiment. This clearly tells that the class information helps to bring relevant response in top-3 and 5 retrieved results.

5 Discussions

From Tables 2 and 3, we can see that the model with class information performs better than the base model. The following example can validate our claim further. Without the class information, the bot is not sure about the context and replied asking about the product name. With class information ("**Refunds**"), the bot replies properly with the details of refund status.

Customer : i have an order but i cancelled because of changing my address but now i didnt get my refund

Without class information: Bot : you are welcome is there anything else that i can help you with today ?

With class information: Bot : according to bank procedure this takes ⟨no of days⟩ to refund the amount as i see that the order get cancelled on the ⟨ date ⟩ so i kindly request you to allow another ⟨no of days⟩ for refund to be done

While we get some good results from the introduction of the classes, on error analysis, we observe that error in classification phase can lead to an incorrect response. In the follwoing example, the context was wrongly classified to the class **"Returns"** instead of **"Refunds"** and the model produces an incorrect response:

Customer : Today please tell filpkart, please today refund

Bot with class info : i would like to inform you that as the product has been delivered to you i request you to please contact the brand for this issue

6 Conclusions and Future Work

This paper presented a framework for automating conversational responses to assist the customers in an e-commerce scenario. In addition to being one of the first studies exploring conversational agents in e-commerce domain, novelty of our approach lies in using the issue type information, as obtained by a multi-instance classifier, to enhance the existing LSTM-based retrieval model. Experimental results suggest that it indeed helps in improving the quality of the responses. Future work would involve experiments with other architectures, as well as adding more information from the chat context, as well as product / sale context.

References

1. Young, S.J.: Talking to machines (statistically speaking). In: INTERSPEECH. (2002)
2. Lowe, R., Pow, N., Serban, I., Pineau, J.: The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. arXiv preprint arXiv:1506.08909 (2015)
3. Yan, Z., Duan, N., Bao, J., Chen, P., Zhou, M., Li, Z., Zhou, J.: Docchat: an information retrieval approach for chatbot engines using unstructured documents, ACL (2016)
4. Yan, R., Song, Y., Zhou, X., Wu, H.: "shall i be your chat companion?": Towards an online human-computer conversation system. In: CIKM. CIKM '16, New York, NY, USA, ACM (2016) 649–658
5. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NIPS. (2014) 3104–3112
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8) (1997) 1735–1780
7. Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., Nie, J.Y., Gao, J., Dolan, B.: A neural network approach to context-sensitive generation of conversational responses. arXiv preprint arXiv:1506.06714 (2015)
8. Yao, K., Zweig, G., Peng, B.: Attention with intention for a neural network conversation model. arXiv preprint arXiv:1510.08565 (2015)
9. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support vector machines for multiple-instance learning. In: NIPS, MIT Press (2003) 561–568
10. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: ACL. (2002) 311–318