When and Who? Conversation Transition Based on Bot-Agent Symbiosis Learning Network

Yipeng Yu[1*†], *Ran Guan*[2*], *Jie Ma*[3], *Zhuoxuan Jiang*[4], *Jingchang Huang*[3]
Tencent, Shanghai, China[1]
University of Cambridge, Cambridge, UK[2]
IBM Research China, Beijing, China[3]
JD AI Research, Shanghai, China[4]
ianyu@tencent.com, ran.guan@cl.cam.ac.uk, jiema.uk@gmail.com
jiangzhuoxuan@jd.com, hjingchang@foxmail.com

Abstract

In online customer service applications, multiple chatbots that are specialized in various topics are typically developed separately and are then merged with other human agents to a single platform, presenting to the users with a unified interface. Ideally the conversation can be transparently transferred between different sources of customer support so that domain-specific questions can be answered timely and this is what we coined as a Bot-Agent symbiosis. Conversation transition is a major challenge in such online customer service and our work formalises the challenge as two core problems, namely, when to transfer and which bot or agent to transfer to and introduces a deep neural networks based approach that addresses these problems. Inspired by the net promoter score (NPS), our research reveals how the problems can be effectively solved by providing user feedback and developing deep neural networks that predict the conversation category distribution and the NPS of the dialogues. Experiments on realistic data generated from an online service support platform demonstrate that the proposed approach outperforms state-of-the-art methods and shows promising perspective for transparent conversation transition.

1 Introduction

Recent years have witnessed a plethora of chatbot-based online customer services (Oraby et al., 2017; Xu et al., 2017; Jiang et al., 2019). Although current chatbots are far from perfect, they are playing increasingly larger roles thanks to more advanced natural language processing (NLP) capabilities and can now respond properly to certain domain-specific problems and do some basic housekeeping tasks before a human agent gets involved. It is also common that the customer support is sourced to multiple agents and bots where the skill sets may vary greatly. Conventionally, users are firstly greeted by a porter bot, and then they could be further transferred to a human agent if they are not happy with the bot's service. Human agent is usually assigned randomly based on availability rather than skill sets and the agent may refer the conversation to another more specialized agent if necessary. Therefore the users may have to navigate themselves among different sources of customer support explicitly or are guided by a customer support, either a bot or a human agent, whose job is quite similar to a switchboard operator before the users can receive corresponding customer service.

However, the aforementioned conversation transition could be largely automated so that the users may not be aware of whether the underneath customer support has been switched, i.e., the conversation transition is transparent to the users. The conversation can also be directed to the customer support with the correct domain knowledge given the questions raised by the users can be well understood automatically. This hybrid system is referred as a *Bot-Agent Symbiosis* online customer service, or a *Symbiosis* for short. In the Symbiosis, the users only

†Corresponding author. *Both authors contributed equally.

Proceedings of the 28th International Conference on Computational Linguistics, pages 4056–4066
Barcelona, Spain (Online), December 8-13, 2020

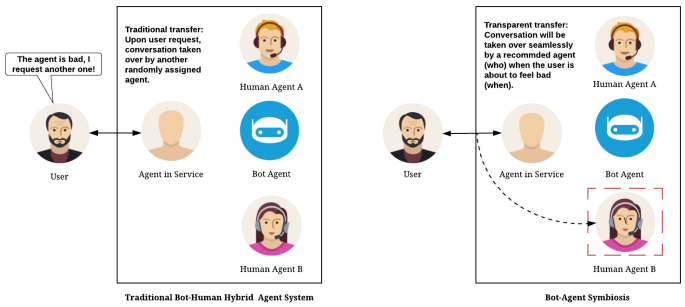


Figure 1: Agent Bot Hybrid and Symbiosis.

need to interact with the system as a whole, instead of figuring out their own ways by speaking to various sources. See Figure 1 for the comparison between a traditional bot-agent hybrid system and a symbiosis.

In our own practice of building towards the symbiosis, two critical questions that are not well answered by the current NLP techniques are identified, namely, ***when*** should the transition occur and ***who*** the conversation session should be transferred to. The who problem is considered to be more difficult than the when problem as the former requires a broader understanding about agent profiles while the later only involves only the current dialogue. In this work, we propose a framework to utilize historical user feedback in order to handle these two problems in one go. The essential idea is to develop a result-driven symbiosis that attempts to maximize user satisfaction so that when the user is predicted to be upset by the current customer support, the transition will be triggered and another (human or bot) agent who is expected to be good at handling the current dialogue will be automatically assigned to take over the current session.

2 Related Works

Different from the call center scheduling which aims to make the calling operation efficient and productive on the basis of knowledge of agents and their schedules (Aksin et al., 2007; Fukunaga et al., 2002; Hashemi et al., 2018; Kiseleva et al., 2016; Sano et al., 2016), the bot-agent symbiosis is a relatively new concept and the conversation transition is a new problem owing to the emerging of chatbot-based customer services. As briefly touched in the last section, the conversation transition contains the ***when*** question, which can be treated as predicting the satisfaction level in real-time, and the ***who*** question, which can be treated as understanding what kind of skills are most needed from the agent based on the current session.

Predicting the satisfaction level of a dialogue can be considered as a scalar regression problem while understanding the required skills is a typical text classification problem. However, due to the fact that current NLP datasets are seldom satisfaction level annotated, the regression problem is little discussed. Traditional methods to assess the satisfaction level is to gather feedback from the customer via survey after the dialogue ends. And some works tried to get the real time feedback through sentiment analysis of the utterances (Bertero et al., 2016; Acosta and Ward, 2011). However, our work shows that sentiment information is not enough to convey the satisfaction level.

Most existing text classification research focus more on monologue text (Zhou et al., 2015; Joulin et al., 2017; Zhou et al., 2016; Liu et al., 2019) rather than multi-turn conversations between multiple speakers. Compared to text classification, multi-turn conversation classification is more complicated due to its nature of interaction. Related works include dialogue act detection in meetings, customer service, online forum, etc. Previous works from (Khanpour et al., 2016; Oraby et al., 2017) commonly aim to understand how utterances from multiple speakers relate to the roles of the speakers. To the best of our knowledge, our work is the first

to formalize the conversation transition problem and the first to attempt the problem with a satisfaction driven approach.

3 Bot-Agent Symbiosis Conversation Transition

The user satisfaction feedback is crucial to most symbiosis design, as the ultimate goal of the symbiosis is well paralleled to maximizing the user satisfaction. A lot of existing systems do employ similar feedback mechanics yet how exactly can the feedback be harnessed to improve the customer service is not very well understood. Suppose the users seek various supports requiring different skill sets from the customer service and there are ways that the user can submit their feedback in the form of a score or indicator, we propose the following framework for building a Bot-Agent symbiosis.

- Build skill set profiles $Z = \{z_1, z_2, \cdots, z_q\}$ for every bot agent and human agent to evaluate if specific customer support is good at answering certain questions;
- Evaluate how satisfied the users is about the current support provided;
- Evaluate the type of customer support that is requested by the users;
- Monitor the predicted satisfaction and trigger conversation transition when it drops below certain threshold and match the current session to the best support available based on the profile and predicted type of the user question.

Here we formalize the type classification and satisfaction level prediction problem, starting from the notations. A conversation or a dialogue is composed of multiple utterances from two speakers, namely, the user and the customer support. The user and the custom support are denoted by A and B respectively. A conversation is denoted by the following set $D^L = \{A_{u_1}, B_{u_1}, \cdots, A_{u_m}, B_{u_n}\}$, where m and n are the numbers of the utterances owned by the user and the custom support respectively and L is the total number of utterances. Without loss of generality, the conversation is tagged with a type label c_i and there are in total k type, so that the set of all types is $C = \{c_1, c_2, \cdots, c_k\}$. It may also be annotated with a scalar satisfaction indicator p where larger p implies better service. Our model aims to predict the type c_i and satisfaction indicator p when a fresh unlabeled utterances of the conversation is provided. In fact the prediction of type and satisfaction indicator can be provided by training two separate yet quite similar deep networks, using a corpus of previous labeled dialogues.

Now the ***when*** question is determined by a function $H(\hat{p}, l)$. And the ***who*** question can be answered by another function $E(\hat{c}, Z, l)$. We call these two functions when and who transition functions which can then be decided dependently and we will show our specific implementation in the following sections.

4 Deep Neural Networks Methods

Figure 2 is the overall architecture of our proposed model. The entire network stacks up three different layers: firstly a convolutional neural network, secondly a bidirectional recurrent neural network and finally an attention layer. Each layer has its unique role in transforming the plain text to either the types label or regressed satisfaction level. We detail how each layer works in the following subsections.

4.1 CNN

We first pad each utterance to the length of l, the maximum length of the utterance supported by the system, and then pad each dialogue with length g, the maximum length of the dialogue supported by the system, so that all dialogues can be represented with a fixed size form $\{x_1, x_2, .., x_n\}$, where $n = l \times g$. Let e_i be the d-dimensional word vectors for the i-th word in a dialogue. Via word embedding, a dialogue is in turn represented as $\{e_1, e_2, ..., e_n\}$. A convolution

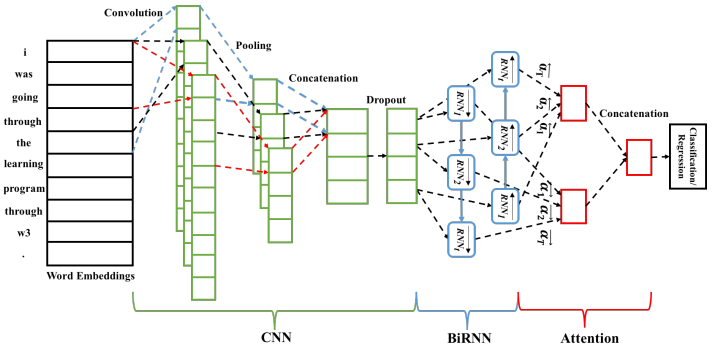


Figure 2: The Architecture of the Model.

operation involves a filter $\omega \in \mathbb{R}^{r \times d}$ and a bias term $b \in \mathbb{R}^r$, which are applied to r $(1 \leq r \leq n)$ continuous words to produce a new value $o_j \in \mathbb{R}$ (Kim, 2014):

$$o_j = \omega^T \cdot e_{i:i+r-1} + b \tag{1}$$

The filter convolves along the height of the dialogue with one stride to produce an output sequence $o = [o_1, o_2, \cdots, o_{n-r+1}]$. To introduce non-linearity, a ReLu activation function is utilized to produce a feature map $f = [f_1, f_2, \cdots, f_{n-r+1}]$.

$$f_j = \max(0, o_j) \tag{2}$$

We then apply a max-over-time pooling operation over the feature map and take the maximum value $\hat{f} = \max\{f\}$ as the feature corresponding to this particular filter (Collobert et al., 2011). A dropout layer is added to the end to prevent networks from over-fitting and produces the CNN output $s = [s_1, s_2, \cdots, s_T]$. The essential idea of this layer is to capture the most dominant features in a single utterance and notice that multiple filters with different window sizes help to learn across diverse phrase structures.

4.2 BiRNN

Given that the input sequences s is in the order of starting from the first symbol s_1 to the last one s_T, we use a bidirectional RNN to abstract the representations of the symbols by summarizing information from both directions. The BiRNN contains a forward $\overrightarrow{RNN}$ which reads the symbol s_i from s_1 to s_T and a backward $\overleftarrow{RNN}$ which reads from s_T to s_1.

$$\begin{aligned} \overrightarrow{h_t} &= \overrightarrow{RNN}(s_t), t \in [1, T], \\ \overleftarrow{h_t} &= \overleftarrow{RNN}(s_t), t \in [T, 1] \end{aligned} \tag{3}$$

By concatenating the forward hidden state $\overrightarrow{h_t}$ and the backward one $\overleftarrow{h_t}$, the representation of the given symbol h_t can be obtained as $h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}]$. And finally that the output of BiRNN layer is $h = [h_1, h_2, \cdots, h_T]$. This layer attempts to correlate a single utterance with the context, this is especially helpful when it comes to multiple turn dialogue.

4.3 Attention

We adopt the attention mechanism by (Yang et al., 2016) to allocate attention weights (α_t) to the outputs (h_t) of the forward layer and backward layer of the BiRNN respectively. The u_s in Equation 5 is a context vector with a predefined length and random initial values to measure the importance of each of h_t.

$$v_t = tanh(W_h h_t + b_h) \tag{4}$$

$$\alpha_t = \frac{exp(v_t^\top u_s)}{\sum_{t=1}^{T} exp(v_t \top u_s)} \tag{5}$$

$$S_\alpha = \sum_{t=1}^{T} \alpha_t h_t \tag{6}$$

According to the above equations, the new representation of the forward layer ($\overrightarrow{S}_\alpha$) and the backward layer ($\overleftarrow{S}_\alpha$) are computed. These two vectors are concatenated to produce the attentive representation (S_α) of the dialogue. This layer can further reward utterance representation that are critical to correctly classifying a dialogue or predicting the satisfaction level.

The final fully connected layer is where we differentiate between regression and classification problem. With the inputs from the attention layer, we devise a fully connected layer with k outputs for classification or a single output for regression. Specially, an additional softmax layer will be attached for classification problem. We then train the network weights by optimizing the loss of either cross entropy or the mean squared error accordingly.

5 Experiment

5.1 Dataset

In order to evaluate our methods, we use a dataset generated by an actual online support service involving only human agents solving online learning and training related problems, which will be referred as the CSD (customer service dialog) dataset hereafter. The customer service is dedicated to a corporation's online learning portal that offers personalized training resources to its employees. The CSD dataset contains in total 15356 sessions of dialogues, the number of utterance for each dialogue varies from 1 to 210, and the number of word for each utterance varies from 1 to 1006. All dialogues took place between April, 2017 and January, 2018. The anonymized dataset is available at GitHub[1].

The CSD dataset has several unique features that are not shared by other existing public datasets. Firstly, around half of the dialogues (6,997) are Net Promoter Score (NPS) labeled by the user. NPS is a 0 to 10 scale indicator that serves as a user satisfaction feedback, albeit somewhat subjective, it can in general reflect if the custom service is sufficiently helpful. Secondly, the dataset is also task-oriented labeled by the human agents. There are 8 types ($k = 8$) in total and types distribution are: *course completion* (3734), *finding contents* (2435), *general help* (1778), *technical issue* (1812), *account issue* (644), *following up* (232), *undirected* (3915), and *other* (806). Unlike most other public datasets that are classified based on the specific domain or speech act, the classifications are designed in such way to facilitate further customer supports. Thirdly, the dataset involves 84 agents and we can easily build a classification-based mean NPS profile for each agent. We observe a huge discrepancy among the skill sets and are convinced that this is common and could be exploited by our symbiosis methods.

5.2 Baselines

Our approach is compared with several baseline methods including convolution neural networks (CNN) (Kim, 2014), long short-term memory networks (LSTM), bidirectional long short-term memory (BiLSTM), a cascade of convolution neural networks and long short-term memory networks (CNN-LSTM) (Zhou et al., 2015), a cascade of convolution neural networks and

[1] https://github.com/paper20/coling

bidirectional long short-term memory networks (CNN-BiLSTM), two bidirectional neural network both with attention mechanism (HAN) (Yang et al., 2016), and the bidirectional encoder representations from transformers model (BERT) (Devlin et al., 2019).

5.3 Model Configurations

Several word vectors and different embedding methods are also compared. The word vectors include: fastText (Bojanowski et al., 2017; Joulin et al., 2016; Joulin et al., 2017), Word2Vec with Skip-gram and CBOW architecture (Mikolov et al., 2013), GloVe (Pennington et al., 2014), predictive text embedding (PTE) which utilizes both labeled and unlabeled information of the training corpus to learn embedding of the words (Tang et al., 2015). The embedding methods include Rand-Static (all words are randomly initialized and then kept static), Rand-Dynamic (all words are randomly initialized and then fine-tuned), Embedding-Static (all words are randomly initialized with pre-trained word vectors and are then kept static), Embedding-Dynamic (all words are randomly initialized with pre-trained word vectors and are then fine-tuned), Embedding-2C (a model with two channels of word vectors in which one of the channel is fine-tuned while the other is kept static).

5.4 Hyperparameters and Training

The data set is split into training set, validation set and testing set with a ratio of 0.8, 0.1 and 0.1 five times. And each set preserves the percentage of samples for each conversation type. For the BERT model, the best epoch number is 5 in our work which is determined by experiments and we use the "BERT-Base, Uncased" pre-trained model for fine-tuning. For other experiments we use the following configurations: 20 epochs, mini-batch size of 32, rectified linear units as CNN activation function, filter windows of 3, 4 and 5 with 32 feature maps each, drop out rate of 0.5, embedding size of 128, hidden units number of 300, max pooling size of 4 and attention size of 100 for HAN model and our model. The hyperparameters of the models are tuned on the validation set with an early stopping at no Micro_F1 increasing during 1000 batches training. We use RMSprop (Tieleman and Hinton, 2012) algorithm to train all models with a learning rate of 0.001 and a discounting factor of 0.9. The mean result of 5 folds cross validation is used as the final result. The source codes of our experiments will be publicly available upon acceptance.

5.5 Results and Discussion

In this section, we first compare our model with other baseline models on the task of dialogue classification. Then we initialize our model with different word vectors and embedding methods. And the NPS evaluated by our model is also discussed. To examine how good the text features can be learned by our model, we plot and analyze the features for a few example cases. At last, we present a straightforward Bot-Agent symbiosis conversation transition policy based on the metric change according to the conversation length.

5.5.1 Model Comparison

Results of our model against other methods are listed in Table 1. Except for BERT, the words fed to the models are all represented by randomly initialized vectors and hyperparameters to train the models are all the same. As can be seen from the Table, our model has a better Micro_F1 and slightly worse Macro_F1 compared to BERT. Because Micro_F1 is commonly considered to be a better metric for the imbalance dataset and training the BERT pre-trained model based on the large public corpus and GPUs is fairly expensive (Devlin et al., 2019), our model is a better choice for practical and resource-constrained or time-constrained applications. Moreover, our model significantly outperforms other homogeneous models in terms of both Micro_F1 and Macro_F1 metric and is better than other mixed models as well.

We attribute the performance of our model to its carefully designed three-layer structure where each layer has its unique role. The CNN layer is responsible for capturing both local and

Table 1: Comparison of Models

	Model							
Metric	CNN	LSTM	BiLSTM	CNN-LSTM	CNN-BiLSTM	HAN	BERT	Ours
Micro_F1	0.7816	0.8052	0.8070	0.8145	0.8201	0.8157	0.8115	**0.8277**
Macro_F1	0.6664	0.7323	0.7301	0.7451	0.7465	0.7385	**0.8048**	0.7668

Table 2: Results with Different Word Vectors

	Word Vector					
Metric	fastText	Skip-Gram	CBOW	GloVe	PTE	Rand-Dynamic
Micro_F1	**0.8430**	0.8380	0.8381	0.8363	0.8305	0.8277
Macro_F1	**0.7850**	0.7779	0.7764	0.7684	0.7598	0.7668
Average Batches	4441	4141	4681	4921	2481	**5741**

Table 3: Results with Different Embedding Methods

	Embedding Method				
Metric	Rand-Static	Rand-Dynamic	fastText-Static	fastText-Dynamic	fastText-2C
Micro_F1	0.8118	0.8277	0.8339	**0.8430**	0.8423
Macro_F1	0.7357	0.7668	0.7659	**0.7850**	0.7806

global context of the multi-turn conversation while the BiRNN layer is responsible for relating utterances from both the users and the agents. CNN or BiRNN alone is clearly not enough as indicated by the outcomes and we believe this is due to the fact that the satisfaction level is dynamically changing as the conversation unfolds. The attention layer prevents the previous two layers from overfitting by correctly directing the learning toward more important parts of the utterances. The order of these three layers is important as well since each layer concentrates lower level features into higher abstraction, reversing the order will lead to significant loss of context. Additionally, reordering them is trivial and the result is not comparable to those presented therefore is omitted.

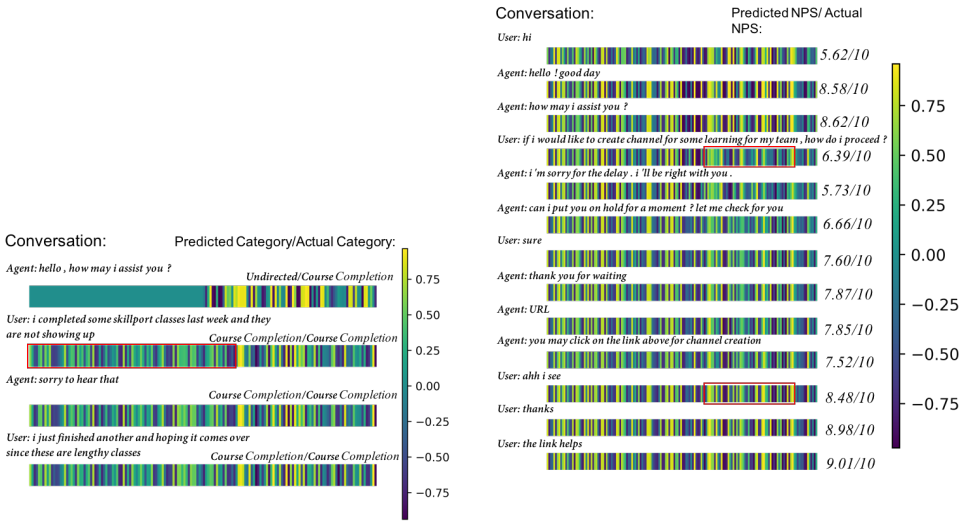
5.5.2 Configuration Comparison

Initializing word vectors with those obtained from an unsupervised neural language model and fine tuning them during the training phase are popular ways to improve performance when a large supervised training set is inaccessible (Kim, 2014). We first initialize our model with word vectors trained by several classical methods and compare their performance for dialogue classification, then we investigate the best way to feed the word vectors to our model

Table 2 shows that initiating model with word vectors can increase the metrics and decrease the average batches needed for model training. The results show that the unlabeled conversation corpus generated from custom service system contains much context informations and should be exploited in Bot-Agent symbiosis transition system. Also, from Table 3 we can see that the best embedding methods are dynamic embedding and two channels embedding in our case.

5.5.3 NPS

Estimating the NPS score at the dialogue level is a challenging task as NPS labels are generated in a rather subjective, if not completely arbitrary, fashion and the conversation itself does not necessarily capture all information needed to predict the NPS score. For example, we found that the NPS is only weakly correlated ($r = 0.117$, r is the pearson correlation coefficient.) to the



(a) (b)

Figure 3: (a) Visualization of representations learned by our model for category classification with dynamic fastText embedding. The output of attention layer (of dimension 160) are plotted in the form of color-bars. (b) Visualization of representations learned by our model for NPS prediction with dynamic fastText embedding. The output of attention layer (of dimension 160) are plotted in the form of color-bars.

sentiment score[2] by IBM Watson NLP API, which is counter-intuitive. However, we managed to achieve an overall root-mean-square error (RMSE) of around 3.0 using other deep neural networks, and our proposed model with the optimal configuration used by the classification also stands out as the best with an RMSE of 2.13, a comparatively salient improvement compared to the baseline RMSE 5.3 of random guessing. The exact threshold of determining when to transfer the conversation could be application specific and a RMSE of around 2 filters most of the unsatisfactory chats according to our observation.

5.5.4 Visualization

To examine how good the features learned by our model are, we analyzed the features for many cases. And here we showcase two intuitive examples in the fashion of visualized feature vector (Mishra et al., 2017; Li et al., 2016). Figure 3a presents an example of test case for the task of dialogue classification as the dialogue progresses. Our model successfully recognizes the correct type from the second utterance as it categorically asked about a course related problem. And as outlined by the red box that the second utterance showcases a unique learned feature that corresponds to the correct category label. Figure 3b presents another example of NPS prediction. The two red boxes again demonstrates the power of our model to capture unique features that are most relevant to deciding the NPS. Notice that from the fourth utterance to the fifth utterance, the agent made the user wait for sometime and the apology from the agent directly results in a drop in the NPS. Similarly in the eleventh utterance where the user replies and acknowledges the help from the agent, the predicted NPS is dramatically raised to another level. In the end the NPS saturates around 9 as the user show his gratitude to the agent. These results show that our model correctly predicts the type and satisfaction score dynamically as the conversation progresses.

[2] An indicator ranging from -1 to 1, the sentiment score suggests if the speaker is feeling positive or negative in general.

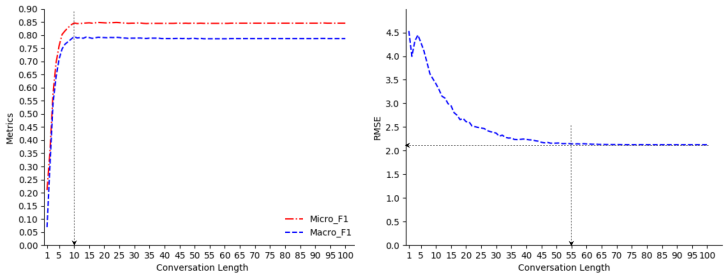


Figure 4: Micro_F1 and Macro_F1 change and RMSE change according to the length of the conversation.

5.5.5 A Straightforward Transition Policy

Premature transition may cause inadequate understanding of the question and late transition may wear our the user's patience. So we analyzed how our models perform with the reference to the conversation length, effectively number of utterances. Dialogue type and NPS prediction are not very stable for the first few utterances as can be seen from Figure 4, and the performance saturate around 10 and 55 utterances, respectively for dialogue type and NPS. Dialogue type emerges quicker than NPS can stabilize as the type normally remain unchanged throughout the dialogue while NPS is highly susceptible to single utterance disturbance. Therefore the conversation session should be protected for the first few turns and we decided that 8 is a better number empirically, since we want the NPS to be more sensitive. Furthermore, the profile of an agent can be represented as the mean NPS for each dialogue type [3], such as $z_i = \{g_{c_1}, g_{c_2}, \cdots, g_{c_8}\}$. And now the when transition function is Equation 7. When $H = 1$ transition takes place and otherwise the session remains, and 4.0 is the minimum NPS our implementation can tolerate before transition.

$$H(\hat{p}, l) = \begin{cases} 1 & if \quad l \geq 8 \quad and \quad \hat{p} \leq 4.0 \\ 0 & if \quad others \end{cases} \tag{7}$$

And the who transition function is Equation 8, effectively the function E mentioned before. Where μ is the input vector to the final softmax layer of the dialogue type network. Meanwhile, we preserve the common design that a user can request a human agent for help when necessary.

$$Agent = \arg\max_i (\sum z_i \mu^T) \tag{8}$$

6 Conclusion and Future Work

In this paper, our work formalizes a new research direction of conversation transition and puts forward an elementary framework for building towards the Bot-Agent symbiosis. Additionally our study based on the CSD dataset shows that the CNN-BiRNN-Attention neural network shows promising results on solving the when and who problems. Furthermore, we argue that the incorporation of user satisfaction feedback is vital to current dialogue system designs since it offers a very strong metric to be learned from. In the future, we will mainly focus on actually implementing an hybrid system that allows complicated interactions between the user, agents and bots, in the hope of collecting realistic data from the hybrid platform. And we will test how the transparent transition can affect the satisfaction level and ultimately bring the symbiosis to reality.

[3] If no NPS is available for the agent under specific type, then the mean NPS for this type across all other agents will be assigned as a placehoder

References

Jaime C Acosta and Nigel G Ward. 2011. Achieving rapport with turn-by-turn, user-responsive emotional coloring. *Speech Communication*, 53(9-10):1137–1148.

Zeynep Aksin, Mor Armony, and Vijay Mehrotra. 2007. The modern call center: A multi-disciplinary perspective on operations management research. *Production and operations management*, 16(6):665–688.

Dario Bertero, Farhad Bin Siddique, Chien-Sheng Wu, Yan Wan, Ricky Ho Yin Chan, and Pascale Fung. 2016. Real-time speech emotion and sentiment recognition for interactive dialogue systems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1042–1047.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Alex Fukunaga, Ed Hamilton, Jason Fama, David Andre, Ofer Matan, and Illah Nourbakhsh. 2002. Staff scheduling for inbound call and customer contact centers. *AI Magazine*, 23(4):30.

Seyyed Hadi Hashemi, Kyle Williams, Ahmed El Kholy, Imed Zitouni, and Paul A Crook. 2018. Measuring user satisfaction on smart speaker intelligent assistants using intent sensitive query embeddings. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1183–1192.

Zhuoxuan Jiang, Jie Ma, Jingyi Lu, Guangyuan Yu, Yipeng Yu, and Shaochun Li. 2019. A general planning-based framework for goal-driven conversation assistant. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9857–9858.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomas Mikolov. 2016. Fasttext.zip: Compressing text classification models.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, April.

Hamed Khanpour, Nishitha Guntakandla, and Rodney Nielsen. 2016. Dialogue act classification in domain-independent conversations using a deep recurrent neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2012–2021.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751, October.

Julia Kiseleva, Kyle Williams, Ahmed Hassan Awadallah, Aidan C Crook, Imed Zitouni, and Tasos Anastasakos. 2016. Predicting user satisfaction with intelligent assistants. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 45–54.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691, June.

Wenqiang Liu, Hongyun Cai, Xu Cheng, Sifa Xie, Yipeng Yu, and Hanyu Zhang. 2019. Learning high-order structural and attribute information by knowledge graph attention networks for enhancing knowledge graph embedding.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Abhijit Mishra, Kuntal Dey, and Pushpak Bhattacharyya. 2017. Learning cognitive features from gaze data for sentiment and sarcasm classification using convolutional neural network. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics: Long Papers*, volume 1, pages 377–387.

Shereen Oraby, Pritam Gundecha, Jalal Mahmud, Mansurul Bhuiyan, and Rama Akkiraju. 2017. How may i help you?: Modeling twitter customer serviceconversations using fine-grained dialogue acts. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, pages 343–355. ACM.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing*, pages 1532–1543.

Shumpei Sano, Nobuhiro Kaji, and Manabu Sassano. 2016. Prediction of prospective user engagement with intelligent assistants. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics : Long Papers*, volume 1, pages 1203–1212.

Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1165–1174. ACM.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31.

Anbang Xu, Zhe Liu, Yufan Guo, Vibha Sinha, and Rama Akkiraju. 2017. A new chatbot for customer service on social media. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 3506–3510. ACM.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2015. A C-LSTM neural network for text classification.

Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. 2016. Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3485–3495, December.