Overview of the Sixth Dialog System Technology Challenge: DSTC6

Chiori Hori[a], Julien Perez[b], Ryuichi Higasinaka[c], Takaaki Hori[a], Y-Lan Boureau[d], Michimasa Inaba[e], Yuiko Tsunomori[f], Tetsuro Takahashi[g], Koichiro Yoshino[h], Seokhwan Kim[i]

[a]*Mitsubishi Electric Research Laboratories, Cambridge, MA, USA*
[b]*Naver Labs Europe, Grenoble, France*
[c]*NTT Corporation, Japan*
[d]*Facebook AI Research, New York, USA*
[e]*Hiroshima City University, Japan*
[f]*NTT DOCOMO, Inc., Japan*
[g]*Fujitsu Laboratories. LTD., Japan*
[h]*Nara Institute of Science and Technology, Ikoma, Nara, Japan*
[i]*Adobe Research, San Jose, CA, USA*

Abstract

This paper describes the experimental setups and the evaluation results of the sixth Dialog System Technology Challenges (DSTC6) aiming to develop end-to-end dialogue systems. Neural network models have become a recent focus of investigation in dialogue technologies. Previous models required training data to be manually annotated with word meanings and dialogue states, but end-to-end neural network dialogue systems learn to directly output natural-language system responses without needing training data to be manually annotated. Thus, this approach allows us to scale up the size of training data and cover more dialog domains. In addition, dialogue systems require a meta-function to avoid deploying inappropriate responses generated by themselves. To challenge such issues, the DSTC6 consists of three tracks, 1. End-to-End Goal Oriented dialogue Learning to select system responses, 2. End-to-End Conversation Modeling to generate system responses using Natural Language Generation (NLG) and 3. Dialogue Breakdown Detection. Since each domain has different issues to be addressed to develop dialogue systems, we targeted restaurant retrieval dialogues to fill slot-value in Track 1, customer services on Twitter by combining goal-oriented dialogues and ChitChat in Track 2 and human-machine dialogue data for ChitChat in Track 3.

DSTC6 had 141 people declaring their interests and 23 teams submit-

ted their final results. 18 scientific papers were presented in the wrap-up workshop. We find the blending end-to-end trainable models associated to meaningful prior knowledge performs the best for the restaurant retrieval for Track 1. Indeed, Hybrid Code Network and Memory Network have been the best models for this task. In Track 2, 78.5% of the system responses automatically generated by the best system were rated better than acceptable by humans and this achieves 89% of the number of the human responses rated in the same class. In Track3, the dialogue breakdown detection technologies performed as well as human agreements, in both data-sets of English and Japanese.

Keywords: DSTC, end-to-end dialogue system, conversation model, sequence-to-sequence model, Natural Language Generation, dialogue breakdown

---

1. Introduction

Recent advancements in artificial intelligence have contributed to closing the gap between the technologies and their uses in our daily life. One of the practical successes is that natural language dialogues have been used as a means of human machine interface implemented in many consumer devices. However, the current dialogue systems still have limited capabilities of conducting natural interactions which is generally taken for granted in human-human conversations.

As a collaborative effort towards further advancements in dialogue technologies, Dialog State Tracking Challenges (DSTCs) have provided common test beds for various research problems focusing on, but not limited to, the task of dialog state tracking. Given the complexity of the dialogue phenomenon and the interest of the research community in a wider variety of dialogue related problems, the DSTC has rebranded itself as Dialog System Technology Challenges for its sixth edition.

Starting as an initiative to provide a common testbed for the task of dialogue state tracking, the first Dialog State Tracking Challenge (DSTC) was organized in 2013, followed by Dialog State Tracking Challenges 2 & 3 in 2014. More recently, Dialog State Tracking Challenge 4 and Dialog State Tracking Challenge 5 have been completed in 2015 and 2016. Since 2014, the challenge as evolved in several ways. First, from human-computer interactions, the challenges started to investigate human-human interactions. Then, the event started to offer pilot tasks on Spoken Language Understanding, Speech Act Prediction, Natural Language Generation and End-to-end System Evaluation which increased the reach of the challenge into the research community of dialogue systems and AI.

Given the remarkable success of the first five editions of the DSTC, and understanding both, the complexity of the dialogue phenomenon and the interest of the research community in a wider variety of dialogue related problems, the DSTC rebrands itself as "Dialog System Technology Challenges" for its sixth edition. In this sixth edition of the DSTC, the call for task proposals has resulted into three tracks, 1. End-to-End Goal Oriented Dialogue Learning, 2. End-to-End Conversation Modeling and 3. Dialogue Breakdown Detection as shown in Table 1. The objective of the tracks is to invite interested organizations conduct dialogue related challenges in specific areas of research and under the umbrella of the DSTC.

These three tasks are selected from the viewpoints of impact and difficulty for dialogue research community. The first track for End-to-End Goal Oriented Dialogue Learning task inherits previous dialogue state tracking

Table 1: Specification of 3 Tracks of DSTC6

Track 1	
Target	Sentence selection
Objective	To select the next utterance in a list of candidates in the context of goal-oriented dialogue management
Dialogue type	Task-oriented dialogue between user and system for restaurant retrieval
#dialogues	40,000 generated dialogues and 4 Knowledge Bases
Evaluation metrics	Mean Reciprocal Rank
Languages	English only

Track 2	
Target	Sentence generation
Objective	System response generation of natural language using models trained from text dialogue data without intention annotation
Dialogue type	Task-oriented dialogue between user and human operator for customer service
#dialogues	We used 1,024 twitter accounts for training and 100 and 116 for test and validation including the domains of Airline, Car, Retail, Fast food chains, etc. The dataset contains 888,201, 107,506, 2,000 dialogues for train, development, test, respectively.
Evaluation metrics	BLEU, Meteor, ROUGE_L, CIDEr, Skip Thought, Embedding Agerage, Vector Exream, Greedy Matching, Human rating using Likert scale for response quality
Languages	English only

Track 3	
Target	Dialogue Breakdown detection
Objective	To detect whether a system utterance causes a dialogue breakdown in a given dialogue context
Dialogue type	Non-task-oriented (Chat-oriented) dialogue between user and system
#dialogues	English: 615 dialogues, Japanese: 150 dialogues (NB. for Japanese, an additional 1,546 dialogues from previous series of DBDCs could be used)
Evaluation metrics	Classification-related metrics: accuracy, precision, recall, F-measure, and distribution-related metrics: Jensen-Shannon Divergence and Mean Squared Error
Languages	English/Japanese

challenges, especially the second challenge, with modern approaches of end-to-end learning that try the direct prediction of next system action to the user utterance and its dialogue history. The second track for End-to-End Conversation Modeling is standing on other recent trends of the dialogue system area, which tries to model a conversation as directly generating sentences given a user query in the open domain. Due to the rises of neural conversation modeling, the task attracts much attentions from the research community of dialogue system. The third track for Dialogue Breakdown De-

tection stands on more practical viewpoint. Controlling statistical dialogue models to suppress unexpected responses becomes an important task due to the development of statistical dialogue models, especially if we want to use dialogue systems on real products.

Since each domain has different issues to be addressed to develop dialogue systems, we targeted restaurant retrieval dialogues to fill slot-value in Track 1, customer services on Twitter by combining goal-oriented dialogues and ChitChat in Track 2 and human-machine dialogue data for ChitChat in Track 3. It is noted that ChitChat doesn not have a specific goal to accomplish such as a slot filling task that sets values in a table of backend systems. Furthermore, the content structure of ChitChat is not as restricted and most answers can be accepted by humans.

1.1. Workshop summary and future DSTCs

The workshop for the Dialog System Technology Challenge (DSTC) was held on December 10th, 2017 at long beach, CA, USA. The organizers had pre-survey to know interests of dialogue community people, and 141 people declared their interests to the proposed three tasks. Finally, 23 teams submitted their final results for tasks and 18 scientific papers are presented in the workshop. The workshop also had 53 participants including on-site registrations. Detailed results are described in the sections below. The workshop also had many supporting organizations including sponsors, and the challenge data was created with their supports.

2. End-to-End Goal Oriented Dialogue Learning (Track 1)

2.1. Introduction

Goal-oriented dialogue requires reasoning competencies that go beyond language modeling. For example, asking questions to clearly define a user request, querying Knowledge Bases (KB's), interpreting results from queries to display options to users or completing a transaction are some of the important competencies a dialogue system has to master in order to be useful. On the one hand, such difficulties make it hard to ascertain how well end-to-end dialogue models would do, and whether they are in a position to replace traditional dialogue methods in a goal-directed setting. On the other hand, because end-to-end dialogue systems make no assumption on the domain or dialogue state structure, they are holding the promise of easily scaling up to new domains. This challenge aims to make it easier to analyze the performance of end-to-end systems in a goal directed setting, using an expanded version of the Facebook AI Research open resource proposed in [4]. The goal of the challenge is to assess the capabilities of the proposed systems to fulfill a set of four basic tasks related to transactional dialogues. The capability of accomplishing all four tasks on a single dialogue corpus has been tested as a final task.

2.2. The task - Restaurant Reservation

The transactional dialogue simulation system is based on an underlying KB. The facts contain the restaurants that can be booked and their properties queried. Each restaurant is defined by a type of cuisine (10 choices, e.g., French, Thai), a location (10 choices, e.g., London, Tokyo), a price range (cheap, moderate or expensive), a rating (from 1 to more than 200), and other characteristics like dietary restrictions and atmosphere. For simplicity, we assume that each restaurant only has availability for a single party size (2, 4, 6 or 8 people). Each restaurant also has an address and a phone number listed in the KB.

The KB can be queried using API calls, which return the list of facts related to the corresponding restaurants. Each query must contain a certain number of slots: a location, a type of cuisine, a price range, a party size, and possibly other required slots like dietary restriction, depending on the set used. Each data file has the same set of required slots for every dialogue. A query can return facts concerning one, several or no restaurant (depending on the party size). Using the KB, conversations are generated in the format shown in Figure 1. Each example is a dialogue comprising utterances from a user and a bot, as well as API calls and the resulting facts. dialogues

are generated after creating a user request by sampling an entry for each of the required slots: e.g. the request in Figure 1 is [cuisine: British, location: London, party size: six, price range: expensive]. We use natural language patterns to create user and bot utterances. There are more patterns for the user than for the bot. Indeed, the user can use several ways to say something, while the bot always uses the same way to make it deterministic. Those patterns are combined with the KB entities to form thousands of different utterances. We split types of cuisine and locations in half, and create two KB's, one with all facts about restaurants within the first halves and one with the rest. In [4], the two KB's had 4,200 facts and 600 restaurants each (5 types of cuisine $\times$ 5 locations $\times$ 3 price ranges $\times$ 8 ratings). The data provided here has been expanded to comprise more slots and thus yield many more restaurants, but the two KB still have disjoint sets of restaurants, locations, types of cuisine, phones and addresses, while sharing all other sets of values. We use one of the KB's to generate train and test dialogues, using only one of the extra slots in the queries. There are 4 sets of test dialogues: (1) one that uses the same KB as for the train dialogues, and the same set of slots in the queries; (2) one that uses the second KB (with disjoint sets of restaurants, locations, cuisines, phones and addresses), termed Out-Of-Vocabulary (OOV), but the same set of slots in the queries; (3) one that uses the same KB as for the train dialogues, but one additional slot for the queries; and (4) one that uses the second KB (OOV) and an additional required slot.

For training, systems have access to the training examples and both KBs. Evaluation is conducted on all four test sets. Beyond the intrinsic difficulty of each task, the challenge on the OOV test sets is for models to generalize to new entities (restaurants, locations and cuisine types) unseen in any training dialogue – something natively impossible for embedding methods. Ideally, models could, for instance, leverage information coming from the entities of the same type seen during training.

We generate five datasets, one per task. Training sets are relatively small (10,000 examples) to create realistic learning conditions. The dialogues from the training and test sets are different, never being based on the same user requests. Thus, we test if models can generalize to new combinations of fields.

2.3. End-to-end dialogue learning as sentence-selection

The task of end-to-end dialogue learning has been recently formalized as next-sentence prediction. One of the main motivation of such approach is the abundance of human-to-human dialogue in industrial systems which

contrasts with the lack of annotated data due to the cost and challenge of such process. Formally, a transaction dialogue system based on a sentence selection model needs to choose, among a potentially large number of available utterances extracted from a corpus of dialogues, the most adequate answer with respect to a current dialogue. Several challenges can be identified (1) dialogue representation (2) Reasoning capabilities (3) Back-end system handling. A series of models have been proposed.

First, Long Short Term Memory (LSTM) [17] is a recurrent neural network that has recently known important success in most of the classic Natural Language Processing task. LSTM has become a common model for sentence encoding. In the context of utterance selection, several models have leverage its expressive capability of learn sentence ranking models [26].

Second, Memory Networks [39] are a recent class of models that have also been applied to a range of natural language processing tasks, including question answering [5], language modeling and non-goal-oriented dialogue [8]. By first writing and then iteratively reading from a memory component (using layers called hops) that can store historical dialogues and short-term context to reason about the required response, they have been shown to perform well on those tasks and to outperform some other end-to-end architectures based on simpler Recurrent Neural Networks.

Then, Hybrid Code Networks [48] (HCNs) learns an recurrent neural network but also allow a developer to express domain knowledge via software and action templates. Indeed, simple operations like sorting a list of database results or updating a dictionary of entities can expressed in a few lines of software, yet may take thousands of dialogues to learn. In addition, this neural network can be trained with supervised learning or reinforcement learning, by changing the gradient update applied.

Regarding the learning strategies, the use of pairwaise ranking loss has been proposed. As an alternative, reinforcement learning has been investigated in order to leverage non-diffentiable loss through policy gradient [47]. More recently, adversarial loss has been studied and compared to human choice [27].

2.4. Description of the Dialogue Dataset

We broke down a goal-directed objective into several sub-tasks to test some crucial capabilities that dialogue systems should have (and hence provide error analysis by design). All the tasks involve a restaurant reservation system, where the goal is to book a table at a restaurant. Solving our tasks requires manipulating both natural language and symbols from a KB. The tasks are generated by a simulation. Grounded with an underlying KB of

restaurants and their properties (location, type of cuisine, etc.), these tasks cover several dialogue stages and test if candidate models can learn various abilities such as performing dialogue management, querying KB's, interpreting the output of such queries to continue the conversation or dealing with new entities not appearing in dialogues from the training set.

Task 1: Issuing API calls. A user request implicitly defines a query that can contain from 0 to 4 of the required fields (sampled uniformly; in Figure 1, it contains 3). The bot must ask questions for filling the missing fields and eventually generate the correct corresponding API call. The bot asks for information in a deterministic order, making prediction possible.

Task 2: Updating API calls. Starting by issuing an API call as in Task 1, users then ask to update their requests. The order in which fields are updated is random. The bot must ask users if they are done with their updates and issue the updated API call.

Task 3: Displaying options. Given a user request, the KB is queried using the corresponding API call and the resulting facts are added to the dialogue history (if too many facts satisfy the call, a random subset is returned to avoid overly lengthy data). The bot must propose options to users by listing the restaurant names sorted by their corresponding rating (from higher to lower) until users accept. For each option, users have a 25% chance of accepting. If they do, the bot must stop displaying options, otherwise propose the next one. Users always accept the option if this is the last remaining one. We only keep examples with API calls retrieving at least 3 options.

Task 4: Providing extra information. Given a user request, we sample a restaurant and start the dialogue as if users had agreed to book a table there. We add all KB facts corresponding to it to the dialogue. Users then ask for the phone number of the restaurant, its address or both, with proportions 25%, 25% and 50%, respectively. The bot must learn to use the KB facts correctly to answer.

Task 5: Conducting full dialogues. For Task 5, we combine Tasks 1-4 to generate full dialogues just as in Figure 1. Unlike in Task 3, we keep examples if API calls return at least 1 option instead of 3.

The dataset is organized in 5 JSON files corresponding to each of the tasks previously mentioned. Basically, a dialogue piece is followed by a list of next-utterance candidates. In the training set, the answer (the single correct candidate) is provided. The goal is to rank the candidates. Precisions @{1,2,5} will be used to evaluate the models. These measures correspond to the probability of the correct utterance to be the 1st best, part of the

2-best and 5-best hypotheses output by each model, respectively. Rank of candidate utterances will be 1-indexed. Evaluation uses per-response accuracies. Evaluation is conducted in a ranking, not a generation, setting: at each turn of the dialogue, the participants have to test whether they can predict bot utterances and API calls by selecting a candidate, not by generating it.[1] Candidates are ranked from a set of candidate utterances and API calls.

Table 2: Provided data for Track 1. Tasks 1-5 were generated using our simulator and share the same KB. Each task have two test sets, one using the vocabulary of the training set and the other using out-of-vocabulary words.

	Tasks	**T1**	**T2**	**T3**	**T4**	**T5**
DIALOGUES *Average statistics*	Number of utterances:	12	17	43	15	55
	- user utterances	5	7	7	4	13
	- bot utterances	7	10	10	4	18
	- outputs from API calls	0	0	23	7	24
DATASETS *Tasks 1-5 share the same data source*	Vocabulary size			3,747		
	Candidate set size			4,212		
	Training dialogues			1,000		
	Validation dialogues			1,000		
	Test dialogues			1,000		

2.5. Results

Table 3 introduces methods proposed during the challenge. End-to-End Memory Network [40], Dynamic Memory Network [20] and Hybrid Code Networks [48] were the main trainable building blocks of the proposed systems. In addition contextual rules were proposed to improve performances.

Table 4 details the results obtained for the participating teams. The two first teams managed to solve the task by obtaining 1.0 precision in the test-set. KB-2 introduced a novel request table slot (ambiance) in the test-set. This slot is available in the knowledge base in both the train and test set.

3. End-to-End Conversation Modeling using NLG (Track 2)

3.1. Introduction

End-to-end training of neural networks is a promising approach to automatic construction of dialogue systems using a human-to-human dialogue corpus. Recently, Vinyals et al. tested neural conversation models using

[1] [28] termed this setting Next-Utterance Classification.

Table 3: Methods implemented in submitted systems for Track 1.

Teams	Method
1/4	Extended Hybrid Code Networks
2	A hierarchical Long Short-Term Memory (LSTM) based ranking module, a Conditional Random Field (CRF)
3	End-to-End Slot-Value Independent Recurrent Entity Network
5	Memory Network with Negative Sample
6	Memory Network with an extra output memory representation named D-Layer with Knowledge based enhancement
7	End-to-End Memory Networks with named entities abstraction and contextual numbering
8	Embedding projection of the text and candidate with rankloss optimization
9	Quantized language model

Table 4: Summaries of the team performances for Track 1 using Precision1, Precision2 and Precision5.

Teams	KB-1	KB-1-OOV	KB-2	KB-2-OOV
team01/04	1.000/1.000/1.000	1.000/1.000/1.000	1.000/1.000/1.000	1.000/1.000/1.000
team02	1.000/1.000/1.000	1.000/1.000/1.000	1.000/1.000/1.000	1.000/1.000/1.000
team03	0.984/0.997/0.999	0.990/0.998/1.000	0.927/0.958/0.990	0.930/0.962/0.991
team05	0.619/0.692/0.831	0.590/0.668/0.797	0.600/0.671/0.822	0.573/0.645/0.782
team06	0.890/0.946/0.995	0.890/0.946/0.994	0.739/0.810/0.932	0.751/0.821/0.914
team07	0.994/0.998/1.000	0.994/0.998/1.000	0.959/0.982/0.986	0.962/0.986/0.990
team08	0.663/0.798/0.912	0.622/0.762/0.923	0.516/0.669/0.835	0.488/0.637/0.833
team09	0.997/0.997/0.998	0.996/0.996/0.998	0.965/0.967/0.980	0.967/0.970/0.978
team10	0.371/0.435/0.652	0.361/0.445/0.644	0.299/0.385/0.639	0.314/0.401/0.648

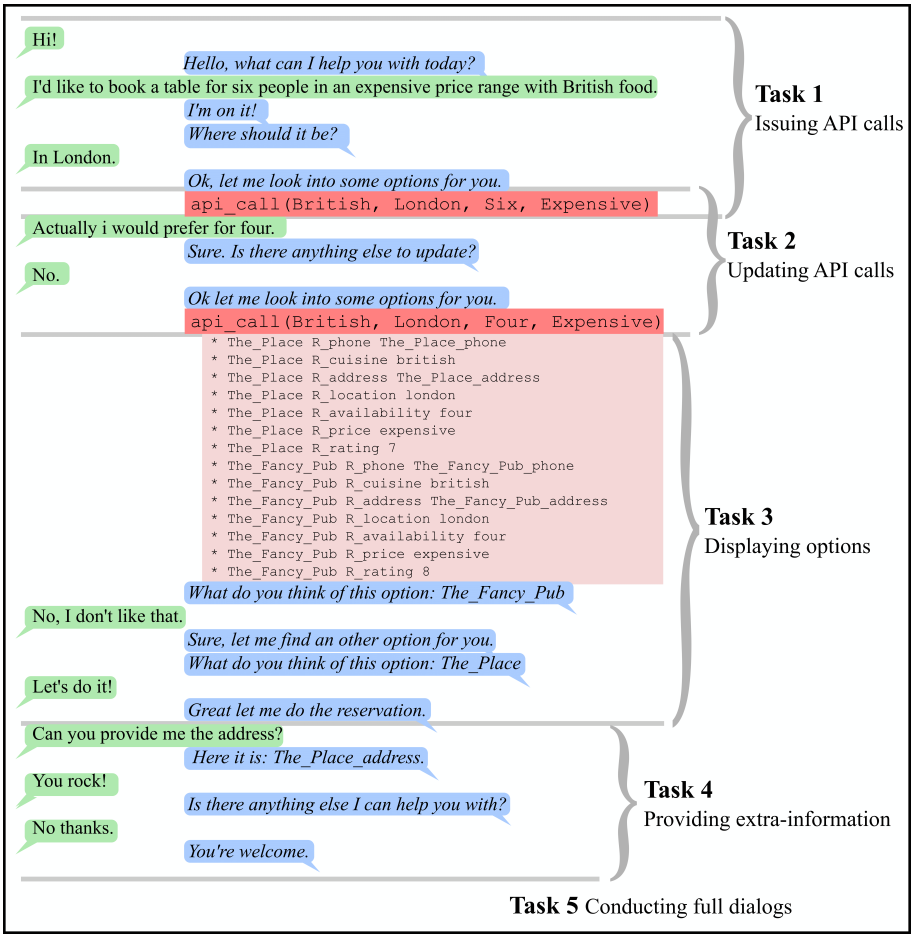

Figure 1: **Task design of goal-oriented dialogue for Track 1.** A user (in green) chats with a bot (in blue) to book a table at a restaurant. Models must predict bot utterances and API calls (in dark red). Task 1 tests the capacity of interpreting a request and asking the right questions to issue an API call. Task 2 checks the ability to modify an API call. Task 3 and 4 test the capacity of using outputs from an API call (in light red) to propose options (sorted by rating) and to provide extra-information. Task 5 combines everything.

OpenSubtitles [43]. Lowe et al. released the Ubuntu Dialogue Corpus [26] for research in unstructured multi-turn dialogue systems. Furthermore, the approach has been extended to accomplish task oriented dialogues to provide information properly with natural conversation. For example, Ghazvininejad et al. proposed a knowledge grounded neural conversation model [10], where the research is aiming at combining conversational dialogues with

task-oriented knowledge using unstructured data such as Twitter data for conversation and Foursquare data for external knowledge. However, the task is still limited to a restaurant information service, and has not yet been tested with a wide variety of dialogue tasks. In addition, it is still unclear how to create intelligent dialogue systems that can respond like a human agent.

In consideration of these problems, we proposed a challenge track to the 6th dialog system technology challenges (DSTC6) [2]. The focus of the challenge track is to train end-to-end conversation models from human-to-human conversation in order to accomplish end-to-end dialogue tasks for a customer service. The dialogue system plays the role of a human agent and generates natural and informative sentences in response to users questions or comments given a dialogue context.

3.2. Tasks

In this challenge track, a system has to generate sentence(s) in response to a user input in a given dialogue context, where it can use external knowledge from public data, e.g. web data. The quality of the automatically generated sentences is evaluated with objective and subjective measures to judge whether or not the generated sentences are natural and informative for the user (see Fig. 2).

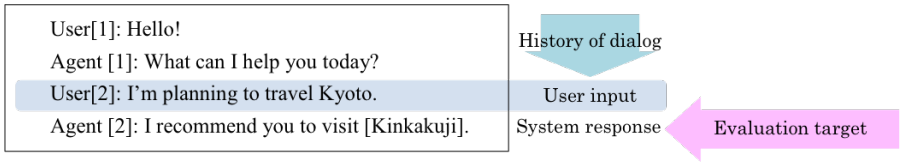

Figure 2: Sentence generation and evaluation in the end-to-end conversation modeling track.

This track aims to generate system responses for Customer service dialogue using Twitter data:

Task A: Full or part of the training data will be used to train conversation models.

Task B: Any open data, e.g. from web, are available as external knowledge to generate informative sentences. But they should not overlap with the training, validation and test data provided by organizers.

[2] http://workshop.colips.org/dstc6/

Challenge attendees can select either A or B, or both. The tools to download Twitter data and extract the dialogue text were provided to all attendees at the challenge track in DSTC6 [18]. The attendees needed to collect the data by themselves. Data collected before Sep. 1st, 2017 was available as trial data, and the official training, development and test data were collected from Sep. 7th to 18th, 2017. The dialogues were used for the test set was not disclosed until Sep. 25th.

3.3. Data Collection

3.3.1. Twitter data

In the Twitter task, we used dialogue data collected from multiple Twitter accounts for customer service. Each dialogue consisted of real tweets between a customer and an agent. A customer usually asked a question or complained something about a product or a service of the company, and an agent responded to the customer accordingly. In this challenge, each participant is supposed to develop a dialogue system that mimics agents behaviors. The system will be evaluated based on the quality of generated sentences in response to customers tweets. For the challenge, we provided a data collection tool to all participants so that they could collect the data by themselves because Twitter does not allow distribution of Twitter data by a third party In this task, it is assumed that each participant continued to collect the data from specific accounts in the challenge period. To acquire a large amount of data, the data collection needed to be done repeatedly, e.g. by running the script once a day, because the amount of data we can download is limited and older tweets cannot be accessed after they expire. At a certain point of time, we provided an additional tool to extract subsets of collected data for training, development (validation), and evaluation so that all the participants were able to use the same data for the challenge. Until the official data sets were fixed, trial data sets were available to develop dialogue systems, which were selected from the data collected by each participant. But once the official data sets were determined, the system needed to be trained from scratch only using the official data sets.

Challenge attendees need to use a common data collection tool included in the provided package. The trial data sets can be extracted from downloaded twitter dialogues using a data extraction script in the package. The official data are collected through the period of Sep. 7th to 18th in 2017, using the data collection tool. The official training, development and test sets can also be extracted using another data extraction script. Finally, the participants are supposed to collect the data sets summarized in Table 5. More

Table 5: Twitter data used for Track 2.

	training	development	test
#dialogue	888,201	107,506	2,000
#turn	2,157,389	262,228	5,266
#word	40,073,697	4,900,743	99,389

information can be found in "https://github.com/dialogtekgeek/DSTC6-End-to-End-Conversation-Modeling".

We trained the data from 1024 twitter accounts for training, 100 accounts for test and 116 account for validation, respectively. The business domain contains Airline, Car, Retail, Fast food chains, etc. The lists of accounts used for the challenge can be found in the following link[3].

3.4. Text Preprocessing

Twitter dialogues contain a lot of noisy text with special expressions and symbols. Therefore, text preprocessing is important to clean up and normalize the text. Moreover, all the participants need to use the same preprocessing at least for target references to assure fair comparisons between different systems in the Challenge.

3.4.1. Twitter data

Twitter data contains a lot of specific information such as Twitter account names, URLs, e-mail addresses, telephone/tracking numbers and hashtags. This kind of information is almost impossible to predict correctly unless we use a lot of training data obtained from the same site. To alleviate this difficulty, we substitute those strings with abstract symbols such as `<URL>`, `<E-MAIL>`, and `<NUMBERS>` using a set of regular expressions. In addition, since each tweet usually starts with a Twitter account name of the recipient, we removed the account name. But if such names appear within a sentence, we leave them because those names are a part of sentence. We leave hashtags as well for the same reason. We also substitute user names with `<USER>` e.g.

```
hi John, can you send me a dm ?
    →  hi <USER>, can you send me a dm ?
```

[3]https://github.com/dialogtekgeek/DSTC6-End-to-End-Conversation-Modeling/tree/master/tasks/twitter official$_a ccount_n ames_\{train|dev|test\}.txt$

Table 6: Submitted systems for Track2.

Team (Entry)	Model type	Objective function	Additional techniques	Paper
baseline	LSTM	Cross entropy		
team_1 (1)				
team_1 (2)				
team_2 (1)	2LSTM+LSTM	Adversarial	Example method	
team_2 (2)	LSTM	Cross entropy	Example method	
team_2 (3)	2LSTM+LSTM	Adversarial + Cosine Similarity	Example method	[46]
team_2 (4)	2LSTM+LSTM	Adversarial		
team_2 (5)	2LSTM+LSTM +HRED	Cross entropy	MBR System combination	
team_3 (1)	LSTM	Cross entropy		
team_3 (2)	2LDTM+Atten.	Cross entropy on diversified data		
team_3 (3)	2LSTM+Atten.	Cross entropy on trial data$^+$		[23]
team_3 (4)	GWGM+Atten.	Cross entropy		
team_3 (5)	SEARG	Cross entropy	Knowledge enhanced model	
team_4 (1)	LSTM	Cross entropy	Word embedding initialization	[2]
team_5 (1)	LSTM	MMI maxdiv	MERT	
team_5 (2)	LSTM	MMI maxBLEU	MERT	
team_5 (3)	LSTM	MMI mixed (maxdiv + maxBLEU)	MERT	[9]
team_5 (4)	LSTM	MMI uniform		
team_5 (5)	LSTM	Cross entropy	Greedy search for decoding	
team_6 (1)				

$^+$Trial data cannot be used for official evaluation. The results are not officially accepted.

Since the user's name can be extracted from the attribute information of each tweet, we can replace it. Note that the text preprocessing is not perfect, and therefore there may remain original phrases, which are not replaced or removed successfully. The text also includes many abbreviations, e.g. "pls hlp", special symbols, e.g. "`(-:`" and wide characters "©♡♣" including 4-byte Emojis. These are left unaltered.

3.5. Submitted Systems

We received 19 sets of system outputs for the Twitter task, from six teams, and four system description papers were accepted [46, 23, 2, 9]. In this section, we summarize the techniques used in the systems, including the baseline system for the challenge track.

The baseline system is an LSTM-based encoder decoder in [18], but this is a simplified version of [43], in which back-propagation is performed only up

to the previous turn from the current turn, although the state information is taken over throughout the dialogue.

Table 6 shows the baseline and submitted systems with their brief specifications including model type, objective function, and additional techniques. An empty specification means that the team did not submit any system description paper to the DSTC6 workshop.

Most systems employed a LSTM or BLSTM (2LSTM) encoder and a LSTM decoder. Some systems used a hierarchical encoder decoder (team_2 (5) and team_3 (5)) and attention-based decoder (team_3 (2–4)). Several types of objective functions were applied for training the models, where cross entropy, adversarial method, cosine similarity, and maximum mutual information (MMI) were used solely or combined. The objective functions except cross entropy were designed to increase the diversity of responses. This hopefully led to more realistic and informative responses.

Furthermore, several additional techniques are introduced to improve the response quality. In [46], an example-based method is used to return real human responses if a similar context exists in the training corpus, and minimum Bayes risk (MBR) decoding is used to improve objective scores. The knowledge enhanced encoder decoder [23] searches for relevant documents in the web using the keywords in the dialogue context, and the relevant documents are used to enhance the decoder. In [2], different types of word-embedding vectors are used for initialization of the models.

3.6. Evaluation

Challenge participants were allowed to submit up to 5 sets of system outputs. The outputs were evaluated with objective measures such as BLEU and METEOR, and also evaluated by rating scores collected by humans using Amazon Mechanical Turk (AMT). The human evaluators rate the system responses in terms of naturalness, informativeness, and appropriateness.

3.6.1. Objective evaluation

For the challenge track, we used `nlg-eval`[4] for objective evaluation of system outputs, which is a publicly available tool supporting various unsupervised automated metrics for natural language generation. The supported metrics include word-overlap-based metrics such as BLEU, METEOR, ROUGE_L, and CIDEr, and embedding-based metrics such as SkipThoughts Cosine Similarity, Embedding Average Cosine Similarity, Vector Extrema Cosine

[4] `https://github.com/Maluuba/nlg-eval`

Similarity, and Greedy Matching Score. Details of these metrics are described in [37].

We prepared 10 more references for a ground truth of each response by humans to operate reliable objective evaluation. The references included a real human response in the Twitter dialogue and 10 human-generated responses. We asked 10 different Amazon Mechanical Turkers for each dialogue to compose a sentence for the final response given the dialogue context. We provided the real human response as an example and asked them to make their responses to be different from the example while keeping to the dialogue topic. We also asked them not to copy and paste the example in their response. When multiple references are available, `nlg-eval` computes the similarity between the prediction and all the references one-by-one, and then selects the maximum value.

3.6.2. Subjective evaluation

We collected human ratings for each system response using 5 point Likert Scale, where 10 different Turkers rated system responses given a dialogue context. We listed randomly 21 responses below the dialogue context, which consists of 19 submitted outputs, a baseline output, and a human response for each dialogue.

The Turkers rated each response by 5 level scores as

Level	Score
Very good	5
Good	4
Acceptable	3
Poor	2
Very poor	1

we instructed to the Turkers to consider naturalness, informativeness, and appropriateness of the response for the given context. If there were identical responses in the list, we reduced them into one response so that they were rated consistently. The average score was computed for each system and reported in Table 7.

3.6.3. Results

Tables 7 and 8 show evaluation results of 21 systems: the 19 submitted systems, the baseline and the reference. Systems are listed as team_M (N), where M is the team index and N is an identifier for a particular system submitted by that team. "Ext. Data" in the table denotes whether or not the system used external data for training and/or testing, where only team_3 (5) used external data (web data) for response generation.

Table 7: Evaluation results with word-overlapping-based objective measures based on 11 references and a subjective measure based on 5-level ratings for Track 2.

Team (Entry)	Ext. Data	BLEU4	METEOR	ROUGE_L	CIDEr	Human Rating
baseline		0.1619	0.2041	0.3598	0.0825	3.3638
team_1 (1)*		0.1598	0.2020	0.3608	0.0780	3.4415
team_1 (2)*		0.1623	0.2039	0.3567	0.0828	3.4297
team_2 (1)		0.1504	0.1826	0.3446	0.0803	3.4453
team_2 (2)		0.2118	0.2140	0.3953	0.1060	3.3894
team_2 (3)		0.1851	0.2040	0.3748	0.0965	3.4777
team_2 (4)		0.1532	0.1833	0.3469	0.0800	3.4381
team_2 (5)		0.2205	**0.2210**	**0.4102**	**0.1279**	3.4332
team_3 (1)		0.1602	0.2016	0.3606	0.0782	3.4503
team_3 (2)		0.1779	0.2085	0.3829	0.0978	3.5239
team_3 (3)**		0.1741	0.2024	0.3703	0.0994	3.5082
team_3 (4)		0.1342	0.1762	0.3366	0.0947	3.5107
team_3 (5)	✓	0.1092	0.1731	0.3201	0.0702	3.3919
team_4 (1)		0.1716	0.2071	0.3671	0.0898	3.4431
team_5 (1)		0.1480	0.1813	0.3388	0.1025	3.5209
team_5 (2)		0.0991	0.1687	0.3146	0.0708	3.3053
team_5 (3)		0.1448	0.1839	0.3375	0.0940	**3.5396**
team_5 (4)		0.1261	0.1754	0.3310	0.0945	3.4545
team_5 (5)		0.1575	0.1918	0.3658	0.1112	3.5097
team_6 (1)*		**0.2762**	0.1656	0.3482	0.1235	2.9906
reference						3.7245

*Results are not officially accepted since any system description paper has not been submitted.
**Results are not officially accepted since the system was tuned with the trial data [23].

Table 9 shows cross-validation results of 11 ground truths generated by humans. Each manually generated sentence was evaluated by comparing with other 10 ground truths using leave-one-out method.

In most objective measures, the system of team_2 (5) achieved highest scores, where the system employed MBR decoding for system combination. This result indicates that explicit maximization of objective measures and the complementarity of multiple systems bring significant improvement for the objective measures.

On the subjective measure with human rating, the system of team_5 (3) achieved the best score (3.5396). Although there were no big differences between the human rating scores, we can see that techniques for improving human rating actually contributed to increase the rating scores. For example, adversarial training (team_2 (3)), use of diversified data (team_3 (2)),

Table 8: Evaluation results with embedding-based objective measures based on 11 references and a subjective measure based on 5-level ratings for Track 2.

Team (Entry)	Ext. Data	Skip Thought	Embedding Average	Vector Extrema	Greedy Matching	Human Rating
baseline		0.6380	0.9132	0.6073	0.7590	3.3638
team_1 (1)*		0.6451	0.9090	0.6039	0.7572	3.4415
team_1 (2)*		0.6386	0.9026	0.6071	0.7587	3.4297
team_2 (1)		0.6451	0.9070	0.5990	0.7534	3.4453
team_2 (2)		**0.7075**	**0.9271**	0.6371	0.7747	3.3894
team_2 (3)		0.6706	0.9116	0.6155	0.7613	3.4777
team_2 (4)		0.6463	0.9077	0.5999	0.7544	3.4381
team_2 (5)		0.6636	0.9251	**0.6449**	**0.7802**	3.4332
team_3 (1)		0.6474	0.9074	0.6031	0.7567	3.4503
team_3 (2)		0.6259	0.9201	0.6106	0.7683	3.5239
team_3 (3)**		0.6348	0.8985	0.6000	0.7573	3.5082
team_3 (4)		0.6127	0.8802	0.5913	0.7412	3.5107
team_3 (5)	✓	0.6132	0.8977	0.5870	0.7420	3.3919
team_4 (1)		0.6529	0.9106	0.6079	0.7596	3.4431
team_5 (1)		0.6131	0.9087	0.5928	0.7433	3.5209
team_5 (2)		0.5952	0.8996	0.5675	0.7257	3.3053
team_5 (3)		0.6025	0.9083	0.5915	0.7433	**3.5396**
team_5 (4)		0.6151	0.8984	0.5814	0.7330	3.4545
team_5 (5)		0.6457	0.9076	0.6075	0.7528	3.5097
team_6 (1)*		0.6989	0.8018	0.5854	0.7316	2.9906
reference						3.7245

*Results are not officially accepted since any system description paper has not been submitted.
**Results are not officially accepted since the system was tuned with the trial data [23].

Table 9: Cross validation results of 11 references for Track2.

	BLEU4	METEOR	ROUGE_L	CIDEr	Skip Thought	Embedding Average	Vector Extrema	Greedy Matching
Original	0.5264	0.3885	0.6559	0.7566	0.7160	0.9483	0.7625	0.8679
Ref (1)	0.2758	0.2357	0.4525	0.3615	0.7091	0.9308	0.6643	0.7958
Ref (2)	0.2626	0.2313	0.4501	0.3477	0.7142	0.9300	0.6646	0.7951
Ref (3)	0.2651	0.2335	0.4499	0.3571	0.7064	0.9328	0.6626	0.7958
Ref (4)	0.2683	0.2358	0.4509	0.3622	0.7039	0.9313	0.6637	0.7951
Ref (5)	0.2682	0.2390	0.4464	0.3611	0.7104	0.9301	0.6632	0.7925
Ref (6)	0.2786	0.2323	0.4476	0.3577	0.7005	0.9291	0.6642	0.7925
Ref (7)	0.2729	0.2382	0.4523	0.3678	0.7049	0.9319	0.6687	0.7971
Ref (8)	0.2593	0.2256	0.4430	0.3488	0.7082	0.9306	0.6604	0.7921
Ref (9)	0.2529	0.2348	0.4436	0.3440	0.7202	0.9325	0.6621	0.7944
Ref (10)	0.2707	0.2364	0.4527	0.3750	0.7105	0.9333	0.6679	0.7968
Average	0.2910	0.2483	0.4677	0.3945	0.7095	0.9328	0.6731	0.8014

and MMI-based objective function with maximum diversity (team_5 (1), (3))

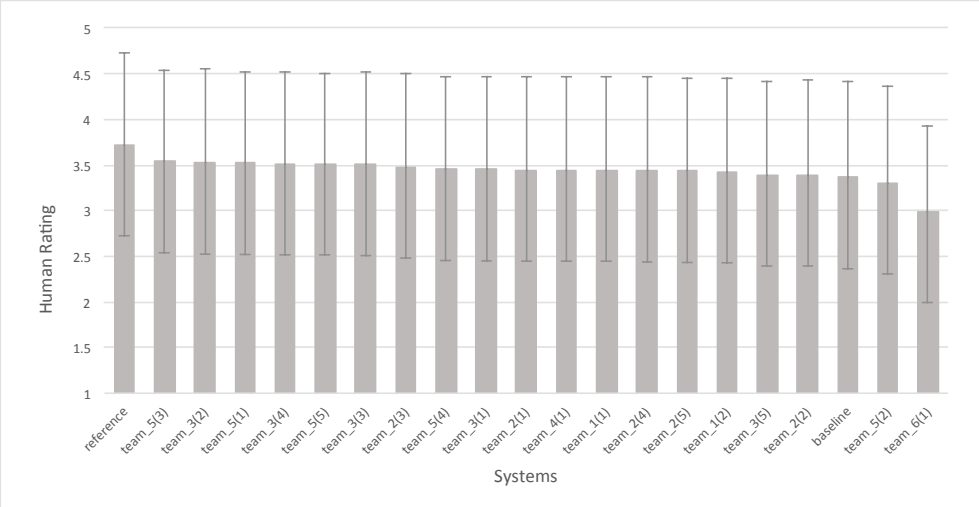

Figure 3: Mean and standard deviation of human rating score for Track 2.

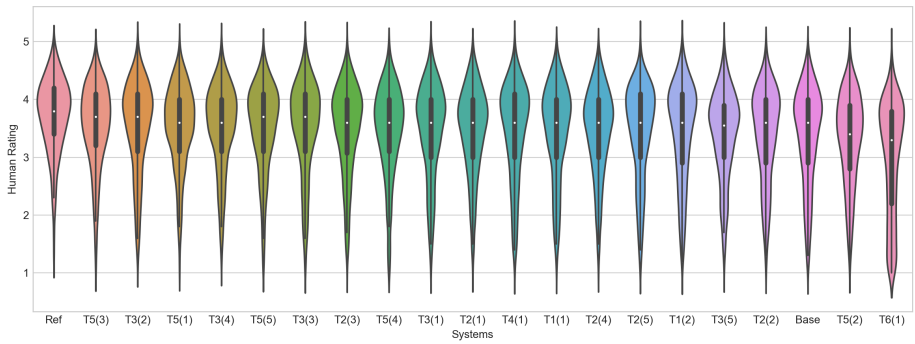

Figure 4: Distribution of human scores averaged sentence by sentence for Track 2.

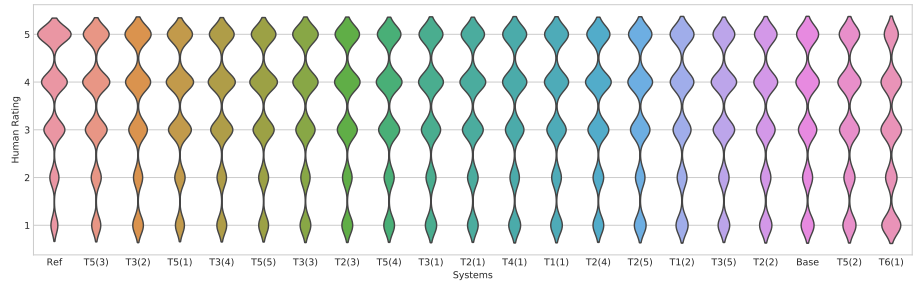

Figure 5: Distribution of human rating score for each level of scores for Track2.

Table 10: Example of System output for Track 2.

Example 1: The worst system does not show sympathy to the user's troubles.

Context		U: it 's currently raining on me inside of a bus @njtransit
Best	4.0	S: we 're sorry to hear this . please submit a report here so we may further assist you . <URL>
Worst	1.0	S: good luck !
Reference	3.9	S: hi <USER> , would you please provide me with your origin , destination , bus line , 4 digit vehicle # & time of travel ?

Example 2: The best system shows much deeper sympathy to show how they will solve the user's troubles than the reference. The worst system totally misunderstands the negative context for the positive one.

Context		U: flight canceled . stuck . rebooked 8hrs later in a middle seat and then asked to pay more to change it back to originally booked aisle
Best	4.5	S: sorry to hear about this , <USER> . could you dm us your confirmation code so we can take a closer look into this for you ? <URL>
Worst	1.2	S: we 're happy to hear that , <USER> . we hope to see you again soon !
Reference	3.3	S: sorry to hear about your flight woes . did you know you may be eligible for compensation ? visit <URL>

Example 3: The best system shows much deeper sympathy using emotional expressions for the user's troubles than the reference. The worst system totally misunderstands the negative context for the positive one.

Context		U: savannah was the worst stay over the weekend !
Best	4.8	S: oh no ! what happened ? we 'd like to look into this for you . please send us a direct message with your reservation number .
Worst	1.5	S: that 's what we like to hear , <USER> ! we 'll be sure to pass this on to the team . have a great day !
Reference	4.0	S: we are sorry to hear of your experience . please email us at <E-MAIL> so we can assist you w / this matter .

had some improvement compared to the other systems. The automatic generation have not yet reached the score of the reference (3.7245), but some of them were better responses than the reference in terms of human rating. The conversation modeling techniques applied to this challenge track for Twitter dialogues are better than we expected.

Figures 3-5 show the human ratings for each system in several ways. The systems are shown in the same order on the X axis for all three figures. Figure 3 shows the mean and the standard deviation of the human ratings for each system (across all responses and all raters for that system). Figure 4 shows the distributions of the mean human rating score for each sentence for each system. Figure 5 shows the distribution of all human rating scores for each system across all sentences. In this Figure, the area for each score of the violinplot shows a count of the number of scores of each level on the Likert scale. The "Reference" system (at the far left of each figure)

Table 11: Example of System output for Track 2.

Example 4: It's difficult to compare the best response sentences assuming the information from a back-end system with the reference waiting for a response from a database.

Context		U: when are the kaws tee shirts gonna restock again online ?
Best	4.7	S: hi <USER> , we don 't have a restock date at this time . please keep an eye on our social media pages for updates .
Worst	1.3	S: it does not .
Reference	3.7	S: we are checking on the restock date for the kaws tee 's rn .

Example 5: The benefits for users and company agents who wants to get more information are different.

Context		U: honestly can 't wait to use my lush bath bomb ♡
Best	4.4	S: hi <USER> , we 're so pleased to hear that you 're happy with your purchase . enjoy your new goodies !
Worst	1.4	S: we have great post-workout meals ! give us a try next ! get 40 % off : <URL> <URL>
Reference	3.6	S: nice which bath bomb ? :D

Example 6: More options are good for users.

Context		U: is queen sold out ? you website seems unable to tell me if it is . desperate to know ... S: hi <USER> , limited tickets still available . thanks U: thanks for the response . so , how does one procure them ?
Best	4.4	S: hi , you can order them from our website <URL> or by calling <NUMBERS> . thanks ,
Worst	1.1	S: hi <USER> ,
Reference	3.9	S: you can follow this link to purchase > > > <URL>

is ratings for the sentences extracted from the original (Twitter) dialogue data. The Reference system had the best human ratings: it had the highest mean rating in Fig. 3, the highest median sentence rating in Fig. 4 and the most sentences rated as level 5 ("Very good") in Fig. 5. The worst system (at the right) had a much lower mean rating, and a long tail of poorly rated sentences.

78.5% (1570/2000) of the system responses automatically generated by the best system were rated better than acceptable and 88.3% (1765/2000) of human's responses were rated in the same class. The best system achieved 89% of the upper-bound given by he human responses.

Tables 10 and 11 show examples of the system outputs. The humans prefer more supportive responses with sympathy. The system responses are sometimes rated better than the original human responses.

3.7. Summary

This article described the end-to-end conversation modeling track of the 6th dialog system technology challenges (DSTC6). We received 19 system outputs from six teams, and evaluated them based on several objective measures and a human-rating-based subjective measure. The final results are summarized in the article. Objective and subjective measures were not always consistent across systems. Some mismatches between objective and subjective measures have also been reported in [22]. Deeper analysis will be necessary on these results to enable us to design a better objective function and perform practical automatic evaluation.

4. Track 3: Dialogue Breakdown Detection

4.1. Introduction

Although voice agent services and smart speakers are beginning to appear on the market, the limited capabilities of these systems mean that humans and machines still cannot converse as naturally as two humans. The main problem is that systems typically make inappropriate utterances that lead to dialogue breakdowns. By dialogue breakdown, we mean a situation in a dialogue where users cannot proceed with the conversation [29]. To avoid this situation, technology for dialogue breakdown detection is essential because such technology will enable systems to avoid the creation of inappropriate utterances and also to identify dialogue breakdowns when they occur and perform the necessary recovery procedures.

The task of dialogue breakdown detection [13] is to detect whether a system utterance causes a dialogue breakdown in a given dialogue context. The participants of the dialogue breakdown detection track (Track 3) of the Dialog System Technology Challenges (DSTC) developed a dialogue breakdown detector that outputs a dialogue breakdown label (B: breakdown, PB: possible breakdown, or NB: not a breakdown) and a distribution of these labels. The definitions of the labels are defined as follows.

NB: It is easy to continue the conversation.

PB: It is difficult to continue the conversation smoothly.

B: It is difficult to continue the conversation.

Similar tasks to detect problematic situations in dialogue have been tackled mainly in task-oriented dialogue systems [45, 21, 25, 30]. In these studies, features, such as speech recognition results, language understanding results, and prosodic information, are extracted from user/system utterances and used to train a model that can detect problematic situations (also called miscommunication or hotspot). On the contrary, there are only a few studies that work on chat-oriented dialogue systems [49, 14], and their reported accuracies are not that high (about 60% accuracy). In addition, these studies use a dataset of a single dialogue system, which raises the generality issue. This track uses the data of multiple dialogue systems in multiple languages in order to obtain more general conclusions. Recently, a study that annotates chat-oriented dialogue systems with three kinds of labels (invalid, acceptable, valid), has been underway in the WOCHAT initiative[5]; however,

[5] `http://workshop.colips.org/wochat/`

the labels are currently used for annotation purposes and not for automatic detection [6, 7].

Dialogue breakdown detection is closely related to user satisfaction estimation in dialogue [44, 31, 15, 36, 1]. The difference is that these approaches aim to estimate user satisfaction after the dialogue so that the system can be improved later by analyzing unsatisfactory dialogues, whereas dialogue breakdown detection aims to detect a system's possible error in real-time so that the system can avoid dialogue breakdown. Since underlying techniques can be similar, we believe methods for user satisfaction estimation can also be useful for dialogue breakdown detection.

In this track, we prepared datasets in two languages: English and Japanese. We used four systems in English (NB. one of the datasets comprises dialogues of several dialogue systems) and three systems in Japanese. Overall, seven teams participated in this track, in which five teams worked on English[6] and four teams on Japanese with two teams working on both. In this paper, we describe the task setting, datasets, and evaluation metrics as well as the results of the submitted runs of the participants.

4.2. Task setting

The task of dialogue breakdown detection is to detect whether a system utterance causes a dialogue to break down in a given dialogue context [13]. For example, consider the following piece of dialogue.

S1: It's nice to go shopping alone.
U1: I agree. That's nice.
S2: Shopping takes time.
U2: Window shopping is also fun.
S3: It's fun to go shopping with somebody.

Here, system utterances are prefixed with S and user utterances with U. The dialogue context is from S1 to U2, and the target utterance for dialogue breakdown detection is S3 (underlined). In this example, S3 is *likely* to be causing a dialogue breakdown because S3 contradicts S1. Therefore, a detector that classifies this as a dialogue breakdown will be regarded as accurate. The reason we say "likely" is that a human's decision on dialogue breakdown is highly subjective, and it is difficult to decide on a single gold

[6] Although six teams submitted their runs in English, one team did not submit a technical paper; therefore, their run is not regarded as an official run and thereby is not included in this paper.

label. For this reason, we use many annotators for dialogue breakdown annotation and opt for majority voting and their probability distribution as references.

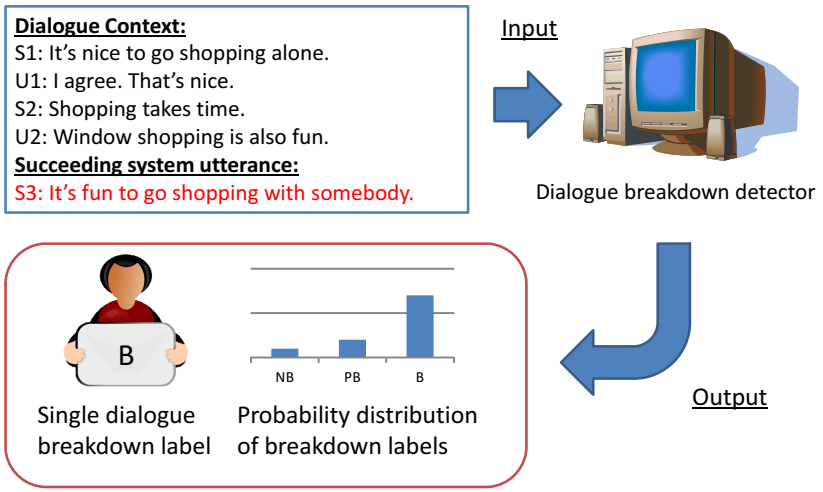

Figure 6: Illustration of task setting for Track 3.

Given pairs of dialogue context and a succeeding system utterance, the participants submit, for each pair, (1) a single dialogue breakdown label and (2) the probability distribution of breakdown labels (see Fig. 6). Note that, although some utterances may exist after the target utterance, they cannot be used for prediction because, for this track, we focus on avoiding dialogue breakdown rather than recovery. Currently many dialogue systems hold multiple utterance candidates for utterance generation; for example, retrieval-based methods typically have top-N candidates retrieved from a database, and generation-based methods have distributions over utterances which can be considered as multiple utterance candidates. We believe, good dialogue breakdown detection technology will make it possible to select appropriate system utterances from the candidates, which will realize smooth conversation with users. We are not making little of recovery, but, we consider that it is more important for the system not to cause severe problems in dialogue (i.e., dialogue breakdown) as a first step.

In this track, each participant can submit up to three runs for each language, so several parameters for dialogue breakdown detection can be tested.

Table 12: Statistics of English datasets for Track 3

	Development data				Evaluation data			
	TKTK-100	IRIS-100	CIC-115	YI-100	TKTK-50	IRIS-50	CIC-50	YI-50
No. of sessions	100	100	115	100	50	50	50	50
No. of annotators	30	30	30	30	30	30	30	30
NB	35.1%	32.9%	28.9%	34.8%	44.3%	34.5%	29.1%	35.4%
PB	27.6%	27.8%	29.8%	36.1%	29.2%	29.3%	39.3%	40.3%
B (Breakdown)	37.3%	39.4%	41.3%	29.1%	26.5%	36.2%	31.6%	24.3%
Fleiss' κ (NB, PB, B)	0.14	0.11	0.054	0.011	0.13	0.090	0.0040	−0.0060
Fleiss' κ (NB, PB+B)	0.21	0.15	0.084	0.020	0.19	0.13	0.0072	−0.0043

Table 13: Statistics of Japanese datasets for Track 3

	Chat dialogue corpus		DBDC1	DBDC2 (DVL/EVL)			DBDC3 (EVL)		
	init100	rest1046	DVL/EVL	DCM	DIT	IRS	DCM	DIT	IRS
No. of sessions	100	1046	20/80	50/50	50/50	50/50	50	50	50
No. of annotators	24	2 or 3	30	30	30	30	30	30	30
NB	59.2%	58.3%	37.1%	39.8%	33.0%	37.4%	34.9%	25.3%	29.3%
PB	22.2%	25.3%	32.2%	30.2%	27.4%	24.3%	34.2%	28.3%	23.8%
B	18.6%	16.4%	30.6%	29.9%	39.5%	38.3%	30.9%	46.4%	46.9%
Fleiss' κ (NB, PB, B)	0.28	0.28	0.20	0.31	0.24	0.36	0.24	0.14	0.27
Fleiss' κ (NB, PB+B)	0.40	0.40	0.27	0.44	0.38	0.48	0.32	0.20	0.37

4.3. Datasets

We distributed two sets of data to participants: one consisting of development (training) data and the other of evaluation (test) data. Tables 12 and 13 show the statistics of the datasets for English and Japanese, respectively [7]. In this section, we first describe the English datasets then the Japanese ones. The datasets are publicly available at `https://dbd-challenge.github.io/dbdc3/data/` .

4.3.1. Datasets for English for Track3

Development data. We provided four datasets: TKTK-100, IRIS-100, CIC-115, and YI-100. The dialogue data for TKTK-100 and IRIS-100 were taken from the WOCHAT TickTock and IRIS datasets[8]. The source of CIC-115 is the human evaluation round of the Conversational Intelligence Challenge

[7]We thank Rafael E. Banchs, Zhou Yu, Valentin Malykh, Idris Yusupov, and Yury Kuratov for graciously providing us with datasets and chatbots to make this track possible. We also thank our sponsors, Denso IT Laboratories, Nextremer, Honda Research Institute Japan, and NTT DOCOMO, Inc., for supporting our data collection. We also thank the Japanese Society for Artificial Intelligence (JSAI) for supporting the event.

[8]`http://workshop.colips.org/wochat/data/index.html`

(CIC)[9]. For YI-100, we newly collected dialogue sessions by crowd-sourcing. All dialogue sessions were 20 or 21 utterances long and included 10 system responses. The four datasets are described below.

TKTK-100 We selected 100 out of 206 sessions in the original WOCHAT dataset. TickTock sessions start from user utterances (see [51] for the details of TickTock). Dialogue breakdown annotation was done using a crowd-sourcing service, CrowdFlower[10]. For each utterance, 30 workers annotated one of the dialogue breakdown labels. Level-2 workers[11] from Australia, Canada, New Zealand, UK, and USA were recruited, and we requested non-native English speakers to refrain from participating in our annotation tasks.

IRIS-100 We selected 100 out of 163 sessions in the original WOCHAT dataset. The dataset was processed in the same manner as TKTK-100. Original IRIS sessions start from system utterances; however, we cut the first system utterances to make the data format identical to that of TKTK-100 for annotation purposes (see [3] for the details of IRIS).

CIC-115 This dataset comes from the human evaluation round of the CIC and DeepHack Turing school-hackathon[12]. The dialogues are available at the CIC site[13]. From all 2,778 dialogue sessions, we selected 115 dialogues performed between a human and a bot. Within the 115 dialogues, 85 dialogues start with a system utterance, and 30 dialogues start with a user utterance. As per the convention of CIC, each dialogue comes with a short paragraph, which is used as the context of the dialogue. The paragraphs are from the SQuAD dataset[14]. Dialogue breakdown annotation was done using a crowd-sourcing service, Amazon Mechanical Turk (AMT)[15], with 30 workers. When recruiting the workers, we specified that the task requires native English skills for the task instructions. The workers saw a short paragraph from the SQuAD before dialogue breakdown annotation.

[9] `http://convai.io`

[10] `https://www.crowdflower.com/`

[11] Higher quality: smaller group of more experienced, higher accuracy contributors based on the definition of CrowdFlower.

[12] `http://turing.tilda.ws`

[13] `http://convai.io/data/`

[14] `https://rajpurkar.github.io/SQuAD-explorer/`

[15] `https://requester.mturk.com`

YI-100 We collected 100 dialogue sessions by using a chatbot developed at the Moscow Institute of Physics and Technology[16] by using AMT. A worker was assigned to have a chat with the system that was more than 10 utterances. Dialogue breakdown annotation was also done using AMT with 30 workers. YI sessions start from system utterances.

Evaluation data. In the formal run, we distributed the following evaluation data.

TKTK-50 We collected 50 dialogue sessions of TickTock by using AMT in the same way as YI-100. Dialogue breakdown annotation was done using CrowdFlower.

IRIS-50 We selected 50 dialogue sessions from the held-out IRIS data graciously provided by the IRIS team. Dialogue breakdown annotation was done using CrowdFlower.

CIC-50 The data source of CIC-50 is held-out dialogue data collected by CIC after the human evaluation round. Dialogue breakdown annotation was done using AMT.

YI-50 We collected 50 dialogue sessions of YI in the same way as YI-100. Dialogue breakdown annotation was done using AMT.

4.3.2. Datasets for Japanese

For the Japanese datasets, we did not create new development data because we had already created several datasets in previous evaluation workshops (two series of dialogue breakdown detection challenges (DBDCs) held in Japan; DBDC1 and DBDC2). We briefly describe the development data and the newly created evaluation data.

Development data.

Chat dialogue corpus This dataset has 1,146 dialogue sessions. The dialogues were collected using NTT DOCOMO's chat API (DCM) [32]. One hundred dialogues (called init100) were annotated by 24 annotators, and the rest of the dialogues (called rest1046) were annotated by 2-3 annotators. Dialogue breakdown annotation was done by the researchers working on chat-oriented dialogue systems in Japan.

[16] `https://www.slideshare.net/sld7700/skillbased-conversational-agent-80976302`

Development data for DBDC1 This dataset has 20 dialogue sessions. The dialogues were collected using DCM via a crowd-sourcing service, CrowdWorks[17], and were annotated by 30 annotators by using another crowd-sourcing service, Yahoo! Crowd-sourcing[18]. All datasets in DBDC1 and DBDC2 were collected and annotated in the same way.

Evaluation data for DBDC1 This dataset contains 80 dialogue sessions. The dialogues were collected using DCM.

Development data for DBDC2 This dataset has 150 dialogue sessions. The dialogues were collected using DCM, DIT (Denso IT Laboratories' system) [42], and IRS (IR-status-based system from [34]) systems.

Evaluation data for DBDC2 This dataset has 150 dialogue sessions, 50 dialogues each were collected from DCM, DIT, and IRS.

Evaluation data. The evaluation data for Japanese contained 150 dialogue sessions. We used the same procedure we used to create the evaluation data for DBDC2.

4.4. Evaluation metrics

For this track, we used two types of evaluation metrics: classification-related and distribution-related.

4.4.1. Classification-related metrics

Classification-related metrics were used to evaluate the accuracy in classifying breakdown labels. The accuracy is calculated by comparing the output of the detector and the gold label determined by majority voting. We use a threshold t to obtain the gold label, that is, we first find the majority label and check if the ratio of that label is above t. If so, the gold label becomes that label and NB otherwise. We used the following metrics.

- Accuracy: the number of correctly classified labels divided by the total number of labels to be classified.
- Precision, Recall, F-measure (B): the precision, recall, and F-measure for the classification of B labels.

[17] `http://crowdworks.jp`
[18] `http://crowdsourcing.yahoo.co.jp`

- Precision, Recall, F-measure (PB+B): The precision, recall, and F-measure for the classification of PB + B labels; that is, PB and B labels are treated as a single label.

These metrics can provide intuitive results about the detection of dialogue breakdowns because they are used to directly evaluate whether dialogue breakdowns are correctly classified. However, the choice of an appropriate t remains an open issue. In this track, we used $t = 0.0$, which means simple majority voting.

4.4.2. Distribution-related metrics

Distribution-related metrics were used to evaluate the similarity of the distribution of breakdown labels, which is calculated by comparing the predicted distribution of the labels with that of the gold labels. We calculate these values for each utterance and use the mean values for evaluation. We used the following metrics.

- Jensen-Shannon Divergence (JSD) (NB,PB,B): distance between the predicted distribution of the three labels and that of the gold labels calculated by using JSD.
- JSD (NB,PB+B): JSD when PB and B are regarded as a single label.
- JSD (NB+PB,B): JSD when NB and PB are regarded as a single label.
- Mean Squared Error (MSE) (NB,PB,B): distance between the predicted distribution of the three labels and that of the gold labels calculated by using MSE.
- MSE (NB,PB+B): MSE when PB and B are regarded as a single label.
- MSE (NB+PB,B): MSE when NB and PB are regarded as a single label.

These metrics are used to compare the distributions of the labels; thus, enabling a direct comparison with the gold labels. However, the results may not be as easily interpretable as the classification-related metrics because they do not directly translate to detection performance.

Table 14: Submitted runs in English summarized by their key features. Bold font indicates the best result. An underline indicates the second best result. MemN2N and ETR denote End-to-End Memory Network and Extra Trees Regressor, respectively.

Run	Model	Word/Sentence embedding	Bag of words	Utterance similarity	Turn index	Acc	JSD
KTH run1 [24]	SVM			✓		0.3375	0.4445
KTH run2	LSTM	✓				**0.4415**	0.0481
KTH run3	LSTM	✓	✓			0.4220	0.3268
PLECO run1 [35]	MemN2N	✓				0.2950	0.0714
PLECO run2	MemN2N	✓				0.2900	0.0774
RSL17BD run1 [19]	ETR	✓		✓	✓	0.4265	0.0432
RSL17BD run2	ETR	✓		✓	✓	<u>0.4310</u>	0.0412
RSL17BD run3	ETR	✓		✓	✓	0.4200	0.0426
NCDS run1 [33]	RNN	✓				0.3605	<u>0.0412</u>
NCDS run2	RNN	✓				0.3655	0.0412
NCDS run3	RNN	✓		✓		0.3565	0.0668
SWPD run1 [50]	Bi-LSTM	✓				0.4295	0.0807
CRF Baseline	CRF		✓			0.4285	0.4409
Majority Baseline						0.3720	**0.0393**

Table 15: Submitted runs in Japanese summarized by their key features for Track 3. Bold font indicates the best result. An underline indicates the second best result. EoR denotes Ensemble of Regressors.

Run	Model	Word/Sentence embedding	Bag of words	Utterance similarity	Turn index	Acc	JSD
PLECO run1 [35]	MemN2N	✓				0.5078	0.1149
PLECO run2	MemN2N	✓				0.5012	0.1011
PLECO run3	MemN2N	✓				0.5345	0.0981
RSL17BD run1 [19]	ETR	✓		✓	✓	0.3890	0.1564
RSL17BD run2	ETR	✓		✓	✓	0.3939	0.1564
RSL17BD run3	ETR	✓		✓	✓	0.4024	0.1582
OUARS run1 [41]	CNN	✓				0.5539	0.0910
OUARS run2	CNN, LSTM	✓				0.5430	0.0989
OUARS run3	CNN, LSTM	✓				0.5593	0.0932
NTTCS run1 [38]	EoR	✓		✓	✓	**0.6036**	**0.0714**
NTTCS run2	EoR	✓		✓	✓	<u>0.5993</u>	<u>0.0717</u>
NTTCS run3	EoR	✓		✓	✓	0.5927	0.0741
CRF Baseline	CRF		✓			0.5296	0.3871
Majority Baseline						0.4721	0.1311

4.5. Results

Overall, seven teams participated in this track: five teams worked on English, four teams on Japanese, and two teams working on both. Each team could submit up to three runs for each language. We had 12 runs for English and 12 for Japanese.

Table 16: Overall Results of Classification (English) for Track 3

Run	Accuracy	Run	F1 (B)	Run	F1 (PB+B)
KTH run2	0.4415	PLECO run1	0.3636	Majority Baseline	0.8927
RSL17BD run2	0.4310	PLECO run2	0.3565	PLECO run1	0.8744
SWPD run1	0.4295	CRF Baseline	0.3543	PLECO run2	0.8708
CRF Baseline	0.4285	KTH run1	0.3487	KTH run1	0.8423
RSL17BD run1	0.4265	KTH run3	0.3373	RSL17BD run2	0.8400
KTH run3	0.4220	Majority Baseline	0.3343	RSL17BD run3	0.8357
RSL17BD run3	0.4200	SWPD run1	0.3210	RSL17BD run1	0.8329
Majority Baseline	0.3720	RSL17BD run2	0.3201	NCDS run3	0.8046
NCDS run2	0.3655	NCDS run3	0.3198	SWPD run1	0.7627
NCDS run1	0.3605	RSL17BD run1	0.3126	CRF Baseline	0.7622
NCDS run3	0.3565	RSL17BD run3	0.3025	KTH run3	0.7592
KTH run1	0.3375	KTH run2	0.2949	KTH run2	0.7440
PLECO run1	0.2950	NCDS run2	0.2097	NCDS run1	0.3458
PLECO run2	0.2900	NCDS run1	0.2076	NCDS run2	0.3397

Table 17: Overall Results of Jensen-Shannon Divergence (JSD) (English) for Track 3

Run	JSD (NB,PB,B)	Run	JSD (NB,PB+B)	Run	JSD (NB+PB,B)
Majority Baseline	0.0393	Majority Baseline	0.0237	RSL17BD run2	0.0225
NCDS run1	0.0412	NCDS run1	0.0248	RSL17BD run3	0.0243
RSL17BD run2	0.0412	NCDS run2	0.0248	RSL17BD run1	0.0247
NCDS run2	0.0412	RSL17BD run2	0.0256	NCDS run2	0.0254
RSL17BD run3	0.0426	RSL17BD run3	0.0258	NCDS run1	0.0254
RSL17BD run1	0.0432	RSL17BD run1	0.0263	Majority Baseline	0.0257
KTH run2	0.0481	KTH run2	0.0267	KTH run2	0.0262
NCDS run3	0.0668	PLECO run1	0.0427	SWPD run1	0.0444
PLECO run1	0.0714	NCDS run3	0.0436	NCDS run3	0.0488
PLECO run2	0.0774	SWPD run1	0.0438	PLECO run1	0.0535
SWPD run1	0.0807	PLECO run2	0.0482	PLECO run2	0.0565
KTH run3	0.3268	KTH run3	0.1892	KTH run1	0.2058
CRF Baseline	0.4409	KTH run1	0.2343	KTH run3	0.2166
KTH run1	0.4445	CRF Baseline	0.2687	CRF Baseline	0.2985

Tables 14 and 15 summarize the submitted runs of the participants in English and Japanese, respectively, together with their accuracy and JSD for reference. The tables indicate that many teams used neural networks (NNs) and word embeddings.

Tables 16, 17, and 18 show the results for English and Tables 19, 20, and 21 for Japanese. The values in these tables are macro-averages over the systems for each language.[19]

We also show the results of two baselines. One is a majority baseline that

[19]See [12] for the detailed results for each dataset in each language. Note that the results of the Japanese runs are slightly different from [12] because, due to a technical trouble, several dialogues were not processed in our evaluation script. This problem was fixed and the tables here show the updated results; there was no change in the order of performance.

Table 18: Overall Results of Mean Squared Error (MSE) (English) for Track 3

Run	MSE (NB,PB,B)	Run	MSE (NB,PB+B)	MSE	MSE (NB+PB,B)
Majority Baseline	0.0224	Majority Baseline	0.0278	RSL17BD run2	0.0246
NCDS run1	0.0237	NCDS run1	0.0287	Majority Baseline	0.0264
NCDS run2	0.0237	NCDS run2	0.0288	NCDS run2	0.0270
RSL17BD run2	0.0241	RSL17BD run2	0.0301	NCDS run1	0.0270
RSL17BD run3	0.0250	RSL17BD run3	0.0301	RSL17BD run3	0.0271
RSL17BD run1	0.0254	RSL17BD run1	0.0307	RSL17BD run1	0.0275
KTH run2	0.0281	KTH run2	0.0315	KTH run2	0.0286
PLECO run1	0.0415	PLECO run1	0.0455	SWPD run1	0.0497
NCDS run3	0.0437	SWPD run1	0.0501	NCDS run3	0.0572
PLECO run2	0.0448	PLECO run2	0.0509	PLECO run1	0.0632
SWPD run1	0.0471	NCDS run3	0.0677	PLECO run2	0.0673
KTH run3	0.1670	KTH run3	0.1664	KTH run1	0.1476
CRF Baseline	0.2185	KTH run1	0.1752	KTH run3	0.2044
KTH run1	0.2240	CRF Baseline	0.2171	CRF Baseline	0.2578

Table 19: Overall Results of Classification (Japanese) for Track 3

Run	Accuracy	Run	F1 (B)	Run	F1 (PB+B)
NTTCS run1	0.6036	NTTCS run1	0.6645	RSL17BD run2	0.8254
NTTCS run2	0.5993	NTTCS run2	0.6612	OUARS run3	0.8195
NTTCS run3	0.5927	NTTCS run3	0.6569	RSL17BD run1	0.8152
OUARS run3	0.5593	OUARS run3	0.6294	PLECO run2	0.8117
OUARS run1	0.5539	OUARS run1	0.6246	PLECO run1	0.8094
OUARS run2	0.5430	OUARS run2	0.6155	RSL17BD run3	0.8066
PLECO run3	0.5345	PLECO run3	0.6143	OUARS run2	0.8039
CRF Baseline	0.5296	PLECO run2	0.6027	OUARS run1	0.7989
PLECO run1	0.5078	PLECO run1	0.6003	NTTCS run3	0.7982
PLECO run2	0.5012	CRF Baseline	0.5777	PLECO run3	0.7941
Majority Baseline	0.4721	Majority Baseline	0.4448	NTTCS run1	0.7836
RSL17BD run3	0.4024	RSL17BD run3	0.2822	NTTCS run2	0.7829
RSL17BD run2	0.3939	RSL17BD run2	0.2762	CRF Baseline	0.7710
RSL17BD run1	0.3890	RSL17BD run1	0.2598	Majority Baseline	0.5549

outputs the most frequent dialogue breakdown label in each system's development data for English and development and evaluation data of DBDC2 for Japanese with averaged probability distributions. The other is a conditional random field (CRF)-based baseline that labels utterance sequences with the three breakdown labels by using CRFs. The features used were words in a target utterance and the previous utterances. For the probability distribution, the probability of 1.0 is given to the label determined by the CRFs.

It can be seen from the tables that, for English, NCDS and RSL17BD performed well in terms of JSD and MSE. In terms of classification-related metrics, KTH, RSL17BD, and PLECO seem to have performed well. For Japanese, except for F1(PB+B), NTTCS outperformed the other teams

Table 20: Overall Results of Jensen-Shannon Divergence (JSD) (Japanese) for Track 3

Run	JSD (NB,PB,B)	Run	JSD (NB,PB+B)	Run	JSD (NB+PB,B)
NTTCS run1	0.0714	NTTCS run1	0.0490	NTTCS run1	0.0428
NTTCS run2	0.0717	NTTCS run2	0.0492	NTTCS run2	0.0429
NTTCS run3	0.0741	NTTCS run3	0.0511	NTTCS run3	0.0447
OUARS run1	0.0910	OUARS run1	0.0635	OUARS run1	0.0555
OUARS run3	0.0932	OUARS run3	0.0657	OUARS run3	0.0569
PLECO run3	0.0981	PLECO run3	0.0701	OUARS run2	0.0605
OUARS run2	0.0989	OUARS run2	0.0704	PLECO run3	0.0606
PLECO run2	0.1011	PLECO run2	0.0722	PLECO run2	0.0627
PLECO run1	0.1149	RSL17BD run2	0.0749	PLECO run1	0.0739
Majority Baseline	0.1311	RSL17BD run3	0.0785	Majority Baseline	0.0752
RSL17BD run2	0.1564	RSL17BD run1	0.0786	RSL17BD run1	0.0965
RSL17BD run1	0.1564	PLECO run1	0.0833	RSL17BD run2	0.0971
RSL17BD run3	0.1582	Majority Baseline	0.1068	RSL17BD run3	0.0982
CRF Baseline	0.3871	CRF Baseline	0.2409	CRF Baseline	0.2798

Table 21: Overall Results of Mean Squared Error (MSE) (Japanese) for Track 3

Run	MSE (NB,PB,B)	Run	MSE (NB,PB+B)	Run	MSE (NB+PB,B)
NTTCS run1	0.0385	NTTCS run1	0.0482	NTTCS run1	0.0473
NTTCS run2	0.0386	NTTCS run2	0.0484	NTTCS run2	0.0474
NTTCS run3	0.0400	NTTCS run3	0.0508	NTTCS run3	0.0490
OUARS run1	0.0475	OUARS run1	0.0604	OUARS run1	0.0584
OUARS run3	0.0480	OUARS run3	0.0613	OUARS run3	0.0588
OUARS run2	0.0510	OUARS run2	0.0659	OUARS run2	0.0624
PLECO run3	0.0529	PLECO run3	0.0693	PLECO run3	0.0660
PLECO run2	0.0558	PLECO run2	0.0744	PLECO run2	0.0690
PLECO run1	0.0644	RSL17BD run2	0.0775	Majority Baseline	0.0750
Majority Baseline	0.0682	RSL17BD run1	0.0815	PLECO run1	0.0828
RSL17BD run1	0.0896	RSL17BD run3	0.0820	RSL17BD run1	0.1032
RSL17BD run2	0.0897	PLECO run1	0.0861	RSL17BD run2	0.1041
RSL17BD run3	0.0907	Majority Baseline	0.1023	RSL17BD run3	0.1046
CRF Baseline	0.2013	CRF Baseline	0.2117	CRF Baseline	0.2477

followed by OUARS.

We conducted a multiple comparison test (Steel-Dwass test) to examine whether the submitted runs were better than the baselines. For English, when we focused on accuracy, KTH run2 (NN-based method) significantly outperformed the majority baseline with $p < 0.01$, followed by RSL17BD run1 RSL17BD run2, SWPD run1 with $p < 0.05$. No runs outperformed the CRF baseline. For F1(B) and F1(PB+B), no runs significantly outperformed the best-performing baseline, that is, the CRF-based baseline for F1(B) and the majority baseline for F1(PB+B). With regards to the distribution-related metrics, only RSL17BD run2 significantly outperformed the majority baseline ($p < 0.05$) for MSE(NB+PB, B). For Japanese, only NTTCS run1 and run2 (methods based on the Ensemble of Regres-

sors (EoR)) significantly outperformed the CRF-based baseline ($p < 0.01$). For F1(B) and F1(PB+B), no runs significantly outperformed the best-performing baseline. With regards to the distribution-related metrics, most runs outperformed both the CRF-based and majority baselines. Overall, we had better results for Japanese compared to those for English, probably because of the low inter-annotator agreement in the English data.

We calculated the upper bound (see Table 22). For accuracy, we left-out 1 annotator from the 30 annotators and calculated whether that left-out annotator can predict the majority label by the other 29 annotators. For JSD and MSE, we split the 30 annotators into two groups and calculated the JSD and MSE between those groups; this process was iterated 100 times to obtain the average. The values were calculated using the datasets of this track.

Table 22: Upper bound (human-level) accuracy, JSD, and MSE. The values are compared against the top run in each language for Track 3.

	English		Japanese	
	Upper bound	Top run	Upper bound	Top run
Accuracy	0.429	0.442	0.625	0.6036
JSD(NB,PB,B)	0.0330	0.0393	0.0289	0.0714
MSE(NB,PB,B)	0.0277	0.0224	0.0206	0.0385

In terms of accuracy, we are already at the upper bound in both languages (English: $0.442/0.429 = 1.03$, Japanese: $0.6036/0.625 = 0.97$). For JSD and MSE in English, since the majority baseline was the top run, it is difficult to discuss the results; we need to improve the quality of the data, such as inter-annotator agreement, so that the proposed method will at least outperform the baselines. For Japanese, there still seems to be some gap between the top run and upper bound.

This analysis indicates that, with current methods, we can detect dialogue breakdown with good accuracy, though there seems to be some room for improvement for JSD and MSE. In other words, it is currently possible to guess the majority decision but it is hard to predict the distribution of dialogue breakdown labels.

4.6. Summary and Future work

We described the dialogue breakdown detection track in the Sixth Dialog System Technology Challenge (DSTC6). We prepared both English and Japanese datasets, and seven teams competed using methods for detecting

dialogue breakdown. We obtained promising results and interesting methods for dialogue breakdown detection. The NN-based method performed the best in English and a method that used EoR was the best in Japanese. It seems that NN-based approaches are struggling for lack of training data, which is the general problem in dialogue processing with neural models. We achieved a human-level accuracy for both languages although there is still some room for improvement in JSD and MSE.

In order to improve JSD and MSE, a more fine-grained understanding of breakdowns/errors in chat-oriented dialogue systems will be important. There have been studies to create a taxonomy of errors in chat-oriented dialogue systems [11, 16]. We want to incorporate such studies so as to improve the prediction of distributions. Another interesting direction will be to focus on a particular type of error, such as contradiction and common sense violation, which may be rare in human-machine dialogue but is difficult to handle by current natural language processing techniques.

5. Conclusion

We reported the results of the sixth Dialog System Technology Challenges (DSTC6). 23 teams challenged with one or two of the 3 tracks aiming to select system responses for restaurant retrieval dialogues to fill slot-value in Track 1, generate system responses using NLG for customer service on twitter by combining goal-oriented dialogues and chitchat in Track 2 and human-machine dialogue data for ChitChat in Track 3. We find the blending end-to-end trainable models associated to meaningful prior knowledge performs the best for the restaurant retrieval in Track 1. Indeed, Hybrid Code Network and Memory Network have been the best models for this task. Regarding customer service response generation using NLG in Track 2, 78.5% of the system responses automatically generated by the best system were rated better than acceptable and this achieves 89% of the number of the human responses rated in the same class. The responses rated as poor and very poor don't show sufficient sympathy to user's troubles. The worst responses generated by systems represented the opposite sentiment and was never observed in the human customer responses. Future works contains customer sentiment understanding. In Track 3 for dialogue breakdown detection, the best system achieved a human-level accuracy for both languages, English and Japanese.

References

[1] Ando, A., Masumura, R., Kamiyama, H., Kobashikawa, S., Aono, Y., 2017. Hierarchical LSTMs with joint learning for estimating customer satisfaction from contact center calls. In: Proc. Interspeech. pp. 1716–1720.

[2] Bairong, Z., Wenbo, W., Zhiyu, L., Chonghui, Z., Shinozaki, T., 2017. Comparative analysis of word embedding methods for DSTC6 end-to-end conversation modeling track. In: Proceedings of the 6th Dialog System Technology Challenges (DSTC6) Workshop.

[3] Banchs, R. E., Li, H., 2012. IRIS: a chat-oriented dialogue system based on the vector space model. In: Proc. ACL 2012 System Demonstrations. pp. 37–42.

[4] Bordes, A., Boureau, Y.-L., Weston, J., 2017. earning end-to-end goal-oriented dialog. arXiv preprint arXiv:1605.07683.

[5] Bordes, A., Usunier, N., Chopra, S., Weston, J., 2015. Large-scale simple question answering with memory networks. arXiv preprint arXiv:1506.02075.

[6] Charras, F., Duplessis, G. D., Letard, V., Ligozat, A.-L., Rosset, S., 2016. Comparing system-response retrieval models for open-domain and casual conversational agent. In: Proc. Workshop on Chatbots and Conversational Agent Technologies.

[7] Curry, A. C., Rieser, V., 2016. A subjective evaluation of chatbot engines.

[8] Dodge, J., Gane, A., Zhang, X., Bordes, A., Chopra, S., Miller, A., Szlam, A., Weston, J., 2016. Evaluating prerequisite qualities for learning end-to-end dialog systems. In: Proc. of ICLR.

[9] Galley, M., Brockett, C., 2017. The MSR-NLP system at dialog system technology challenges 6. In: Proceedings of the 6th Dialog System Technology Challenges (DSTC6) Workshop.

[10] Ghazvininejad, M., Brockett, C., Chang, M.-W., Dolan, B., Gao, J., Yih, W.-t., Galley, M., 2017. A knowledge-grounded neural conversation model. arXiv preprint arXiv:1702.01932.

[11] Higashinaka, R., Funakoshi, K., Araki, M., Tsukahara, H., Kobayashi, Y., Mizukami, M., 2015. Towards taxonomy of errors in chat-oriented dialogue systems. In: Proc. SIGDIAL. pp. 87–95.

[12] Higashinaka, R., Funakoshi, K., Inaba, M., Tsunomori, Y., Takahashi, T., Kaji, N., 2017. Overview of dialogue breakdown detection challenge 3. In: Proc. Dialog System Technology Challenges Workshop (DSTC6).

[13] Higashinaka, R., Funakoshi, K., Kobayashi, Y., Inaba, M., 2016. The dialogue breakdown detection challenge: Task description, datasets, and evaluation metrics. In: Proc. LREC. pp. 3146–3150.

[14] Higashinaka, R., Meguro, T., Imamura, K., Sugiyama, H., Makino, T., Matsuo, Y., 2014. Evaluating coherence in open domain conversational systems. In: Proc. Interspeech. pp. 130–133.

[15] Higashinaka, R., Minami, Y., Dohsaka, K., Meguro, T., 2010. Modeling user satisfaction transitions in dialogues from overall ratings. In: Proc. SIGDIAL. pp. 18–27.

[16] Higashinaka, R., Mizukami, M., Funakoshi, K., Araki, M., Tsukahara, H., Kobayashi, Y., 2015. Fatal or not? Finding errors that lead to dialogue breakdowns in chat-oriented dialogue systems. In: Proc. EMNLP. pp. 2243–2248.

[17] Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural computation 9 (8), 1735–1780.

[18] Hori, T., 2017. DSTC6 end-to-end conversation modeling track: tools and baseline system. `https://github.com/dialogtekgeek/DSTC6-End-to-End-Conversation-Modeling`.

[19] Kato, S., Sakai, T., 2017. RSL17BD at DBDC3: Computing utterance similarities based on term frequency and word embedding vectors. In: Proc. Dialog System Technology Challenges Workshop (DSTC6).

[20] Kumar, A., Irsoy, O., Su, J., Bradbury, J., English, R., Pierce, B., Ondruska, P., Gulrajani, I., Socher, R., 2015. Ask me anything: Dynamic memory networks for natural language processing. CoRR abs/1506.07285.
URL `http://arxiv.org/abs/1506.07285`

[21] Lendvai, P., Van Den Bosch, A., Krahmer, E., Swerts, M., 2002. Improving machine-learned detection of miscommunications in human-machine dialogues through informed data splitting. In: Proc. the ESSLLI Workshop on Machine Learning Approaches in Computational Linguistics. pp. 1–15.

[22] Liu, C.-W., Lowe, R., Serban, I. V., Noseworthy, M., Charlin, L., Pineau, J., 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. arXiv preprint arXiv:1603.08023.

[23] Long, Y., Wang, J., Xu, Z., Wang, Z., Wang, B., 2017. A knowledge enhanced generative conversational service agent. In: Proceedings of the 6th Dialog System Technology Challenges (DSTC6) Workshop.

[24] Lopes, J., 2017. How generic can dialogue breakdown detection be? the KTH entry to DBDC3. In: Proc. Dialog System Technology Challenges Workshop (DSTC6).

[25] Lopes, J., Chorianopoulou, A., Palogiannidi, E., Moniz, H., Abad, A., Louka, K., Iosif, E., Potamianos, A., 2016. The SpeDial datasets: datasets for spoken dialogue systems analytics. In: Proc. LREC.

[26] Lowe, R., Pow, N., Serban, I., Pineau, J., 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In: SIGDIAL Conference. The Association for Computer Linguistics, pp. 285–294.
URL `http://aclweb.org/anthology/W/W15/W15-4640.pdf`

[27] Lowe, R., Serban, I. V., Noseworthy, M., Charlin, L., Pineau, J., 2016. On the evaluation of dialogue systems with next utterance classification. In: SIGDIAL Conference. The Association for Computer Linguistics, pp. 264–269.
URL `http://aclweb.org/anthology/W/W16/W16-3634.pdf`

[28] Lowe, R., Serban, I. V., Noseworthy, M., Charlin, L., Pineau, J., 2016. On the evaluation of dialogue systems with next utterance classification. arXiv preprint arXiv:1605.05414.

[29] Martinovsky, B., Traum, D., 2003. The error is the clue: Breakdown in human-machine interaction. In: Proc. ISCA Workshop on Error Handling in Spoken Dialogue Systems. pp. 11–16.

[30] Meena, R., Lopes, J., Skantze, G., Gustafson, J., 2015. Automatic detection of miscommunication in spoken dialogue systems. In: Proc. SIGDIAL. pp. 354–363.

[31] Möller, S., Engelbrecht, K.-P., Schleicher, R., 2008. Predicting the quality and usability of spoken dialogue services. Speech Communication 50 (8-9), 730–744.

[32] Onishi, K., Yoshimura, T., 2014. Casual conversation technology achieving natural dialog with computers. NTT DOCOMO Technical Jouranl 15 (4), 16–21.

[33] Park, C., Kim, K., Kim, S., 2017. Attention-based dialog embedding for dialog breakdown detection. In: Proc. Dialog System Technology Challenges Workshop (DSTC6).

[34] Ritter, A., Cherry, C., Dolan, W. B., 2011. Data-driven response generation in social media. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing.

[35] Saito, A., Iki, T., 2017. End-to-end character-level dialogue breakdown detection with external memory models. In: Proc. Dialog System Technology Challenges Workshop (DSTC6).

[36] Schmitt, A., Schatz, B., Minker, W., 2011. Modeling and predicting quality in spoken human-computer interaction. In: Proc. SIGDIAL. pp. 173–184.

[37] Sharma, S., El Asri, L., Schulz, H., Zumer, J., 2017. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. CoRR abs/1706.09799.
URL `http://arxiv.org/abs/1706.09799`

[38] Sugiyama, H., 2017. Dialogue breakdown detection based on estimating appropriateness of topic transition. In: Proc. Dialog System Technology Challenges Workshop (DSTC6).

[39] Sukhbaatar, S., Szlam, A., Weston, J., Fergus, R., 2015. End-to-end memory networks. Proceedings of NIPS.

[40] Sukhbaatar, S., Szlam, A., Weston, J., Fergus, R., 2015. Weakly supervised memory networks. CoRR abs/1503.08895.
URL `http://arxiv.org/abs/1503.08895`

[41] Takayama, J., Nomoto, E., Arase, Y., 2017. Dialogue breakdown detection considering annotation biases. In: Proc. Dialog System Technology Challenges Workshop (DSTC6).

[42] Tsukahara, H., Uchiumi, K., 2015. System utterance generation by label propagation over association graph of words and utterance patterns for open-domain dialogue systems. In: Proc. PACLIC. pp. 323–331.

[43] Vinyals, O., Le, Q., 2015. A neural conversational model. arXiv preprint arXiv:1506.05869.

[44] Walker, M., Kamm, C., Litman, D., 2000. Towards developing general models of usability with PARADISE. Natural Language Engineering 6 (3-4), 363–377.

[45] Walker, M., Langkilde, I., Wright, J., Gorin, A., Litman, D., 2000. Learning to predict problematic situations in a spoken dialogue system: Experiments with How May I Help You? In: Proc. NAACL. pp. 210–217.

[46] Wang, W., Koji, Y., Harsham, B., Hori, T., Hershey, J. R., 2017. Sequence adversarial training and minimum Bayes risk decoding for end-to-end neural conversation models. In: Proceedings of the 6th Dialog System Technology Challenges (DSTC6) Workshop.

[47] Wen, T.-H., Miao, Y., Blunsom, P., Young, S. J., 2017. Latent intention dialogue models. In: Precup, D., Teh, Y. W. (Eds.), Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017. Vol. 70 of Proceedings of Machine Learning Research. PMLR, pp. 3732–3741.
URL `http://jmlr.org/proceedings/papers/v70/`

[48] Williams, J. D., Asadi, K., Zweig, G., 2017. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. CoRR abs/1702.03274.
URL `http://arxiv.org/abs/1702.03274`

[49] Xiang, Y., Zhang, Y., Zhou, X., Wang, X., Qin, Y., 2014. Problematic situation analysis and automatic recognition for chinese online conversational system. In: Proc. CLP. pp. 43–51.

[50] Xie, Z., Ling, G., 2017. Dialogue breakdown detection using hierarchical bi-directional LSTMs. In: Proc. Dialog System Technology Challenges Workshop (DSTC6).

[51] Yu, Z., Xu, Z., Black, A. W., Rudnicky, A. I., 2016. Strategy and policy learning for non-task-oriented conversational systems. In: Proc. SIGDIAL. pp. 404–412.