Dialogue Breakdown Detection Considering Annotation Biases

Junya Takayama[1], *Eriko Nomoto*[2], *Yuki Arase*[1]

[1]Graduate school of information science and technology, Osaka University, Japan
[2]School of Engineering, Osaka University, Japan

takayama.junya@ist.osaka-u.ac.jp, nomoto.eriko@ist.osaka-u.ac.jp, arase@ist.osaka-u.ac.jp

Abstract

We propose a method for dialogue breakdown detection based on ensemble of detectors trained by different sets of annotators. Our approach clusters the training data by the annotation distribution of each annotator and constructs a detector for each cluster. A detector uses a neural network that encodes an utterance-reply pair to label it either *Breakdown*, *Not a breakdown*, or *Possible breakdown*. Specifically, a network consists of two parallel/series combined encoders, a hidden layer, and a softmax layer. Then the ensemble results of all detectors predict a breakdown. We submitted our models to the Dialogue Breakdown Detection Challenge 3 (DBDC3) Japanese task. The results confirm the effectiveness of considering biases among annotators.

Index Terms: ensemble learning, clustering, convolutional neural network, recurrent neural network

1. Introduction

As chat-oriented dialogue systems, which are known as chatbots, implemented by generation-based and example-based approaches gain popularity, output utterances that collapse the dialogue context become an issue. The Dialogue Breakdown Detection Challenge (DBDC) [1] is a shared task to detect inappropriate utterances, which cause breakdowns in user-system dialogue.

Numerous types of conventional methods for dialogue breakdown detection exist. Sugiyama [2], who achieved the best performance in DBDC2, analyzed the breakdown patterns of each data to design the features and built a detector using the Extra Trees Regressor [3]. Although a deep neural network (DNN) was not applied in this study, it was discussed as a promising approach if the amount of resources increases. Today, many conventional methods employ DNNs to build a detector. Inaba et al. [4] built a detector using a recurrent neural network (RNN). Their model encodes the current and past pairs of user and system utterances using an RNN encoder. Then the probability of breakdown is computed using a softmax classifier. Kubo et al. [5] use only the current pair of user and system utterances. They trained an encoder-decoder model by utterance-reply pairs of a user and a system, and input the last state of the decoder to a support vector machine for dialogue breakdown detection.

DBDC data consists of user-system dialogue, where each system's utterance is annotated with breakdown labels (O: Not a breakdown, T: Possible breakdown, X: Breakdown) by some annotators. Such annotation tasks are inevitability subjective. Hence, the distribution of annotated labels could be biased among annotators, *i.e.*, one annotator group is more sensitive while another one is more generous. Given this observation, we propose a method focusing on such differences between annotators distributions. First, the k-means clustering is employed to cluster annotators based on their distributions of annotation labels. Second, a dialogue breakdown detector is constructed for each cluster using the annotation labels. Finally, the breakdown probability is estimated using the averaged results for all detectors.

We propose three models as detectors employing DNNs: one uses two series Long Short-Term Memory (LSTM) encoders, another uses two parallel Convolutional Neural Network (CNN) encoders, and the other ensembles these two. We participated in the DBDC3 [6] Japanese task whose results show that our method achieves 63.6% F-measure outperforming the baseline detector by 5.6%. This confirms the effectiveness of our approach considering annotator biases.

2. Our Approach

Our method involves three steps to construct a detector. First, the training data is clustered by the label distributions of each annotator. Second, words of all sentences in the training data are converted into vectors using the CBoW in Word2Vec [7] method. Third, a dialogue breakdown detector is constructed for each cluster. This detector is then used to predict the dialogue breakdown probability ensembled with other detectors. In the following, we describe details of the first and the third steps.

2.1. Training data clustering

The DBDC training data was annotated by multiple annotators. Typical conventional studies [4, 5] use a probability distribution of annotated labels or a majority label for training. Herein we focus on the difference in the annotation distribution of each annotator to create a model that simulates the human voting. For that, we first group annotators having similar tendency for judgment by clustering.

All annotators are clustered by the k-means++ [8] method using the label distributions. Then user-system utterance pairs are distributed to each cluster and the average label of annotations is assigned to them, *i.e.*, the same pair may have different labels at different clusters. We use these user-system utterances assigned aggregated labels to build a detector.

2.2. Design of detectors

Detailed observation of DBDC training data let us assume that dialogue breakdown happens locally and/or contextually, *i.e.*, there are local (word or phrase) level breakdown and contextual (sentence or dialogue context) level breakdown. Thus, we construct two kinds of detectors. A detector employing Convolutional neural networks (CNN) aims to detect local level breakdown. Another detector employing RNN aims to detect contextual level breakdown. Both detectors receive two sequences of vector representations of words converted from a pair of user-

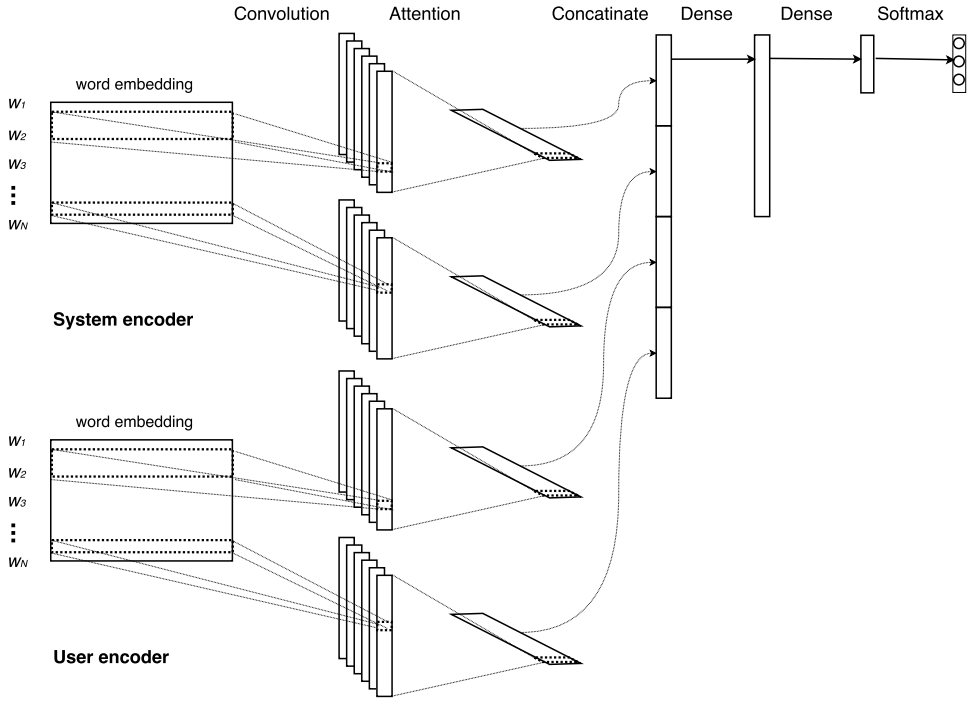

Figure 1: *Design of Parallel-CNN dialogue breakdown detector.*

system utterances as the input and output the probability of each breakdown label (O: Not a breakdown, T: Possible breakdown, X: Breakdown). Below, each model is described in detail.

2.2.1. Parallel-CNN detector

CNN can extract local features by performing time-independent convolution operations on series data. CNN is used to model sentences for classification tasks. Some recent studies have applied the attention mechanism to CNN for text classification [9] and relation extraction [10] tasks. Using the attention mechanism on CNN, a network can learn which feature should be emphasized in the local features obtained by the convolution operation.

We design the first detector using two parallel-connected CNN encoders (Parallel-CNN detector, Figure 1). The parallel connection is chosen because local level breakdowns are within an utterance. A pair of CNN encoders encodes sequences of vector representations of words converted from the user and system utterances. Then the three-layered perceptron receives the concatenated vector comprised of the outputs of two encoders and outputs probabilities of each breakdown label (O, T, X).

2.2.2. Series-LSTM detector

RNN is a kind of neural network model to process time sequence data using previously hidden states as the current input. Long short-term memory (LSTM), which is an extension of RNN, is then developed to process the long-term context.

We design the second detector using two series-connected LSTM encoders (Series-LSTM detector, 2). The series connection is chosen to consider the previous utterance in order to detect the context-level breakdown in an utterance and its reply. Firstly, an LSTM encoder (user encoder) encodes the sequence of vector representations of words converted from the user utterance. Then another encoder (system encoder) receives the

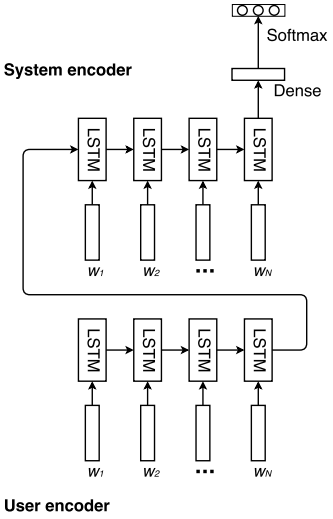

Figure 2: *Design of Series-LSTM dialogue breakdown detector.*

output of the user encoder as the initial state and encodes the sequence of vector representations of words converted from the system utterance. Finally, the softmax layer receives the output of the system encoder and computes the probability of each breakdown label (O, T, X).

2.3. Ensemble

We construct dialogue breakdown detectors per each cluster. In addition, each detector can have two network designs (Parallel-CNN and Series-LSTM). Each of them may capture different aspects of breakdowns. To make use of their prediction powers, we ensemble them using the normalized average of all predictions.

3. Preliminary experiment

We conducted a preliminary experiment to determine the appropriate cluster size for the formal run.

3.1. Datasets

The DBDC3 provides different datasets as development data for the Japanese task. All dialogues in each dataset consist of 21 user-system utterances, which are initiated by a system. The names, number of data, and description of each development data are listed below:

Chat dialogue corpus (1,146 dialogues)
Consists of dialogues collected using the chat API provided by NTT Docomo (DCM). The initial 100 dialogues are annotated by 24 annotators, and the remaining 1,046 dialogues are annotated by 23 annotators.

Developement data for DBDC1 (200 dialogues)
Consists of dialogues collected by DCM. Each dialogue is annotated by 30 annotators.

Evaluation data for DBDC1 (20 dialogues)
Consists of dialogues collected by DCM. Each dialogue is annotated by 30 annotators.

Developement data for DBDC2 (150 dialogues)
Consists of dialogues collected by DCM, DIT (Denso IT Laboratories system), and IRS (IR-status based system[11]). Each dialogue is annotated by 30 annotators.

Evaluation data for DBDC2 (150 dialogues)
Consists of dialogues collected by DCM, DIT and IRS. Each dialogue is annotated by 30 annotators.

In our preliminary experiment, we used the evaluation data for DBDC2 for testing and the rest for development. We assessed models for each test data of DCM, DIT, and IRS.

We trained word embedding using the CBoW in Word2Vec with Japanese Wikipedia data [1]. However, because the articles of Wikipedia are not written in colloquial style, they are insufficient to train word embedding in domains such as chat dialogue. Hence, we also used some play scripts crawled by ourselves from the web.

3.2. Evaluation criteria

The DBDC has two types of evaluation metrics: classification-related and distribution-related. Classification-related metrics consist of Accuracy, Recall, Precision, and F-measure. Distribution-related metrics consist of mean squared error (MSE) and Jensen-Shannon divergence (JSD). Both MSE and JSD are calculated by the predicted distribution of the output labels (O, T, X) and the gold labels.

3.3. Settings

MeCab is used for morphological analysis. Then words with a frequency of 50 or less in the corpus are replaced with tokens representing each part-of-speech when training word embedding. The dimension of word embedding is set to 512. We set the length of sliding window of CNN to 3 and 5, and the number of feature maps to 256. The hidden state dimensions of LSTM cells are 512. To train both of the Series-LSTM detector and Parallel-CNN detector, MSE and Adam are used as the loss function and the optimizer, respectively.

In the experiment, the number of k-means clusters is changed from one to ten in an incremental manner to determine the optimal number of clusters in the formal run. Although the DBDC task has many metrics, we considered only MSE and F-measure.

3.4. Results

Figures 3 and 4, which show the relationships between the number of clusters and F-measure or MSE on a Series-LSTM detector and a Parallel-CNN detector, respectively. Both Series-LSTM and Parallel-CNN have a larger F value and a smaller MSE when there are two or more clusters. This effect is especially significant for DCM.

For the Parallel-CNN, the average F value is maximized (0.642) when there are seven clusters, and the average MSE is minimized (0.0467) when there are six clusters. For the Series-LSTM detector, the average F value is maximized (0.626) when there are five clusters, and the average MSE is minimized (0.0489) when there are seven clusters.

By training different detectors per a group of similar annotators and then ensembling their outputs, we can significantly improve both F-measure and MSE values. When we observed the training data, we found that majority of labels to DIT were X (Breakdown) and a detector trained by the entire data can easily overfit to them. We can avoid to be overly affected by such biased annotations due to our approach.

4. Formal run

We participated in the DBDC3. In this section, we describe the submitted models and their detection results.

4.1. The submission model

We submitted dialogue breakdown detection results using three models as below.

Run1 The Parallel-CNN detector achieved the best F-measure in the preliminary experiment

Run2 The Series-LSTM detector achieved the best F-measure in the preliminary experiment

Run3 The ensemble (average values of predictions) of Run1 and Run2

4.2. Results

Table 1 and 2 show F-measure and MSE values of our runs as well as the baseline, respectively[2]. The baseline detector is constructed using the conditional random fields (CRF) [12] where features are words in user and system utterances. As you can see from the results, all of our runs outperform the baseline. Notably, Run3, which ensembles Run1 and Run2, achieved the highest F-measure of 63.6%.

Table 3 shows Precisions, Recalls and F-measures of each run. Interestingly, Run1 (Parallel-CNN) achieves the highest Precision while Run2 (Series-LSTM) achieves the highest Recall. These results supports our assumption that dialogue breakdotwns can be locally and contextually, and these models can capture these. Finally, Run3 successfully takes the advantages

[1] https://dumps.wikimedia.org/jawiki/20170920/jawiki-20170920-pages-articles.xml.bz2

[2] Please refer to the DBDC3 overview paper [6] for more detailed results.

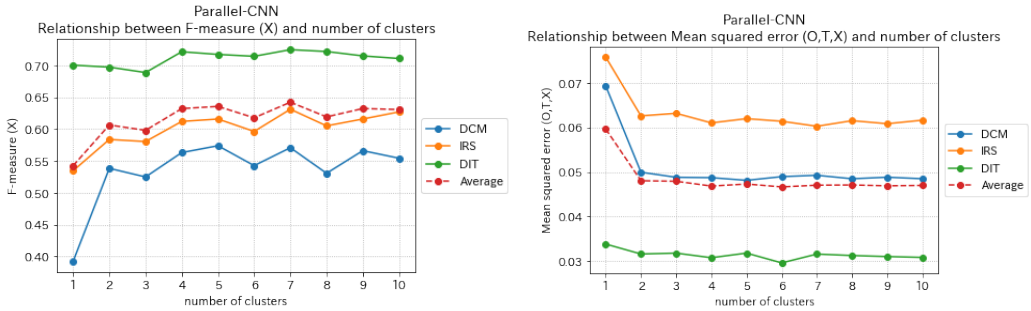

Figure 3: *Results of preliminary experiment: Relationships between the number of clusters and F-measures (left) or MSE (right) on Parallel-CNN detector*

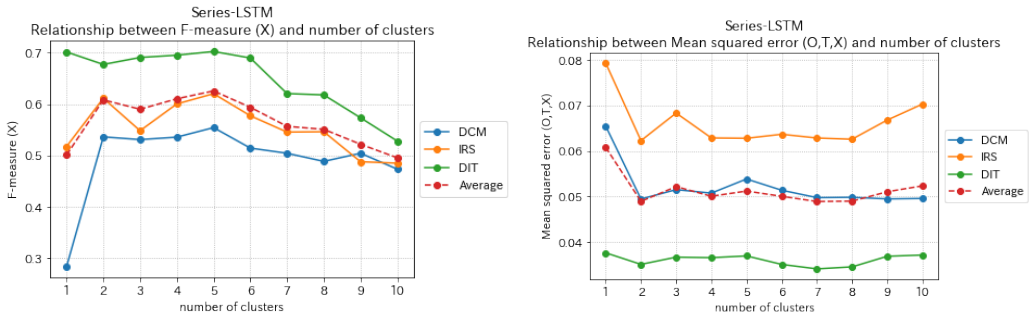

Figure 4: *Results of preliminary experiment: Relationships between the number of clusters and F-measures (left) or MSE (right) on Series-LSTM detector*

Runs	Average	DCM	DIT	IRS
Baseline (CRF)	0.5796	0.4502	0.6634	0.6252
Run1	0.6316	**0.4785**	0.7165	0.6998
Run2	0.6230	0.4413	0.7307	0.6968
Run3	**0.6364**	0.4629	**0.7323**	**0.7140**

Table 1: *Results on F-measure of X (Breakdown) label for DBDC3 Formal Run*

Runs	Average	DCM	DIT	IRS
Baseline (CRF)	0.1996	0.2169	0.1870	0.1950
Run1	**0.0465**	**0.0519**	**0.0352**	0.0525
Run2	0.0498	0.0545	0.0407	0.0543
Run3	0.0469	0.0525	0.0365	**0.0517**

Table 2: *Results of MSE on X (Breakdown) for DBDC3 Formal Run*

of these two models by ensembling as shown by its highest Recall without a significant loss in Precision, which results in the highest F-measure.

5. Conclusion

We focused on the differences of annotation distributions of each annotator and built clustering and ensemble based dialogue breakdown detectors. The official DBDC3 Japanese task results show that our models significantly outperform the CRF-based baseline. We participated in only the Japanese task because of low resources provided in the English task. However, our models are language independent that are applicable to dialogue

Runs	Precision	Recall	F-measure
Run1 (Parallel-CNN)	**0.5113**	0.8386	0.6316
Run2 (Series-LSTM)	0.4939	**0.8498**	0.6230
Run3 (Ensemble)	0.5055	0.8687	**0.6364**

Table 3: *Overall Precisions, Recalls, F-measures of our three runs*

breakdown detection in any languages if a sufficient amount of training data is available. As future work, we will analyze the errors and characteristics of our models, especially about the difference of Series-LSTM detector and Parallel-CNN detector to improve their detection ability.

Acknowledgements

This project is funded by Microsoft Research Asia.

6. References

[1] R. Higashinaka, K. Funakoshi, Y. Kobayashi, and M. Inaba, "The dialogue breakdown detection challenge: Task description, datasets, and evaluation metrics," in *Proceedings of The Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, 2016.

[2] H. Sugiyama, "Chat-oriented dialogue breakdown detection based on the analysis of error patterns in utterance generation (in japanese)," in *Proceedings of SIG-SLUD 78*, 2016.

[3] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machile Learning Journal*, vol. 63, pp. 3–42, 2006.

[4] M. Inaba and K. Takahashi, "Dialogue breakdown detection using rnn encoders (in japanese)," in *Proceedings of SIG-SLUD 78*, 2016.

[5] T. Kubo and H. Nakayama, "Learning dialog and its breakdowns simultaneously by neural conversational model (in japanese)," in *Proceedings of SIG-SLUD 78*, 2016.

[6] R. Higashinaka, K. Funakoshi, M. Inaba, Y. Tsunomori, T. Takahashi, and N. Kaji, "Overview of dialogue breakdown detection challenge 3," in *Proceedings of Dialog System Technology Challenge 6 (DSTC6) Workshop*, 2017.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of the Workshop of International Conference on Learning Representations*, 2013.

[8] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proceedings of The Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2007, pp. 1027–1035.

[9] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2016)*, 2016, pp. 1480–1489.

[10] L. Wang, Z. Cao, G. De Melo, and Z. Liu, "Relation classification via multi-level attention cnns," in *Proceedings of The 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, 2016, pp. 1298–1307.

[11] A. Ritter, C. Cherry, and W. B. Dolan, "Data-driven response generation in social media," in *Proceedings of 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, 2011, pp. 583–593.

[12] J. Lafferty, A. Mccallum, F. C. Pereira, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of The 18th International Conference on Machine Learning (ICML 2001)*, 2001, pp. 282–289.