Dialog response ranking is used to rank response candidates by considering their relation to the dialog history. Although researchers have addressed this concept for open-domain dialogs, little attention has been focused on task-oriented dialogs. Furthermore, no previous studies have analyzed whether response ranking can improve the performance of existing dialog systems in real human–computer dialogs with speech recognition errors. In this paper, we propose a context-aware dialog response re-ranking system. Our system reranks responses in two steps: (1) it calculates matching scores for each candidate response and the current dialog context; (2) it combines the matching scores and a probability distribution of the candidates from an existing dialog system for response re-ranking. By using neural word embedding-based models and handcrafted or logistic regression-based ensemble models, we have improved the performance of a recently proposed end-to-end task-oriented dialog system on real dialogs with speech recognition errors.
filtering was also implemented in [5] by integrating the developer’s hand-crafted rules into end-to-end networks.

To obtain consistent behavior, a model that uses response candidates directly as input features is also useful. The task of choosing a correct response among the candidate responses is called response ranking. Generally, there are two types of approaches: query-response pair base [16, 17, 18] and context-response pair base [19, 20, 21, 22, 23]. The former matches a user query with candidates without using the dialog history and handles task-oriented dialogs. The latter uses a dialog history-candidate pair and addresses open-domain dialogs. However, these systems have not been used for response re-ranking.

This paper proposes a context-aware dialog re-ranking system that employs two additional models: Match and ReRank. Match considers the response candidates to be related to the dialog context. ReRank combines different outputs, i.e., predictions from a dialog system and matching scores from Match for better prediction. We employ an existing dialog system into our framework.

2. RESPONSE RE-RANKING MODEL

In this section, we describe the construction of the response re-ranking system shown in Fig. 1. Our re-ranking system consists of three main modules: the base dialog system (BDS), Match, and ReRank.

2.1. Base Dialog System

BDS (Base Dialog System) is an arbitrary dialog system which predicts a system response with a probability distribution. In our re-ranking model, BDS is a fixed model, i.e., it does not require training for re-ranking as we assume that existing dialog systems are reused as BDS. This reusability allows use of various existing models without additional training for the BDS. In this work, we used end-to-end memory networks [24, 4] as our BDS.

Memory Networks: Memory Networks are neural networks that read and store events for solving tasks in the areas of natural language processing. Let \{x_1, ..., x_T\}, q, and \{a_1, ..., a_N\} represent the words of the input dialog history, a query, and candidates for system action, respectively, where \(T\) is the number of turns of the dialog history, and \(N\) the number of all possible system actions. Each variable constitutes one sentence. Further, \(x_i\) is embedded in a \(d\)-dimensional vector using matrix \(A \in \mathbb{R}^{d \times V}\), (where \(V\) is the vocabulary size) and is encoded into a memory vector \(m_i \in \mathbb{R}^d\) using position encoding, a technique reported in [24] where \(m_i\) is affected by word order. The query \(q\) is also embedded into query vector \(u\) using embedding \(B \in \mathbb{R}^{d \times V}\). With the memory vectors and \(u\), an attention score for each item of memory \(m_i\) is given by

\[
p_i = \text{Softmax}(u^Tm_i), \quad (i = 1, \ldots, T).
\]  

We also have a vector \(c_i\) from \(x_i\) using matrix \(C \in \mathbb{R}^{d \times V}\). The memories are read by taking their weighted sum:

\[
o = \sum_i p_i c_i.
\]  

We employ \(K\) hop operations. Thus, the input to the \(k + 1\)-th layer is updated by the following equation:

\[
u^{k+1} = u^k + o^k.
\]  

We also apply adjacent weight tying, e.g., \(A^{k+1} = C^k\) and \(B = A^1\). Finally, the model predicts the system actions \(\hat{a}\) using the weight matrix \(W\), as follows:

\[
\hat{a} = \text{argmax}(\text{Softmax}(W u^{K+1})).
\]

2.2. Match

Match calculates matching scores between the dialog history and the candidates generated by BDS. While BDS predicts a system response given the dialog history, Match uses the candidate responses directly for the matching scores. Generally, these matching methods are used as response selection models such as [16, 17, 18, 19, 20, 21, 22, 23]. In our re-ranking task, we use the scores as input features of re-ranking.

To evaluate the effect of Match, we prepared five models which include two non-machine learning models. The first three models are identical to those described in [4]. The fourth model is based on Memory Networks for predicting matching scores. The last model, QA-LSTM, is an response selection model developed by [17]. We now describe each of Match in detail.

TF-IDF: This model uses bag-of-words features to represent inputs and targets: the whole dialog history including the last utterance and the candidate responses, respectively. The matching score is the TF-IDF weighted cosine similarity between the inputs and the targets.

Nearest Neighbor (NN): In this model, we consider (last utterance - response) pairs for the scoring method. By considering the pairs, this model attempts to find the most similar conversation in the training set. Word overlap is used as the scoring mechanism. The pairs are sorted by decreasing the co-occurrence frequency when multiple responses are linked to the same utterance in the training set.

Supervised Embedding (SLEmb): This is a supervised word embedding method. Let \(x\) represent the words of the input dialog history including the last utterance, \(y\) represents the candidate response to the input. Then, the scoring function is given by: \(f(x, y) = (Ax)^T(By)\), where \(A\) and \(B\) are \(d \times V\) word embedding matrices, (where \(d\) is embedding size and \(V\) size of the vocabulary). The embedding model is trained with a margin ranking loss with negative samples.

Match Memory Networks (MMNs): We also used Memory Networks for Match, referred to as Match Memory Networks (MMNs) to distinguish them from BDS. We
modified two equations of the original Memory Networks model described in [21]. We first modified Equation 1 by taking the L2 norm for attention, as our preliminary analysis showed that attention is biased to one or two dialog turns without the normalization. The second modification was the last layer (Eq. 4). While the original memory networks predict system actions using a weight matrix, MMNs calculate a cosine similarity between the dialog context and the candidate response \( a_j \). \( a_j \) is embedded into vector \( v_j \) via matrix \( A^{K+1} \) in the same manner as the \( q \) embedding. Thus,

\[
match_j = \cos(a^{K+1}, v_j), \quad (1 \leq j \leq N),
\]

where \( match_j \) represents the extent to which a given response candidate is related to the given dialog context.

**QA-LSTM:** QA-LSTM [17] is a simple and strong model for response selection tasks (Fig. 2). The dialog history and the candidate responses are encoded into the same word representations as Memory Networks and Supervised Embedding. Each input and target is fed to bidirectional long short-term memory (BiLSTM), which can generate word-level representations. Each output is aggregated in one of three simple ways: (1) average pooling; (2) max pooling; (3) concatenation of the last vectors of both directions. In this study, we used a max pooling method to aggregate each word representation. Finally, the cosine similarity is calculated between both aggregated representations. For the loss function, we used the same margin ranking loss function as for Supervised Embedding. The margin loss is calculated for each dialog turn:

\[
L_{\text{match}} = \max(0, m - match_+ + match_-),
\]

where \( m \) represents a constant margin (a hyperparameter) between the scores of correct and incorrect system action pairs. \( match_+ \) and \( match_- \) return a matching score for correct and incorrect action candidates, respectively, using the same dialog context given in Eq. 5.

### 2.3. ReRank

**ReRank** reranks the candidate responses by considering the two predictions from BDS and Match. To combine the two outputs, we developed two ReRank models: a rule-based and a logistic regression model.

**Rule:** Rule is a heuristically-designed method for combining the two different BDS and Match outputs. This model calculates \( \text{score}_i \) as a re-ranked score:

\[
\text{score}_i = \text{Normalize}(p_i) \times (\alpha_i \times \text{matching_score}_i),
\]

where \( p_i \) is the probability of BDS candidate \( a_i \), and \( \alpha_i \) is a scalar reflecting of the rank of \( a_i \) of the matching score of Match. The highest scoring response is selected as the final response.

**Stacking and logistic regression:** We tackled this problem as ensemble learning, a technique that uses multiple learning algorithms to obtain better prediction. Among the various techniques, we used *Stacking* [25] to combine two models. *Stacking* gives the predictions of multiple models given as input to a second-level learning model. Our Stack- ing model has two learning layers: (1) training of base-level classifiers; (2) training of meta-classifiers (Fig. 3). We can use this meta-classifier as our ReRank. We explain this model with Fig. 3.

Both BDS and Match are treated as base classifiers in Stacking system. To train the models, the training data are split into folds (subsets). Match is trained on training subset 1 and 2. The trained Match output is the prediction for training subset 3. This process is repeated until all subsets are used for the predictions. In our experiments, BDS was trained on the entire training set at once since we took this model as a fixed model. Note that it is not necessary for BDS to be trained in the first stacking layer; i.e., our system can afford to reuse existing models trained on different datasets. After training the base classifiers, we obtain two predictions from BDS and Match; the former is a probability distribution over system responses \( y_{\text{BDS}} \), and the latter is matching scores \( y_{\text{match}} \) between the dialog history and the candidate responses.

The meta classifiers use the predictions from the base classifiers for the final prediction. We apply multiple logistic regression (LR) models as the classifier. There are many similar api call system actions in the bAbI dialog dataset (Section 3.1) [4], which are different from slot entities. Note that api call is a special system action for searching restaurant information, taking multiple slots as its arguments. To capture similarity in the meta classifiers, we developed multiple LR models and simplified the api call actions as one action. The first LR predicts the simplified system actions. The remaining LRs predict each slot in api call, e.g., cuisine type, location, and price. The output from the slot LRs is used to reconstruct the original api call actions, where the first LR predicting the current response is the api call with the highest score. In LR training, the cost function is the sum of the cross entropy of each LR.
We found that using multiple LRs for system actions is effective for predicting similar api call actions. In the dataset employed in this work, some api call actions in the test data do not appear in the training data. On the other hand, all slots appear in the training data. Therefore, separately handling arguments works well.

We also use ranking masks \( m_{bds} \) and \( m_{mat} \) (for BDS and Match, respectively) to obtain the predictions, to focus on the high-score candidates. The mask values are set to 1.0 if the candidate is within the top \( H \) predictions; otherwise, 0. The masks are used for BDS and Match separately.

We use additional features for the LR input: dialog context embedding \( e_{ctx} \), answer embedding \( e_{ans} \), and the length of the dialog history \( l \) for QA-LSTM and MMNs. Here, \( e_{ctx} \) is a dialog history embedding vector. In MMNs, this is relevant to \( u_{K+1} \) in Eq. 5. In QA-LSTM, this vector is aggregated from vectors of the BiLSTM of the dialog history. Similarly, \( e_{ans} \) is also an embedding vector; however, it differs in that it embeds the candidate response. We choose the response embedding with the highest matching score from the Match output. Finally, \( l \) is a one-hot vector which represents the number of turns of the current dialog history. Then, the input of the meta classifier is:

\[
input = [y_{bds} \odot m_{bds}, y_{mat} \odot m_{mat}, e_{ctx} + e_{ans}, l],
\]

where \( \odot \) denotes element-wise multiplication, + the element-wise plus, and \([\cdot, \cdot] \) the concatenation. The input is shared for all LRs.

3. EXPERIMENTS

In this section, we report experiments to determine whether our re-ranker improves the performance of a recently proposed end-to-end dialog system \([4]\) by considering the relation between the dialog context and the response candidates generated by the dialog system.

3.1. Dataset

We used the following datasets to assess our system:

bAbI dialog dataset: The bAbI dialog dataset \([4]\) is a set of six tasks from dialogs about restaurant reservations. Although Tasks 1–5 were systematically generated using the same dialog dataset, they were different in that each required different dialog skills. In contrast, Task 6 is a human–computer restaurant reservation dialog dataset created by converting the Second Dialog State Tracking Challenge dataset \([1]\) into the bAbI dataset format. Thus, Task 6 is based on a real dialog dataset and incorporates speech transcription. We found that the speech recognition error rate was high, as shown on Table 1. Therefore, to use real dialog data containing the ASR errors, we reproduced a Task 6 generator by following the configurations reported in \([4]\). We chose a 1-best hypothesis of the ASR results; we call the resulting Task 6 “ASR-Task 6”.

Table 1. Speech recognition error analysis of DSTC2. (WER: word error rate). Although both recognizers are based on the same model, Recog 0 had artificially degraded acoustic models. Note that “filtered” indicates the evaluations in which stop-words were removed from the text before the evaluation.

|          | Average | Recog 0 | Recog 1 |
|----------|---------|---------|---------|
| WER      | 25.57%  | 31.82%  | 19.24%  |
| WER (filtered) | 24.17%  | 30.03%  | 18.58%  |

bAbI+: bAbI+ \([26]\) was created systematically by adding natural disfluencies, such as self-corrections, hesitations, and restarts to the bAbI dialog dataset. bAbI+ was limited to Task 1 as it focused on the capability of the system to ask users about their restaurant preferences.

3.2. Setup

We conducted two experiments on bAbI/bAbI+ involving ASR-Task 6.

bAbI/bAbI+: In this experiment, our re-ranker was trained on the bAbI dialog Task 1 and tested on the bAbI+ dataset. We used the same experiment settings as in \([26]\). It is known that Memory Networks can perfectly answer questions for Task 1 \([4]\). However, the Task 1 accuracy decreases dramatically if the models are tested with bAbI+ \([26]\). Furthermore, we are interested in how our models are robust to the disfluencies. That is why we used bAbI+ for the testing.
ASR-Task 6: Our re-ranker was trained on the newly created ASR-Task 6.

We used training/validation/test sets that were already split by the dataset providers on both bAbI+ and DSTC2.

3.3. Implementation details

**BDS**: We use Memory Networks as BDS. In the bAbI/bAbI+ task, we employed their implementation. In the ASR-Task 6 task, we reproduced Memory Networks by following the configurations reported in [4]. Basically, both models are the same.

**Match**: We followed the configurations reported in [4] for TF-IDF, Nearest Neighbor, and Supervised Embedding. For QA-LSTM, we used a bidirectional gated recurrent unit (GRU) to obtain vectors of the dialog context and the candidate response. We used 64 GRU dimensions and shared the GRU for encoding of the dialog context and of the candidate. For MMNs, we used a word embedding size of 128 and set the number of hops (K) to three. For QA-LSTM and MMNs, the following parameters were common: a word embedding size of 128, a margin M of 0.5, and a training set split into five folds. Negative sampling was performed 100 times, on the condition that the loss L was positive for every batch. These models were trained with the Adam optimizer for 20 epochs in each fold and a batch size of 32. For brevity, the Match was shared for both ReRank models; however, the rule-based ReRank does not necessarily require training on fold data.

**ReRank**: For Rule, to calculate scores, we used a sigmoid function to normalize the BDS probabilities. The probabilities of most candidates were almost zero, because the base model used softmax over thousands of candidates. As the probability of a correct response has been close to zero, we were required to change the range of the probabilities. α was set to 1.0 if the score of a given candidate was within the top five, as arranged by Match; otherwise, it was set to 0. For Stacking, all the meta classifiers with the hidden dimension of 700 (one-layer perceptron) were used, with a H of 10 for the input masks. The training batch size was 64 for 20 epochs with the Adam.

3.4. Main results

The accuracy results for the dialog turns are presented in Table 2. Here, the accuracy corresponds to the ratio of correct response selection for the entire dataset. “MAT” represents the accuracy based on the score of Match, “RR1” corresponds to rule-based ReRank, and “RR2” is for the stacking ReRank.

TF-IDF and NN yielded poor results for both datasets. It is difficult for TF-IDF to handle dialog features, such as dialog flow, as bag-of-words does not consider word order. The results show that choosing a correct response without considering the current dialog context is almost impossible. NN uses (last utterance – candidate response) pairs to choose the response. bAbI/bAbI+ has synthesized simple dialogs; therefore, the performance is better than that of TF-IDF. However, NN is not effective for ASR-Task 6 since it is quite rare for exactly the same pair to be found in the training dialog.

SLEmb obtained almost the highest accuracy on the bAbI/bAbI+ while its accuracy on ASR-Task 6 dropped dramatically. It can be presumed that this change was due to the difference in vocabulary size. The vocabulary size of ASR-Task 6 is 1490, whereas that of bAbI/bAbI+ is 111. Moreover, the (context – response) pairs were not simple since ASR-Task 6 is a corpus of real dialogs. Therefore, SLEmb only works for limited vocabulary and dialog patterns, e.g., synthetically generated dialogs.

Unlike the previous models, MMNs and QA-LSTM sufficiently improved the accuracy of BDS on both datasets.

All MMN models outperformed BDS. Unlike the previous models, Memory Networks have the ability to read dialog context. Further, the cosine similarity between the dialog context and the candidate responses was effective in improving predictions.

QA-LSTM can also understand the dialog context from the Memory Networks, i.e., the former uses recurrent neural networks, the latter uses memory reading and writing components. For QA-LSTM, RR1 and RR2 had the highest total accuracy scores on both datasets. Both the RR1 and RR2 models could combine different outputs to achieve better predictions.

Overall, for both MMNs and QA-LSTM, the accuracy of api_call was dramatically boosted. We presume that word-wise embedding is effective since the arguments (slots) in api_call appear in the dialog history. We analyzed the effect of word-wise embedding in Section 3.5.

3.5. Analysis of improvement

The API call improvement was much higher than that for other system response types (e.g., asking slots). Therefore, we analyzed the api_call results by focusing on Match (QA-LSTM and MMNs) and ReRank.

**Match**: Match can use response candidates directly for the matching scores, whereas BDS simply uses the dialog history and the current user query, as shown in Fig 1. Thus, Match has an advantage when some words are shared between the dialog history and the response candidate. api_call action contains slot entities that are mentioned by a user in the dialog history. That is why Match yields improvement in api_call. Furthermore, Match can reflect the number of slots matching with the score, as apparent from Table 3. Note that Match uses the cosine similarity, whereas BDS uses softmax to calculate scores.
Table 2. Dialog turn accuracy. “Total” represents the accuracy for an entire test dataset and “API” indicates the accuracy of the API call action that the dialog system decided to take. The best-performing models are formatted in bold, while underscores indicate the best score for each model and dataset combination. A yellow background indicates a result superior to that of BDS.

|       | bAbI/bAbI+ | ASR-Task-6 |
|-------|------------|------------|
|       | Total API  | Total API  |
| BDS   | 82.1       | 21.7       |
|       |            | 42.5       |
| TF-IDF| 6.0        | 2.3        |
|       | 3.8        | 0.0        |
| MMNs  | 8.2        | 15.6       |
|       | 4.3        | 0.4        |
| NN    | 51.6       | 0.4        |
|       | 38.6       | 30.2       |
| SLEmb | MAT        | 41.8       |
|       | 0.0        | 0.3        |
|       | 10.4       | 0.0        |
|       | 38.0       | 27.9       |
| MMNs  | MAT        | 91.3       |
|       | 51.6       | 25.9       |
|       | 12.7       |            |
|       | RR1        | 90.7       |
|       | 47.9       | 33.5       |
|       | 27.2       |            |
|       | RR2        | 82.3       |
|       | 17.6       | 25.0       |
|       | 5.4        |            |
| QA-LSTM| MAT       | 84.7       |
|       | 38.1       | 43.7       |
|       | 50.6       |            |
| MMNs  | RR1        | 86.3       |
|       | 38.3       | 45.6       |
|       | 53.9       |            |
|       | RR2        | 86.0       |
|       | 36.6       | 44.8       |
|       | 43.2       |            |
| QA-LSTM| MAT       | 92.0       |
|       | 60.9       | 46.2       |
|       | 70.3       |            |
|       | RR1        | 88.8       |
|       | 49.6       | 48.7       |
|       | 68.1       |            |
|       | RR2        | 92.2       |
|       | 60.8       | 46.7       |
|       | 50.5       |            |

RR1 (Rule-based): This ReRank strongly depends on the result of Match result, i.e., $\alpha$ is only set to 1 for the top $N$ candidates in Match as Eq. [7] shows. Table 4 indicates that Match has good accuracy results for the top K candidates compared to BDS. This coverage contributes to the success of RR1.

RR2 (Stacking and logistic regression): As mentioned above, our word-based matching approach works well for re-ranking. It uses both the predictions and the word features as input. Table 5 presents the results of an ablation study to check whether the strength of word features of Match propagates to ReRank. While general stacking uses predictions from base classifiers only, we use additional features, i.e., the dialog context word features and the candidate response. If we exclude either context or answer embeddings, the accuracy of Total changes only slightly; however the API accuracy significantly decreases significantly. We presume that word-embedding features are effective for system actions that use slot entities, as removing both word features causes the Total accuracy to decrease dramatically.

### 4. CONCLUSIONS AND FUTURE WORK

This paper describes a context-aware dialog response re-ranking system for task-oriented dialog systems based on two key re-ranking modules; Match and ReRank. We have boosted the performance of an existing task-oriented dialog system.

Table 3. Sample comparison of top three candidates of BDS and Match for ASR-Task 6. The underscores indicate the correct slots. “R_slot” means that the user does not care about the restaurant preference.

| Score | Predicted Answer |
|-------|------------------|
| BDS   | 8.2              |
|       | api_call R_cuisine east expensive |
| MMNs  | 8.0              |
|       | api_call R_cuisine south expensive |
| QA-LSTM | 8.7            |
|       | api_call R_cuisine R_price |

Table 4. Accuracy of top $K$ candidates. The ground truth label was included in the top $K$ predictions.

| bAbI/bAbI+ | ASR-Task 6 |
|------------|------------|
| BDS        | 82.1       |
| MMNs       | 92.0       |
| QA-LSTM    | 82.1       |

Table 5. Ablation study on ASR-Task 6, where “ctx” indicates $e_{ctx}$ and “ans” indicates $e_{ans}$ in Eq. [8]

| MMNs | ASR-Task 6 |
|------|------------|
| BDS  | 82.1       |
| MMNs | 92.0       |
| QA-LSTM | 82.1     |

RR1 (Rule-based): This ReRank strongly depends on the result of Match result, i.e., $\alpha$ is only set to 1 for the top $N$ candidates in Match as Eq. [7] shows. Table 4 indicates that Match has good accuracy results for the top K candidates compared to BDS. This coverage contributes to the success of RR1.

Match (Word-based matching). This re-ranking system works well for re-ranking. It uses both the predictions and the word features as input. Table 5 presents the results of an ablation study to check whether the strength of word features of Match propagates to ReRank. While general stacking uses predictions from base classifiers only, we use additional features, i.e., the dialog context word features and the candidate response. If we exclude either context or answer embeddings, the accuracy of Total changes only slightly; however the API accuracy significantly decreases significantly. We presume that word-embedding features are effective for system actions that use slot entities, as removing both word features causes the Total accuracy to decrease dramatically.

We evaluated five Match models, and two ReRank models on the real human–computer restaurant search dialogs with speech recognition errors. Our framework improved the existing dialog system by using neural-based Match models and both ReRank models.

To our knowledge none of the previous studies presented a re-ranking response task that uses response ranking to validate response candidates. We have presented a simple and effective re-ranking module to reorder response candidates.

Our research should be extended to natural dialogs including hesitations and corrections since applying word-level attention may be key to resolving disfluencies. We should also apply this technique to other types of dialog systems (e.g., chit-chat and question answering).

### 5. REFERENCES

[1] Matthew Henderson, Blaise Thomson, and Jason D Williams, “The second dialog state tracking challenge.”
in Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), 2014, pp. 263–272.

[2] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig, “The microsoft 2016 conversational speech recognition system,” in Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on. IEEE, 2017, pp. 5255–5259.

[3] Chung-Cheng Chiu, Tara Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J. Weiss, Kanishka Rao, Katya Gornina, Navdeep Jaitly, Bo Li, Jan Chorowski, and Michiel Bacchiani, “State-of-the-art speech recognition with sequence-to-sequence models,” 2018.

[4] Antoine Bordes and Jason Weston, “Learning end-to-end goal-oriented dialog,” CoRR, vol. abs/1605.07683, 2016.

[5] Jason D. Williams, Kavosh Asadi, and Geoffrey Zweig, “Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning,” in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers, 2017, pp. 665–677.

[6] Ananlada Chotimongkol and Alexander I Rudnicky, “N-best speech hypotheses reordering using linear regression,” 2001.

[7] Sibel Yaman, Li Deng, Dong Ye, Ye-Yi Wang, and Alex Acero, “A discriminative training framework using n-best speech recognition transcriptions and scores for spoken utterance classification,” in Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on. IEEE, 2007, vol. 4, pp. IV–5.

[8] Gökhan Tür, Anoop Deoras, and Dilek Hakkani-Tür, “Semantic parsing using word confusion networks with conditional random fields.” in INTERSPEECH, 2013, pp. 2579–2583.

[9] Antoine Raux, Brian Langner, Dan Bohus, Alan W Black, and Maxine Eskenazi, “Let’s go public! taking a spoken dialog system to the real world,” in Ninth European Conference on Speech Communication and Technology, 2005.

[10] Steve Young, “Using pomdps for dialog management,” in Spoken Language Technology Workshop, 2006. IEEE. IEEE, 2006, pp. 8–13.

[11] Tiancheng Zhao and Maxine Eskenazi, “Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning.” in Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue. 2016, pp. 1–10, Association for Computational Linguistics.

[12] Xiujuan Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz, “End-to-end task-completion neural dialogue systems,” in Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers). 2017, pp. 733–743, Asian Federation of Natural Language Processing.

[13] Rebecca Jonson, “Dialogue context-based re-ranking of asr hypotheses,” in Spoken Language Technology Workshop. 2006. IEEE. IEEE, 2006, pp. 174–177.

[14] Mikio Nakano, Shun Sato, Kazunori Komatani, Kyoko Matsuyama, Kotaro Funakoshi, and Hiroshi G Okuno, “A two-stage domain selection framework for extensible multi-domain spoken dialogue systems,” in Proceedings of the SIGDIAL 2011 Conference. Association for Computational Linguistics, 2011, pp. 18–29.

[15] Seonghan Ryu, Donghyeon Lee, Injae Lee, Sangdo Han, Gary Geunbae Lee, Myungjae Kim, and Kyung-duk Kim, “A hierarchical domain model-based multi-domain selection framework for multi-domain dialog systems,” Proceedings of COLING 2012: Posters, pp. 1049–1058, 2012.

[16] Zhengdong Lu and Hang Li, “A deep architecture for matching short texts.” in Advances in Neural Information Processing Systems 26, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., pp. 1367–1375. Curran Associates, Inc., 2013.

[17] Ming Tan, Bing Xiang, and Bowen Zhou, “Lstm-based deep learning models for non-factoid answer selection,” CoRR, vol. abs/1511.04108, 2015.

[18] Fedorenko Denis, Smetanin Nikita, and Artem Rodichev, “Avoiding echo-responses in a retrieval-based conversation system,” 2017.

[19] Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau, “The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems,” in Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue. 2015, pp. 285–294, Association for Computational Linguistics.

[20] Rudolf Kadlec, Martin Schmid, and Jan Kleindienst, “Improved deep learning baselines for ubuntu corpus dialogs,” CoRR, vol. abs/1510.03753, 2015.
[21] Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan, “Multi-view response selection for human-computer conversation,” in Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. 2016, pp. 372–381, Association for Computational Linguistics.

[22] Bowen Wu, Baoxun Wang, and Hui Xue, “Ranking responses oriented to conversational relevance in chatbots,” in Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. 2016, pp. 652–662, The COLING 2016 Organizing Committee.

[23] Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li, “Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots,” in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2017, pp. 496–505, Association for Computational Linguistics.

[24] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus, “End-to-end memory networks,” in Advances in Neural Information Processing Systems 28, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., pp. 2440–2448. Curran Associates, Inc., 2015.

[25] David H Wolpert, “Stacked generalization,” Neural networks, vol. 5, no. 2, pp. 241–259, 1992.

[26] Igor Shalyminov, Arash Eshghi, and Oliver Lemon, “Challenging Neural Dialogue Models with Natural Data: Memory Networks Fail on Incremental Phenomena,” in Proceedings of the 21st SemDial Workshop on the Semantics and Pragmatics of Dialogue (SaarDial), 2017, pp. 98–106.

[27] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” arXiv preprint arXiv:1406.1078, 2014.

[28] Diederik P. Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” CoRR, vol. abs/1412.6980, 2014.