SEEntNet: Source-aware Recurrent Entity Network for Dialogue Response Selection

Jiahuan Pei\textsuperscript{1}, Arent Stienstra\textsuperscript{1}, Julia Kiseleva\textsuperscript{2} and Maarten de Rijke\textsuperscript{1}

\textsuperscript{1}University of Amsterdam
\textsuperscript{2}Microsoft Research AI

{j.pei, derijke}@uva.nl, {arent.stienstra, julianakiseleva}@gmail.com

Abstract

Dialogue response selection is an important part of Task-oriented Dialogue Systems (TDSs); it aims to predict an appropriate response given a dialogue context. Obtaining key information from a complex, long dialogue context is challenging, especially when different sources of information are available, e.g., the user’s utterances, the system’s responses, and results retrieved from a knowledge base (KB). Previous work ignores the type of information source and merges sources for response selection. However, accounting for the source type may lead to remarkable differences in the quality of response selection. We propose the Source-aware Recurrent Entity Network (SEntNet), which is aware of different information sources for the response selection process. SEntNet achieves this by employing source-specific memories to exploit differences in the usage of words and syntactic structure from different information sources (user, system, and KB). Experimental results show that SEntNet obtains 91.0% accuracy on the Dialog bAbI dataset, outperforming prior work by 4.7%. On the DSTC2 dataset, SEntNet obtains an accuracy of 41.2%, beating source unaware recurrent entity networks by 2.4%.

1 Introduction

Task-oriented Dialogue Systems (TDSs) have attracted a lot of attention recently for their practical applications, e.g., booking flight tickets or scheduling meetings [Williams et al., 2017, Young et al., 2013]. Unlike open-ended dialogue systems [Zhao and Eskenazi, 2016], TDSs aim to assist users to achieve specific goals through multiple dialogue turns. Recently introduced end-to-end approaches for TDSs improve over traditional ones [Bordes and Weston, 2017, Chen et al., 2017a, Young et al., 2013] due to their ability to deal with global optimization, which facilitates easier adaptation to new domains [Chen et al., 2017a]. End-to-end approaches can be classified into two categories: response generation and response selection. We focus on response selection methods because they show more convincing performance than response generation ones [Eric et al., 2017, Wen et al., 2017].

Different information sources may contribute to the response selection process. It is known that the combination of specialized models may be more effective than a single general model [Dietterich, 2000]. Thus, we hypothesize that the combination of source-specific expert models outperforms a single general model in a response selection process. Let us consider an example of dialogue response selection for a five-turn dialogue shown in Figure 1. In turn 5 ($t = 5$), the system should give a recommendation in response to the user utterance “No, thanks!” This recommendation has to include an entity, i.e., the name of a restaurant (“Suzette”), that has not occurred yet in the dialogue history and, hence, has to come from the knowledge base (KB) that feeds the dialogue. So, a model that decides which response to select should prefer to consider the KB rather than the user utterances or past system responses so as to obtain the entity (“Suzette”) for its...
response. Awareness of the source of information is essential for dialogue language understanding [Chen et al., 2017b]. However, state-of-the-art response selection methods [Bordes and Weston, 2017, Liu and Perez, 2017, Sakai et al., 2017, Wu et al., 2018] are not explicitly aware of the different information sources that play a role in response selection.

We propose an end-to-end response selection model, named Source-aware Recurrent Entity Network (SEntNet) that is source-aware. As an extension of EntNet [Henaff et al., 2017], SEntNet provides dynamic long-term memory blocks to maintain and update latent concepts and attributes. SEntNet is designed to improve EntNet by introducing source-specific memories to exploit differences in the usage of words and syntactic structure in different information sources. Moreover, SEntNet employs a source-aware attention mechanism to dynamically capture the importance of different sources. SEntNet is able to choose responses more effectively by exploiting source-specific features derived from the dialogue history and KB. Furthermore, the computational costs implied by the use of extra memory modules can be offset by a parallel update mechanism design.

Our main contribution is the SEntNet model that significantly outperforms EntNet in terms of the turn-level accuracy of response selection. We carry out extensive experiments on the Dialog bAbl and modified DSTC2 datasets. Our experimental analysis shows the following properties of SEntNet:

- an ability to capture the semantics of dialogue context and learn word embeddings during the training process;
- a tolerance against sparse data, that is, it displays a stable performance, even when trained on a small amount of training data; and
- an ability to handle different degrees of lexical diversity, which may be affected by noise.

2 Problem Definition

TDS can be framed as a collection of search constraints that are denoted by slots \{s_1, \ldots, s_l\} and values \{v_1, \ldots, v_l\} that each slot can take. Result snippets can be obtained by issuing a symbolic query \texttt{api\_call} = (v_1, \ldots, v_l) to a KB that is generated by those search constraints. Each retrieved result is an entry from the KB that can be presented as a triple \texttt{(entity}_1, relation, \texttt{entity}_2). E.g., in the restaurant booking domain, such a triple could look like \texttt{(Noma, cuisine-type, Indian)} and the relations include rating, cuisine-type and location, etc.

The dialogue context is used for response selection. It consists of alternating utterances with three main sources, i.e., the user, the system and retrieved results from the KB. Formally, the dialogue context at turn \(t\) is defined as a tuple \((U_t, S_{t-1}, B_t)\) where:

- \(U_t = (u_1, u_2, \ldots, u_t)\) are user utterances, which are highlighted in blue in Figure 1;
- \(S_{t-1} = (s_1, s_2, \ldots, s_{t-1})\) are system responses, which are highlighted in green in Figure 1;
- \(B_t = (b_1^t, b_2^t, \ldots, b_n^t)\) is a sequence of \(\lambda\)-best retrieved results from an external KB, which are highlighted in yellow in Figure 1.

Therefore, we consider three sources of information \(S \in \{S_U, S_S, S_B\}\) for response selection.

Figure 1 shows an example of a dialogue turn where the system needs to select a restaurant suggestion satisfying the user’s needs. In this case, only two sources are useful: \(U\) because it contains user preferences and \(B\) because it has information about available restaurants.

We aim to learn a response selection model \(\psi\), parameterized by \(\Theta\), that predicts a candidate response \(s_t\) by taking as input a dialogue context \((U_t, S_{t-1}, B_t)\) and is able to decide which sources of information are most useful at turn \(t\):

\[
\psi_\Theta(U_t, S_{t-1}, B_t) \rightarrow s_t. 
\] (1)

3 Preliminaries: Recurrent Entity Networks

Unlike Memory Networks [Weston et al., 2014], EntNet is able to manage entities that are contained in KB triples to track the state of the dialogue. EntNet uses an attention mechanism in combination with Recurrent Neural Networks (RNNs) to store and retrieve memories in parallel rather than sequentially. EntNet’s functions depend on three modules described below.

Input module. An utterance representation is obtained by multiplying the embedding vector of constituent words with a positional mask and then averaging the results. Each utterance has an index \(i\) to present its temporal position in the sequential dialogue history; \(e_i \in \mathbb{R}^d\) represents the embedding vector of the \(i\)-th utterance.

Let \(w_x^j \in \mathbb{R}^d\) be the embedding vector of the \(x\)-th word of the \(i\)-th utterance, where the hyper-parameter \(d\) is the dimension of the embedding vector. The learnable parameter \(f_x \in \mathbb{R}^d\) is the mask that is multiplied with the word embedding vector at position \(x\). The embedding vector of the \(i\)-th utterance is then calculated as:

\[
e_i = \sum_x f_x \odot w_x^j \in \mathbb{R}^d.
\] (2)

Dynamic memory module. The state of entities is learned via memory blocks, each of which is a gated RNN that encodes the information of one entity. The memory module has \(m\) memory blocks and each block learns an embedding vector of the dialogue history with \(n\) utterances. The \(j\)-th memory block of the \(i\)-th utterance has an embedding vector of a slot \(k_j^i \in \mathbb{R}^d\) and a hidden state \(h_j^i \in \mathbb{R}^d\). The gate \(g_j^i \in \mathbb{R}^d\) determines how much information from the \(i\)-th utterance should influence the state of the \(j\)-th memory. The learnable matrices \(G \in \mathbb{R}^{d \times d}, V \in \mathbb{R}^{d \times d}\) and \(W \in \mathbb{R}^{d \times d}\) are shared among different blocks; \(\odot\) denotes the element-wise product; \(\sigma\) and \(\phi\) denote the sigmoid function and ReLU function, respectively. For the \(i\)-th utterance in a dialogue, the memory block of the \(j\)-th entity is updated by:

\[
g_j^i = \sigma(e_j^i \odot k_j^i - 1 + e_j^i \odot k_j^i - 1) \in \mathbb{R}^d
\] (3)

\[
\hat{h}_j^i = \phi(Gh_j^i - 1 + V k_j^i - 1 + W e_i) \in \mathbb{R}^d
\] (4)

\[
h_j^i = \frac{h_j^{i-1} + g_j^i \odot \hat{h}_j^i}{\|h_j^{i-1} + g_j^i \odot \hat{h}_j^i\|} \in \mathbb{R}^d.
\] (5)

The final hidden state \(h_j^i\) of the \(j\)-th memory block is the concatenation of the hidden states \(\{h_j^1, h_j^2, \ldots, h_j^n\}\). The state for each memory block is initialized with the slot, i.e., \(h_j^0 = k_j^i\).
Output module. The weighted sum of the final states \( \{ h_1, h_2, \ldots, h_m \} \) is used to predict the output, where the attention between those states and the representation of the query is used as the weights. Let \( q \in \mathbb{R}^d \) be the embedding vector of the user utterance \( u_t \) for the current turn \( t \). Also, \( z \in \mathbb{R}^d \) is the weighted sum over the previous final states, where \( p_j \) is the weight for the \( j \)-th state. The learnable matrix \( L \in \mathbb{R}^{c \times d} \) is used to map the intermediate representation to the output distribution, where \( c \) is the number of the candidate responses. The weight matrix \( H \in \mathbb{R}^{d \times d} \) is a learnable parameter. The intermediate prediction \( \hat{y} \in \mathbb{R}^c \) is a weighted sum between \( q \) and \( z \), normalized by the ReLU function \( \phi \). The output \( y \in \mathbb{R}^c \) is a distribution over the candidate responses, which can be calculated as follows:

\[
P_j = \frac{\exp(q^T h_j)}{\sum_k \exp(q^T h_k)} \in \mathbb{R}^1 \tag{6}
\]

\[
z = \sum_j h_j p_j \in \mathbb{R}^d \tag{7}
\]

\[
\hat{y} = L \phi(q + Hz) \in \mathbb{R}^c \tag{8}
\]

\[
y = \frac{\exp(\hat{y}_i)}{\sum_k \exp(\hat{y}_k)} \in \mathbb{R}^c. \tag{9}
\]

4 Source-Aware Recurrent Entity Networks

We propose the Source-aware Recurrent Entity Network (SEntNet), which aims to learn an intermediate representation for each source of information with a separately parameterized memory module; see Figure 2. The source-specific memory modules share the embedding matrix \( E \). The final state of every memory module is a weighted sum of the value vectors of the blocks in the module. The weights are calculated with attention between the query and the slots of the memory blocks. The output is then calculated by concatenating the attention weighted states of each memory module.

Input module. For a specific information source \( S \), each utterance has a new index \( i \) to represent its temporal position in the sequential history originating from the source \( S \). Let \( w^i_j \) be the word embedding and \( l^i_j \) be an optional item – the Part-Of-Speech (POS) tag embedding of the \( x \)-th word in the \( i \)-th utterance of the source \( S \); \( f_x \in \mathbb{R}^d \) is the mask that is multiplied with the word embedding at position \( x \). The embedding of the \( i \)-th utterance \( e_i(S) \) for source \( S \) is:

\[
e_i(S) = \sum_x f_x \odot w^i_x + l^i_x \in \mathbb{R}^d. \tag{10}
\]

Dynamic memory module. For the \( i \)-th utterance with a specific information source \( S \) in the dialogue, the memory block for the \( j \)-th entity is updated as the following equations:

\[
g^i_j(S) = \sigma(e^T_i(S) h^{i-1}_j(S) + e^T_i(S) h^{i-1}_j(S)) \in \mathbb{R}^d \tag{11}
\]

\[
\hat{h}^i_j(S) = \phi(G_S h^{i-1}_j(S) + V_S k^{i-1}_j(S) + W_S e_i(S)) \in \mathbb{R}^d \tag{12}
\]

\[
h^i_j(S) = \frac{h^{i-1}_j(S) + g^i_j(S) \odot \hat{h}^i_j(S)}{\|h^{i-1}_j(S) + g^i_j(S) \odot \hat{h}^i_j(S)\|} \in \mathbb{R}^d. \tag{13}
\]

The final hidden state \( h_j(S) \) of the \( j \)-th memory block is the concatenation of the hidden states \( \{ h^1_j(S), h^2_j(S), \ldots, h^n_j(S) \} \). The gate \( g^i_j(S) \) controls how much the \( i \)-th utterance of source \( S \) should contribute to the content of the \( j \)-th memory. \( U_S, V_S \) and \( W_S \) are trainable weight matrices that are separately parameterized among all the blocks.

Output module. Let \( q \in \mathbb{R}^d \) be the embedding of the user utterance \( u_t \) for the current turn \( t \). The output module is de-
Datasets and Evaluation. We evaluate SEntNet and EntNet’s performance based on the prediction of each of the experts. Here, we use concatenation to produce a single output vector : $z$. The dimensionality of the learnable matrix $H$ is changed to $d \times 3d$.

To sum up, SEntNet enhances EntNet by taking advantage of specialized predictions of “experts” that are modelled as source-specific memory modules. We used dialogue history information for the prediction of each expert. The dialogue history information can be taken into account when the list of restaurants retrieved from the KB are considered together. SEntNet is trained on the full training set using the Random strategy. We then obtain oracle embeddings that are optimized by all available training data. To answer RQ3, we train SEntNet based on various fractions of the datasets and test how SEntNet performs compared to the baselines even when limited training data is available. RQ4 introduces the baselines for response selection on the DSTC2 dataset [Liang and Yang, 2018].

EntNet. We reproduced EntNet, which was originally introduced for question answering and is reported to have strong reasoning abilities [Henaff et al., 2017].

(E2) To answer RQ2, we compare the turn-level accuracy of SEntNet with different pre-trained embeddings, i.e., Glove [Pennington et al., 2014], Paragram [Wieting et al., 2016] and NumberBatch [Speet et al., 2017], with three embedding initialization strategies:

1. Random strategy. The word embeddings are initialized by a zero-mean Gaussian distribution with a standard deviation of 0.1. They are optimized during training.
2. Fixed strategy. This initialization method is similar to the above, but the embeddings are not optimized during training. It tests whether the model requires an embedding space that encodes the semantics between words.
3. Oracle strategy. The model is trained on the full training set using the Random strategy. We then obtain oracle embeddings that are optimized by all available training data.

(E3) To answer RQ3, we train SEntNet based on various fractions of the datasets and test how SEntNet performs compared to the baselines even when limited training data is available.

(E4) To answer RQ4, we simulate the case when SEntNet is trained on a dataset with low lexical diversity. Specifically, we use SpaCy¹ to get the POS tag of each word. Then the embeddings a tag is added to that of the word and can be optimized in the learning process using the random strategy.

Training details. The training loss is measured by the log of the cross-entropy between the one-hot encoded golden label and the predicted output. Our objective is to minimize the loss over all of the $n$ samples with $c$ response candidates. All weights of the network are initialized with a zero-mean Gaussian distribution using a standard deviation of 0.1. A grid-search is performed to find the optimal hyper-parameter settings for each model. We use the Adam optimizer [Kingma and Ba, 2014] with a learning rate of 0.01 and decay frequency of 10. We set the memory block to 5, maximum epoch to 50, and the dimension of embeddings to 50. The gradient is restricted to be at most 40 to prevent gradient explosion; $l_2$ regularization is set to 0.001 and the dropout ratio to 0.5.

6 Results

6.1 E1: Comparison with baselines

Table 1 reports the average turn-level accuracy of SEntNet and the baselines on the dialog bAbI and mDSTC2 datasets.²

¹https://spacy.io/usage/linguistic-features
²We reimplemented TF-IDF, Q2A, EntNet and SEntNet. The scores for DQMemNN and HHCN were taken from Wu et al. [2018] and Liang and Yang [2018], respectively; unfortunately, the authors reported performance on only one dataset.
SEntNet model significantly outperforms all baseline models. Furthermore, the EntNet model significantly outperforms the simple baselines, i.e., TF-IDF and Q2A, by a large margin. Q2A model achieves a surprisingly high turn-level accuracy given the simplicity of the method. Furthermore, the large gap between the performance of SEntNet and the baseline models suggest that non-trivial relations between the dialogue history and responses are found. SEntNet is outperformed by HHCN on the mDSTC2 dataset. This is mainly because HHCN uses a stronger language model (i.e., a word-character RNN) and integrates a NN-based selection for domain knowledge.

Table 1: Comparison with baselines on the dialog bAbI and mDSTC2 datasets. The results are the best turn-level accuracy in 10 runs. Bold results indicate a statistically significant improvement over the strongest baseline (paired t-test, $p < 0.01$).

<table>
<thead>
<tr>
<th>Model</th>
<th>dialog bAbI</th>
<th>mDSTC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-IDF</td>
<td>0.040</td>
<td>0.030</td>
</tr>
<tr>
<td>Q2A</td>
<td>0.570</td>
<td>0.220</td>
</tr>
<tr>
<td>EntNet</td>
<td>0.850</td>
<td>0.388</td>
</tr>
<tr>
<td>DQModNN</td>
<td>0.863</td>
<td>–</td>
</tr>
<tr>
<td>HHCN</td>
<td>–</td>
<td>0.661</td>
</tr>
<tr>
<td>SEntNet</td>
<td><strong>0.910</strong></td>
<td>0.412</td>
</tr>
</tbody>
</table>

### 6.2 E2: The effect of embeddings

To assess the effect of word embeddings, we conduct experiments on the initialization strategies described in Section 5. Figure 3 shows the accuracy for different embedding spaces tested on the two datasets. For both datasets, the Oracle strategy outperforms the other strategies when trained on few example dialogues. The performance gain diminishes when the number of used training dialogues reaches its maximum. This indicates that there exists an embedding space that is more effective than randomly initializing embeddings and optimizing during training. Additionally, the random strategy outperforms the fixed strategy for both datasets; this indicates that the model can learn useful semantics during training.

For the dialog bAbI dataset, the random strategy has the best performance for any fraction of the dataset. This is not the case for the mDSTC2 dataset, where the optimal embedding strategy depends on the number of available training dialogues. The embedding strategies perform equally well when very few example dialogues (less than 200) are available. Pre-trained embeddings outperform the randomly initialized embeddings when more dialogues are available.

Pre-trained embeddings are more useful for the mDSTC2 dataset than the dialog bAbI dataset. This is as expected: the overlap in vocabulary between mDSTC2 and the pre-trained embeddings is much larger.

### 6.3 E3: The ability to handle sparse data

Figure 4 shows the results of the models we consider when trained on different fractions of the dialog bAbI and mDSTC2 datasets. SEntNet outperforms EntNet for any dialogue size. This further supports our hypothesis that SEntNet is a better TDS. The standard deviation of the results seems to show that the difference in performance is significant. It shows that SEntNet finds a meaningful relationship between the dialogue history and KB on the one hand and the response on the other hand, even when few training dialogues are available.

### 6.4 E4: The effect of lexical diversity

To investigate the effect of lexical diversity, we design a simulation as stated in Section 5. In this way, we can use POS information to generalize concrete words to abstract symbols and get a new synthetic dataset with lower diversity.

As the results in Table 2 show, SEntNet outperforms SEntNet with POS on the dialog bAbI dataset, while EntNet with POS reaches a similar accuracy as the regular EntNet. It indicates that SEntNet has the potential to handle different degrees of lexical diversity. This matters as it may help with adaptation for end-to-end TDSs. In the results on the mDSTC2 dataset we do not obtain significant differences; noise in the data may be a crucial factor behind this observation.

### 7 Related Work

There are two dominant paradigms for end-to-end dialogue systems: *generation* and *selection.*
Figure 4: Turn-level accuracy of SEntNet on both datasets, when trained with different volumes of training dialogues. Please note that the scales on the x-axes and y-axes differ.

**Generation.** In this paradigm, a response is generated word by word given the dialogue context. Recent improvements have been made by adding an attention mechanism [Li et al., 2016, Vinayals and Le, 2015] or by modeling the hierarchical structure of dialogues [Serban et al., 2016] based on sequential models like RNNs [Sordoni et al., 2015]. Early attempts [Vinayals and Le, 2015, Yao et al., 2015] to apply these approaches to TDSs neglect aggregate information from an external KB. This is problematic when the response contains out-of-dialogue entities. To address this issue, separate state tracking [Wen et al., 2017] and soft-lookup methods [Eric et al., 2017] have been adopted to interact with a KB.

**Selection.** In this paradigm, a response is selected from a list of candidate responses. Important improvements mainly rely on sequential models with attention mechanisms, especially on memory networks. State-of-the-art models [Bordes and Weston, 2017, Liu and Perez, 2017, Sakai et al., 2017, Wu et al., 2018] are built on top of a MEMN2N architecture [Sukhbaatar et al., 2015] to select a response, where delexicalization methods are introduced to enrich an entity with its generic type. This helps the TDS in finding exact matches between entities in the dialogue and the KB, and alleviates the out-of-vocabulary problem. However, it still explicitly relies on a handcrafted ontology. Recent models [Shin and Cha, 2019, Wu et al., 2017] evaluate EntNet [Henaff et al., 2017] on DSTC6 dataset [Perez et al., 2017]. The models only select the response from a small list of pre-selected candidates. Obviously, this does not reflect real-world conversational agents. Wu et al. [2018] enhance MEMN2N by dynamic query components originated from EntNet, so as to capture the sequential dependencies of dialogue utterances. However, their highest performance still relies on delexicalization methods. EntNet can be seen as a group of gated RNNs with shared parameters, but its hidden states are updated only when they receive new information related to their entities [Henaff et al., 2017]. In this way, latent concepts and attributes can be learned by its hidden states [Wu et al., 2018].

The key distinctions of our work compared to previous efforts are: we introduce a new architecture, SEntNet, that adds separate parameterized memory modules to exploit source-specific information.

**8 Conclusion**

We have proposed a dialogue response selection model, Source-aware Recurrent Entity Network (SEntNet), that is built on top of a memory network architecture and is able to select responses aware of source-specific history for end-to-end TDSs. Experimental results suggest that SEntNet consistently outperforms the baselines for end-to-end TDS. Optimizing embeddings while training SEntNet is found to be useful for end-to-end task performance. SEntNet is more tolerant of sparse data than baselines and has the potential to handle different degrees of lexical diversity.

One limitation of SEntNet is the increase of learnable parameters with introducing extra memory modules. However, the parallel update mechanism design inherited from EntNet can offset the use of the computational resources. This mechanism makes SEntNet scalable to real-world systems that have to deal with even more sources of information. In future work we plan to apply the source-aware context idea that underlies SEntNet to other variant memory networks.

**Acknowledgments.** This research was partially supported by Ahold Delhaize, the Association of Universities in the Netherlands (VSNU), the China Scholarship Council (CSC), and the Innovation Center for Artificial Intelligence (ICAI). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

Table 2: The effect of lexical diversity on SEntNet and EntNet, on the dialogue bAbI and mDSTC2 datasets. The results are the best turn-level accuracy in 10 runs. Bold results indicate a statistically significant improvement over the strongest baseline (paired t-test, \( p < 0.01 \)).

<table>
<thead>
<tr>
<th>Model</th>
<th>dialogue bAbI</th>
<th>mDSTC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>EntNet</td>
<td>0.850</td>
<td>0.388</td>
</tr>
<tr>
<td>EntNet + POS</td>
<td>0.850</td>
<td>0.398</td>
</tr>
<tr>
<td>SEntNet</td>
<td><strong>0.910</strong></td>
<td><strong>0.412</strong></td>
</tr>
<tr>
<td>SEntNet + POS</td>
<td>0.890</td>
<td>0.409</td>
</tr>
</tbody>
</table>
References


### A Notation

Table 3: Notation used in the paper.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Glossary</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbb{U}$</td>
<td>Set of user utterances</td>
</tr>
<tr>
<td>$\mathbb{S}$</td>
<td>Set of system responses</td>
</tr>
<tr>
<td>$\mathbb{B}$</td>
<td>Set of retrieved results</td>
</tr>
<tr>
<td>$u, s, b$</td>
<td>A user utterance, system response, retrieved result</td>
</tr>
<tr>
<td>$S$</td>
<td>Source of information</td>
</tr>
<tr>
<td>$k_l, v_l$</td>
<td>$l$-th slot $k_l$ and its corresponding value $v_l$</td>
</tr>
<tr>
<td>$\psi_{\Theta}$</td>
<td>Dialogue selection model parameterized by $\Theta$</td>
</tr>
<tr>
<td>$\mathbb{R}^d$</td>
<td>Vector space with $d$ dimension</td>
</tr>
<tr>
<td>$w_{ix}$</td>
<td>Embedding vector of $x$-th word of $i$-th utterance</td>
</tr>
<tr>
<td>$e_i$</td>
<td>Embedding vector of $i$-th utterance</td>
</tr>
<tr>
<td>$q$</td>
<td>Embedding vector of query $u_t$</td>
</tr>
<tr>
<td>$f_x$</td>
<td>Vector of learnable mask at position $x$</td>
</tr>
<tr>
<td>$\tilde{h}_i^j, h_i^j$</td>
<td>Vector of hidden state $\tilde{h}_i^j$ and its normalization $h_i^j$ of $j$-th entity in the $i$-th utterance</td>
</tr>
<tr>
<td>$h_j$</td>
<td>Vector of hidden state $h_j$ of $j$-th entity</td>
</tr>
<tr>
<td>$z$</td>
<td>Vector of sum of all hidden states</td>
</tr>
<tr>
<td>$y$</td>
<td>Predicted distribution of response candidates</td>
</tr>
<tr>
<td>$\hat{y}$</td>
<td>True distribution of response candidates</td>
</tr>
<tr>
<td>$c$</td>
<td>Number of all candidate responses</td>
</tr>
<tr>
<td>$d$</td>
<td>Number of embedding dimensions</td>
</tr>
<tr>
<td>$l$</td>
<td>Number of the required slots</td>
</tr>
<tr>
<td>$m$</td>
<td>Number of all memory blocks</td>
</tr>
<tr>
<td>$n$</td>
<td>Number of all utterances</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>Number of retrieved results</td>
</tr>
<tr>
<td>$t$</td>
<td>Index of the current dialogue turn</td>
</tr>
<tr>
<td>$E$</td>
<td>Matrix of word embeddings for $\mathbb{U}$</td>
</tr>
</tbody>
</table>