Model	Embedding Average Score	Embedding Greedy Score	Embedding Extrema Score
Seq2seq	0.4733 +/- 0.00010	0.4728 +/- 0.00005	0.3067 +/- 0.00009
Seq2seq (last utte)	0.5969 +/- 0.00009	0.4751 +/- 0.00005	0.3185 +/- 0.00007
HRED	0.5151 +/- 0.00012	0.9053 +/- 0.00011	0.3252 +/- 0.00011
GSN No-speaker (1-iter)	0.7224 +/- 0.00004	1.0209 +/- 0.00008	0.5624 +/- 0.00005
GSN No-speaker (2-iter)	0.7318 +/- 0.00005	1.1199 +/- 0.00010	0.5796 +/- 0.00005
GSN No-speaker (3-iter)	0.7244 +/- 0.00004	0.9495 +/- 0.00006	0.5630 +/- 0.00004
GSN W-speaker (1-iter)	0.7133 +/- 0.00005	1.1193 +/- 0.00011	0.5860 +/- 0.00005
GSN W-speaker (2-iter)	0.7695 +/- 0.00004	1.0400 +/- 0.00006	0.6508 +/- 0.00005
GSN W-speaker (3-iter)	0.7473 +/- 0.00006	1.1610 +/- 0.00012	0.5890 +/- 0.00006

Table 5: Experimental results measured using three embedding-based metrics. Each score is followed by its standard deviation.

Algorithm 1: Neural Graph Model

Input: 1) a session; 2) alpha α; 3) iteration no. N
Output: Generated response
Represent the input session with state $\mathbf{S}$ using Eq. 2;
Build state matrix $\mathbb{S}$ with $\mathbf{S}$ (Section 3.3);
for $i = 0$; $i < N$; $i += 1$ **do**
  Compute $\Delta\mathbf{E}$ and $\Delta\mathbf{U}$ using Eq. 12;
  // Backward Information Flow
  Update $\mathbb{S}$ using Eqs. 10 and 11;
end
for $i = 0$; $i < N$; $i += 1$ **do**
  Compute $\Delta\mathbf{E}$ and $\Delta\mathbf{U}$ using Eq. 9;
  // Forward Information Flow
  Update $\mathbb{S}$ using Eqs. 10 and 11;
end
Decode/generate response using Eq. 13;
return *The generated response*;

A More Details about Experiments

A.1 Human Evaluation Instructions

Human judges were asked to rate the generated responses and the ground truth (human responses) by given these instructions (initial training and test runs with the judges were also conducted):

a) We evaluate based on naturalness, which includes 1) grammaticality, 2) fluency, and 3) rationality. To make the judgement process easier and to reduce the bias for each aspect, we use binary scores: good or bad. This means a response will be given three binary scores (good or bad) by each judge.

b) If a response gets three good ratings by a judge, 4 is awarded to the response from the judge. If it gets two goods and one bad, 3 is awarded. If it gets two bads and one good, it gets 2. If a response gets three bads, 1 is given. For those responses with the score of 1, the judge is further requested to separate them into two groups if one group is clearly better than the other. Each response in the better group gets the final score of 1, and each response in the other group gets the final score of 0.

A.2 Detailed Results in Embedding-based Metrics for Automated Evaluation

Experimental results measured using three embedding-based metrics under different settings for different methods with sequential data and graph data are given in Table 5. 'Seq2seq (last utte)' model is trained by only using the last utterance before the final response as the input (all utterances before it are ignored). 'n-iter' means the results are obtained after n iterations. 'No-speaker' is our GSN model without speaker information flow while 'W-speaker' has it.

A.3 Detailed Comparison with GCN

Prior to the proposed GSN, Graph Convolutional Networks (GCN) [Kipf and Welling, 2017] encode the graph structure directly using a neural network and train on a supervised target for all nodes with labels, thereby avoiding explicit graph-based regularization in the loss function. The main difference between GSN and GCN is the information integration mechanism. GSN uses gated updating (calculating the update directly) while GCN uses propagation rules (estimating the output after updating). GSN uses dynamic iteration to consider long distance dependency while GCN uses multi-layers. We tested GCN by using a 3-layers GCN (corresponding to 3 iterations in GSN; 1 and 2-layers are poorer. There is a little change from 4-layers for GCN (we use only one layer). The GCN code is from https://github.com/tkipf/gcn. The experimental results for the six metrics in Table 2 are 11.86, 4.33, 2.39, 1.54, 4.36, 9.86, which are poorer than GSN.

B Case Study

Table 6 presents some sample responses generated by our model and the baselines. From the first example, we see that the responses generated by our model are more consistent with the given utterance. The utterance in the first example (on the left in Table 6) is asking for help, not offering an answer. However, HRED mistakenly takes it as a useful answer and gives a response "that's what i'm looking for", which may not be appropriate. The seq2seq model provides a meaningless utterance "try that. i'm not sure how to do it", which addresses nothing. In contrast to these methods, our model tries to give some useful suggestions, e.g., "you can mount it with the mount command." and "what is the output of: uname -r?", etc, which we believe are more effective.

In the second column (Table 6), based on utterances 1, 2 and 3, the speaker tried to run a 64-bit application but the application failed with an error message ".txt files unknown". In this case, our models generate meaningful responses from

Utterance 1: my memory card for my phone won't allow me to delete files or put files on it, it says "read only file system". **Human**: have you tried—cd filepath ls -l? **No-speaker (1-iter)**: please don't flood, use url to paste, don't use enter as punctuation. **No-speaker (3-iter)**: what is the output of: uname -r? **W-speaker (1-iter)**: does it show up in the terminal? **W-speaker (3-iter)**: you can mount it with the mount command. **Seq2seq**: try that. i'm not sure how to do it. **HRED**: that's what i'm looking for.	**SESSION**: **utterance 1:** i'm running on amd athlon 64 4000. **utterance 2:** then it will support 64bit if you wish to use it. **utterance 3:** i understand that, but how i make .txt files not unknown? **utterance 4:** should i use it? whats the difference? **RESPONSES**: **No-speaker (3-iter)**: i'm not sure what you're looking for. **W-speaker (1-iter)**: in a terminal, type "sudo apt-get update" **W-speaker (3-iter)**: you need to install it from source. **Seq2seq**: i'm not sure what you ' re trying to do. **HRED**: you ' ll need to find out what you ' re doing .

Table 6: Sample responses generated by human, GSN, and baselines. All the responses in column 1 are for utterance 1. Column 2 is a session with the graph structure in Figure 5.

the training corpus: giving a suggestion to update the system (W-speaker 1-iter); advising the speaker to try another way to install the software (W-speaker 3-iter). Thus, our models (with the speaker information flow) generate responses relevant to the entire session and give substantive advice for the raised questions. These responses are more appropriate and preferred from the human perspective.