Predicting Success in Goal-Driven Human-Human Dialogues

Michael Noseworthy, Jackie Chi Kit Cheung, Joelle Pineau
School of Computer Science
McGill University
michael.noseworthy@mail.mcgill.ca
{jcheung,jpineau}@cs.mcgill.ca

Abstract

In goal-driven dialogue systems, success is often defined based on a structured definition of the goal. This requires that the dialogue system be constrained to handle a specific class of goals and that there be a mechanism to measure success with respect to that goal. However, in many human-human dialogues the diversity of goals makes it infeasible to define success in such a way. To address this scenario, we consider the task of automatically predicting success in goal-driven human-human dialogues using only the information communicated between participants in the form of text. We build a dataset from *stackoverflow.com* which consists of exchanges between two users in the technical domain where ground-truth success labels are available. We then propose a turn-based hierarchical neural network model that can be used to predict success without requiring a structured goal definition. We show this model outperforms rule-based heuristics and other baselines as it is able to detect patterns over the course of a dialogue and capture notions such as gratitude.

1 Introduction

In this paper, we investigate goal-driven dialogues in large open-ended domains where one participant engages in a conversation with another participant in order to gain information or complete some task. Such dialogues are common in online communication channels where users help each other complete tasks with various requirements. For instance, many corporations have online chat systems where users can talk to a representative, and there are countless online forums (both technical and non-technical) where people go for help.

Current dialogue agents learn to assist users to complete tasks in relatively constrained domains such as restaurant reservation booking (see Table 1 of Serban et al., 2015 for a list of these domains). In such domains, agents can measure success by referring to a structured goal definition or ontology and learn to maximize this score (Young et al., 2013). However, in less-constrained domains, success can be difficult to define as it is often dependent on the specific dialogue and participants.

One difficulty arises when participants enter a conversation with intrinsically different goals which we cannot anticipate in advance. For example, on *stackoverflow*, a popular forum for programming-related help, users can ask for help fixing a bug (in which case success occurs when the bug is resolved), or ask for a recommendation (in which case success occurs when the user is satisfied with a recommendation). On top of this, different users may have differing definitions of success (e.g., a novice may require more information than an expert).

The aforementioned difficulties suggest that the definition of success is highly specific to the user who initiates the dialogue. Even in constrained domains it has been observed that a user's perception of success is more indicative of user satisfaction than an objective measure (Walker et al., 2000; Williams and Young, 2004). Thus, we aim to let the original participant be the judge of success and build models that can predict success based on information communicated rather than enforcing a rigorous definition in our models.

An impediment in building models that predict success (or interactive agents) in these domains is the lack of success labels in current datasets (Kim et al., 2010; Lowe et al., 2015). These labels can be difficult to collect as forums often do not pro-

Proceedings of the SIGDIAL 2017 Conference, pages 253–262,
Saarbrücken, Germany, 15-17 August 2017.

vide any structured process of indicating whether a problem was solved or not. Our model is trained to predict success in these interactions using only the dialogue text, which can then be used as automatic feedback to improve the quality of the dialogues and enable automatic dialogue agents to learn from large, previously unlabeled corpora.

We address the challenge of predicting success in goal-driven human-human dialogues with three contributions. First, we present a new dataset of human-human goal-driven dialogues in the technical domain.[1] The use of human-human dialogues allows our dataset to reach a size needed to work in and be representative of large domains. We focus on dialogues from *stackoverflow.com*, where we have success labels available. These dialogues consist of one participant asking a programming-related question and other participants interacting with them to come up with a solution. This dataset will allow the community to work in an open-ended domain with success labels.

Our second contribution consists of an investigation of new models to predict success using only the raw text of the dialogue history. Our most successful model is a turn-based hierarchical recurrent neural network (H-RNN). This model is inspired by the observation that dialogues consist of multi-level sequences. At the higher-level, we have a sequence of turns, which is commonly abstracted as a dialogue act, or intent (Traum and Hinkelman, 2011). For each turn, we also have a lower-level sequence of words which are a natural language realization of the dialogue act. We show that the H-RNN outperforms alternative models, and in particular can capture the semantics of a user expressing their gratitude.

Our final contribution is an analysis of the salient features for success prediction. We show that our models' performances significantly increase when they explicitly model the entire dialogue history (and learn more complicated indicators of success along with gratitude). Although our models only use each turn in their raw text form (as opposed to the dialogue act type such as *Confirmation* or *Rejection*), they implicitly benefit from this natural structure that arises in dialogue.

[1] Available at https://mike-n-7.github.io/stackoverflow

2 Related Work

There has been much work on automatically evaluating dialogue success. This work has largely focused on small domains where one can manually define every task the system or participants can perform and what it means to complete the task.

Success, as defined by task completion, is easier to evaluate in traditional dialogue systems which have been highly scripted. These systems are designed for restricted domains in which the relevant ontology and language generation prompts or templates have been specified. Such systems include the *Let's Go* Pittsburgh Bus System (Raux et al., 2003), the Cambridge Restaurant System (Thomson and Young, 2010), and the *ELVIS* email assistant (Walker et al., 1998). Scaling these systems to larger domains, such as those found in online forums, is difficult because expanding their ontologies becomes infeasible.

The PARADISE framework (Walker et al., 1997) was proposed to automatically evaluate dialogues where the quality of a dialogue can be seen to consist of task success and costs such as the dialogue length. Here, task success has a rigid definition, where for each dialogue system using this framework, the designer must specify what attributes need to be communicated by the system to achieve a goal. This definition makes it clear what success looks like; however, it is not clear how to apply PARADISE to open-ended human-human dialogues, where each dialogue could have a different goal that we cannot anticipate before the conversation, or to large domains, where it is not feasible to design such a reward.

Instead of requiring the designer of a dialogue system to specify what information needs to be communicated between users, work has been done that tries to learn this. Su et al. (2015) propose neural network models that operate on dialogue acts to learn what success means in a constrained domain with knowledge of the true goal. A limitation to this work is that it requires a domain specific feature vector. We consider domains where it is infeasible to acquire such features and instead work directly from text input. Vandyke et al. (2015) extend this model to work in unseen domains but still require we parse our input into slots.

Recently, *Amazon Mechanical Turk* (AMT) has been used to crowd-source the success of a dialogue (Yang et al., 2010; Jurcıcek et al., 2011). The first method presents the transcript of a dia-

logue to a worker and asks them to fill out a questionnaire rating success. The latter has users interact with a dialogue system by giving them a goal and asking them to evaluate the dialogue after its completion. It is unclear how to extend these methods to an open-ended domain as AMT workers are unlikely to have enough expertise to evaluate success or start a conversation on each possible dialogue topic (Lowe et al., 2016). Furthermore, although AMT is both faster and cheaper than running in-person experiments, we search for an automatic evaluation method with near zero costs.

Work has been done that aims to make data from online forums more accessible to both other users and computer models. The Ubuntu dataset (Lowe et al., 2015) was proposed for dialogue modeling from the *Ubuntu Internet Relay Chat* channel and the CNET (Kim et al., 2010) dataset was proposed to learn dialogue structure from these often unstructured forums. We build on this work by offering a way to provide success labels with this type of data.

Complementary work has been done that shares a common goal of extending dialogue systems to open-ended domains. One area of research focuses on extending intent detection to open-ended domains, where an intent is defined as an action a user wants to perform in the dialogue (e.g. request information or make a reservation). These methods look for semantic similarity with existing intents (Chen et al., 2016) or exploit the structure of knowledge graphs (El-Kahky et al., 2014). Another line of research is on extending natural language generation to multiple or open-ended domains. Domain adaptation techniques have proved useful to generate responses for unseen dialogues (Wen et al., 2016).

The *Community Question Answering* (CQA) literature has investigated predicting the success of answers posed on online forums but typically in a different scenario. Whereas we are interested with predicting the success of a single question and answer, CQA often looks to predict user satisfaction based on several answers (Liu et al., 2008). Furthermore, we restrict our models to only consider the text of the questions, answers, and comments (we do not include any information about users or votes). Kim and Oh (2009) observe that comments often contain useful information for predicting success. Our work investigates this hypothesis.

Question
User A (Lee): I accidentally closed the Stack Trace window in the Visual Studio 2008 debugger. How do I redisplay this window?
Answer
User B (Brian): While debugging: Debug\Windows\Call stack
Comment
User A (Lee): Thanks, I don't know how I overlooked it.

Figure 1: Example dialogue from our *stackoverflow* dataset. Post taken from *http://stackoverflow.com/questions/612123/redisplay-stack-trace-window*.

3 Dataset

Our first contribution is a dataset from *stackoverflow.com* curated to allow training a success prediction function. *Stackoverflow* is a community-based website where users post programming-related questions and other users can respond with answers. Multiple users can provide answers to the same question and users can comment on any potential answer. This format allows us to extract dialogues from the website that consist of the aforementioned exchange. To limit the complexity, we restrict our dataset to dialogues between two users. These dialogues are goal-driven as each is an attempt to solve the question initially posted. Figure 1 is an example of a question, answer, and comment found on *stackoverflow*.

In addition, the user who posed a question can mark an answer as accepted if that answer successfully solved their problem. Only the original user can mark an answer as accepted and they can only mark a single answer. Any user can vote ($+1$ or -1) on answers based on how helpful they are.

Our goal when creating the dataset is to collect a label for dialogue success that is representative of the original user's goal. Note that their true goal may differ slightly from what they express in their question (for example, due to a poor explanation). Votes have a high variance that depend on how popular a question is and the difficulty of the question. Furthermore, users who vote for an answer may not be experiencing the exact same problem as the original user. For this reason, we do not use the vote count alone to judge dialogue success (only to ensure a high quality dataset as described below).

3.1 Collection

In our work, we are concerned with dialogues that consist of only two participants and are complete dialogues in the sense that either the initial user's question was accepted or rejected. We define an accepted dialogue to be one in which the original user's question was successfully answered. Similarly, a rejected dialogue is one in which the dialogue did not solve the original question. Because of the open-ended nature of *stackoverflow*, many posts do not conform to these requirements and we must perform filtering to collect a high quality dataset.

We use the *stackoverflow* posts from the *Stack Exchange Data Dump*.[2] A candidate dialogue consists of a question, answer, and series of comments. In order to be considered, this series of exchanges must take place only between two unique users. We do not consider dialogues where the comments consist of other users.[3] We require at least one comment so that the dialogue extends beyond just question answering. For this reason, all dialogues are at least three turns long where a single turn is one or more utterances by a single user.

On *stackoverflow*, it is possible for a user to edit their posts after seeing answers or comments. We exclude any dialogue where the question or answer was edited to ensure a linear structure.

To ensure a question was either accepted or rejected by the original user, we use information from *stackoverflow* outside of the text. As mentioned above, there are two methods to express satisfaction with an answer. For a post to be accepted, we require both of the following to hold:

1. The answer be marked as accepted by the user who posed the question.
2. There be a strictly positive score from the votes attributed to the answer.

On the contrary, for an answer to be rejected, we require:

3. No answer associated with the question be marked as accepted.
4. There be a non-positive score from the votes assigned to the answer.

[2] http://archive.org/details/stackexchange

[3] Note that for questions with multiple answers, we treat each answer as a new dialogue.

# Accepted Dialogues	667,777
# Rejected Dialogues	297,145
Avg. # of Turns	4
Avg. Question Length	110 words
Avg. Answer Length	60 words
Avg. Comment Length	31 words

Table 1: Statistics of the *stackoverflow.com* dataset. *Accepted* and *Rejected* are the two class labels.

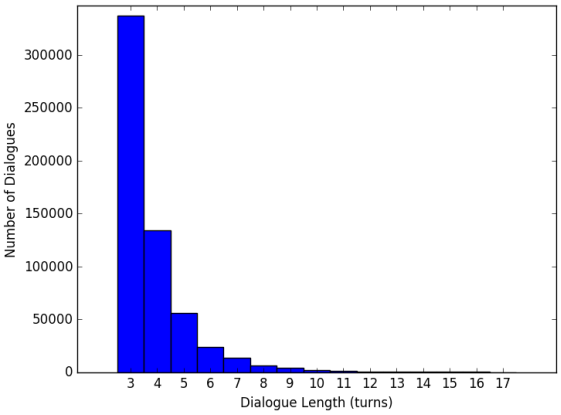

Figure 2: Turn distribution for the *stackoverflow.com* dataset.

Point (3) requires that no other answer be marked as accepted, in order to prevent less popular but nevertheless correct answers from being labelled as rejected. This situation often occurs because *stackoverflow* only allows one answer to be accepted by the user.

Note that (2) and (4) act as a form of validation as it requires the user and crowd be in agreement about the success of an answer. We performed label validation by blindly labelling randomly sampled dialogues from our dataset.

3.2 Statistics

After filtering through the *Data Dump* as described in the previous section, we have a dataset with 964,922 dialogues (reduced from 7,990,787 unfiltered posts). More statistics can be found in Table 1.

It is worth noting that the first and second turns will often be longer in these dialogues due to their question-answer nature. These types of dialogues are of particular interest in the tech-support (Kim et al., 2010; Lowe et al., 2015) and e-mail (Ulrich

et al., 2008) domains where the bulk of the information is communicated in the first few turns (e.g. to explain a problem). As we will see, most of our models take advantage of features that appear in the comments which are more characteristic of traditional dialogues.

Figure 2 shows the distribution of number of turns in a dialogue. A single turn consists or a group of sentences by one user. For example, a question will be a single turn, as will an answer. Consecutive comments by the same user are considered as a single turn.

3.3 Preprocessing

Before continuing with any experiments, we preprocess each question, answer, and comment by removing HTML tags, and replacing numbers and code with generic tags.

4 Recurrent Neural Network Models for Dialogue Success Prediction

Presented with a dialogue that consists of a series of turns (question, answer, and comments), we would like to classify whether that dialogue was successful or not. We will denote the true labels as y and the model predictions as $success$. We refer to successful dialogues as *accepted* ($y = 1$) and unsuccessful dialogues as *rejected* ($y = 0$) as previously defined.

Formally, our input is the sequence of turns, $d = t_1, \ldots, t_n$, and the sequence of words within each turn, $t_i = w_{i,1}, \ldots, w_{i,n}$. The first turn will be a question, q, the second turn an answer, a, and the remaining turns a series of comments, $c_1, \ldots, c_n$.

We consider two recurrent neural network (RNN) models: a flat one that operates over the concatenated sequence of words from each turn, and a hierarchical one that explicitly models multi-level sequences.

No further information from *stackoverflow* is included in the models such as a user's reputation, tags, or the number of views. We want our models to be usable in other scenarios where this data may not be available. Thus, the only features the models have to work with are text from the dialogues.

A motivating example behind using RNNs is their ability to predict complex non-linear discourse features. For example, consider the following comments:

Rejected: *Thanks for the advice. I tried this change, but I am still encountering the same error.*

Accepted: *Hmm, I thought I already tried that but there probably were more errors in the regex at that time. It did the trick, thanks!*

Both these examples require reasoning across the complete utterance. A model that could capture longer dependencies within a discourse would be useful for differentiating these two examples.

4.1 Flat Recurrent Neural Network

The Flat RNN works by first converting each word of a dialogue into its word embedding. After seeing each word embedding, the RNN updates its hidden state. We insert a special token, $\langle t \rangle$ (with its own embedding), to denote the separation between turns of the dialogue. At the end of the dialogue, the RNN makes a prediction using a logistic regression unit on the final hidden state of the network. This allows us to learn discourse features beyond bag-of-words (BOW) as we maintain word ordering. See Figure 3(a) for the architecture.

In our implementation, we use *Long Short Term Memory* (LSTM) units (Hochreiter and Schmidhuber, 1997) to account for long-term dependencies. We use *Theano* (Bastien et al., 2012) and pretrained *GloVe* word embeddings (Pennington et al., 2014). Optimization is done using the *ADAM* optimizer (Kingma and Ba, 2014) to minimize cross-entropy between the model predictions, $success$, and the actual success labels, y.

4.2 Turn-Based Hierarchical Recurrent Neural Network

In this model, we extend the Flat RNN to model the natural hierarchy that occurs in dialogues. This allows our model to separate the flow of content throughout the dialogue from the natural language realization of each turn. Our model is similar to the encoder models used in previous work (Sordoni et al., 2015; Li et al., 2015).

Similar to the Flat RNN, each word is projected into its vectorized word embedding. Then for each turn t_i, we feed its word embeddings through the same RNN (the turn-level RNN) which outputs an encoded version of that turn, $t_{en,i}$. We feed all these encoded turn vectors into a higher-level RNN (the dialogue-level RNN) which takes into account the context of the dialogue.

We also use *LSTM* units (Hochreiter and Schmidhuber, 1997) with *GloVe* embeddings (Pennington et al., 2014). Refer to Figure 3 (b) for the model architecture.

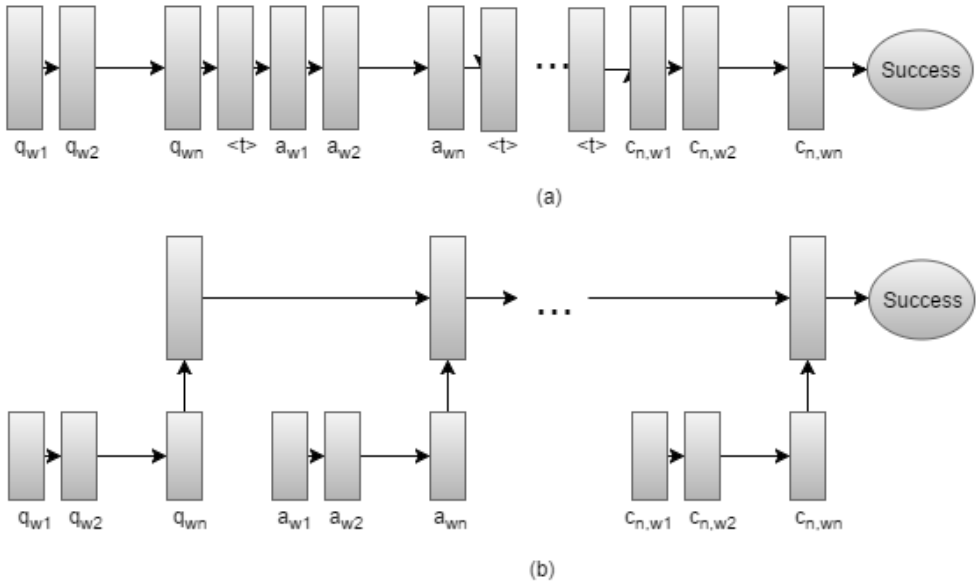

Figure 3: Flat (a) and Turn-Based Hierarchical (b) RNN models including features for a full dialogue.

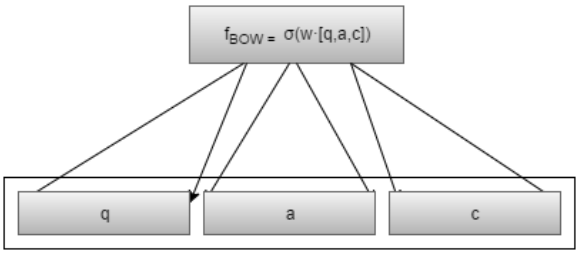

Figure 4: Logistic Regression BOW model including question, answer, and comment features.

5 Experiments

5.1 Baselines

We compare our two neural network models to several baselines ranging in complexity.

5.1.1 Majority Class

As our dataset suffers from class imbalance, we consider a majority class model which always predicts accepted.

5.1.2 Thanks Heuristic

The "Thanks" baseline operates on the intuition that users who ask a question will express gratitude for accepted answers in terms of thanking the user in a comment.

This method simply looks at the last comment, c_{-1}, by the user who posed the question and looks for the appearance of the word "thanks" or common variations of that word (we denote these words by the set $\mathbb{TH} = \{thx, thanks, ty, thankyou, tx\}$). Our classification rule is:

$$f_{baseline}(c_{-1}) = \mathbb{1}_{\mathbb{TH}}(c_{-1}) \quad (1)$$

5.1.3 Logistic Regression Classifier

We extend the "Thanks" baseline by considering the bag-of-word (BOW) vectors for each turn, and learning their respective weights when classifying a dialogue. In this model, we represent the dialogue as three concatenated BOW-vectors for the question, q, answer, a, and the sum of the comments, c. Together, these make up an input vector that is fed to a logistic regression classifier:

$$f_{BOW} = \sigma(w \cdot [q, a, c]) \quad (2)$$

We learn the parameters by minimizing cross-entropy. See Figure 4 for a depiction of this model.

5.2 Evaluation

We performed multiple experiments to gain intuition about what our models are learning and their performance at predicting dialogue success. We divided our dataset into training, validation, and testing sets using a $60\%/20\%/20\%$ split (there is equal class imbalance across sets). We present precision, recall, and F1 metrics for each class.

For the RNN models, we used the cross-validation set to optimize the model parameters. For the Flat-RNN this resulted in word embeddings of dimension 50 and hidden states of dimension 256. For the hierarchical model, we used

	Accepted			Rejected		
Model	Precision	Recall	F1	Precision	Recall	F1
Majority	69.20 ± 0.20	100 ± 0.0	81.66 ± 0.14	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
Thanks	82.45 ± 0.20	73.51 ± 0.22	77.72 ± 0.17	52.13 ± 0.34	64.83 ± 0.36	57.79 ± 0.30
LR	81.95 ± 0.19	89.57 ± 0.17	85.59 ± 0.14	70.38 ± 0.43	55.68 ± 0.39	62.17 ± 0.35
Flat RNN	**87.08 ± 0.18**	87.39 ± 0.18	87.23 ± 0.14	71.42 ± 0.36	**70.85 ± 0.37**	71.13 ± 0.28
H-RNN	**87.28 ± 0.17**	**90.06 ± 0.17**	**88.65 ± 0.13**	**75.95 ± 0.35**	**70.51 ± 0.37**	**73.13 ± 0.31**

Table 2: 95% confidence intervals for Precision, Recall, and F1 for both classes for models trained using the entire dialogue. The highest metrics are bolded.

word embeddings of dimension 50, turn-level hidden states of dimension 100, and dialogue-level hidden states of dimension 200.

We can see the performance of various models and feature sets in Table 2. Confidence intervals were calculated using the bootstrap method (Efron and Tibshirani, 1994).

5.3 Results

We start with a comparison of the various models. From Table 2, we see that our RNN models, which can capture more complex dependencies, have the best F1-scores for each class. Their performances exceed that of the Logistic Regression model, the strongest baseline.

It is also interesting to note that the Turn-Based Hierarchical RNN has significantly higher F1-scores than the Flat RNN. This suggests that this model can better represent both a turn, and an entire dialogue. By explicitly building in the structure to represent a turn, we allow the model to learn the important information in a single turn in regards to predicting success. The lower-level RNN can learn the semantics of a single turn which leaves the higher-level RNN to focus solely on the aspects of each turn relevant to predicting success. We go on to examine this model further in the next section.

6 Discussion

6.1 Turn-Embedding Analysis

The Turn-Based Hierarchical RNN model allows us to inspect the turn embeddings through the turn-level RNN. We extract the final hidden layer embedding from this lower-level RNN which represents the last fully-encoded turn of a dialogue. These turn-vectors represent the part of the turn that is relevant to predicting dialogue success (as the model was optimized for this). We then use t-SNE (Maaten and Hinton, 2008) to visualize these embeddings for the last comment by the initial user in two dimensions (see Figure 5). Here the circles denote a successful dialogue whereas the triangles denote a rejected one.

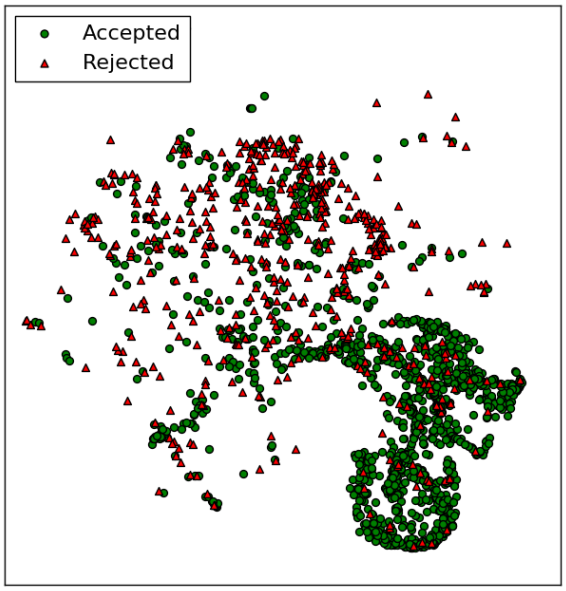

Figure 5: t-SNE visualization of the embedding of the last comment from the initial user from the H-RNN model.

We see a cluster in the lower right of the visualization. In Table 3, we show examples sampled at random from the cluster, and from elsewhere in the visualization. We can see the cluster represents ways for the user to show their gratitude. This supports the hypothesis that the H-RNN model is picking up on various ways for a user to express their satisfaction with a proposed answer.

By incorporating a hierarchical structure the model was able to learn a useful embedding of a given turn before incorporating information from previous turns. We can see the turn-level RNN as a way of extracting the information relevant to the success prediction task from its natural language representation.

Lower Right Cluster	Elsewhere
thank you! this is perfect and youre my bestfriend.	for bordering the cells of a table as <code> do try the following. add a css class for the <code>. let say the class name be <code>. then add the style as below
ah alright thanks for the quick reply	selenium webdriver code to reply a mail in gmail. i tried writing code for replying a mail in gmail but was trapped in between i want to perform below task open ...
really great helpful!! thanks a lot	make an absolutely positioned div stretch to <num> of the document height with no javascript. is there any neat cssonly way to make an absolutely positioned div element stretch ...
actually it worked. i missed the whereraw at first read. thanks!	the viewpager is for going between detail views. i want a custom action particularly to hide the list view.

Table 3: Example comments from the lower right cluster and everywhere else in the t-SNE plot. Comments are sampled at random from their respective clusters.

	Accepted			Rejected		
Model	Precision	Recall	F1	Precision	Recall	F1
LR (c_{-1})	79.86 ± 0.19	89.71 ± 0.17	84.50 ± 0.14	68.00 ± 0.46	49.13 ± 0.39	57.05 ± 0.36
LR (d_{-1})	73.07 ± 0.21	92.39 ± 0.14	81.60 ± 0.15	57.85 ± 0.64	23.49 ± 0.34	33.41 ± 0.40
LR (d)	81.95 ± 0.19	89.57 ± 0.17	85.59 ± 0.14	70.38 ± 0.43	55.68 ± 0.39	62.17 ± 0.35
RNN (c_{-1})	85.48 ± 0.18	87.43 ± 0.19	86.45 ± 0.13	70.23 ± 0.39	66.62 ± 0.39	68.38 ± 0.31
RNN (d_{-1})	69.23 ± 0.20	100.0 ± 0.0	81.81 ± 0.14	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
RNN (d)	87.08 ± 0.18	87.39 ± 0.18	87.23 ± 0.14	71.42 ± 0.36	70.85 ± 0.37	71.13 ± 0.28
H-RNN (c_{-1})	85.48 ± 0.18	87.43 ± 0.19	86.45 ± 0.13	70.23 ± 0.39	66.62 ± 0.39	68.38 ± 0.31
H-RNN (d_{-1})	74.11 ± 0.22	93.75 ± 0.12	82.79 ± 0.16	65.30 ± 0.60	26.43 ± 0.34	37.63 ± 0.41
H-RNN (d)	87.28 ± 0.17	90.06 ± 0.17	88.65 ± 0.13	75.95 ± 0.35	70.51 ± 0.37	73.13 ± 0.31

Table 4: 95 % confidence intervals for Precision, Recall, and F1 for models trained using subsets of the dialogue that include or exclude the last turn by the initial user.

6.2 Feature Ablation

We now show that our models exploit features throughout the entire dialogue history to make their predictions. We define feature sets based on whether or not the dialogue contains the last comment by the initial user. We are interested in this comment in particular as it is often the comment where a user will express their satisfaction with the proposed answer. In human-human dialogues, people are generally polite, even if an answer wasn't helpful. It is important for our models to learn the difference between a true expression of gratitude, and that of just being polite - which the baseline methods fail to do.

We can define a dialogue as $d = q, a, c_1, \ldots, c_n$ and let c_{-1} be the last comment by the user who asked the question. Then we will let d_{-1} be the dialogue with c_{-1} removed.

For each model, we re-calculate the above metrics using just c_{-1} or d_{-1} as features. The Hierarchical RNN reduces to the Flat RNN when using c_{-1} features. The results can be seen in Table 4 (we include results from Table 2 for comparison).

We see that the models that utilize the entire dialogue outperform the respective models that use just the last comment in F1-score. This suggests that these models can pick up on more complicated indicators of success than just gratitude such as whether an answer was irrelevant or a question was ill-posed.

It is worth noting that the models that use just the last comment still significantly outperform the baselines (Table 2). We can see the importance of the last comment by observing that the Thanks Heuristic has a higher F1 score for the Rejected class than models that do not include the last turn by the initial user (d_{-1}).

When removing the last comment (going from d to d_{-1}), the models see a drop in precision for the *accepted* class but a rise in recall. This is likely a result of removing discriminative features (c_{-1}) which causes the model to predict the majority class more often (we see that the performance for the *rejected* class greatly drops). The Flat RNN predicts all answers as accepted when

the last comment is removed.

Removing the last comment makes the problem more difficult as we no longer use the user's expression of gratitude. In this case, the hierarchical model improves upon the Logistic Regression and Flat RNN models. This can potentially be because it tries to model different dialogue acts such as clarification, or requests for information.

7 Conclusion

In this paper, we collected a *stackoverflow* dataset that consists of dialogues labeled with whether that dialogue was accepted or not. This dataset will allow the community to work in open-ended domains with a clear notion of success. We used this dataset to build models that accurately predict success in open-ended human-human dialogues.

Our Turn-Based Hierarchical RNN model takes advantage of the natural structure that occurs in dialogues by recognizing both expressions of gratitude and more complex indicators of success found throughout the entire dialogue history.

An extension of this work will apply similar methods to human-computer dialogues. Our methods will become more relevant as human-computer interactions become more naturalistic. To minimize the dependence on users expressing their gratitude, we can focus on improving our models that remove the last comment by the initial user from the dataset.

Our methods can also be used to label similar human-human corpora which can then be used to train a dialogue system. Success offers a notion of reward and can be used as such in dialogue systems trained with reinforcement learning.

Acknowledgements

The authors gratefully acknowledge financial support for this work by the Samsung Advanced Institute of Technology (SAIT) and the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. 2012. Theano: new features and speed improvements. In *Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop*.

Yun-Nung Chen, Dilek Hakkani-Tur, and Xiaodong He. 2016. Zero-shot learning of intent embeddings for expansion by convolutional deep structured semantic models. In *Proceedings of the 2016 International Conference on Acoustics, Speech and Signal Processing*. IEEE, pages 6045–6049.

Bradley Efron and Robert J Tibshirani. 1994. *An introduction to the bootstrap*. CRC press.

Ali El-Kahky, Xiaohu Liu, Ruhi Sarikaya, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2014. Extending domain coverage of language understanding systems via intent transfer between domains using knowledge graphs and search query click logs. In *Proceedings of the 2014 International Conference on Acoustics, Speech and Signal Processing*. IEEE, pages 4067–4071.

Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

F. Jurcıcek, S. Keizer, M. Gašic, F. Mairesse, B. Thomson, K. Yu, and S. Young. 2011. Real user evaluation of spoken dialogue systems using amazon mechanical turk. In *Proceedings of the 12th Annual Conference of the International Speech Communication Association*. pages 3061–3064.

Soojung Kim and Sanghee Oh. 2009. Users' relevance criteria for evaluating answers in a social Q&A site. *Journal of the American society for information science and technology* 60(4):716–727.

Su Nam Kim, Li Wang, and Timothy Baldwin. 2010. Tagging and linking web forum posts. In *Proceedings of the 14th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, pages 192–202.

Diederik. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference for Learning Representations*.

Jiwei Li, Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, pages 1106–1115.

Yangdon Liu, Jiang Bian, and Eugene Agichtein. 2008. Predicting information seeker satisfaction in community question answering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, pages 483–490.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. 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. Association for Computational Linguistics, pages 285–294.

Ryan Lowe, Iulian Vlad Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. On the evaluation of dialogue systems with next utterance classification. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Association for Computational Linguistics, pages 264–269.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9:2579–2605.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1532–1543.

Antoine Raux, Brian Langner, Alan W. Black, and Maxine Eskenazi. 2003. LET'S GO: Improving spoken dialog systems for the elderly and non-natives. In *Proceedings of the 8th European Conference on Speech Communication and Technology*. pages 753–756.

Iulian V. Serban, Ryan Lowe, Laurent Charlin, and Joelle Pineau. 2015. A survey of available corpora for building data-driven dialogue systems. *arXiv preprint arXiv:1512.05742* .

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob G. Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, pages 553–562.

Pei-Hao Su, David Vandyke, Mrksic Gasic, Dongho Kim, Nikola Mrksic, Tsung-Hsien Wen, and Steve Young. 2015. Learning from real users: Rating dialogue success with neural networks for reinforcement learning in spoken dialogue systems. In *Proceedings of the 16th Annual Conference of the International Speech Communication Association*. pages 2007–2011.

Blaise Thomson and Steve Young. 2010. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. *Computer Speech & Language* 24(4):562–588.

David R. Traum and Elizabeth A. Hinkelman. 2011. Conversation acts in task-oriented spoken dialogue. Technical Report 425, Computer Science Department, University of Rochester.

Jan Ulrich, Gabriel Murray, and Giuseppe Carenini. 2008. A publicly available annotated corpus for supervised email summarization. In *Proceedings of the 2008 AAAI Enhanced Messaging Workshop*. AAAI, pages 77–82.

David Vandyke, Pei-Hao Su, Milica Gasic, Nikola Mrksic, Tsung-Hsien Wen, and Steve Young. 2015. Multi-domain dialogue success classifiers for policy training. In *In Proceedings of the 2015 IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, pages 763–770.

Marilyn Walker, Candace Kamm, and Diane Litman. 2000. Towards developing general models of usability with paradise. *Natural Language Engineering* 6(3-4):363–377.

Marilyn A. Walker, Jeanne C. Fromer, and Shrikanth Narayanan. 1998. Learning optimal dialogue strategies: A case study of a spoken dialogue agent for email. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*. Association for Computational Linguistics, pages 1345–1351.

Marilyn A. Walker, Diane J. Littman, Candace A. Kamm, and Alicia Abella. 1997. PARADISE: A framework for evaluating spoken dialogue agents. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 271–280.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, and Steve Young. 2016. Multi-domain neural network language generation for spoken dialogue systems. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 120–129.

Jason D. Williams and Steve Young. 2004. Characterizing task-oriented dialog using a simulated asr chanel. In *Proceedings of the 2004 International Conference on Spoken Language Processing*. pages 185–188.

Zhaojun Yang, Baichuan Li, Yi Zhu, Irwin King, Gina Levow, and Helen Meng. 2010. Collection of user judgments on spoken dialog system with crowdsourcing. In *Proceedings of the 2010 Spoken Language Technology Workshop*. IEEE, pages 277–282.

Steve Young, Milica Gasic, Blaise Thomson, and Jason D. Williams. 2013. POMDP-based statistical spoken dialog systems: A review. *IEEE* 101(5):1160–1179.