Sentiment Adaptive End-to-End Dialog Systems

Weiyan Shi
[24]7.ai
`weiyan.shi@247.ai`

Zhou Yu
University of California, Davis
`joyu@ucdavis.edu`

Abstract

End-to-end learning framework is useful for building dialog systems for its simplicity in training and efficiency in model updating. However, current end-to-end approaches only consider user semantic inputs in learning and under-utilize other user information. Therefore, we propose to include user sentiment obtained through multimodal information (acoustic, dialogic and textual), in the end-to-end learning framework to make systems more user-adaptive and effective. We incorporated user sentiment information in both supervised and reinforcement learning settings. In both settings, adding sentiment information reduced the dialog length and improved the task success rate on a bus information search task. This work is the first attempt to incorporate multimodal user information in the adaptive end-to-end dialog system training framework and attained state-of-the-art performance.

1 Introduction

Most of us have had frustrating experience and even expressed anger towards automated customer service systems. Unfortunately, none of the current commercial systems can detect user sentiment and let alone act upon it. Researchers have included user sentiment in rule-based systems (Acosta, 2009; Pittermann et al., 2010), where there are strictly-written rules that guide the system to react to user sentiment. Because traditional modular-based systems are harder to train, to update with new data and to debug errors, end-to-end trainable systems are more popular. However, no work has tried to incorporate sentiment information in the end-to-end trainable systems so far to create sentiment-adaptive systems that are easy to train. The ultimate evaluators of dialog systems are users. Therefore, we believe dialog system research should strive for better user satisfaction. In this paper, we not only included user sentiment information as an additional context feature in an end-to-end supervised policy learning model, but also incorporated user sentiment information as an immediate reward in a reinforcement learning model. We believe that providing extra feedback from the user would guide the model to adapt to user behaviour and learn the optimal policy faster and better.

There are three contributions in this work: 1) an audio dataset[1] with sentiment annotation (the annotators were given the complete dialog history); 2) an automatic sentiment detector that considers conversation history by using dialogic features, textual features and traditional acoustic features; and 3) end-to-end trainable dialog policies adaptive to user sentiment in both supervised and reinforcement learning settings. We believe such dialog systems with better user adaptation are beneficial in various domains, such as customer services, education, health care and entertainment.

2 Related Work

Many studies in emotion recognition (Schuller et al., 2003; Nwe et al., 2003; Bertero et al., 2016) have used only acoustic features. But there has been work on emotion detection in spoken dialog systems incorporating extra information as well (Lee and Narayanan, 2005; Devillers et al., 2003; Liscombe et al., 2005; Burkhardt et al., 2009; Yu et al., 2017). For example, Liscombe et al. (2005) explored features like users' dialog act, lexical context and discourse context of the previous turns. Our approach considered accumulated di-

---

[1]The dataset is available here.

alogic features, such as total number of interruptions, to predict user sentiment along with acoustic and textual features.

The traditional method to build dialog system is to train modules such as language understanding component, dialog manager and language generator separately (Levin et al., 2000; Williams and Young, 2007; Singh et al., 2002). Recently, more and more work combines all the modules in an end-to-end training framework (Wen et al., 2016; Li et al., 2017; Dhingra et al., 2016; Williams et al., 2017; Liu and Lane, 2017a). Specifically related to our work, Williams et al. (2017) built a model, which combined the traditional rule-based system and the modern deep-learning-based system, with experts designing actions masks to regulate the neural model. Action masks are bit vectors indicating allowed system actions at certain dialog state. The end-to-end framework made dialog system training simpler and model updating easier.

Reinforcement learning (RL) is also popular in dialog system building (Zhao and Eskenazi, 2016; Liu and Lane, 2017b; Li et al., 2016). A common practice is to simulate users. However, building a user simulator is not a trivial task. Zhao and Eskenazi (2016) combines the strengths of reinforcement learning and supervised learning to accelerate the learning of a conversational game simulator. Li et al. (2016) provides a standard framework for building user simulators, which can be modified and generalized to different domains. Liu and Lane (2017b) describes a more advanced way to build simulators for both the user and the agent, and train both sides jointly for better performance. We simulated user sentiment by sampling from real data and incorporated it as immediate rewards in RL, which is different from common practice of using task success as delayed rewards in RL training.

Some previous module-based systems integrated user sentiment in dialog planning (Acosta, 2009; Acosta and Ward, 2011; Pittermann et al., 2010). They all integrated user sentiment in the dialog manager with manually defined rules to react to different user sentiment and showed that tracking sentiment is helpful in gaining rapport with users and creating interpersonal interaction in the dialog system. In this work, we include user sentiment into end-to-end dialog system training and make the dialog policy learn to choose dialog actions to react to different user sentiments

automatically. We achieve this through integrating user sentiment into reinforcement reward design. Many previous RL studies used delayed rewards, mostly task success. However, delayed rewards make the converging speed slow, so some studies integrated estimated per-turn immediate reward. For example, Ferreira and Lefèvre (2013) explored expert-based reward shaping in dialog management and Ultes et al. (2017) proposed *Interaction Quality (IQ)*, a less subjective variant of user satisfaction, as immediate reward in dialog training. However, both methods are not end-to-end trainable, and require manual input as prior, either in designing proper form of reward, or in annotating the *IQ*. Our approach is different as we detect the multimodal user sentiment on the fly and does not require manual input. Because sentiment information comes directly from real users, our method will adapt to user sentiment as the dialog evolves in real time. Another advantage of our model is that the sentiment scores come from a pre-trained sentiment detector, so no manual annotation of rewards is required. Furthermore, the sentiment information is independent of the user's goal, so no prior domain knowledge is required, which makes our method generalizable and independent of the task.

3 Dataset

We experimented our methods on DSTC1 dataset (Raux et al., 2005), which has a bus information search task. Although DSTC2 dataset is a more commonly-used dataset in evaluating dialog system performance, the audio recordings of DSTC2 are not publicly available and therefore, DSTC1 was chosen. There are a total of 914 dialogs in DSTC1 with both text and audio information. Statistics of this dataset are shown in Table 1. We used the automatic speech recognition (ASR) as the user text inputs instead of the transcripts, because the system's action decisions heavily depend on ASR. There are 212 system action templates in this dataset. Four types of entities are involved, `<place>`, `<time>`, `<route>`, and `<neighborhood>`.

4 Annotation

We manually annotated 50 dialogs consisting of 517 conversation turns for user sentiment. Sentiment is categorized into `negative`, `neutral` and `positive`. The annotator had access to the

Category	Total
total dialogs	914
total dialogs in train	517
total dialogs in test	397

Statistics	Total
avg dialog len	13.8
vocabulary size	685

Table 1: Statistics of the text data.

Category	Total
total dialogs	50
total audios	517
total audios in train	318
total audios in dev	99
total audios in test	100

Category	Total
neutral	254
negative	253
positive	10

Table 2: Statistics of the annotated audio set.

entire dialog history in the annotation process because the dialog context gives the annotators a holistic view of the interactions, and annotating user sentiment in a dialog without the context is really difficult. Some previous studies have also performed similar user information annotation given context, such as Devillers et al. (2002). The annotation scheme is described in Table 10 in Appendix A.2. To address the concern that dialog quality may bias the sentiment annotation, we explicitly asked the annotators to focus on users' behaviour instead of the system, and hid all the details of multimodal features from the annotators. Moreover, two annotators were calibrated on 37 audio files, and reached an inter-annotator agreement (kappa) of 0.74. The statistics of the annotation results are shown in Table 2. The skewness in the dataset is due to the data's nature. In the annotation scheme, `positive` is defined as "excitement or other positive feelings", but people rarely express obvious excitement towards automated task-oriented dialog systems. What we really want to distinguish is neutral and positive cases from negative cases so as to avoid the negative sentiment, and the dataset is balanced for these two cases. To the best of our knowledge, our dataset is the first publicly available dataset that annotated user sentiment with respect to the entire dialog history. There are similar datasets with emotion annotations (Schuller et al., 2013) but are not labeled under dialog contexts.

5 Multimodal Sentiment Classification

To detect user sentiment, we extracted a set of acoustic, dialogic and textual features.

5.1 Acoustic features

We used openSMILE (Eyben et al., 2013) to extract acoustic features. Specifically, we used the paralinguistics configuration from Schuller et al. (2003), which includes 1584 acoustic features, such as pitch, volume and jitter. In order to avoid possible overfitting caused by the large number of acoustic features, we performed tree-based feature selection (Pedregosa et al., 2011) to reduce the size of acoustic features to 20. The selected features are listed in Table 12 in Appendix A.4.

5.2 Dialogic features

Four categories of dialogic features are selected according to previous literature (Liscombe et al., 2005) and the statistics observed in the dataset. We used not only the per-turn statistics of these features, but also the accumulated statistics of them throughout the entire conversation so that the sentiment classifier can also take the entire dialog context into consideration.

Interruption is defined as the user interrupting the system speech. Interruptions occurred fairly frequently in our dataset (4896 times out of 14860 user utterances).

Button usage When the user is not satisfied with the ASR performance of the system, he/she would rather choose to press a button for "yes/no" questions, so the usage of buttons can be an indication of negative sentiment. During DSTC1 data collection, users were notified about the option to use buttons, so this kind of information is available in the data.

Repetitions There are two kinds of repetitions: the user asks the system to repeat the previous sentence, and the system keeps asking the same question due to failures to catch some important entity. In our model, we combined these two situations as one feature because very few user repetitions occur in our data (<1%). But for other data, it might be helpful to separate them.

Start over is active when the user chooses to restart the task in the middle of the conversation. The system is designed to give the user an option to start over after several turns. If the user takes this offer, he/she might have negative sentiment.

5.3 Textual features

We also noticed that the semantic content of the utterance was relevant to sentiment. So we used the entire dataset as a corpus and created a tf-idf vector for each utterance as textual features.

5.4 Classification results

The sentiment classifier was trained on the 50 dialogs annotated with sentiment labels. The predictions made by this classifier were used for the supervised learning and reinforcement learning in the later sections. We used random forest as our classifier (an implementation from scikit-learn (Pedregosa et al., 2011)), as we had limited annotated data. We separated the data to be 60% for training, 20% for validation and 20% for testing. Due to the randomness in the experiments, we ran all the experiments 20 times and reported the average results of different models in Table 4. We also conducted unpaired one-tailed t-test to assess the statistical significance.

We extracted 20 acoustic features, eight dialogic features and 164 textual features. From Table 4, we see that the model combining all the three categories of features performed the best (0.686 in F-1, $p < 1e{-}6$ compared to acoustic baseline). One interesting observation is that by only using eight dialogic features, the model already achieved 0.596 in F-1. Another interesting observation is that using 164 textual features alone reached a comparable performance (0.664), but the combination of acoustic and textual features actually brought down the performance to 0.647. One possible reason is that the acoustic information has noise that confused the textual information when combined. But this observation doesn't necessarily apply to other datasets. The significance tests show that adding dialogic features improved the baseline significantly. For example, the model with both acoustic features and dialogic features are significantly better than the one with only acoustic features ($p < 1e{-}6$). In Table 3, we listed the dialogic features with their relative importance rank, which were obtained from ranking their feature importance scores in the classifier. We observe that "total interruptions so far" is the most useful dialogic features to predict user sentiment. The sentiment detector trained will be integrated in the end-to-end learning described later.

Dialogic Features	Relative Rank of importance
total interruptions so far	1
interruptions	2
total button usages so far	3
total repetitions so far	4
repetition	5
button usage	6
total start over so far	7
start over	8

Table 3: Dialogic features' relative importance rank in sentiment detection.

Model	Avg. of F-1	Std. of F-1	Max of F-1
Acoustic features only	0.635	0.027	0.686
Dialogic features only	0.596	0.001	0.596
Textual features only *	0.664	0.010	0.685
Textual + Dialogic *	0.672	0.011	0.700
Acoustic + Dialogic *	0.680	0.019	0.707
Acoustic + Textual	0.647	0.025	0.686
Acoustic + Dialogic + Text *	**0.686**	**0.028**	**0.756**

Table 4: Results of sentiment detectors using different features. The best result is highlighted in **bold** and * indicates statistical significance compared to the baseline, which is using acoustic features only. ($p < 0.0001$)

6 Supervised Learning (SL)

We incorporated the detected user sentiment from the previous section into a supervised learning framework for training end-to-end dialog systems. There are many studies on building a dialog system in a supervised learning setting (Bordes and Weston (2016); Eric and Manning (2017); Seo et al. (2016); Liu and Lane (2017a); Li et al. (2017); Williams et al. (2017)). Following these approaches, we treated the problem of dialog policy learning as a classification problem, which is to select actions among system action templates given conversation history. Specifically, we decided to adopt the framework of *Hybrid Code Network (HCN)* introduced in Williams et al. (2017), because it is the current state-of-the-art model. We reimplemented *HCN* and used it as the baseline system, given the absence of direct comparison on DSTC1 data. One caveat is that *HCN* used action masks (bit vectors indicating allowed actions at certain dialog states) to prevent impossible system actions, but we didn't use hand-crafted action masks in the supervised learning setting because manually designing action masks for 212 action templates is very labor-intensive. This makes our method more general and adaptive to different tasks. All the dialog modules were trained

together instead of separately. Therefore, our method is end-to-end trainable and doesn't require human expert involvement.

We listed all the context features used in Williams et al. (2017) in Table 11 in Appendix A.3. In our model, we added one more set of context features, the user-sentiment-related features. For entity extraction, given that the entity values in our dataset form a simple unique fixed set, we used simple string matching. We conducted three experiments: the first one used entity presences as context features, which serves as the baseline; the second one used entity presences in addition to all the raw dialogic features mentioned in Table 3; the third experiment used the baseline features plus the predicted sentiment label by the pre-built sentiment detector (converted to one-hot vector) instead of the raw dialogic features. We used the entire DSTC1 dataset to train the supervised model. The input is the normalized natural language and the contextual features, and the output is the action template id. We kept the same experiment setting in Williams et al. (2017), e.g. `last_action_taken` was also used as a feature, along with word embeddings (Mikolov et al., 2013) and bag-of-words; LSTM with 128 hidden-units and AdaDelta optimizer (Zeiler, 2012) were used to train the model.

The results of different models are shown in Table 5. We observe that using the eight raw dialogic features did not improve turn-level F-1 score. One possible reason is that a total of eight dialogic features were added to the model, and some of them might contain noises and therefore caused the model to overfit. However, using predicted sentiment information as an extra feature, which is a more condensed information, outperformed the other models both in terms of turn-level F-1 score and dialog accuracy which indicates if all turns in a dialog are correct. The difference in absolute F-1 score is small because we have a relatively large test set (5876 turns). But the unpaired one-tailed t-test shows that $p < 0.01$ for both the F-1 and the dialog accuracy. This suggests that including user sentiment information in action planning is helpful in a supervised learning setting.

7 Reinforcement Learning (RL)

In the previous section, we discussed including sentiment features directly as a context feature in a supervised learning model for end-to-end dialog

Model	Weighted F-1	Dialog Acc.
HCN	0.4198	6.05%
HCN + raw dialogic features	0.4190	5.79%
HCN + predicted sentiment label∗	**0.4261**	**6.55%**

Table 5: Results of different SL models. The best result is highlighted in **bold**. ∗ indicates that the result is significantly better than the baseline ($p < 0.01$). Dialog accuracy indicates if all turns in a dialog are correct, so it's low. For DSTC2 data, the state-of-art dialog accuracy is 1.9%, consistent with our results.

system training, which showed promising results. But once a system operates at scale and interacts with a large number of users, it is desirable for the system to continue to learn autonomously using reinforcement learning (RL). With RL, each turn receives a measurement of goodness called *reward* (Williams et al., 2017). Previously, training task-oriented systems mainly relies on the delayed reward about task success. Due to the lack of informative immediate reward, the training takes a long time to converge. In this work, we propose to include user sentiment as immediate rewards to expedite the reinforcement learning training process and create a better user experience.

To use sentiment scores in the reward function, we chose the *policy gradient* approach (Williams, 1992) and implemented the algorithm based on Zhu (2017). The traditional reward function uses a positive constant (e.g. 20) to reward the success of the task, 0 or a negative constant to penalize the failure of the task after certain number of turns, and gives -1 to each extra turn to encourage the system to complete the task sooner. However, such reward function doesn't consider any feedback from the end-user. It is natural for human to consider conversational partner's sentiment in planning dialogs. So, we propose a set of new reward functions that incorporate user sentiment to emulate human behaviors.

The intuition of integrating sentiment in reward functions is as follows. The ultimate evaluator of dialog systems is the end-users. And user sentiment is a direct reflection of user satisfaction. Therefore, we detected the user sentiment scores from multimodal sources on the fly, and used them as immediate rewards in an adaptive end-to-end dialog training setting. This sentiment information came directly from real users, which made the system adapt to individual user's sentiment as the

dialog proceeds. Furthermore, the sentiment information is independent of the task, so our method doesn't require any prior domain knowledge and can be easily generalized to other domains. There have been works that incorporated user information into reward design (Su et al., 2015; Ultes et al., 2017). But they used information from one single channel and sometimes required manual labelling of the reward. Our approach utilizes information from multiple channels and doesn't involve manual work once a sentiment detector is ready.

We built a simulated system in the same bus information search domain to test the effectiveness of using sentiment scores in the reward function. In this system, there are 3 entity types - <departure>, <arrival>, and <time> - and 5 actions, asking for different entities, and giving information. A simple action mask was used to prevent impossible actions, such as giving information of an uncovered place. The inputs to the system are the simulated user's dialog acts and the simulated sentiment sampled from a subset of DSTC1, the $CleanData$, which will be described later. The output of the system is the system action template.

7.1 User simulator

Given that reinforcement learning requires feedback from the environment - in our case, the users - and interacting with real users is always expensive, we created a user simulator to interact with the system. At the beginning of each dialog, the simulated user is initiated with a goal consisting of the three entities mentioned above and the goal remains unchanged throughout the conversation. The user responds to system's questions with entities, which are placeholders like <departure> instead of real values. To simulate ASR errors, the simulated user's act type occasionally changes from "informing slot values" to "making noises" at certain probabilities set by hand (10% in our case). Some example dialogs along with their associated rewards are shown in Table 8 and 9 in Appendix A.1.

We simulated user sentiment by sampling from real data, the DSTC1 dataset. There are three steps involved. First, we cleaned the DSTC1 dialogs by removing the audio files with no ASR output and high ASR errors. This resulted in a dataset $CleanData$ with 413 dialogs and 1918 user inputs. We observed that users accumulate their

sentiment as the conversation unfolds. When the system repeatedly asks for the same entity, they express stronger sentiment. Therefore, summary statistics that record how many times certain entities have been asked during the conversation is representative of users' accumulating sentiment. We designed a set of summary statistics S that record the statistics of system actions, e.g. how many times the arrival place has been asked or the schedule information has been given.

The second step is to create a mapping between the five simulated system actions and the DSTC1 system actions. We do this by calculating a vector s_{real} consisting of the values in S for each user utterance in $CleanData$. s_{real} is used to compare the similarity between the real dialog and the simulated dialog.

The final step is to sample from $CleanData$. For each simulated user utterance, we calculated the same vector s_{sim} and compared it with each s_{real}. There are two possible results. If there are s_{real} equal to s_{sim}, we would randomly sample one from all the matched user utterances to represent the sentiment of the simulated user. But if there is no s_{real} matching s_{sim}, different strategies would be applied based on the reward function used, which will be described in details later. Once we have a sample, the eight dialogic features of the sample utterance are used to calculate the sentiment score. We didn't use the acoustic or the textual features because in a simulated setting, only the dialogic features are valid.

7.2 Experiments

We designed four experiments with different reward functions. A discount factor of 0.9 was applied to all the experiments. And the maximum number of turns is 15. Following Williams et al. (2017), we used LSTM with 32 hidden units for the RNN in the *HCN* and AdaDelta for the optimization, and updated the reinforcement learning policy after each dialog. The ϵ-greedy exploration strategy (Tokic, 2010) was applied here. Given that the entire system was simulated, we only used the presence of each entity and the last action taken by the system as the context features, and didn't use bag-of-words or utterance embedding features.

In order to evaluate the method, we froze the policy after every 200 updates, and ran 500 simulated dialogs to calculate the task success rate. We

repeated the process 20 times and reported the average performance in Figure 1, 2 and Table 6.

7.2.1 Baseline

We define the baseline reward as follows without any sentiment involvement.

Reward 1 Baseline

  if success **then**
    $R_1 = 20$
  else if failure **then**
    $R_1 = -10$
  else if each proceeding turn **then**
    $R_1 = -1$
  end if

7.2.2 Sentiment reward with random samples (SRRS)

We designed the first simple reward function with user sentiment as the immediate reward: sentiment with random samples (SRRS). We first drew a sample from real data with matched context; if there was no matched data, a random sample was used instead. Because the amount of $CleanData$ is relatively small, so only 36% turns were covered by matched samples. If the sampled dialogic features were not all zeros, the sentiment reward (SR) was calculated as a linear combination with tunable parameters. We chose it to be $-5P_{neg}-P_{neu}+10P_{pos}$ for simplicity. When the dialogic features were all zero, in most cases it meant the user didn't express an obvious sentiment, we set the reward to be -1.

Reward 2 SRRS

  if success **then**
    $R_2 = 20$
  else if failure **then**
    $R_2 = -10$
  else if sample with all-zero dialogic features **then**
    $R_2 = -1$
  else if sample with non-zero dialogic features **then**
    R_2=-5P$_{neg}$-P$_{neu}$+10P$_{pos}$
  end if

7.2.3 Sentiment reward with repetition penalty (SRRP)

Random samples in SRRS may result in extreme sentiment data. So we used dialogic features to approximate sentiment for the unmatched data. Specifically, if there were repetitions, which correlate with negative sentiment (see Table 3), we assigned a penalty to that utterance. See Reward 3 Formula below for detailed parameters. 36% turns were covered by real data samples, 15% turns had no match in real data and had repetitions, and 33% turns had no match and no repetition.

Moreover, we experimented with different penalty weights. When we increased the repetition penalty to 5, the success rate was similar to penalty of 2.5. However, when we increased the penalty even further to 10, the success rate was brought down by a large margin. Our interpretation is that increasing the repetition penalty to a big value made the focus less on the real sentiment samples but more on the repetitions, which did not help the learning.

Reward 3 SRRP

  if success **then**
    $R_3 = 20$
  else if failure **then**
    $R_3 = -10$
  else
    if match **then**
      if all-zero dialogic features **then**
        $R_3 = -1$
      else if non-zero dialogic features **then**
        R_3=-5P$_{neg}$-P$_{neu}$+10P$_{pos}$
      end if
    else if repeated question **then**
      $R_3 = -2.5$
    else
      $R_3 = -1$
    end if
  end if

7.2.4 Sentiment reward with repetition and interruption penalties (SRRIP)

We observed in Section 5 that `interruption` is the most important feature in detecting sentiment, so if an interruption existed in the simulated user input, we assumed it had a negative sentiment and added an additional penalty of -1 to the previous sentiment reward SRRP to test the effect of interruption. 7.5% turns have interruptions.

Reward 4 SRRIP

  if success **then**
    $R_4 = 20$
  else if failure **then**
    $R_4 = -10$
  else
    $R_4 = R_3(SRRP)$
    if interruption **then**
      $R_4 = R_4 - 1$
    end if
  end if

7.3 Experiment results

We evaluated every model on two metrics: dialog lengths and task success rates. We observed in Figure 1 that all the sentiment reward functions, even SRRS with random samples, reduced the average length of the dialogs, meaning that the system finished the task faster. The rationale behind is that by adapting to user sentiment, the model can avoid unnecessary system actions to make systems more effective.

In terms of success rate, sentiment reward with both repetition and interruption penalties (SRRIP) performed the best (see Figure 2). In Figure 2, SR-RIP is converging faster than the baseline. For example, around 5000 iterations, it outperforms the baseline by 5% in task success rate (60% vs 55%) with statistical significance ($p < 0.01$). It also converges to a better task success rate after 10000 iterations (92.4% vs 94.3%, $p < 0.01$).
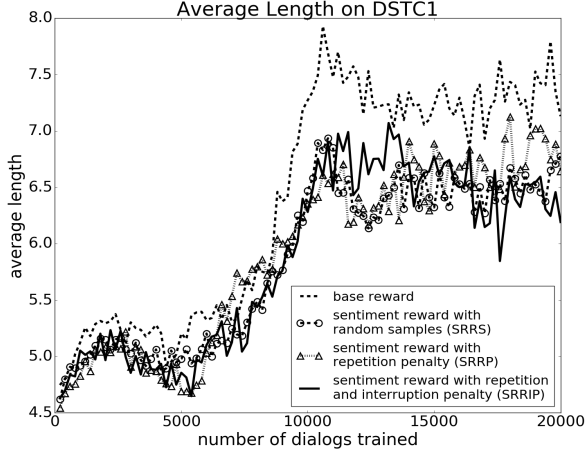

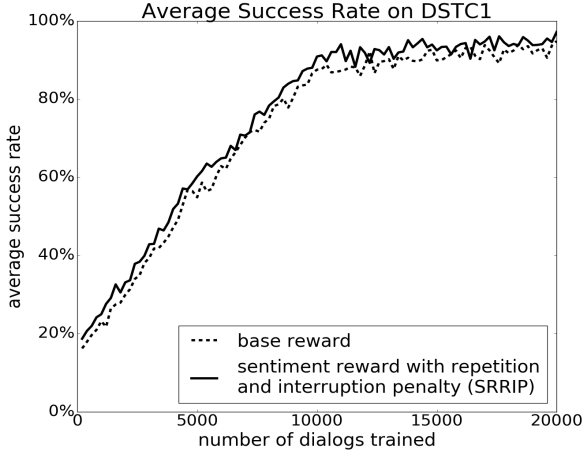Figure 1: Average dialog length of RL models with different reward functions.

Figure 2: Average success rate of the baseline and the best performing model, SRRIP.

We describe all models' performance in Table 6 in terms of the convergent success rate calculated as the mean success rate after 10000 dialogs. We observed that incorporating various sentiment rewards improved the success rate and expedited the training process overall with statistical significance. We found that even sentiment reward with random samples (SRRS) outperformed the baseline after convergence. By adding penalties for

Model	Convergent success rate
Baseline	0.924
SRRS	0.938∗
SRRP	0.941∗
SRRIP	**0.943**∗

Table 6: Convergent success rate of RL models with different reward functions. It is calculated as the mean success rate after 10000 dialogs. The best result is highlighted in **bold**. ∗ indicates that the result is significantly better than the baseline ($p < 0.01$).

repetition, the algorithm covered more data points, and therefore, the task success rate and the convergence speed improved. We also found that penalizing interruption and repetition together (SR-RIP) achieved a slightly better performance compared to penalizing repetition only (SRRP). This suggests that interruptions is another factor to consider when approximating sentiment. But the performances between SRRP and SRRIP is not significant. Our guess is that only 7.5% turns in our data contains interruption and the penalty is just an extra -1, so the model confused this signal with noises. But given more interruptions in the data, interruptions could still be helpful.

8 Discussion and Future Work

The intuition behind the good performance of models with user sentiment is that the learned policy is in general more sentiment adaptive. For example, there are some system actions that have the same intention but with different surface forms, especially for error-handling strategies. By analyzing the results, we found that when the sentiment adaptive system detected a negative sentiment from the user, it chose to respond the user with a more detailed error-handling strategy than a general one. For example, it chose the template "Where are you leaving from? For example, you can say, <place>", while the baseline model would respond with "Where would you like to leave from?", which doesn't provide details to help the user compared with the previous template. As we all know, dealing with a disappointed user to proceed, providing more details is always better. One example dialog is shown in Table 7. There was no written rules to force the model to choose one specific template under certain situ-

Sentiment Adaptive System	Baseline System without Sentiment
SYS: The `<route>`. Where would you like to leave from?	**SYS**: The `<route>`. Where would you like to leave from?
USR: *Yeah [negative sentiment]*	**USR**: *Yeah*
SYS: Where are you leaving from? For example, you can say, `<place>`.	**SYS**: Right. Where would you like to leave from?

Table 7: An example dialog by different systems in the supervised learning setting. The sentiment-adaptive system gives a more detailed error-handling strategy than the baseline system.

ations, so the model learned these subtle differences on its own. Some may argue that the system could always use a more detailed template to better guide the user instead of distinguishing between two similar system templates. But this is not necessarily true. Ideally, we want the system to be succinct initially to save users' time, because we observe that users, especially repeated users, tend to interrupt long and detailed system utterances. If the user has attempted to answer the system question but failed, then it's beneficial to provide detailed guidance.

The performance of the sentiment detector is a key factor in our work. So in the future, we plan to incorporate features from more channels such as vision to further improve the sentiment predictor's performance, and potentially further improve the performance of the dialog system. We also want to explore more in user sentiment simulation, for example, instead of randomly sampling data for the uncovered cases, we could use linear interpolation to create a similarity score between s_{sim} and s_{real}, and choose the user utterance with the highest score. Furthermore, reward shaping (Ng et al., 1999; Ferreira and Lefèvre, 2013) is an important technique in RL. Specifically, Ferreira and Lefèvre (2013) talked about incorporating expert knowledge in reward design. We also plan to integrate information from different sources into reward function and apply reward shaping. Besides, creating a good user simulator is also very important in the RL training. There are some more advanced methods to create user simulators. For example, Liu and Lane (2017b) described how to optimize the agent and the user simulators jointly using RL. We plan to apply our sentiment reward functions in this framework in the future.

9   Conclusion

We proposed to detect user sentiment from multimodal channels and incorporate the detected sentiment as feedback into adaptive end-to-end dialog system training to make the system more effective and user-adaptive. We included sentiment information directly as a context feature in the supervised learning framework and used sentiment scores as immediate rewards in the reinforcement learning setting. Experiments suggest that incorporating user sentiment is helpful in reducing the dialog length and increasing the task success rate in both SL and RL settings. This work proposed an adaptive methodology to incorporate user sentiment in end-to-end dialog policy learning and showed promising results on a bus information search task. We believe this approach can be easily generalized to other domains given its end-to-end training procedure and task independence.

Acknowledgments

References

Jaime C Acosta. 2009. Using emotion to gain rapport in a spoken dialog system. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Student Research Workshop and Doctoral Consortium*, pages 49–54. Association for Computational Linguistics.

Jaime C Acosta and Nigel G Ward. 2011. Achieving rapport with turn-by-turn, user-responsive emotional coloring. *Speech Communication*, 53(9-10):1137–1148.

Dario Bertero, Farhad Bin Siddique, Chien-Sheng Wu, Yan Wan, Ricky Ho Yin Chan, and Pascale Fung. 2016. Real-time speech emotion and sentiment recognition for interactive dialogue systems. In *EMNLP*, pages 1042–1047.

Antoine Bordes and Jason Weston. 2016. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683*.

Felix Burkhardt, Markus Van Ballegooy, Klaus-Peter Engelbrecht, Tim Polzehl, and Joachim Stegmann. 2009. Emotion detection in dialog systems: applications, strategies and challenges. In *Affective Computing and Intelligent Interaction and Workshops, 2009. ACII 2009. 3rd International Conference on*, pages 1–6. IEEE.

Laurence Devillers, Lori Lamel, and Ioana Vasilescu. 2003. Emotion detection in task-oriented spoken dialogues. In *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, volume 3, pages III–549. IEEE.

Laurence Devillers, Ioana Vasilescu, and Lori Lamel. 2002. Annotation and detection of emotion in a task-oriented human-human dialog corpus. In *proceedings of ISLE Workshop*.

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2016. End-to-end reinforcement learning of dialogue agents for information access. *arXiv preprint arXiv:1609.00777*.

Mihail Eric and Christopher D Manning. 2017. A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue. *arXiv preprint arXiv:1701.04024*.

Florian Eyben, Felix Weninger, Florian Gross, and Björn Schuller. 2013. Recent developments in opensmile, the munich open-source multimedia feature extractor. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 835–838, New York, NY, USA. ACM.

Emmanuel Ferreira and Fabrice Lefèvre. 2013. Expert-based reward shaping and exploration scheme for boosting policy learning of dialogue management. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pages 108–113. IEEE.

C. M. Lee and Shrikanth Narayanan. 2005. Toward Detecting Emotions in Spoken Dialogs. In *IEEE Transactions on Speech and Audio Processing*, volume 12, pages 293–303.

Esther Levin, Roberto Pieraccini, and Wieland Eckert. 2000. A stochastic model of human-machine interaction for learning dialog strategies. *IEEE Transactions on speech and audio processing*, 8(1):11–23.

Xiujun Li, Zachary C Lipton, Bhuwan Dhingra, Lihong Li, Jianfeng Gao, and Yun-Nung Chen. 2016. A user simulator for task-completion dialogues. *arXiv preprint arXiv:1612.05688*.

Xuijun Li, Yun-Nung Chen, Lihong Li, and Jianfeng Gao. 2017. End-to-end task-completion neural dialogue systems. *arXiv preprint arXiv:1703.01008*.

Jackson Liscombe, Giuseppe Riccardi, and Dilek Hakkani-Tür. 2005. Using context to improve emotion detection in spoken dialog systems. In *Ninth European Conference on Speech Communication and Technology*.

Bing Liu and Ian Lane. 2017a. An end-to-end trainable neural network model with belief tracking for task-oriented dialog. *arXiv preprint arXiv:1708.05956*.

Bing Liu and Ian Lane. 2017b. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. *arXiv preprint arXiv:1709.06136*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Andrew Y Ng, Daishi Harada, and Stuart Russell. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, volume 99, pages 278–287.

Tin Lay Nwe, Say Wei Foo, and Liyanage C De Silva. 2003. Speech emotion recognition using hidden markov models. *Speech communication*, 41(4):603–623.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Johannes Pittermann, Angela Pittermann, and Wolfgang Minker. 2010. Emotion recognition and adaptation in spoken dialogue systems. *International Journal of Speech Technology*, 13(1):49–60.

Antoine Raux, Brian Langner, Dan Bohus, Alan W Black, and Maxine Eskenazi. 2005. Lets go public! taking a spoken dialog system to the real world. In *in Proc. of Interspeech 2005*. Citeseer.

Björn Schuller, Gerhard Rigoll, and Manfred Lang. 2003. Hidden markov model-based speech emotion recognition. In *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, volume 1, pages I–401. IEEE.

Björn Schuller, Stefan Steidl, Anton Batliner, Alessandro Vinciarelli, Klaus Scherer, Fabien Ringeval, Mohamed Chetouani, Felix Weninger, Florian Eyben, Erik Marchi, et al. 2013. The interspeech 2013 computational paralinguistics challenge: social signals, conflict, emotion, autism. In *Proceedings INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France*.

Minjoon Seo, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Query-regression networks for machine comprehension. *arXiv preprint arXiv:1606.04582*.

Satinder Singh, Diane Litman, Michael Kearns, and Marilyn Walker. 2002. Optimizing dialogue management with reinforcement learning: Experiments with the njfun system. *Journal of Artificial Intelligence Research*, 16:105–133.

Pei-Hao Su, David Vandyke, Milica 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. *arXiv preprint arXiv:1508.03386*.

Michel Tokic. 2010. Adaptive ε-greedy exploration in reinforcement learning based on value differences. In *Annual Conference on Artificial Intelligence*, pages 203–210. Springer.

Stefan Ultes, Paweł Budzianowski, Inigo Casanueva, Nikola Mrkšic, Lina Rojas-Barahona, Pei-Hao Su, Tsung-Hsien Wen, Milica Gašic, and Steve Young. 2017. Domain-independent user satisfaction reward estimation for dialogue policy learning. In *Proc. Interspeech*, pages 1721–1725.

Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2016. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*.

Jason Williams, Kavosh Asadi, and Geoffrey Zweig. 2017. Hybrid code networks: Practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *Proceedings of 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*. Association for Computational Linguistics.

Jason D Williams and Steve Young. 2007. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Zhou Yu, Alan W Black, and Alexander I Rudnicky. 2017. Learning conversational systems that interleave task and non-task content. *IJCAI*.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Tiancheng Zhao and Maxine Eskenazi. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. *arXiv preprint arXiv:1606.02560*.

Yuke Zhu. 2017. tensorflow-reinforce. https://github.com/yukezhu/tensorflow-reinforce.

A Supplemental Material

A.1 Example dialogs with different reward functions

Dialogs	Category	Reward
USR: *I am at* `<uncovered_departure>`.		
SYS: what time do you want to travel?	proceeding turn	-1
USR: *At* `<time>`.		
SYS: what time do you want to travel?	proceeding turn	-1
USR: *You already knew that!*		
SYS: Where do you want to go?	proceeding turn	-1
USR: *I want to go to* `<uncovered_arrival>`.		
SYS: Let me look that up for you. Sorry, there is no result that matches your request.	success	20
	Total reward	17

Table 8: An example of the simulated dialog using the baseline reward.

Dialogs	Category	Reward
USR: *I am at* `<uncovered_departure>`		
SYS: Where do you want to go?	no match	-1
USR: `<arrival>`.		
SYS: where do you want to go?	matched	-4.01
USR: *You already knew that!*		
SYS: where are you leaving from?	repetition	-2.5
USR: *You already knew that!*		
SYS: what time do you want to travel?	no match	-1
USR: *At* `<time>`.		
SYS: Let me look that up for you. Sorry, there is no result that matches your request.	success	20
	Total reward	11.49

Table 9: An example of the simulated dialog using the sentiment reward with repetition penalty (SRRP).

A.2 Annotation scheme

Label	Definition	Description
-1	negative	Shows impatience, disappointment, anger or other negative feelings in voice.
0	neutral	Shows neither positive nor negative feelings in voice.
1	positive	Shows excitement, happiness or other positive feelings in voice.

Table 10: Sentiment annotation scheme.

A.3 Context features used in supervised learning

Model	Context features
baseline HCN	Presence of each entity in dialog state
HCN + dialogic features	Presence of each entity + dialogic features in Table 3
HCN + predicted sentiment label	Presence of each entity + predicted sentiment label in one-hot vector

Table 11: Context features in different SL models.

A.4 Selected acoustic features in Section 5.1

Selected acoustic features after feature selection
pcm_loudness_sma_iqr1-3
pcm_fftMag_mfcc_sma[0]_pctlrange0-1
pcm_fftMag_mfcc_sma[0]_upleveltime90
pcm_fftMag_mfcc_sma[7]_upleveltime75
logMelFreqBand_sma[1]_quartile3
logMelFreqBand_sma[1]_upleveltime75
logMelFreqBand_sma[2]_kurtosis
logMelFreqBand_sma[3]_amean
logMelFreqBand_sma[3]_linregc2
logMelFreqBand_sma[5]_upleveltime90
lspFreq_sma[7]_minPos
lspFreq_sma[7]_skewness
pcm_loudness_sma_de_kurtosis
pcm_fftMag_mfcc_sma_de[5]_amean
pcm_fftMag_mfcc_sma_de[6]_linregerrQ
pcm_fftMag_mfcc_sma_de[9]_kurtosis
pcm_fftMag_mfcc_sma_de[10]_iqr1-2
logMelFreqBand_sma_de[5]_quartile2
lspFreq_sma_de[4]_percentile99.0
jitterLocal_sma_de_iqr1-2

Table 12: Selected acoustic features in Section 5.1