Evaluating and Enhancing the Robustness of Dialogue Systems: A Case Study on a Negotiation Agent

Minhao Cheng
University of California Los Angeles
mhcheng@cs.ucla.edu

Wei Wei
Google Cloud AI
wewei@google.com

Cho-Jui Hsieh
University of California Los Angeles
chohsieh@cs.ucla.edu

Abstract

Recent research has demonstrated that goal-oriented dialogue agents trained on large datasets can achieve striking performance when interacting with human users. In real world applications, however, it is important to ensure that the agent performs smoothly interacting with not only regular users but also those malicious ones who would attack the system through interactions in order to achieve goals for their own advantage. In this paper, we develop algorithms to evaluate the robustness of a dialogue agent by carefully designed attacks using adversarial agents. Those attacks are performed in both black-box and white-box settings. Furthermore, we demonstrate that adversarial training using our attacks can significantly improve the robustness of a goal-oriented dialogue system. On a case-study of the negotiation agent developed by (Lewis et al., 2017), our attacks reduced the average advantage of rewards between the attacker and the trained RL-based agent from 2.68 to -5.76 on a scale from -10 to 10 for randomized goals. Moreover, with the proposed adversarial training, we are able to improve the robustness of negotiation agents by 1.5 points on average against all our attacks.

1 Introduction

Crafting an intelligent agent to communicate in the dialogue system using natural languages has been a long-standing problem in AI. It requires designing an agent to understand, plan and generate natural language to achieve different goals such as question-answering, cooperation, negotiation etc (Vinyals and Le, 2015; Li et al., 2017; Serban et al., 2016; Dhingra et al., 2016; Serban et al., 2016). Inspired by recent successes in deep neural networks, (Lewis et al., 2017) has recently developed an end-to-end learning framework to train a recurrent neural network (RNN)-based negotiation agent in goal-oriented dialogue systems. This NN-based technique has been identified as one of the state-of-the-arts and has been applied to several other tasks (Bahdanau et al., 2014; Luong et al., 2015; Rush et al., 2015; Chan et al., 2016).

Although NN-based dialogue agents have shown convincing performance on several tasks, it is not clear whether they also work well when facing malicious users or agents. To answer this question, we study how to evaluate the robustness of a goal-oriented dialogue system. For simplicity, we consider a goal-oriented agent A that aims to maximize some score, and define the "robustness" of A as the worst-case performance under any feasible agent A'. We also call A' an **adversarial agent** that tries to "attack" A since it aims to minimize A's score. The problem of evaluating the robustness of A can then be solved by designing an adversarial agent to attack A. For instance, considering a negotiation agent that can decide when to make a deal, we say the agent is not robust if an adversarial agent can fool the target agent to make a deal with significant lower scores. Ideally, before deploying an agent into real systems, we need to ensure it performs smoothly under strong adversarial attacks.

The concept of adversarial agent is related to recent studies on adversarial examples for image classifiers—it has been shown that a carefully designed small perturbation can easily make neural networks mis-classify (Goodfellow et al., 2014; Szegedy et al., 2013; Moosavi Dezfooli et al., 2016; Carlini and Wagner, 2017; Cheng et al., 2019), and several recent works has extended these attacks to natural language processing models such as sentiment analysis (Gao et al., 2018; Yang et al., 2018) and machine translation (Ebrahimi et al., 2018; Cheng et al., 2018). However, all of the previous work consider attacking a static model, where except input image/sentence there is no interaction between the attacker and the target model. Instead, we investigate a much more challenging problem, where there can be many turns of interactions between adversarial and target agents. This leads to several difficulties including 1) How to lead the target agent to a bad state and 2) how to force the target agent to make a wrong decision. Therefore, previous methods for attacking static models cannot be directly applied.

In this paper, we tackle the aforementioned challenges by proposing several novel ways to design an adversarial agent to evaluate the robustness of goal-oriented dialogue systems. We highlight our major contributions as follows:

- We propose a framework to generate adversarial agents in both black-box and white-box settings. To the best of our knowledge, this is the first work on crafting adversarial agents instead of adversarial examples in an interactive dialogue system.
- We conduct a series of studies on the negotiation agent proposed in (Lewis et al., 2017). We demonstrate that the proposed strategies can successfully attack existing negotiation agents to significantly reduce their average score. For instance, our attacks can reduce the average advantage of the RL-based negotiation agent from 2.68 to -5.76 on random problems with the total value of 10.
- We also show that through the proposed iterative adversarial training procedure, we could significantly improve the robustness of a goal-oriented agent against various attacks.

2 Related work

2.1 Goal-oriented dialogue agent

Goal-oriented dialogue systems aim at building a conversation model that is capable of accomplishing tasks through the interactions with human using natural language (Li et al., 2017; Eric and Manning, 2017; Wen et al., 2016; Wei et al., 2018; Bordes et al., 2016). Traditional approaches to learn a goal-oriented intelligent agent relies heavily on dialogue states annotated in the training data (Wen et al., 2016; Henderson et al., 2014). The use of state annotations allows a cleaner separation of the reasoning and natural language aspects of dialogues. However, it is very expensive to annotate every state in a large amount of training data. (Bordes et al., 2016) explores end-to-end goal orientated dialogue with a supervised model. And (He et al., 2017) uses task-specific rules to combine the task input and dialogue history into a more structured state representation. Recently, reinforcement learning has been widely used in dialogue systems to increase the agent versatility (Mordatch and Abbeel, 2017) and improve the agent's performance in goal-oriented tasks such as cooperative bot-bot dialogues (Das et al., 2017) and negotiation tasks (Lewis et al., 2017).

2.2 Adversarial examples in NLP applications

Algorithms have been proposed to craft adversarial sentences in NLP applications. (Papernot et al., 2016) uses Fast Gradient Sign method to generate adversarial example on RNN/LSTM based model. (Li et al., 2016) learns the importance of words by deleting them in sentiment analysis task and then use reinforcement learning to locate such words. (Samanta and Mehta, 2017) and (Liang et al., 2017) generate adversarial sequences by inserting or replacing existing words with typos and synonyms. (Gao et al., 2018) aims to attack sentiment classification models in a black-box setting. It develops some scoring functions to find the most important words to modify. (Jia and Liang, 2017) aims to fool the SQuAD reading comprehension system by adding crafted sentences. (Yang et al., 2018) proposes a greedy algorithm to swap the word/character and uses a Gumbel softmax function to reduce the computation. (Ebrahimi et al., 2018) aims to generate adversarial examples on character CNN model in machine translation problem by using Jacobian matrix to determine which word/character should be replaced or deleted. (Zhao et al., 2017) generated natural adversarial example using Generative Adversarial Networks (GANs). (Cheng et al., 2018) proposed a framework to conduct non-overlapping and targeted keyword attack on seq2seq model.

All the above-mentioned work focus on the static setting, i.e., the input does not depend on the model's output. However, in our work, one agent's input depends on the other agent's output, which makes the input undecidable in the beginning. Therefore, an adversarial sentence or example is not enough to conduct attack in dialogue systems. Instead, for the first time, we propose novel ways to construct a adversarial agent, which can bait the target agent to step to a wrong state and make a bad decision.

2.3 Defense against adversarial examples

Many defense algorithms have been proposed recently to enhance the robustness of classification models. Among them, adversarial training (Madry et al., 2018; Goodfellow et al., 2014) has become one of the most successful methods, which uses both clean and adversarial examples to train a robust model. (Wong and Kolter, 2018) proposed another kind of adversarial training to improve the verification lower bound of neural networks; (Liu and Hsieh, 2019) combines the idea of generative adversarial network (GAN) and adversarial training to further boost the robustness on test images. Another promising way to enhance robustness is by adding randomness to the model. (Liu et al., 2018) shows adding randomness to both input and intermediate layers of neural networks can improve robustness; (Liu et al., 2019; Ye and Zhu, 2018) show that combining Bayesian neural network (with randomized weights) and adversarial training can achieve state-of-the-art adversarial error under attacks. However, all the existing defense methods only work for static models (usually for classification tasks). In this paper, we propose an adversarial training algorithm for an agent using RL with an "adversarial agent". This is to our knowledge the first algorithm for improving the robustness of an agent.

3 Competitive Negotiation Dialogues

We use the negotiation agent developed in (Lewis et al., 2017) as the running example in this paper. Note that our algorithm can be generalized to other goal-oriented dialogue systems by designing a different scoring function according to the task.

In a competitive negotiation dialogue setting, two agents are negotiating with each other over a set of items. We adopt the same setting as (Lewis et al., 2017), in which case items can be categorized into either a ball, a hat or a book. Each agent is given the goal of the conversation (denoted by g), which contains the initial values and the quantities of each of the three items. Agents then negotiate to maximize the total value of their possessed items. Agents are allowed to negotiate up to a maximum of 10 turns. Scores will be granted to agents based on the total value of the items if they reach an agreement. If they choose not to agree, 0 score will be granted to both agents. A competitive negotiation dialogue example played by human and agent could be found in Table 1.

Input		
Human	3x**book** value 2 3x**hat** value 1 1x**ball** value 1	
Agent	3x**book** value 1 3x**hat** value 2 1x**ball** value 1	
Human	I'd like the books and the hats.	
Agent	I will take the books and hats.	
Human	I need the books and at least one other object.	
Agent	Ok, you can have one book and one ball	
Human	If I can have all books, I can leave you the rest.	
Agent	Deal	
Human	$\langle selection \rangle$	
Output		**Reward**
Human	3x**hat** 1x**ball**	7/10
Agent	3x**book**	6/10

Table 1: Competitive negotiation dialogue generated between agent and human.

We assess the robustness of a trained end-to-end negotiation agent used in (Lewis et al., 2017). In the negotiation chatbot setting, agents first chat using natural language and then make a selection based on what they have chatted with. We refer to the first phase as **negotiation phase** and the second phase as **decision phase**.

In the negotiation phase, conversation response at time t, x_t is generated word by word based on chat history $x_{0..t-1}$ and the goal of the conversation g. The conversation model is controlled by a speaking module θ and tokens are randomly sampled from probability distribution p_θ. This process continues recursively until an end-of-sentence token $\langle EOS \rangle$ or selection token $\langle selection \rangle$ token is generated. When $\langle EOS \rangle$ is encountered, the turn terminates and the conversation is handled to another agent. When $\langle selection \rangle$ is encountered, the negotiation phase terminates and the negotiation will reach the decision phase.

$$x_t \sim p_\theta(x_t | x_{0...t-1}, g) \tag{1}$$

In the decision phase, both agents will output a decision o based on a decision module probability distribution p'_θ. Agents' decisions will be based on conversation history $x_{0...T}$ up to the current time step T and the goal of the conversation g. Here O is a set of all legitimate selections, which is defined to be a space of where each selection must be greater or equal than 0 and the sum of selections for the same item must be equal to its original quantity. Since we only have a few items, it is possible to enumerate all the possibilities to get the set O.

$$o^* = \arg\max_{o \in O} \prod_i p'_\theta(o_i | x_{0...T}, g) \qquad (2)$$

Agents will then collect rewards (i.e. scores) from the environment (which will be 0 if they output conflicted decisions, e.g. the total number of items are different from the initial amount). It is important to keep the agent producing sentences that are correct both grammatically and semantically and keeping them competitive at the same time. Therefore, a common strategy is to train agents using supervised learning to learn natural language and to use reinforcement learning to optimize models' performance using on goal-oriented learning. We measure two statistics **score** and **agreement**. **score** is the average score for each agent (0-10). **agreement** is the percentage of dialogues where both agents agreed on the same decision. To measure the extent of success of our adversarial agent, we use **advantage** which is easy to compute directly from adversarial agent score minus target agent score, i.e. $S_{adv} - S_{ori}$.

4 Proposed Black-box Attack Algorithms

We first build our adversarial agent in black-box setting. Black-box setting in goal-oriented dialogue system is defined where the target agent is unknown to the attacker, but it is possible to make queries to obtain the final decision made by the target agent. To be noted, our aim is to test the robustness of the target agent. Therefore, in the decision phase we let adversarial agent chooses the complementary of target agent's choice, so those two agents will always reach agreement. The adversarial agent thus only has the speaking module and there is no decision network needed. In this section we proposed two adversarial agents in the black-box setting.

4.1 Reinforcement learning attack

Inspired by the procedure of goal-based reinforcement learning, we modified the reward function of our adversarial agent with the advantage instead of the score he got:

$$r^{adv} = S_{adv} - S_{ori} \qquad (3)$$

where S_{adv} and S_{ori} are adversarial agent score and target agent score respectively. After a complete dialogue has been generated, we update adversarial agent's parameters based on the outcome of the negotiation.

To learn the adversarial agent's speaking network by reinforcement learning, we denote the subset of tokens generated by the adversarial agent as X^{adv}. In the completed dialogue, γ is the discount factor that rewards actions at the end of the dialogue more strongly, and μ is a running average of completed dialogue rewards so far. We define the future reward R for an action $x_t \in X^{adv}$ as follows:

$$R(x_t) = \sum_{x_t \in X^{adv}} \gamma^{T-t}(r^{adv} - \mu). \qquad (4)$$

Then by a standard policy gradient algorithm, we could train our adversarial agent. Note that this attack doesn't require the knowledge on the target agent's structure/weights, and the experimental results demonstrate significant attack performance over regular agents.

4.2 Transfer attack

Transfer attack is a popular idea for attacking black-box models (Papernot et al., 2017). In dialogue systems, we can also consider the following transfer process: a sentence that leads to low r^{adv} in one dialogue might also lead to similar results in another dialogue. To implement this idea, we first collect a list of last sentences spoken by the adversarial agent from dialogues with high reward, denoted by L. In the conversations, we let our adversarial agent and the target agent negotiate n turns using the regular speaking module, and then plug in one sentence in L at the $(n+1)$-th turn. Our experimental results show that this transfer attack does not work well in practice.

5 Proposed White-box Attack Algorithms

In the white-box setting, we assume that the attacker can access every part of the target agent, including the weights of both speaking and decision models, and the decision output in every dialogue. Similar to the black-box attacks, we let the adversarial agent choose the complementary of target agent's choices to ensure 100% agreement. By exploiting the knowledge of the target agent's model, white-box attacks can achieve much higher advantage than black-box attacks.

5.1 Force target agent to select at a fixed turn

To begin with, we consider a simplified strategy where we first let our adversarial agent and the target agent negotiate n turns using regular speaking module. For the $(n+1)$-th turn, we propose the following two ways to modify the output of regular speaking module to maximize the rewards of adversarial agent.

5.1.1 Reactive attack

The first strategy is that the adversarial agent produces a sentence that forces the target agent to say $\langle selection \rangle$. The conversation will then enter the decision phase. At the same time, the sentence produced by the adversarial agent should guide the target agent to make a bad selection that would be in favor of the adversarial agent. We call this method **reactive attack**.

We formulate this strategy as an optimization problem. Let $\hat{\mathbf{x}} = x_{t_n \ldots T-1}$ be the output sentence generated by adversarial agent in the speaking model after n -th turn. Specifically, we define $x_{0 \ldots T-1}$ as all the tokens in the dialogue history before $\langle selection \rangle$. $Z_r(x_{0 \ldots T-1})$ indicates the logit layer outputs for predicting x_T based on chat history $x_{0 \ldots T-1}$ in the speaking model. $Z_o(x_{0 \ldots T})$ indicates the logit layer outputs on conversation history $x_{0 \ldots T}$ in the decision model. Because we have a constraint to force the target agent to say the end-of-dialog token $\langle selection \rangle$, we could format this constraint as

$$[Z_r(x_{0 \ldots T-1})]_{k_{sel}} - \max_{i \neq k_{sel}} [Z_r(x_{0 \ldots T-1})]_i \geq 0 \quad (5)$$

where k_{sel} is the corresponding index of end-of-dialog token $\langle selection \rangle$.

At the same time, the score of output o should be in favor of our adversarial agent. Assume the original decision output is o',

$$L(\hat{\mathbf{x}}) = \max\{[Z_o(x_{0 \ldots T})]_{o'} - \max_{o \in \bar{O}} [Z_o(x_{0 \ldots T})]_o, -\kappa\} \quad (6)$$

where $\bar{O}$ is the set of outputs that score of adversarial agent is greater than target agent i.e. $\bar{O} = \{o \in \bar{O} | S_{adv}(o) > S_{ori}(o)\}$, and $\kappa \geq 0$ denotes the confidence margin parameter. Note that $\hat{\mathbf{x}}$ is a sub-sequence in $x_{0 \ldots T}$, so the right hand side of (6) is a function of $\hat{\mathbf{x}}$.

Combining these two equations together, we can get our final objective function:

$$\begin{aligned} \min_{\hat{\mathbf{x}}} \quad & L(\hat{\mathbf{x}}) \\ \text{s.t. } & [Z_r(x_{0 \ldots T-1})]_{k_{sel}} - \max_{i \neq k_{sel}} [Z_r(x_{0 \ldots T-1})]_i \geq 0 \end{aligned} \quad (7)$$

Eq (7) is a discrete optimization problem since $\hat{x}$ is the sentence produced by adversarial agent.

In this paper, we use a modified version of the greedy algorithm to optimize (7). Although the original algorithm proposed in (Yang et al., 2018) only considered the unconstrained discrete problem, we show that the following slightly modified version performs well for solving (7). At each iteration, we try to replace each word in $\hat{x}$ by the special token $\langle PAD \rangle$. A word that achieves minimal loss after swapping with $\langle PAD \rangle$ is then selected as the word to be replaced. Then we try to replace the selected word with each word in the vocabulary. For all the trials that satisfy the constraint, we choose the one with minimal loss and conduct the actual change. We run this procedure iteratively to minimize (7). In the experiments, we only replace two words in $\hat{x}$ to ensure the fluency and correctness of the adversarial sentences.

5.1.2 Preemptive attack

The other attack strategy is to produce a sentence to guide the target agent to lower its demand in the reply instead of making target agent say end-of-dialog token. And after the reply from target agent, the adversarial agent speaks the end-of-dialogue token to enter the decision phase. Similar to the reactive attack, adversarial agent's score should be greater than target agent's score in the decision phase. Clearly, this strategy is more challenging than the previous one because there is an intermediate sentence spoken by the target agent before end-of-dialogue. We call this preemptive attack.

Let $\hat{\mathbf{x}} = x_{t_n \ldots t_{n_T}}$ be the output sentence generated by adversarial agent in the speaking model after turn n, where t_n is the first word and t_{n_T} is the last word of the sentence. Similarly, we could formally turn the intuition into optimization problem as follows:

$$L(\hat{\mathbf{x}}) = \max\{[Z_o(x_{0 \ldots T})]_{o'} - \max_{o \in \bar{O}} [Z_o(x_{0 \ldots T})]_o, -\kappa\} \quad (8)$$

Since we do not need to force target agent to say end-of-dialogue, the problem becomes an unconstrained discrete optimization problem. We then

Algorithm 1 Arbitrary turn attack algorithm

```
Input: Target agent B, Input goal $g$
Output: Dialogue $x_{0...T}$, Agent score $S_{adv}$ and $S_{ori}$
while $\langle selection \rangle$ is not generated do
  Set the loss $L(\cdot)$ to be (7)
  Optimize the Loss $L(\cdot)$
  if $L(\cdot) < 0$ then
    Add the output into the dialogue
  else
    Set the loss $L(\cdot)$ in to be (8)
    Optimize the Loss $L(\cdot)$
    if $L(\cdot) < 0$ then
      Add the output into the dialogue
    else
      if Transfer Attack then
        Randomly add a sentence in $L$ (malicious sentences) into the dialogue.
      else
        Add the sentence generated by regular speaking model into the dialogue (delayed attack).
      end if
    end if
  end if
end while
Generate $o$ using dialogue $x_{0...T}$
Calculate $S_{adv}$ and $S_{ori}$
Return: $x_{0...T}$,$S_{adv}$,$S_{ori}$
```

directly apply the unconstrained version of greedy algorithm (Yang et al., 2018) to solve it.

5.2 Force target agent to select at arbitrary turn

While we could let our adversarial agent and the target agent negotiate n turns, it is still unknown which n should be chosen to get the best performance. In other words, we aim to not only know what to say but also when to say to fool the target agent.

We propose two strategies to force target agent to make bad decisions at arbitrary turn. The details are presented in Algorithm 1. When it is the turn for adversarial agent to speak, we first try to apply reactive and preemptive attacks. If both attacks couldn't make the loss $L(\cdot)$ less than 0, there are two strategies: 1) just output the sentence generated by the regular speaking module (delayed attack), and 2) conduct transfer attack. The comparisons can be found in the experiments.

6 Adversarial Training

Adversarial training is a popular method to improve the robustness of machine learning models (Miyato et al., 2016; Madry et al., 2018). In this section, we use the agents designed in the previous sections to improve the robustness of the target agent.

In standard adversarial training for neural network models (Goodfellow et al., 2014; Jia and Liang, 2017), adversarial examples (images or sentences) generated by an attack are added to the training procedure to refine the model. Since our setting is interactive and there is no fixed data used in selfplay, we should conduct training with *adversarial agents* instead of adversarial examples. Moreover, as pointed out by (Jia and Liang, 2017), training on the examples generated by a single attack will lead to over-fitting to a particular attack, so we should do adversarial training iteratively.

Taking the black-box RL agent as an example, we consider the following min-max formulation:

$$\min_{\theta^{ori}}\{\max_{\theta^{adv}} S_{adv} - S_{ori}\}, \qquad (9)$$

where θ^{ori} is the weights for the target agent and θ^{adv} is the weights for the adversarial black-box agent. We solve (9) by the following alternating minimization procedure. At each iteration, we first update the target agent (θ^{ori}) using the standard policy gradient algorithm, and then use our RL algorithm in Section 4.1 to update adversarial agent to counter the target model. We iteratively conduct these updates until convergence. The experiments show that the adversarial training procedure can improve the robustness not only under RL attack but also under other white-box attacks.

7 Experimental Results

We perform extensive experiments on evaluating the robustness of the negotiation agents developed in (Lewis et al., 2017). Furthermore, we show that the robustness of negotiation agents can be significantly improved using the proposed adversarial training procedure. Our codes are publicly available at `https://github.com/cmhcbb/Robustness-of-Dialogue-systems`.

7.1 Experimental Setup

We use the code released by the authors (Lewis et al., 2017) and follow their instructions to get the target end-to-end negotiation agents. More specifically, we first train the model on 5808 dialogues, based on 2236 unique scenarios in supervised way to imitate the actions of human users. We call this model supervised model (SV agent). Then we use reinforcement learning to conduct goal-oriented training in order to maximize the agent' reward. The second model is called the reinforcement learning model (RL agent). As a result, when doing selfplay between RL agent and SV agent, we could get RL agent with 5.86 perplexity, 89.57% agreement and 7.23 average score, while SV agent achieves 5.47 perplexity and 4.55 average score. These numbers are similar to the numbers reported in (Lewis et al., 2017).

To evaluate the robustness of these agents, we conduct all the proposed attacks on both supervised model (SV agent) and reinforcement learning model (RL agent). The successfulness of an attack is measured by average score advantage and positive advantage rate (PAR). Average score advantage is defined by averaged adversarial agent's score minus average target agent's score. The value is in the region of $[-10, 10]$ since the total values are controlled to be 10 for both sides, and a larger advantage indicates a more successful attack. Also, we define positive advantage rate (PAR) as the ratio of dialogues that the adversarial agent gets a higher score than the target agent. We will see that most attacks developed in this paper will improve both average score advantage and PAR. Note that this is the first work on attacking a goal-oriented dialogue agent so there is no previous method that could be included in the comparisons.

7.2 Results on Black-box Attacks

As introduced in Section 4, we have two black-box attacks: reinforcement learning attack (RL attack) and Transfer attack.

RL Attack. In the reinforcement learning attack, we use a learning rate of 0.1, clip gradients above 1.0, and set the discount factor $\gamma = 0.95$ in (4). We train the adversarial agent for 4 epochs on all scenarios. From Table 2, we observe that with 100% agreement rate, our adversarial agent could gain 2.32 score advantage against the RL agent and 4.25 advantage against the SV agent. Also, our agent achieves a relatively high positive advantage rate at 84.45% and 69.35% respectively. We show some adversarial dialogues played by adversarial agent and target agent in Table 3. It shows that RL agent is able to identify the weak point of target agent by saying ”take book you get rest”, which could easily let the agent accept the deal and make a bad selection that is inconsistent with the context of dialogue.

Transfer attack. In transfer attack, we first let our adversarial agent speak the sentence generated by the speaking model with target agent for 3 turns. If the end-of-dialog token has never been mentioned, in the 4th turn, the adversarial agent speaks the sentence generated by our RL agent. The detailed results are shown in Table 2. We observe that the transfer attack is not successful—only -0.13 and -1.189 score advantage are achieved. We found that transferring sentences between dialogues is not successful because the item values and conversation histories are quite different between dialogues.

7.3 Results on White-box Attacks

We conduct the white-box attacks introduced in Section 5.

Force target agent to select at a fixed turn. There are two types of algorithms (reactive attack and preemptive attack) introduced in Section 5.1. The detailed results are shown in Table 2. We observe that the reactive attack could achieve better results than black-box method with 5.40 score advantage against SV agent and 4.98 score advantage against RL agent. On the other hand, preemptive attack is not that successful—it gets 2.81 advantage against SV agent and 0.77 score advantage against RL agent. Furthermore, we have included some adversarial dialogues played by white-box adversarial agent and target agent in Table 4. From these examples, we could see that white-box adversarial agent could generate the adversarial sentences, slightly unnatural however still readable, that could fool the target agent to make terrible decisions.

Force target agent to select at arbitrary turn. To determine when should we begin the attack, we design combinations of reactive attack, preemptive attack and transfer attack or delayed attack in Section 5.2. Here, we conduct experiments to validate the effectiveness of these two attack combinations. From Table 2, the combinations achieve better results than all the previous attacks. The best result is achieved by the combination of reactive attack, preemptive attack and delayed attack

	vs SV agent			vs RL agent		
Model	**PAR%**	**Score(advantage)**	**Agreement%**	**PAR%**	**Score(advantage)**	**Agreement%**
RL model(w/o attack)	75.79	7.23 vs 4.55 (2.68)	89.57	44.70	5.05 vs 5.00 (0.05)	76.36
Transfer attack	44.43	6.41 vs 6.54 (-0.13)	100	36.10	5.65 vs 6.84 (-1.19)	100
RL attack	84.45	8.28 vs 4.03 (4.25)	100	69.35	7.11 vs 4.79 (2.32)	100
Reactive attack	87.00	8.83 vs 3.43 (5.40)	100	90.23	8.72 vs 3.77 (4.95)	100
Preemptive attack	71.86	7.76 vs 4.95 (2.81)	100	69.23	6.78 vs 6.01 (0.77)	100
RA+PA+DA	84.33	8.79 vs 2.96 (5.83)	100	86.93	8.73 vs 2.95 (5.78)	100
RA+PA+TA	83.12	8.67 vs 3.05 (5.62)	100	89.74	8.62 vs 2.92 (5.70)	100

Table 2: Negotiation task evaluation with different adversarial agent on 2000 randomly generated scenarios, against the supervised model and reinforcement learning model. The maximum socre is 10. When agents failed to agree, all agents get 0 score. PAR stands for positive advantage rate. RA+PA+DA stands for the combination of reactive attack, preemptive attacka and delayed attack. RA+PA+TA stands for the combination of reactive attack, preemptive attacka and transfer attack.

Input		
Adv agent	1x**book** value 1 4x**hat** value 1 1x**ball** value 5	
RL agent	1x**book** value 2 4x**hat** value 1 1x**ball** value 4	
Adv agent	i want the hats and 2 balls	
RL agent	i need the balls and the hat	
Adv agent	take book you get rest	
RL agent	deal	
Adv agent	$\langle selection \rangle$	
Output		**Reward**
Adv agent	4x**hat** 1x**ball**	9/10
RL agent	1x**book**	2/10

Table 3: Dialogue example generated by black-box RL attack agent against RL agent.

Input		
Adv agent	1x**book** value 0 1x**hat** value 1 3x**ball** value 3	
RL agent	1x**book** value 1 1x**hat** value 0 3x**ball** value 3	
Adv agent	i would like the balls and the hat	
RL agent	i need the balls and the book	
Adv agent	i need the balls and fine book	
RL agent	$\langle selection \rangle$	
Output		**Reward**
Adv agent	1x**hat** 1x**book** 3x**ball**	10/10
RL agent		0/10

Table 4: Dialogue example generated by reactive attack agent against RL agent.

Input		
Adv agent	1x**book** value 4 2x**hat** value 1 2x**ball** value 2	
RL agent	1x**book** value 8 2x**hat** value 0 2x**ball** value 1	
RL agent	i would like the book and the hat.	
Adv agent	i want reasonable balls and book	
RL agent	$\langle selection \rangle$	
Output		**Reward**
Adv agent	1x**book** 2x**ball**	8/10
RL agent	2x**hat**	0/10

Table 5: Dialogue example generated by RA+PA+DA attack agent against RL agent.

(RA+PA+DA), which gets 5.83 advantage against SV agent and 5.78 score advantage against RL agent, with very high positive advantage rates at 84.33% and 86.93% respectively. We have included some adversarial dialogues played by this adversarial agent and the target agent in Table 5. We observe that with the delayed attack, the adversarial agent can decide **when to attack**, thus achieves much better performance than attacking at a fixed turn.

7.4 Adversarial Training

Using the algorithm proposed in Section 6, we conduct adversarial training using the black-box RL attack model. The results are shown in Table 6. First, we observe that the adversarial trained model achieves much better performance against black-box RL attack; the advantage of RL attack drops from 2.32 to -1.8. Moreover, the model achieves consistently better performance against other white-box attacks. For instance, the advantage of the strongest RA+PA+DA attack is reduced from 5.78 to 3.98.

	vs advtrain model		
Model	**PAR%**	**Score(advantage)**	**Agreement%**
RL model(w/o attack)	48.67	6.51 vs 6.64 (-0.13)	91.75
Transfer attack	23.05	4.93 vs 7.59 (-2.66)	100
RL attack	62.61	5.71 vs 7.51 (-1.80)	100
Reactive attack	80.76	8.83 vs 4.31 (4.52)	100
Preemptive attack	34.39	5.64 vs 7.41 (-1.77)	100
RA+PA+DA	73.96	8.05 vs 4.07 (3.98)	100
RA+PA+TA	73.45	8.06 vs 4.13 (3.93)	100

Table 6: Negotiation task evaluation with different adversarial agent on 2000 randomly generated scenarios, against adversarial trained model.

7.5 Analysis and Discussions

RL agents are more robust than SV agents. From Table 2, we could see that all the attack methods perform better when facing SV agents than RL agents. It is because that SV agents only learn to mimic human's action and are trained only on human data. Therefore, it is reasonable that RL agents are more robust than SV agents.

The importance of arbitrary turns. In reactive attack and preemptive attack, we begin our attack at the n-th turn and we set $n = 2$ in the experiments. Here we show the results with different n in Table 7. We observe that the performance of white-box attacks are quite consistent with different choices of n. This probably indicates that there the best n varies for different cases. Therefore, if we could change the n from case to case adaptively, which is done by delayed attack, we could see a performance boost.

n	PAR%	Score(advantage)	Agreement%
1	94.02	8.84 vs 3.32 (5.52)	100
2	90.23	8.72 vs 3.77 (4.95)	100
3	92.02	8.81 vs 3.62 (5.19)	100
4	90.35	8.71 vs 3.87 (4.84)	100

Table 7: Negotiation task evaluation with different choices of n against RL model.

Adversarial training helps to improve the robustness. We then try to investigate the robustness of the adversarial trained model. We found that in the original model, it is easy for an attacker to find a sentence to quickly end the dialogue. However, after adversarial training, it becomes much harder to find such sentences. Moreover, although we only conduct adversarial training on black-box RL model, the adversarial trained model still achieves better performance against other white-box attacks. This indicates that the adversarial trained model could probably detect the slight unnaturalness of those sentences and thus have a better reading comprehension ability.

8 Conclusion

In this paper, we develop adversarial agents to evaluate the robustness of a goal-oriented dialogue system. Our experimental results show that the current NN-based models are not robust against our adversarial agents. Furthermore, by iterative adversarial training using our black-box RL agent, we can significantly improve the robustness of the dialogue system.

Acknowledgments

We acknowledge the support by NSF IIS1719097, Intel faculty award, Google Cloud and AITRICS.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

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

Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4960–4964. IEEE.

Minhao Cheng, Thong Le, Pin-Yu Chen, Jinfeng Yi, Huan Zhang, and Cho-Jui Hsieh. 2019. Query-efficient hard-label black-box attack: An optimization-based approach. In *ICLR*.

Minhao Cheng, Jinfeng Yi, Huan Zhang, Pin-Yu Chen, and Cho-Jui Hsieh. 2018. Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. *arXiv preprint arXiv:1803.01128*.

Abhishek Das, Satwik Kottur, José MF Moura, Stefan Lee, and Dhruv Batra. 2017. Learning cooperative visual dialog agents with deep reinforcement learning. *arXiv preprint arXiv:1703.06585*.

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

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 31–36.

Mihail Eric and Christopher D Manning. 2017. Key-value retrieval networks for task-oriented dialogue. *arXiv preprint arXiv:1705.05414*.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. *arXiv preprint arXiv:1801.04354*.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. 2017. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. *arXiv preprint arXiv:1704.07130*.

Matthew Henderson, Blaise Thomson, and Jason D Williams. 2014. The second dialog state tracking challenge. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 263–272.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*.

Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning of negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2443–2453.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

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

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2017. Deep text classification can be fooled. *arXiv preprint arXiv:1704.08006*.

Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. 2018. Towards robust neural networks via random self-ensemble. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 369–385.

Xuanqing Liu and Cho-Jui Hsieh. 2019. Adv-gan: Generator, discriminator, and adversarial attacker. In *CVPR*.

Xuanqing Liu, Yao Li, Chongruo Wu, and Cho-Jui Hsieh. 2019. Ad v-bnn: Improved adversarial defense through robust bayesian neural network. In *ICLR*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *ICLR*.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Seyed Mohsen Moosavi Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, EPFL-CONF-218057.

Igor Mordatch and Pieter Abbeel. 2017. Emergence of grounded compositional language in multi-agent populations. *arXiv preprint arXiv:1703.04908*.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM.

Nicolas Papernot, Patrick McDaniel, Ananthram Swami, and Richard Harang. 2016. Crafting adversarial input sequences for recurrent neural networks. In *Military Communications Conference, MILCOM 2016-2016 IEEE*, pages 49–54. IEEE.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*.

Suranjana Samanta and Sameep Mehta. 2017. Towards crafting text adversarial samples. *arXiv preprint arXiv:1707.02812*.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, volume 16, pages 3776–3784.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Wei Wei, Quoc Le, Andrew Dai, and Jia Li. 2018. Airdialogue: An environment for goal-oriented dialogue research. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3844–3854.

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*.

Eric Wong and J Zico Kolter. 2018. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*.

Puyudi Yang, Jianbo Chen, Cho-Jui Hsieh, Jane-Ling Wang, and Michael I Jordan. 2018. Greedy attack and gumbel attack: Generating adversarial examples for discrete data. *arXiv preprint arXiv:1805.12316*.

Nanyang Ye and Zhanxing Zhu. 2018. Bayesian adversarial learning. In *NIPS*.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2017. Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342*.