Adversarial Over-Sensitivity and Over-Stability Strategies for Dialogue Models

Tong Niu and **Mohit Bansal**
UNC Chapel Hill
{tongn, mbansal}@cs.unc.edu

Abstract

We present two categories of model-agnostic adversarial strategies that reveal the weaknesses of several generative, task-oriented dialogue models: *Should-Not-Change* strategies that evaluate over-sensitivity to small and semantics-preserving edits, as well as *Should-Change* strategies that test if a model is over-stable against subtle yet semantics-changing modifications. We next perform adversarial training with each strategy, employing a max-margin approach for negative generative examples. This not only makes the target dialogue model more robust to the adversarial inputs, but also helps it perform significantly better on the original inputs. Moreover, training on all strategies combined achieves further improvements, achieving a new state-of-the-art performance on the original task (also verified via human evaluation). In addition to adversarial training, we also address the robustness task at the model-level, by feeding it subword units as both inputs and outputs, and show that the resulting model is equally competitive, requires only 1/4 of the original vocabulary size, and is robust to one of the adversarial strategies (to which the original model is vulnerable) even without adversarial training.

1 Introduction

Adversarial evaluation aims at filling in the gap between potential train/test distribution mismatch and revealing how models will perform under real-world inputs containing natural or malicious noise. Recently, there has been substantial work on adversarial attacks in computer vision and NLP. Unlike vision, where one can simply add in imperceptible perturbations without changing an image's meaning, carrying out such subtle changes in text is harder since text is discrete in nature (Jia and Liang, 2017). Thus, some previous works have either avoided modifying original source inputs and only resorted to inserting distractive sentences (Jia and Liang, 2017), or have restricted themselves to introducing spelling errors (Belinkov and Bisk, 2018) and adding non-functioning tokens (Shalyminov et al., 2017). Furthermore, there has been limited adversarial work on generative NLP tasks, e.g., dialogue generation (Henderson et al., 2017), which is especially important because it is a crucial component of real-world virtual assistants such as Alexa, Siri, and Google Home. It is also a challenging and worthwhile task to keep the output quality of a dialogue system stable, because a conversation usually involves multiple turns, and a small mistake in an early turn could cascade into bigger misunderstanding later on.

Motivated by this, we present a comprehensive adversarial study on dialogue models – we not only simulate imperfect inputs in the real world, but also launch intentionally malicious attacks on the model in order to assess them on both over-sensitivity and over-stability. Unlike most previous works that exclusively focus on Should-Not-Change adversarial strategies (i.e., non-semantics-changing perturbations to the source sequence that *should not change* the response), we demonstrate that it is equally valuable to consider Should-Change strategies (i.e., semantics-changing, intentional perturbations to the source sequence that *should change* the response).

We investigate three state-of-the-art models on two task-oriented dialogue datasets. Concretely, we propose and evaluate five naturally motivated and increasingly complex Should-Not-Change and five Should-Change adversarial strategies on the VHRED (Variational Hierarchical Encoder-Decoder) model (Serban et al., 2017b) and the RL (Reinforcement Learning) model (Li et al., 2016) with the Ubuntu Dialogue Cor-

We publicly release all our code and data at https://github.com/WolfNiu/AdversarialDialogue

Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL 2018), pages 486–496
Brussels, Belgium, October 31 - November 1, 2018. ©2018 Association for Computational Linguistics

pus (Lowe et al., 2015), and Dynamic Knowledge Graph Network with the Collaborative Communicating Agents (CoCoA) dataset (He et al., 2017).

On the Should-Not-Change side for the Ubuntu task, we introduce adversarial strategies of increasing linguistic-unit complexity – from shallow word-level errors, to phrase-level paraphrastic changes, and finally to syntactic perturbations. We first propose two rule-based perturbations to the source dialogue context, namely Random Swap (randomly transposing neighboring tokens) and Stopword Dropout (randomly removing stopwords). Next, we propose two data-level strategies that leverage existing parallel datasets in order to simulate more realistic, diverse noises: namely, Data-Level Paraphrasing (replacing words with their paraphrases) and Grammar Errors (e.g., changing a verb to the wrong tense). Finally, we employ Generative-Level Paraphrasing, where we adopt a neural model to automatically generate paraphrases of the source inputs.[1] On the Should-Change side for the Ubuntu task, we propose the Add Negation strategy, which negates the root verb of the source input, and the Antonym strategy, which changes verbs, adjectives, or adverbs to their antonyms. As will be shown in Section 6, the above strategies are effective on the Ubuntu task, but not on the collaborative-style, database-dependent CoCoA task. Thus for the latter, we investigate additional Should-Change strategies including Random Inputs (changing each word in the utterance to random ones), Random Inputs with Entities (like Random Inputs but leaving mentioned entities untouched), and Normal Inputs with Confusing Entities (replacing entities in an agent's utterance with distractive ones) to analyze where the model's robustness stems from.

To evaluate these strategies, we first show that (1) both VHRED and the RL model are vulnerable to most Should-Not-Change and all Should-Change strategies, and (2) DynoNet's robustness to Should-Change inputs shows that it does not pay any attention to natural language inputs other than the entities contained in them. Next, observing how our adversarial strategies 'successfully' fool the target models, we try to expose these models to such perturbation patterns early on during training itself, where we feed adversarial input context and ground-truth target pairs as training data. Importantly, we realize this adversarial training via a maximum-likelihood loss for Should-Not-Change strategies, and via a max-margin loss for Should-Change strategies. We show that this adversarial training can not only make both VHRED and RL more robust to the adversarial data, but also improve their performances when evaluated on the original test set (verified via human evaluation). In addition, when we train VHRED on all of the perturbed data from each adversarial strategy together, the performance on the original task improves even further, achieving the state-of-the-art result by a significant margin (also verified via human evaluation).

Finally, we attempt to resolve the robustness issue directly at the model-level (instead of adversarial-level) by feeding subword units derived from the Byte Pair Encoding (BPE) algorithm (Sennrich et al., 2016) to the VHRED model. We show that the resulting model not only reduces the vocabulary size by around 75% (thus trains much faster) and obtains results comparable to the original VHRED, but is also naturally (i.e., without requiring adversarial training) robust to the Grammar Errors adversarial strategy.

2 Tasks and Models

For a comprehensive study on dialogue model robustness, we investigate both semi-task-based troubleshooting dialogue (the Ubuntu task) and the new important paradigm of collaborative two-bot dialogue (the CoCoA task). The former focuses more on natural conversations, while the latter focuses more on the knowledge base. Consequently, the model trained on the latter tends to ignore the natural language context (as will be shown in Section 6.2) and hence requires a different set of adversarial strategies that can directly reveal this weakness (e.g., Random Inputs with Entities). Overall, adversarial strategies on Ubuntu and CoCoA reveal very different types of weaknesses of a dialogue model. We implement two models on the Ubuntu task and one on the CoCoA task, each achieving state-of-the-art result on its respective task. Note that although we employ these two strong models as our testbeds for the proposed adversarial strategies, these adversarial strategies are not specific to the two models.

[1] A real example of Generative-Paraphrasing: context "*You can find xorg . conf in /etc/X11 . It 's not needed unless it is . ;-) You may need to create one yourself .*" is paraphrased as "*You may find xorg . conf in /etc/X11 . It 's not necessary until it is . You may be required to create one .*"

2.1 Ubuntu Dialogue

Dataset and Task: The Ubuntu Dialogue Corpus (Lowe et al., 2015) contains 1 million 2-person, multi-turn dialogues extracted from Ubuntu chat logs, used to provide and receive technical support. We focus on the task of generating fluent, relevant, and goal-oriented responses.

Evaluation Method: The model is evaluated on F1's for both activities (technical verbs, e.g., "download", "install") and entities (technical nouns, e.g., "root", "web"). These metrics are computed by mapping the ground-truth and model responses to their corresponding activity-entity representations using the automatic procedure described in Serban et al. (2017a), who found that F1 is "particularly suited for the goal-oriented Ubuntu Dialogue Corpus" based on manual inspection of the extracted activities and entities. We also conducted human studies on the dialogue quality of generated responses (see Section 5 for setup and Section 6.1 for results).

Models: We reproduce the state-of-the-art Latent Variable Hierarchical Recurrent Encoder-Decoder (VHRED) model (Serban et al., 2016), and a Deep Reinforcement Learning based generative model (Li et al., 2016). For the VHRED model, we apply additive attention mechanism (Bahdanau et al., 2015) to the source sequence while keeping the remaining architecture unchanged. For the RL-based model, we adopt the mixed objective function (Paulus et al., 2018) and employ a novel reward: during training, for each source sequence S, we sample a response G on the decoder side, feed the encoder with a random source sequence S_{R} drawn from the train set, and use $-\log P(G|S_{\mathrm{R}})$ as the reward. Intuitively, if S_{R} stands a high chance of generating G (which corresponds to a large negative reward), it is very likely that G is dull and generic.

2.2 Collaborative Communicating Agents

Dataset and Task: The collaborative CoCoA[2] dialogue task involves two agents that are asymmetrically primed with a private Knowledge Base (KB), and engage in a natural language conversation to find out the unique entry shared by the two KBs. For a bot-bot chat of the CoCoA task, a bot is allowed one of the two actions each turn: performing an UTTERANCE action, where it generates an utterance, or making a SELECT action, where it chooses an entry from the KB. Note that each bot's SELECT action is visible to the other bot, and each is allowed to make multiple SELECT actions if the previous guess is wrong.

Evaluation Method: One of the major metrics is Completion Rate, the percentage of two bots successfully finishing the task.

Models: We focus on DynoNet, the best-performing model for the CoCoA task (He et al., 2017). It consists of a dynamic knowledge graph, a graph embedding over the entity nodes, and a Seq2seq-based utterance generator.

3 Adversarial Strategies

3.1 Adversarial Strategies on Ubuntu

For Ubuntu, we introduce adversarial strategies of increasing linguistic-unit complexity – from shallow word-level errors such as Random Swap and Stopword Dropout, to phrase-level paraphrastic changes, and finally to syntactic Grammar Errors.

Should-Not-Change Strategies

(1) Random Swap: Swapping adjacent words occurs often in the real world, e.g., transposition of words is one of the most frequent errors in manuscripts (Headlam, 1902; Marqués-Aguado, 2014); it is also frequently seen in blog posts.[3] Thus, being robust to swapping adjacent words is useful for chatbots that take typed/written text as inputs (e.g., virtual customer support on a airline/bank website). Even for speech-based conversations, non-native speakers can accidentally swap words due to habits formed in their native language (e.g., SVO in English vs. SOV in Hindi, Japanese, and Korean). Inspired by this, we also generate globally contiguous but locally "time-reversed" text, where positions of neighboring words are swapped (e.g., "*I don't want you to go*" to "*I don't want to you go*").

(2) Stopword Dropout: Stopwords are the most frequent words in a language. The most commonly-used 25 words in the Oxford English corpus make up one-third of all printed material in English, and these words consequently carry less information than other words do in a sentence.[4]

[2] https://stanfordnlp.github.io/cocoa/

[3] E.g., "*he would give **to it** me*" in https://talk.drugabuse.com/threads/his-behavior-this-week.4347/

[4] One could also use closed-class words (prepositions, determiners, coordinators, and pronouns), but we opt for stopwords because a majority of stopwords are indeed closed-class words, and secondly, closed-class words usually require a very accurate POS-tagger, which is not available for low-resource or noisy domains and languages (e.g., Ubuntu).

Inspired by this observation, we propose randomly dropping stopwords from the inputs (e.g., "*Ben ate the carrot*" to "*Ben ate carrot*").

(3) Data-level Paraphrasing: We repurpose PPDB 2.0 (Pavlick et al., 2015) and replace words and phrases in the original inputs with their paraphrases (e.g., "*She bought a bike*" to "*She purchased a bicycle*").

(4) Generative-level Paraphrasing: Although Data-level Paraphrasing provides us with semantic-preserving inputs most of the time, it still suffers from the fact that the validity of a paraphrase depends on the context, especially for words with multiple meanings. In addition, simply replacing word-by-word does not lead to new compositional sentence-level paraphrases, e.g., "*How old are you*" to "*What's your age*". We thus also experiment with generative-level paraphrasing, where we employ the Pointer-Generator Networks (See et al., 2017), and train it on the recently published paraphrase dataset ParaNMT-5M (Wieting and Gimpel, 2017) which contains 5 millions paraphrase pairs.

(5) Grammar Errors: We repurpose the AESW dataset (Daudaravicius, 2015), text extracted from $9,919$ published journal articles with data before/after language editing. This dataset was used for training models that identify and correct grammar errors. Based on the corrections in the edits, we build a look-up table to replace each correct word/phrase with a wrong one (e.g., "*He doesn't like cakes*" to "*He don't like cake*").

Should-Change Strategies

(1) Add Negation: Suppose we add negation to the source sequence of some task-oriented model — from "*I want some coffee*" to "*I don't want some coffee*". A proper response to the first utterance could be "*Sure, I will bring you some coffee*", but for the second one, the model should do anything but bring some coffee. We thus assume that if we add negation to the root verb of each source sequence and the response is unchanged, the model must be ignoring important linguistic cues like negation. Hence this qualifies as a Should-Change strategy, i.e., if the model is robust, it should change the response.

(2) Antonym: We change words in utterances to their antonyms to apply more subtle meaning changes (e.g., "*You need to install Ubuntu*" to "*You need to uninstall Ubuntu*").[5]

3.2 Adversarial Strategies on CoCoA

We applied all the above successful strategies used for the Ubuntu task to the UTTERANCE actions in a bot-bot-chat setting for the CoCoA task, but found that none of them was effective on DynoNet. This is surprising considering that the model's language generation module is a traditional Seq2seq model. This observation motivated us to perform the following analysis. The high performance of bot-bot chat may have stemmed from two sources: information revealed in an utterance, or entries directly disclosed by a SELECT action.

To investigate which part the model relies on more, we experiment with different Should-Change strategies which introduce obvious perturbations that have minimal word or semantic meaning overlap with the original source inputs:

(1) Random Inputs: Turn both bots' utterances into random inputs. This aims at investigating how much the model depends on the SELECT action.

(2) Random Inputs with Kept Entities: Replace each bot's utterance with random inputs, but keep the contained entities untouched. This further investigates how much entities alone contribute to the final performance.

(3) Confusing Entity: Replace entities mentioned in bot A's utterances with entities that are present in bot B's KB but not in their shared entry (and vice versa). This aims at coaxing bot B into believing that the mentioned entities come from their shared entry. By intentionally making the utterances misleading, we expect DynoNet's performance to be lower – hence this qualifies as a Should-Change strategy.

4 Adversarial Training

To make a model robust to an adversarial strategy, a natural approach is exposing it to the same pattern of perturbation during training (i.e., *adversarial training*). This is achieved by feeding adversarial inputs as training data. For each strategy, we report results under three train/test combinations: (1) trained with normal inputs, tested on adversarial inputs (*N-train* + *A-test*), which evaluates whether the adversarial strategy is effective at

[5] Note that Should-Change strategies may lead to contexts that do not correspond to any legitimate task completion action, but the purpose of such a strategy is to make sure that the model at least should not respond the same way as it responded to the original context, i.e., even for the no-action state, the model should respond with something different like "*Sorry, I cannot help with that.*" Our semantic similarity results in Table 4 capture this intuition directly.

fooling the model and exposing its robustness issues; (2) trained with adversarial inputs, tested on adversarial inputs (*A-train* + *A-test*), which next evaluates whether adversarial training made the model more robust to that adversarial attack; and (3) trained with adversarial inputs, tested on normal inputs (*A-train* + *N-test*), which finally evaluates whether the adversarial training also makes the model perform equally or better on the original normal inputs. Note that (3) is important, because one should not make the model more robust to a strategy at the cost of lower performance on the original data; also when (3) improves the performance on the original inputs, it means adversarial training successfully teaches the model to recognize and be robust to a certain type of noise, so that the model performs better when encountering similar patterns during inference. Also note that we use perturbed train set for adversarial training, and perturbed test set for adversarial testing. There is thus no overlap between the two sets.

4.1 Adversarial Training for Should-Not-Change Strategies

For each Should-Not-Change strategy, we take an already trained model from a certain checkpoint,[6] and train it on the adversarial inputs with maximum likelihood loss for K epochs (Shalyminov et al., 2017; Belinkov and Bisk, 2018; Jia and Liang, 2017; Iyyer et al., 2018). By feeding "adversarial source sequence + ground-truth response pairs" as regular positive data, we teach the model that these pairs are also valid examples despite the added perturbations.

4.2 Adversarial Training for Should-Change Strategies

For Should-Change strategies, we want the F1's to be lower with adversarial inputs after adversarial training, since this shows that the model becomes sensitive to subtle yet semantic-changing perturbations. This cannot be achieved by naively training on the perturbed inputs with maximum likelihood loss, because the "perturbed source sequence + ground-truth response pairs" for Should-Change strategies are negative examples which we need to train the model to avoid from generating. Inspired by Mao et al. (2016) and Yu et al. (2017), we instead use a linear combination of maximum likelihood loss and max-margin loss:

Model	Activity F1	Entity F1
LSTM	1.18	0.87
HRED	4.34	2.22
VHRED	4.63	2.53
VHRED (w/ attn.)	**5.94**	3.52
Reranking-RL	5.67	**3.73**

Table 1: F1 results of previous works as compared to our models. LSTM, HRED and VHRED are results reported in Serban et al. (2017a). VHRED (w/ attn.) and Reranking-RL are our results. Top results are bolded.

$$
\begin{aligned}
L &= L_{\mathrm{ML}} + \alpha L_{\mathrm{MM}} \\
L_{\mathrm{ML}} &= \sum_i \log P(t_i|s_i) \\
L_{\mathrm{MM}} &= \sum_i \max\left(0, M + \log P(t_i|a_i) - \log P(t_i|s_i)\right)
\end{aligned}
$$

where L_{ML} is the maximum likelihood loss, L_{MM} is the max-margin loss, α is the weight of the max-margin loss (set to 1.0 following Yu et al. (2017)), M is the margin (tuned be to 0.1), and t_i, s_i and a_i are the target sequence, normal input, and adversarial input, respectively.[7]

5 Experimental Setup

In addition to datasets, tasks, models and evaluation methods introduced in Section 2, we present training details in this section (see Appendix for a comprehensive version).

Models on Ubuntu: We implemented VHRED and Reranking-RL in TensorFlow (Abadi et al., 2016) and employed greedy search for inference. As shown in Table 1, for both models we obtained Activity and Entity F1's higher than the VHRED results reported in Serban et al. (2017a). Hence, each of these two implementations serves as a solid baseline for adversarial testing and training.

Should-Not-Change Strategies on Ubuntu: For Random Swap, we allow up to 1 swap of neighboring words per 4 words in each utterance. For Stopword Dropout, we allow up to 8 words to be dropped in each turn. For Data-level Paraphrasing, we use the small version of PPDB 2.0. For Generative-level Paraphrasing, we use the publicly available Pointer-Generator Networks code (See Appendix for some random samples of the generated paraphrases).[8] For Grammar Errors, in addition to those extracted from the AESW dataset,

[6] We do not train from scratch because each model (for each strategy) takes several days to converge.

[7] Please refer to supp. about greedy sampling based max-margin setup and CoCoA discussion for adversarial training.

[8] https://github.com/becxer/pointer-generator

Strategy Name	N-train + A-test	A-train + A-test	A-train + N-test	N-train + N-test
Normal Input	-	-	-	5.94, 3.52
Random Swap	6.10*, 3.42	6.47*, 3.64*	6.42*, 3.74*	-
Stopword Dropout	5.49*, 3.44	6.23*, 3.82*	6.29*, 3.71*	-
Data-Level Para.	5.38*, 3.18*	6.39*, 3.83*	6.32*, 3.87*	-
Generative-Level Para.	4.25*, 2.48*	5.89 , 3.60	6.11*, 3.66*	-
Grammar Errors	5.60*, 3.09*	5.93 , 3.67*	6.05 , 3.69*	-
All Should-Not-Change	-	-	6.74*, 3.97*	-
Add Negation	6.06 , 3.42	5.01*, 3.12*	6.07 , 3.46	-
Antonym	5.85 , 3.56	5.43*, 3.43	5.98 , 3.56	-

Table 2: Activity and Entity F1 results of adversarial strategies on the **VHRED** model. Numbers marked with * are stat. significantly higher/lower than their counterparts obtained with Normal Input (upper-right corner of table).

we also add a heuristic where an inflected verb is replaced with its respective infinitive form, and a plural noun with its singular form. Note that for all strategies we only keep an adversarial token if it is within the original vocabulary set.

Should-Change Strategies on Ubuntu: For Add Negation, we negate the first verb in each utterance. For Antonym, we modify the first verb, adjective or adverb that has an antonym.

Human Evaluation: We also conducted human studies on MTurk to evaluate adversarial training (pairwise comparison for dialogue quality) and generative paraphrasing (five-point Likert scale). The utterances were randomly shuffled to anonymize model identity, and we used MTurk with US-located human evaluators with approval rate $> 98\%$, and at least $10,000$ approved HITs. Results are presented in Section 6.1. Note that the human studies and automatic evaluation are complementary to each other: while MTurk annotators are good at judging how natural and coherent a response is, they are usually not experts in the Ubuntu operating system's technical details. On the other hand, automatic evaluation focuses more on the technical side (i.e., whether key activities or entities are present in the response).

Model on CoCoA: We adopted the publicly available code from He et al. (2017),[9] and used their already trained DynoNet model.

6 Results

6.1 Adversarial Results on Ubuntu

Result Interpretation For Table 2 and 3 with Should-Not-Change strategies, lower is better in the first column (since a successful adversarial testing strategy will be effective at fooling the model), while higher is better in the second column (since successful adversarial training should bring the performance back up). However, for Should-Change strategies, the reverse holds.[10] Lastly, in the third column, higher is better since we want the adversarially trained model to perform better on the original source inputs.

Results on Should-Not-Change Strategies Table 2 and 3 present the adversarial results on F1 scores of all our strategies for VHRED and Reranking-RL, respectively. Table 2 shows that VHRED is robust to none of the Should-Not-Change strategies other than Random Swap, while Table 3 shows that Reranking-RL is robust to none of the Should-Not-Change strategies other than Stopword Dropout. For each effective strategy, at least one of the F1's decreases statistically significantly[11] as compared to the same model fed with normal inputs. Next, all adversarial trainings on Should-Not-Change strategies not only make the model more robust to adversarial inputs (each *A-train* + *A-test* F1 is stat. significantly higher than that of *N-train* + *A-test*) , but also make them perform better on normal inputs (each *A-train* + *N-test* F1 is stat. significantly higher than that of *N-train* + *N-test*, except for Grammar Errors's Activity F1). Motivated by the success in adversarial training on each strategy alone, we also experimented with training on all Should-Not-Change strategies combined, and obtained F1's stat. significantly higher than any single strategy (the *All Should-Not-Change* row in Table 2), except that *All-Should-Not-Change*'s Entity F1 is stat. equal to that of Data-Level Paraphrasing, showing that these strategies are able to compensate for each other to further improve performance. An inter-

[9] https://worksheets.codalab.org/worksheets/0xc757f29f5c794e5eb7bfa8ca9c945573/

[10] Higher is better in the first column, because this shows that the model is not paying attention to important semantic changes in the source inputs (and is maintaining its original performance); while lower is better in the second column, since we want the model to be more sensitive to such changes after adversarial training.

[11] We obtained stat. significance via the bootstrap test (Noreen, 1989; Efron and Tibshirani, 1994) with 100K samples, and consider $p < 0.05$ as stat. significant.

Strategy Name	N-train + A-test	A-train + A-test	A-train + N-test	N-train + N-test
Normal Input	-	-	-	5.67, 3.73
Random Swap	5.49*, 3.56*	6.20*, 4.28*	6.36*, 4.39*	-
Stopword Dropout	5.51*, 4.09*	-	-	-
Data-Level Para.	5.28*, 3.07*	5.53*, 3.69	5.79*, 3.87*	-
Generative-Level Para.	4.47*, 2.63*	5.30*, 3.35*	5.86*, 3.90*	-
Grammar Errors	5.33*, 3.25*	5.55*, 3.92*	5.93*, 4.04*	-
Add Negation	5.61 , 3.79	4.92*, 2.78*	6.10*, 3.93*	-
Antonym	5.68 , 3.70	5.30*, 2.95*	5.80*, 3.71	-

Table 3: Activity and Entity F1 results of adversarial strategies on the **Reranking-RL** model. Numbers marked with * are stat. significantly higher/lower than their counterparts obtained with Normal Input (upper-right corner).

Strategy Name	VHRED		Reranking-RL	
	Cont.	Resp.	Cont.	Resp.
Random Swap	1.00	0.71	1.00	0.86
Stopword Dropout	0.61	0.50	0.76	0.68
Data-Level Para.	0.96	0.58	0.96	0.74
Gen.-Level Para.	0.70	0.40	0.76	0.55
Grammar Err.	0.96	0.58	0.97	0.74
Add Negation	0.96	0.69	0.97	0.81
Antonym	0.98	0.66	0.98	0.74

Table 4: Textual similarity of adversarial strategies on the VHRED and Reranking-RL models. "Cont." stands for "Context", and "Resp." stands for "Response".

esting strategy to note is Random Swap: although it itself is not effective as an adversarial strategy for VHRED, training on it does make the model perform better on normal inputs.

Results on Should-Change Strategies Table 2 and 3 show that Add Negation and Antonym are both successful Should-Change strategies, because no change in *N-train* + *A-test* F1 is stat. significant compared to that of *N-train* + *N-test*, which shows that both models are ignoring the semantic-changing perturbations to the inputs. From the last two rows of *A-train* + *A-test* column in each table, we also see that adversarial training successfully brings down both F1's (stat. significantly) for each model, showing that the model becomes more sensitive to the context change.

Semantic Similarity In addition to F1, we also follow Serban et al. (2017a) and employ cosine similarity between average embeddings of normal and adversarial inputs/responses (proposed by Liu et al. (2016)) to evaluate how much the inputs/responses change in semantic meaning (Table 4). This metric is useful in three ways. Firstly, by comparing the two columns of context similarity, we can get a general idea of how much change is perceived *by each model*. For example, we can see that Stopword Dropout leads to more evident changes from VHRED's perspective than from Reranking-RL's. This also agrees with the F1 results in Table 2 and 3, which indicate that Reranking-RL is much more robust to this strategy than VHRED is. The high context similarity of Should-Change strategies shows that although we have added "not" or replaced antonyms in every utterance of the source inputs, from the model's point of view the context has not changed much in meaning. Secondly, for each Should-Not-Change strategy, the cosine similarity of context is much higher than that of response, indicating that responses change more significantly in meaning than their corresponding contexts. Lastly, The high semantic similarity for Generative Paraphrasing also partly shows that the Pointer-Generator model in general produces faithful paraphrases.

Compared to Baseline	Win(%)	Tie(%)	Loss(%)
Random Swap	49	19	32
Stopword Dropout	45	19	36
Data-Level Para.	37	22	41
Generative-Level Para.	41	26	33
Grammar Errors	41	29	30
All Should-Not-Change	49	22	28
Add Negation	34	25	41
Antonym	40	29	31

Table 5: Human evaluation results on comparison between VHRED baseline trained on normal inputs vs. VHRED trained on each Should-Not-Change strategy (incl. one with all Should-Not-Change strategies combined) and each Should-Change strategy for Ubuntu.

	Pointer-Generator	ParaNMT-5M
Avg. Score	3.26	3.54

Table 6: Human evaluation scores on paraphrases generated by Pointer-Generator Networks and ground-truth pairs from ParaNMT-5M.

Human Evaluation As introduced in Section 5, we performed two human studies on adversarial training and Generative Paraphrasing. For the first study, Table 5 indicates that models trained on each adversarial strategy (as well as on all Should-Not-Change strategies combined) indeed on average produced better responses, and mostly agrees with the adversarial training results in Table 2.[12]

[12] Note that human evaluation does not show improvements with the Data-Level-Paraphrasing and Add-Negation strate-

Context	Response
N: ... you could save your ubuntu files and reinstall Windows , then install ubuntu as a dual boot option __eou__ __eot__ aight buddy , so how do i get that **unknown** space back__eou__ **Random Swap:** ... you could your save ubuntu and files Windows reinstall , then install ubuntu as dual a option boot __eou__ __eot__ aight buddy , so do how i that get space **unknown** back __eou__	**NN:** you can use the Live CD , you can install Ubuntu on the same partition as the Windows partition __eou__ **NA:** I am using ubuntu . __eou__ **AA:** you can use Windows XP on the Windows partition , and then install Ubuntu on the same drive __eou__

Table 7: VHRED output example before and after adversarial training on the Random Swap strategy.

For the second study, Table 6 shows that on average the generated paraphrase has roughly the same semantic meaning with the original utterance, but may sometimes miss some information. Its quality is also close to that of the ground-truth in ParaNMT-5M dataset.

Output Examples of Generated Responses We present a selected example of generated responses before and after adversarial training on the Random Swap strategy with the VHRED model in Table 7 (more examples in Appendix on all strategies with both models). First of all, we can see that it is hard to differentiate between the original and the perturbed context (*N-context* and *A-context*) if one does not look very closely. For this reason, the model gets fooled by the adversarial strategy, i.e., after adversarial perturbation, the *N-train* + *A-test* response (NA-Response) is worse than that of *N-train* + *N-test* (NN-Response). However, after our adversarial training phase, *A-train* + *A-test* (AA-Response) becomes better again.

6.2 Adversarial Results on CoCoA

Table 8 shows the results of Should-Change strategies on DynoNet with the CoCoA task. The Random Inputs strategy shows that even without communication, the two bots are able to locate their shared entry 82% of the time by revealing their own KB through SELECT action. When we keep the mentioned entities untouched but randomize all other tokens, DynoNet actually achieves state-of-the-art Completion Rate, indicating that the two agents are paying zero attention to each other's utterances other than the entities contained in them. This is also why we did not apply Add Negation and Antonym to DynoNet — if Random Inputs does not work, these two strategies will also make no difference to the performance (in other words Random Inputs subsumes the other two Should-Change strategies). We can also see that even with the Normal Inputs with Confusing Entities strategy, DynoNet is still able to finish the task 77% of the time, and with only slightly more turns. This again shows that the model mainly relies on the SELECT action to guess the shared entry.

Strategy	Completion Rate	Num. of Turns
Norm. Inputs	0.94	16.06
Rand. Inputs	0.82	22.87
Rand. w/ Entity	0.95	17.19
Confusing Entity	0.77	24.11

Table 8: Adversarial Results on DynoNet.

7 Byte-Pair-Encoding VHRED

Although we have shown that adversarial training on most strategies makes the dialogue model more robust, generating such perturbed data is not always straightforward for diverse, complex strategies. For example, our data-level and generative-level strategies all leverage datasets that are not always available to a language. We are thus motivated to also address the robustness task on the model-level, and explore an extension to the VHRED model that makes it robust to Grammar Errors even without adversarial training.

Model Description: We performed Byte Pair Encoding (BPE) (Sennrich et al., 2016) on the Ubuntu dataset.[13] This algorithm encodes rare/unknown words as sequences of subword units, which helps segmenting words with the same lemma but different inflections (e.g., "showing" to "show + ing", and "cakes" to "cake + s"), making the model more likely to be robust to grammar errors such as verb tense or plural/singular noun confusion. We experimented BPE with 5K merging operations, and obtained a vocabulary size of 5121.

Results: As shown in Table 9, BPE-VHRED achieved F1's (5.99, 3.66), which is stat. equal to (5.94, 3.52) obtained without BPE. To our best knowledge, we are the first to apply BPE to a gen-

gies, though the latter does agree with F1 trends. Overall, we provide both human and F1 evaluations because they are complementary at judging naturalness/coherence vs. key Ubuntu technical activities/entities.

[13] We employed code released by the authors on https://github.com/rsennrich/subword-nmt

	VHRED	BPE-VHRED
Normal Input	5.94, 3.52	5.99, 3.66
Grammar Errors	5.60, 3.09	5.86, 3.54

Table 9: Activity, Entity F1 results of VHRED model vs. BPE-VHRED model tested on normal inputs.

erative dialogue task. Moreover, BPE-VHRED achieved (5.86, 3.54) on Grammar Errors based adversarial test set, which is stat. equal to the F1's when tested with normal data, indicating that BPE-VHRED is more robust to this adversarial strategy than VHRED is, since the latter had (5.60, 3.09) when tested with perturbed data, where both F1's are stat. signif. lower than when fed with normal inputs. Moreover, BPE-VHRED reduces the vocabulary size by 15K, corresponding to 4.5M fewer parameters. This makes BPE-VHRED train much faster. Note that BPE only makes the model robust to one type of noise (i.e. Grammar Errors), and hence adversarial training on other strategies is still necessary (but we hope that this encourages future work to build other advanced models that are naturally robust to diverse adversaries).

8 Related Works

Model-Dependent vs. Model-Agnostic Strategies: Many adversarial strategies have been applied to both Computer Vision (Biggio et al., 2012; Szegedy et al., 2013; Goodfellow et al., 2015; Mei and Zhu, 2015; Papernot et al., 2016; Narodytska and Kasiviswanathan, 2017; Liu et al., 2017; Carlini and Wagner, 2017; Papernot et al., 2017; Mironenco et al.; Wong, 2017; Gao et al., 2018) and NLP (Jia and Liang, 2017; Zhao et al., 2018; Belinkov and Bisk, 2018; Shalyminov et al., 2017; Mironenco et al.; Iyyer et al., 2018). Previous works have distinguished between model-aware strategies, where the adversarial algorithms have access to the model parameters, and model-agnostic strategies, where the adversary does not have such information (Papernot et al., 2017; Liu et al., 2017; Narodytska and Kasiviswanathan, 2017). We however, observed that within the model-agnostic category, there are two subcategories. One is *half-model-agnostic*, where although the adversary has no access to the model parameters, it is allowed to probe the target model and observe its output as a way to craft adversarial inputs (Biggio et al., 2012; Szegedy et al., 2013; Goodfellow et al., 2015; Mei and Zhu, 2015; Papernot et al., 2017; Mironenco et al.). On the other hand, a *pure-model-agnostic* adversary, such as works by Jia and Liang (2017) and Belinkov and Bisk (2018), does not have any access to the model outputs when creating adversarial inputs, and is thus more generalizable across models/tasks. We adopt the pure-model-agnostic approach, only drawing inspiration from real-world noise, and testing them on the target model.

Adversarial in NLP: Text-based adversarial works have targeted both classification models (Weston et al., 2016; Jia and Liang, 2017; Wong, 2017; Liang et al., 2017; Samanta and Mehta, 2017; Shalyminov et al., 2017; Gao et al., 2018; Iyyer et al., 2018) and generative models (Hosseini et al., 2017; Henderson et al., 2017; Mironenco et al.; Zhao et al., 2018; Belinkov and Bisk, 2018). To our best knowledge, our work is the first to target generative goal-oriented dialogue systems with several new adversarial strategies in both Should-Not-Change and Should-Change categories, and then to fix the broken models through adversarial training (esp. using max-margin loss for Should-Change), and also achieving model robustness without using any adversarial data.

9 Conclusion

We first revealed both the over-sensibility and over-stability of state-of-the-art models on Ubuntu and CoCoA dialogue tasks, via Should-Not-Change and Should-Change adversarial strategies. We then showed that training on adversarial inputs not only made the models more robust to the perturbations, but also helped them achieve new state-of-the-art performance on the original data (with further improvements when we combined strategies). Lastly, we also proposed a BPE-enhanced VHRED model that not only trains faster with comparable performance, but is also robust to Grammar Errors even without adversarial training, motivating that if no strong adversary-generation tools (e.g., paraphraser) are available (esp. in low-resource domains/languages), we should try alternative model-robustness architectural changes.

Acknowledgments

We thank the anonymous reviewers for their helpful comments and discussions. This work was supported by DARPA (YFA17-D17AP00022), Facebook ParlAI Research Award, Google Faculty Research Award, Bloomberg Data Science Research Grant, and Nvidia GPU awards. The views contained in this article are those of the authors and not of the funding agency.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *Proceedings of ICLR*.

Battista Biggio, Blaine Nelson, and Pavel Laskov. 2012. Poisoning attacks against support vector machines. In *Proceedings of ICML*.

Nicholas Carlini and David Wagner. 2017. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14. ACM.

Vidas Daudaravicius. 2015. Automated evaluation of scientific writing data set (version 1.2)[data file]. *VTeX, Vilnius, Lithuania*.

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

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *Deep Learning and Security Workshop*.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *Proceedings of ICLR*.

He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. 2017. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. In *Proceedings of ACL*.

W. Headlam. 1902. Transposition of words in mss. *The Classical Review*, 16(5):243–256.

Peter Henderson, Koustuv Sinha, Nicolas Angelard-Gontier, Nan Rosemary Ke, Genevieve Fried, Ryan Lowe, and Joelle Pineau. 2017. Ethical challenges in data-driven dialogue systems. *arXiv preprint arXiv:1711.09050*.

Hossein Hosseini, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. 2017. Deceiving google's perspective api built for detecting toxic comments. *arXiv preprint arXiv:1702.08138*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of NAACL*.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of EMNLP*.

Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. 2016. Deep reinforcement learning for dialogue generation. In *Proceedings of EMNLP*.

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*.

Chia-Wei Liu, Ryan Lowe, Iulian V. Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of EMNLP*.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. 2017. Delving into transferable adversarial examples and black-box attacks. In *Proceedings of ICLR*.

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. *arXiv preprint arXiv:1506.08909*.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11–20.

Teresa Marqués-Aguado. 2014. Errors, corrections and other textual problems in three copies of a middle english antidotary. *Nordic Journal of English Studies*, 13(1):53–77.

Shike Mei and Xiaojin Zhu. 2015. Using machine teaching to identify optimal training-set attacks on machine learners. In *Proceedings of AAAI*, pages 2871–2877.

Mircea Mironenco, Dana Kianfar, Ke Tran, Evangelos Kanoulas, and Efstratios Gavves. Examining cooperation in visual dialog models. In *Proceedings of NIPS*.

Nina Narodytska and Shiva Prasad Kasiviswanathan. 2017. Simple black-box adversarial perturbations for deep networks. In *Proceedings of CVPR*.

Eric W. Noreen. 1989. *Computer-intensive methods for testing hypotheses*. Wiley New York.

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, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 372–387. IEEE.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *Proceedings of ICLR*.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 425–430.

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

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of ACL*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of ACL*.

Iulian Vlad Serban, Tim Klinger, Gerald Tesauro, Kartik Talamadupula, Bowen Zhou, Yoshua Bengio, and Aaron C. Courville. 2017a. Multiresolution recurrent neural networks: An application to dialogue response generation. In *AAAI*, pages 3288–3294.

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*, pages 3776–3784.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2017b. A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*, pages 3295–3301.

Igor Shalyminov, Arash Eshghi, and Oliver Lemon. 2017. Challenging neural dialogue models with natural data: Memory networks fail on incremental phenomena. In *Proceedings of SemDial*.

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*.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. 2016. Towards ai-complete question answering: A set of prerequisite toy tasks. In *Proceedings of ICLR*.

John Wieting and Kevin Gimpel. 2017. Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. *arXiv preprint arXiv:1711.05732*.

Catherine Wong. 2017. Dancin seq2seq: Fooling text classifiers with adversarial text example generation. *arXiv preprint arXiv:1712.05419*.

Licheng Yu, Hao Tan, Mohit Bansal, and Tamara L. Berg. 2017. A joint speakerlistener-reinforcer model for referring expressions. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2018. Generating natural adversarial examples. In *Proceedings of ICLR*.