ANSWER-BASED ADVERSARIAL TRAINING FOR GENERATING CLARIFICATION QUESTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose a generative adversarial training approach for the problem of clarification question generation. Our approach generates clarification questions with the goal of eliciting new information that would make the given context more complete. We develop a Generative Adversarial Network (GAN) where the generator is a sequence-to-sequence model and the discriminator is a utility function that models the value of updating the context with the answer to the clarification question. We evaluate on two datasets, using both automatic metrics and human judgments of usefulness, specificity and relevance, showing that our approach outperforms both a retrieval-based model and ablations that exclude the utility model and the adversarial training.

1 INTRODUCTION

A goal of natural language processing is to develop techniques that enable machines to process naturally occurring language. However, not all language is clear and, as humans, we may not always understand each other (Grice, 1975); in cases of gaps or mismatches in knowledge, we tend to ask questions (Graesser et al., 2008). In this work, we focus on the task of automatically generating clarification questions: questions that ask for information that is *missing* from a given linguistic context. Our clarification question generation model builds on the sequence-to-sequence approach that has proven effective for several language generation tasks (Sutskever et al., 2014; Serban et al., 2016; Yin et al., 2016; Du et al., 2017). Unfortunately, training a sequence-to-sequence model directly on context/question pairs yields generated questions that are highly generic[1], corroborating a common finding in dialog systems (Li et al., 2016b). Our goal is to be able to generate questions that are useful *and* specific.

To achieve this, we begin with a recent observation of Rao & Daumé III (2018), who considered the task of question reranking: the system should learn to generate clarification questions whose answers have high *utility*, which they defined as the likelihood that this question would lead to an answer that will make the context more complete (§2.3). Inspired by this, we construct a question generation model that first generates a question given a context, and then generates a hypothetical answer to that question. Given this (context, question, answer) tuple, we train a utility calculator to estimate the usefulness of this question. We then show that this utility calculator can be generalized using ideas for generative adversarial networks (Goodfellow et al., 2014) for text (Yu et al., 2017), wherein the utility predictor plays the role of the "discriminator" and the question generator is the "generator" (§2.2), which we train using the MIXER algorithm (Ranzato et al., 2015).

We evaluate our approach on two question generation datasets: for posts on Stack Exchange and for Amazon product descriptions (Figure 1). Using both automatic metrics and human evaluation, we demonstrate that our adversarially trained model generates a more diverse set of questions than all the baseline models. Furthermore, we find that although all models generate questions that are relevant to the context at hand, our adversarially-trained model generates questions that are more specific to the context.[2]

[1]For instance, in the context of asking questions about home appliances, frequently asking like "What are the dimensions?" or "Is it made in China?"

[2]**Code and data release:** All code will be released under a license at least as permissive as MIT; all data will be made available after publication subject to allowance by the original licenses.

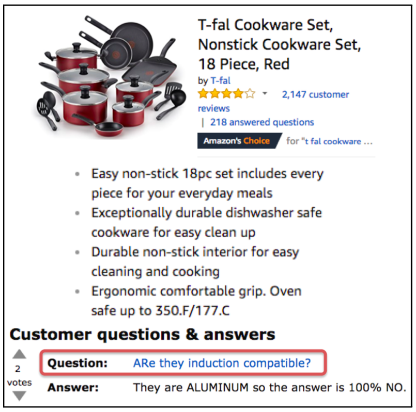

Figure 1: Sample product description paired with a clarification question from amazon.com.

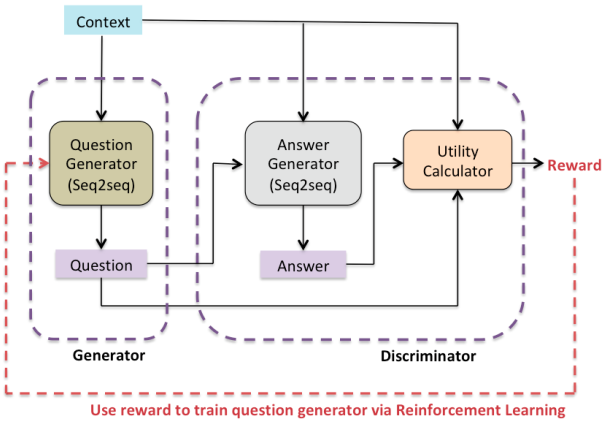

Figure 2: Overview of our GAN-based clarification question generation model (refer preamble of §2)

2 Training a Clarification Question Generator

Our goal is to build a model that, given a context, can generate an appropriate clarification question. As a running example, we will use the Amazon setting: where the dataset consists of (context, question, answer) triples where the context is the product description, question is clarification question about that product that (preferably) is not already answered in the description and answer is the seller's (or other users') reply to the question. Representationally, our question generator is a standard sequence-to-sequence model with attention (§2.1). The *learning* problem is: how to train the sequence-to-sequence model to produce good question.

An overview of our training setup is shown in Figure 2. Given a context, our question generator outputs a question. In order to evaluate the usefulness of this question, we then have a second sequence-to-sequence model called the "answer generator" that generates a hypothetical answer based on the context and the question (§2.5). This (context, question and answer) triple is fed into a Utility calculator, whose initial goal is to estimate the probability that this question/answer pair is useful in this context (§2.3). This Utility is treated as a reward, which is used to update the question generator using the Mixer (Ranzato et al., 2015) algorithm (§2.2). Finally, we reinterpret the answer-generator-plus-utility-calculator component as a *discriminator* for differentiating between true (context, question, answer) triples and synthetic triples (§2.4), and optimize this adversarial objective using Mixer.

2.1 Sequence-to-sequence Model for Question Generation

We use a standard attention based sequence-to-sequence model (Luong et al., 2015) for our question generator. Given an input sequence (context) $c = (c_1, c_2, ..., c_N)$, this model generates an output sequence (question) $q = (q_1, q_2, ..., q_T)$. The architecture of this model is an encoder-decoder with attention. The encoder is a recurrent neural network (RNN) operating over the input word embeddings to compute a source context representation $\tilde{c}$. The decoder uses this source representation to generate the target sequence one word at a time:

$$p(q|\tilde{c_t}) = \prod_{t=1}^{T} p(q_t|q_1, q_2, ..., q_{t-1}, \tilde{c_t}) = \prod_{t=1}^{T} softmax(W_s\tilde{h}_t) \quad ; \quad \text{where } \tilde{h}_t = \tanh(W_c[\tilde{c_t}; h_t]) \tag{1}$$

In Eq 1, $\tilde{h}_t$ is the attentional hidden state of the RNN at time t and W_s and W_c are parameters of the model (details in Appendix A). The predicted token q_t is the token in the vocabulary that is assigned the highest probability using the softmax function. The standard training objective for sequence-to-sequence model is to maximize the log-likelihood of all (c, q) pairs in the training data D which is

equivalent to minimizing the loss,

$$L_{\text{mle}}(D) = -\sum_{(c,q)\in D}\sum_{t=1}^{T}\log p(q_t|q_1, q_2, ..., q_{t-1}, c) \tag{2}$$

2.2 TRAINING THE GENERATOR TO OPTIMIZE QUESTION UTILITY

Training sequence-to-sequence models for the task of clarification question generation (with context as input and question as output) using maximum likelihood objective unfortunately leads to the generation of highly generic questions, such as *"What are the dimensions?"* when asking questions about home appliances. This issue has been observed in dialog generation as well (Li et al., 2016b). Recently Rao & Daumé III (2018) observed that usefulness of a question can be better measured as the *utility* that would be obtained if the context were updated with the answer to the proposed question. We use this observation to define a UTILITY based reward function and train the question generator to optimize this reward. We train the UTILITY reward to predict the likelihood that a question would generate an answer that would increase the utility of the context by adding useful information to it (see §2.3 for details).

Similar to optimizing metrics like BLEU and ROUGE, this UTILITY function also operates on discrete text outputs, which makes optimization difficult due to non-differentiability. A successful recent approach dealing with the non-differentiability while also retaining some advantages of maximum likelihood training is the Mixed Incremental Cross-Entropy Reinforce (Ranzato et al., 2015) algorithm (MIXER). In MIXER, the overall loss L is differentiated as in REINFORCE (Williams, 1992):

$$L(\theta) = -\mathbb{E}_{q^s\sim p_\theta} r(q^s) \quad ; \quad \nabla_\theta L(\theta) = -\mathbb{E}_{q^s\sim p_\theta} r(q^s)\nabla_\theta \log p_\theta(q^s) \tag{3}$$

where y^s is a random output sample according to the model p_θ, where θ are the parameters of the network. We then approximate the expected gradient using a single sample $q^s = (q_1^s, q_2^s, ..., q_T^s)$ from the model distribution (p_θ). In REINFORCE, the policy is initialized random, which can cause long convergence times. To solve this, MIXER starts by optimizing maximum likelihood and slowly shifts to optimizing the expected reward from Eq 3. For the initial Δ time steps, MIXER optimizes L_{mle} and for the remaining $(T - \Delta)$ time steps, it optimizes the external reward.

In our model, we minimize the UTILITY-based loss $L_{\text{max-utility}}$ defined as:

$$L_{\text{max-utility}} = -(r(q^p) - r(q^b))\sum_{t=1}^{T}\log p(q_t|q_1, q_2, ..., q_{t-1}, c_t) \tag{4}$$

where $r(q^p)$ is the UTILITY based reward on the predicted question and $r(q^b)$ is a baseline reward introduced to reduce the high variance otherwise observed when using REINFORCE.

In MIXER, the baseline is estimated using a linear regressor that takes in the current hidden states of the model as input and is trained to minimize the mean squared error $(||r(q^p) - r(q^b)||)^2$. Instead we use a self-critical training approach Rennie et al. (2017) where the baseline is estimated using the reward obtained by the current model under greedy decoding during test time.

2.3 ESTIMATING A UTILITY FUNCTION FROM HISTORICAL DATA

Given a (context, question, answer) triple, Rao & Daumé III (2018) introduce a utility function $\text{UTILITY}(c, q, a)$ to calculate the value of updating a context c with the answer a to a clarification question q. The inspiration for thier utility function is to estimate the probability that an *answer* would be a meaningful addition to a context, and treat this as a binary classification problem where the positive instances are the true (context, question, answer) triples in the dataset whereas the negative instances are contexts paired with a random (question, answer) from the dataset. The model we use is to first embed of the words in the context c, then use an LSTM (long-short term memory) (Hochreiter & Schmidhuber, 1997) to generate a neural representation $\bar{c}$ of the context by averaging the output of each of the hidden states. Similarly, we obtain a neural representation $\bar{q}$ and $\bar{a}$ of q and a respectively using question and answer LSTM models. Finally, a feed forward neural network $F_{\text{UTILITY}}(\bar{c}, \bar{q}, \bar{a})$ predicts the usefulness of the question.

2.4 UTILITY GAN FOR CLARIFICATION QUESTION GENERATION

The UTILITY function trained on true vs random samples from real data (as described in the previous section) can be a weak reward signal for questions generated by a model due to the large discrepancy between the true data and the model's outputs. In order to strengthen the reward signal, we reinterpret the UTILITY function (coupled with the answer generator) as a discriminator in an adversarial learning setting. That is, instead of taking the UTILITY calculator to be a fixed model that outputs the expected quality of a question/answer pair, we additionally optimize it to distinguish between true question/answer pairs and model-generated ones. This reinterpretation turns our model into a form of a generative adversarial network (GAN) (Goodfellow et al., 2014).

A GAN is a training procedure for "generative" models that can be interpreted as a game between a generator and a discriminator. The generator is an arbitrary model $g \in \mathcal{G}$ that produces outputs (in our case, questions). The discriminator is another model $d \in \mathcal{D}$ that attempts to classify between true outputs and model-generated outputs. The goal of the generator is to generate data such that it can fool the discriminator; the goal of the discriminator is to be able to successfully distinguish between real and generated data. In the process of trying to fool the discriminator, the generator produces data that is as close as possible to the real data distribution. Generically, the GAN objective is:

$$L_{\text{GAN}}(\mathcal{D}, \mathcal{G}) = \max_{d \in \mathcal{D}} \min_{g \in \mathcal{G}} \mathbb{E}_{x \sim \hat{p}} \log d(x) + \mathbb{E}_{z \sim p_z} \log(1 - d(g(z))) \tag{5}$$

where x is sampled from the true data distribution $\hat{p}$, and z is sampled from a prior defined on input noise variables p_z.

Although GANs have been successfully used for image tasks, training GANs for text generation is challenging due to the discrete nature of outputs in text. The discrete outputs from the generator make it difficult to pass the gradient update from the discriminator to the generator. Recently, Yu et al. (2017) proposed a sequence GAN model for text generation to overcome this issue. They treat their generator as an agent and use the discriminator as a reward function to update the generative model using reinforcement learning techniques. Our GAN-based approach is inspired by this sequence GAN model with two main modifications: a) We use the MIXER algorithm as our generator (§2.2) instead of policy gradient approach; and b) We use the UTILITY function (§2.3) as our discriminator instead of a convolutional neural network (CNN).

In our model, the answer is an *latent variable*: we do not actually use it anywhere except to train the discriminator. Because of this, we train our discriminator using (context, true question, generated answer) triples as positive instances and (context, generated question, generated answer) triples as the negative instances. Formally, our objective function is:

$$L_{\text{GAN-U}}(\mathcal{U}, \mathcal{M}) = \max_{u \in \mathcal{U}} \min_{m \in \mathcal{M}} \mathbb{E}_{q \sim \hat{p}} \log u(c, q, \mathcal{A}(c, q)) + \mathbb{E}_{c \sim \hat{p}} \log(1 - u(c, m(c), \mathcal{A}(c, m(c)))) \tag{6}$$

where $\mathcal{U}$ is the UTILITY discriminator, $\mathcal{M}$ is the MIXER generator, $\hat{p}$ is our data of (context, question, answer) triples and $\mathcal{A}$ is our answer generator.

2.5 PRETRAINING

Question Generator. We pretrain our question generator using the sequence-to-sequence model§2.1 where we define the input sequence as the context and the output sequence as the question. This answer generator is trained to maximize the log-likelihood of all ([context+question], answer) pairs in the training data. Parameters of this model are updated during adversarial training.

Answer Generator. We pretrain our answer generator using the sequence-to-sequence model§2.1 where we define the input sequence as the concatenation of the context and the question and the output sequence as the answer. This answer generator is trained to maximize the log-likelihood of all (context, question) pairs in the training data. Unlike the question generator, the parameters of the answer generator are kept fixed during the adversarial training.

Discriminator. We pretrain the discriminator using (context, question, answer) triples from the training data. For positive instances, we use a context and its true question, answer and for negative instances, we use the same context but randomly sample a question from the training data (and use the answer paired with that random question).

3 EXPERIMENTAL RESULTS

We base our experimental design on the following research questions:

1. Do generation models outperform simpler retrieval baselines?
2. Does optimizing the UTILITY reward improve over maximum likelihood training?
3. Does using adversarial training improve over optimizing the pretrained UTILITY?
4. How do the models perform when evaluated for nuances such as specificity and usefulness?

3.1 DATASETS

We evaluate our model on two datasets. The first is from StackExchange and was curated by Rao & Daumé III (2018); the second is from Amazon, curated by McAuley & Yang (2016), and has not previously been used for the task of question generation.

StackExchange. This dataset consists of posts, questions asked to that post on stackexchange.com (and answers) collected from three related subdomains on stackexchage.com (askubuntu, unix and superuser). Additionally, for 500 instances each from the tune and the test set, the dataset includes 1 to 5 other questions identified as valid questions by expert human annotators from a pool of candidate questions. This dataset consists of $61,681$ training, 7710 validation and 7709 test examples.

Amazon. Each instance consists of a question asked about a product on amazon.com combined with other information (product ID, question type "Yes/No", answer type, answer and answer time). To obtain the description of the product, we use the metadata information contained in the amazon reviews dataset (McAuley et al., 2015). We consider at most 10 questions for each product. This dataset includes several different product categories. We choose the `Home and Kitchen` category since it contains a high number of questions and is relatively easy category for human based evaluation. This dataset consists of $19,119$ training, 2435 validation and 2305 test examples, and each product description contains between 3 and 10 questions (average: 7).

3.2 BASELINES AND ABLATED MODELS

We compare three variants (ablations) of our proposed approach, together with an information retrieval baseline: **GAN-Utility** is our full model which is a UTILITY function based GAN training (§ 2.4) including the UTILITY discriminator, a MIXER question generator and a sequence-to-sequence based answer generator. **Max-Utility** is our reinforcement learning baseline with a pretrained question generator described model (§ 2.2) without the adversarial training. **MLE** is the question generator model pretrained on context, question pairs using maximum likelihood objective (§ 2.1). **Lucene**[3] is a TF-IDF (term frequency-inverse document frequency) based document ranking system which given a document, retrieves N other documents that are most similar to the given document. Given a context, we use Lucene to retrieve top 10 contexts that are most similar to the given context. We randomly choose a question from the 10 questions paired with these contexts to construct our Lucene baseline[4]. Experimental details of all our models are described in Appendix B.

3.3 EVALUATION METRICS

We evaluate initially with several automated evaluation metrics, and then more substantially based on crowdsourced human judgments.

Automatic metrics include: **Diversity**, which calculates the proportion of unique trigrams[5] in the output to measure the diversity as commonly used to evaluate dialogue generation (Li et al., 2016b); **BLEU** (Papineni et al., 2002), which evaluate n-gram precision between a predicted sentence and reference sentences; and **METEOR** (Banerjee & Lavie, 2005), which is similar to BLEU but includes stemmed and synonym matches when measuring the similarity between the predicted sequence and the reference sequences.

[3] `https://lucene.apache.org/`

[4] For the Amazon dataset, we ignore questions asked to products of the same brand as the given product since Amazon replicates questions across same brand allowing the true question to be included in that set.

[5] We report trigrams, but bigrams and unigrams follow similar trends.

	Amazon			StackExchange		
Model	DIVERSITY	BLEU	METEOR	DIVERSITY	BLEU	METEOR
Reference	0.6934	—	—	0.7509	—	—
Lucene	0.6289	4.26	10.85	0.7453	1.63	7.96
MLE	0.1059	**17.02**	12.72	0.2183	3.49	8.49
Max-Utility	0.1214	16.77	12.69	**0.2508**	3.89	8.79
GAN-Utility	**0.1296**	15.20	**12.82**	0.2256	**4.26**	**8.99**

Table 1: DIVERSITY as measured by the proportion of unique trigrams in model outputs. BLEU and METEOR scores using up to 10 references for the Amazon dataset and up to six references for the StackExchange dataset. Numbers in bold are the highest among the models. All results for Amazon are on the entire test set whereas for StackExchange they are on the 500 instances of the test set that have multiple references.

Human judgments involve showing contexts and generated questions to crowdworkers[6] and asking them to evaluate the questions along several axes. Roughly, we ask for the following five judgments for each question (exact wordings in Appendix C): Is it **relevant** (yes/no); Is it **grammatical** (yes/comprehensible/incomprehensible); How **specific** is it to this product (four options from "specific to only this product" to "generic to any product"); Does this question ask for **new information** not contained in the discription (completely/somewhat/no); and How **useful** is this question to a potential buyer (four options from "should be included in the description" to "useful only to the person asking"). For the last three questions, we also allowed a "not applicable" response in the case that the question was either ungrammatical or irrelevant.

3.4 AUTOMATIC METRIC RESULTS

Table 1 shows the results on the two datasets when evaluated according to automatic metrics.

In the Amazon dataset, GAN-Utility outperforms all ablations on DIVERSITY, suggesting that it produces more diverse outputs. Lucene, on the other hand, has the highest DIVERSITY since it consists of human generated questions, which tend to be more diverse because they are much longer compared to model generated questions. This comes at the cost of lower match with the reference as visible in the BLEU and METEOR scores. In terms of BLEU and METEOR, there is inconsistency. Although GAN-Utility outperforms all baselines according to METEOR, the fully ablated MLE model has a higher BLEU score. This is because BLEU score looks for exact n-gram matches and since MLE produces more generic outputs, it is much more likely that it will match one of 10 references compared to the specific/diverse outputs of GAN-Utility, since one of those ten is highly likely to itself be generic.

In the StackExchange dataset GAN-Utility outperforms all ablations on both BLEU and METEOR. Unlike in the Amazon dataset, MLE does not outperform GAN-Utility in BLEU. This is because the MLE outputs in this dataset are not as generic as in the amazon dataset due to the highly technical nature of contexts in StackExchange. As in the Amazon dataset, GAN-Utility outperforms MLE on DIVERSITY. Interestingly, the Max-Utility ablation achieves a higher DIVERSITY score than GAN-Utility. On manual analysis we find that Max-Utility produces longer outputs compared to GAN-Utility but at the cost of being less grammatical.

3.5 HUMAN JUDGEMENTS ANALYSIS

Table 2 shows the numeric results of human-based evaluation performed on the reference and the system outputs on 500 random samples from the test set of the Amazon dataset.[7] These results overall show that the GAN-Utility model successfully generates the most specific questions, while being equally good at seeking new information and being useful to potential buyers. All approaches produce relevant, grammatical questions. All our models are all equally good at seeking new information, but are weaker than Lucene, which performs better according to new information but at

[6]We use Figure-Eight, `https://www.figure-eight.com`. We paid crowdworkers 5 cents per judgment.

[7]We could not ask crowdworkers evaluate the StackExchange data due to its highly technical nature.

Model	Relevant [0-1]	Grammatical [0-2]	Specific [0-4]	New Info [0-3]	Useful [0-1]
Reference	0.96	1.97	3.07	2.68	0.79
Lucene	**0.90**	**1.95**	2.80	**2.56**	0.77
MLE	**0.94**	**1.91**	2.84	2.48	**0.93**
Max-Utility	**0.94**	**1.91**	2.88	2.47	**0.90**
GAN-Utility	**0.95**	**1.91**	**2.99**	2.51	**0.94**

Table 2: Results of human judgments on model generated questions on 500 sample Home & Kitchen product descriptions. The options described in §3.3 are converted to corresponding numeric range (as described in Appendix C). The difference between the bold and the non-bold numbers is statistically insignificant with p <0.001. Reference is excluded in the significance calculation.

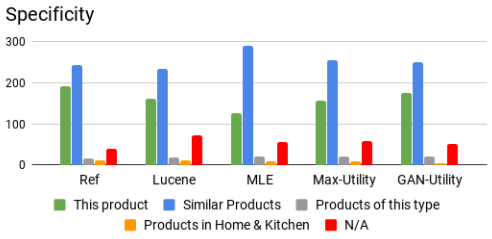

Figure 3: Results of human judgements on the specificity criteria.

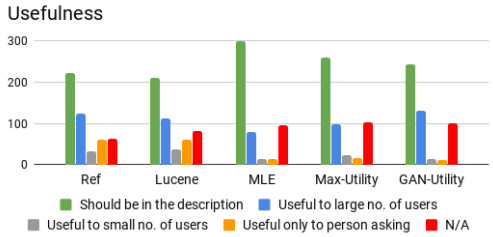

Figure 4: Results of human judgements on the usefulness criteria.

the cost of much lower specificity and slightly lower relevance. Our models are all equally good also at generating useful questions: their usefulness score is significantly better than both Lucene and Reference, largely because Lucene and Reference tend to ask questions that are more often useful only to the person asking the question, making them less useful for potential other buyers (see Figure 4). Our full model, GAN-Utility, performs significantly better when measured by specificity to the product, which aligns with the higher DIVERSITY score obtained by GAN-Utility under automatic metric evaluation.

4 RELATED WORK

Question Generation. Most previous work on question generation has been on generating reading comprehension style questions i.e. questions that ask about information present in a given text (Heilman, 2011; Rus et al., 2010; 2011; Duan et al., 2017). Outside reading comprehension questions, Labutov et al. (2015) use crowdsourcing to generate question templates, Liu et al. (2010) use templated questions to help authors write better related work sections, Mostafazadeh et al. (2016) introduced visual question answer tasking that focuses on generating natural and engaging questions about an image. Mostafazadeh et al. (2017) introduced an extension of this task called the Image Grounded Conversation task where they use both the image and some initial textual context to generate a natural follow-up question and a response to that question. Buck et al. (2017) propose an active question answering model where they build an agent that learns to reformulate the question to be asked to a question-answering system so as to elicit the best possible answers. Duan et al. (2017) extract large number of question-answer pairs from community question answering forums and use them to train a model that can generate a natural question given a passage.

Neural Models and Adversarial Training for Text Generation. Neural network based models have had significant success at a variety of text generation tasks, including machine translation (Bahdanau et al., 2015; Luong et al., 2015), summarization (Nallapati et al., 2016), dialog (Serban et al., 2016; Bordes et al., 2016; Li et al., 2016a; Serban et al., 2017), textual style transfer (Jhamtani et al., 2017; Kabbara & Cheung, 2016; Rao & Tetreault, 2018) and question answering (Yin et al., 2016; Serban et al., 2016).Our task is most similar to dialog, in which a wide variety of possible outputs are acceptable, and where lack of specificity in generated outputs is common. We addresses

Title	HOST Freeze Cooling Wine Glass		
Product Description	Say goodbye to lukewarm drinks with the FREEZE! Just pop in the freezer, pour and enjoy perfectly chilled wine. No need to preplan by chilling your wine or drink beforehand.		
		Specific	Useful
Reference	ours have an odd odor to them. how do we get rid of that	4	1
Lucene	where is the vinoice wine pourer with chill rod made ? usa ?	3	3
MLE	does this come with a straw ?	3	4
Max-Utility	does this come with a straw ?	3	1
GAN-Utility	does this come with a lid ?	4	4
Title	Miracle Whisk Self Mixer Stainless Steel No Batteries Hand Push		
Product Description	Put an end to cramped hands and splattered messes! The Miracle Whisk is a cordless, battery-free hand operated whisk. Simply push down on the handle and the whisk smoothly and easily rotates to mix, froth or whip your recipes to perfection.		
		Specific	Useful
Reference	dimensions pl	1	4
Lucene	does the power button stay at the " on " position when you push it , or do you have to keep holding it to make the attachment spin ?	4	3
MLE	does it work on 220v ?	2	3
Max-Utility	does this work on a glass top mixer ?	4	4
GAN-Utility	does it come with the beaters ?	3	4

Table 3: Example outputs from each of the systems for a single product description

this challenge using an adversarial network approach (Goodfellow et al., 2014), a training procedure that can generate natural-looking outputs, which have been effective for natural image generation (Denton et al., 2015). Due to the challenges in optimizing over discrete output spaces like text, Yu et al. (2017) introduced a Seq(uence)GAN approach where they overcome this issue by using REINFORCE to optimize. Li et al. (2017) train an adversarial model similar to SeqGAN for generating next utterance in a dialog given a context. However, unlike our work, their discriminator is a binary classifier trained to distinguish between human and machine generated utterances. Finally, Fedus et al. (2018) introduce an actor-critic conditional GAN for filling in missing text conditioned on the surrounding context.

5 CONCLUSION

In this work, we describe a novel approach to the problem of clarification question generation. Given a context, we use the observation of Rao & Daumé III (2018) that the usefulness of a clarification question can be measured by the value of updating the context with an answer to the question. We use a sequence-to-sequence model to generate a question given a context and a second sequence-to-sequence model to generate an answer given the context and the question. Given the (context, predicted question, predicted answer) triple we calculator the utility of this triple and use it as a reward to retrain the question generator using reinforcement learning based MIXER model. Further, to improve upon the utility function, we reinterpret it as a discriminator in an adversarial setting and train both the utility function and the MIXER model in a minimax fashion. We find that our adversarial training approach produces more diverse questions compared to both a model trained using maximum likelihood objective and a model trained using utility reward based reinforcement learning. There are several avenues of future work in this area. Following Mostafazadeh et al. (2016), we could combine text input with image input to generate more relevant questions. Because some questions can be answered by looking at the product image in the Amazon dataset (McAuley & Yang, 2016), this could help generate more relevant and useful questions. As in most One significant research challenge in the space of free text generation problems when the set of possible outputs is large, is that of automatic evaluation (Lowe et al., 2016): in our results we saw some correlation between human judgments and automatic metrics, but not enough to trust the automatic metrics completely. Lastly, integrating such a question generation model into a real world platform like StackExchange or Amazon to understand the real utility of such models and to unearth additional research questions.

REFERENCES

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

Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pp. 65–72, 2005.

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

Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Andrea Gesmundo, Neil Houlsby, Wojciech Gajewski, and Wei Wang. Ask the right questions: Active question reformulation with reinforcement learning. *arXiv preprint arXiv:1705.07830*, 2017.

Emily L Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. Deep generative image models using a Laplacian pyramid of adversarial networks. In *Advances in Neural Information Processing Systems*, pp. 1486–1494, 2015.

Xinya Du, Junru Shao, and Claire Cardie. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pp. 1342–1352, 2017.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 866–874, 2017.

William Fedus, Ian Goodfellow, and Andrew M Dai. MaskGAN: Better text generation via filling in the _. *arXiv preprint arXiv:1801.07736*, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

Art Graesser, Vasile Rus, and Zhiqiang Cai. Question classification schemes. In *Proc. of the Workshop on Question Generation*, 2008.

H Paul Grice. Logic and conversation. *1975*, pp. 41–58, 1975.

Michael Heilman. *Automatic factual question generation from text*. PhD thesis, Carnegie Mellon University, 2011.

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

Harsh Jhamtani, Varun Gangal, Eduard Hovy, and Eric Nyberg. Shakespearizing modern language using copy-enriched sequence to sequence models. In *Proceedings of the Workshop on Stylistic Variation*, pp. 10–19, 2017.

Jad Kabbara and Jackie Chi Kit Cheung. Stylistic transfer in natural language generation systems using recurrent neural networks. In *Proceedings of the Workshop on Uphill Battles in Language Processing: Scaling Early Achievements to Robust Methods*, pp. 43–47, 2016.

Igor Labutov, Sumit Basu, and Lucy Vanderwende. Deep questions without deep understanding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pp. 889–898, 2015.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pp. 994–1003, 2016a.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1192–1202, 2016b.

Jiwei Li, Will Monroe, Tianlin Shi, Sėbastien Jean, Alan Ritter, and Dan Jurafsky. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2157–2169, 2017.

Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pp. 366–376. Association for Computational Linguistics, 2010.

Ryan Lowe, Iulian V. Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. On the evaluation of dialogue systems with next utterance classification. In *SIGDIAL*, 2016.

Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421, 2015.

Julian McAuley and Alex Yang. Addressing complex and subjective product-related queries with customer reviews. In *Proceedings of the 25th International Conference on World Wide Web*, pp. 625–635. International World Wide Web Conferences Steering Committee, 2016.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 43–52. ACM, 2015.

Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pp. 1802–1813, 2016.

Nasrin Mostafazadeh, Chris Brockett, Bill Dolan, Michel Galley, Jianfeng Gao, Georgios Spithourakis, and Lucy Vanderwende. Image-grounded conversations: Multimodal context for natural question and response generation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pp. 462–472, 2017.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pp. 280–290, 2016.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318. Association for Computational Linguistics, 2002.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

Sudha Rao and Hal Daumé III. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 2018.

Sudha Rao and Joel Tetreault. Dear Sir or Madam, May I introduce the GYAFC Dataset: Corpus, Benchmarks and Metrics for Formality Style Transfer. In *HLT-NAACL*. The Association for Computational Linguistics, 2018.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *CVPR*, volume 1, pp. 3, 2017.

Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Cristian Moldovan. The first question generation shared task evaluation challenge. In *Proceedings of the 6th International Natural Language Generation Conference*, pp. 251–257. Association for Computational Linguistics, 2010.

Vasile Rus, Paul Piwek, Svetlana Stoyanchev, Brendan Wyse, Mihai Lintean, and Cristian Moldovan. Question generation shared task and evaluation challenge: Status report. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pp. 318–320. Association for Computational Linguistics, 2011.

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

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

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pp. 3104–3112, 2014.

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

Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li. Neural generative question answering. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pp. 2972–2978. AAAI Press, 2016.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. SeqGAN: Sequence generative adversarial nets with policy gradient. In *arxiv*, 2017.

Answer-based Adversarial Training for Generating Clarification Questions [[Supplementary Material]]

A Details of sequence-to-sequence model

In this section, we describe the attention based sequence-to-sequence model introduced in §2.1 of the main paper. In Eq 1, $\tilde{h}_t$ is the attentional hidden state of the RNN at time t obtained by concatenating the target hidden state h_t and the source-side context vector $\tilde{c}_t$, and W_s is a linear transformation that maps h_t to an output vocabulary-sized vector. The predicted token q_t is the token in the vocabulary that is assigned the highest probability using the softmax function. Each attentional hidden state $\tilde{h}_t$ depends on a distinct input context vector $\tilde{c}_t$ computed using a global attention mechanism over the input hidden states as:

$$\tilde{c}_t = \sum_{n=1}^{N} a_{nt} h_n \tag{7}$$

$$a_{nt} = \text{align}(h_n, h_t) = \exp\left[h_t^T W_a h_n\right] \Big/ \sum_{n'} \exp\left[h_t^T W_a h_{n'}\right] \tag{8}$$

The attention weights a_{nt} is calculated based on the alignment score between the source hidden state h_n and the current target hidden state h_t.

B Experimental Details

In this section, we describe the details of our experimental setup. We preprocess all inputs (context, question and answers) using tokenization and lowercasing. We set the max length of context to be 100, question to be 20 and answer to be 20. Our sequence-to-sequence model (§ 2.1) operates on word embeddings which are pretrained on in domain data using Glove (Pennington et al., 2014). We use embeddings of size 200 and a vocabulary with cut off frequency set to 10. During train time, we use teacher forcing. During test time, we use beam search decoding with beam size 5. We use a hidden layer of size two for both the encoder and decoder recurrent neural network models with size of hidden unit set to 100. We use a dropout of 0.5 and learning ratio of 0.0001 In the MIXER model, we start with $\Delta = T$ and decrease it by 2 for every epoch (we found decreasing Δ to 0 is ineffective for our task, hence we stop at 2).

C Details of human based evaluation

In this section, we describe in detail the human based evaluation methodology introduced in §3.3 of the main paper.

Relevance: We ask a Yes-No question: *"Is the question on topic"*

Grammaticality: We ask *"Is the question grammatical?"*, and let workers choose from: [Grammatical, Comprehensible and Incomprehensible].

Specificity: We ask *"How specific is the question?"* and let workers choose from:
1. Specific pretty much only to this product
2. Specific to this and other very similar products (same product from different manufacturer)
3. Generic enough to be applicable to many other products of this type
4. Generic enough to be applicable to any product under Home and Kitchen
5. N/A (Not applicable): Question is not on topic OR is incomprehensible

Seeking new information: We ask *"Does the question ask for new information currently not included in the description?"* and let workers choose from: [Completely, Somewhat, No, N/A]

Usefulness: We ask *"How useful is the question to a potential buyer (or a current user) of the product?"* and let workers choose from:

1. Useful enough to be included in the product description
2. Useful to a large number of potential buyers (or current users)
3. Useful to a small number of potential buyers (or current users)
4. Useful only to the person asking the question
5. N/A (Not applicable): Not on topic OR incomprehensible OR not asking new information