Rationalization: A Neural Machine Translation Approach to Generating Natural Language Explanations

Upol Ehsan *[†], **Brent Harrison** *[‡], **Larry Chan**[†], and **Mark Riedl**[†]
[†]Georgia Institute of Technology, Atlanta, GA, USA
[‡]University of Kentucky, Lexington, KY, USA
ehsanu@gatech.edu, harrison@cs.uky.edu, larrychan@gatech.edu, riedl@cc.gatech.edu

Abstract

We introduce *AI rationalization*, an approach for generating explanations of autonomous system behavior as if a human had performed the behavior. We describe a rationalization technique that uses neural machine translation to translate internal state-action representations of an autonomous agent into natural language. We evaluate our technique in the Frogger game environment, training an autonomous game playing agent to rationalize its action choices using natural language. A natural language training corpus is collected from human players thinking out loud as they play the game. We motivate the use of rationalization as an approach to explanation generation and show the results of two experiments evaluating the effectiveness of rationalization. Results of these evaluations show that neural machine translation is able to accurately generate rationalizations that describe agent behavior, and that rationalizations are more satisfying to humans than other alternative methods of explanation.

Introduction

Autonomous systems must make complex sequential decisions in the face of uncertainty. *Explainable AI* refers to artificial intelligence and machine learning techniques that can provide human understandable justification for their behavior. With the proliferation of AI in everyday use, explainability is important in situations where human operators work alongside autonomous and semi-autonomous systems because it can help build rapport, confidence, and understanding between the agent and its operator. For instance, a non-expert human collaborating with a robot for a search and rescue mission requires confidence in the robot's action. In the event of failure—or if the agent performs unexpected behaviors—it is natural for the human operator *to want to know why*. Explanations help the human operator understand why an agent failed to achieve a goal or the circumstances whereby the behavior of the agent deviated from the expectations of the human operator. They may then take appropriate remedial action: trying again, providing more training to machine learning algorithms controlling the agent, reporting bugs to the manufacturer, etc.

Explanation differs from *interpretability*, which is a feature of an algorithm or representation that affords inspection for the purposes of understanding behavior or results. While there has been work done recently on the interpretability of neural networks (Yosinski et al. 2015; Zeiler and Fergus 2014), these studies mainly focus on interpretability for experts on non-sequential problems. Explanation, on the other hand, focuses on sequential problems, is grounded in natural language communication, and is theorized to be more useful for non-AI-experts who need to operate autonomous or semi-autonomous systems.

In this paper we introduce a new approach to explainable AI: *AI rationalization*. AI rationalization is a process of producing an explanation for agent behavior *as if a human had performed the behavior*. AI rationalization is based on the observation that there are times when humans may not have full conscious access to reasons for their behavior and consequently may not give explanations that literally reveal how a decision was made. In these situations, it is more likely that humans create plausible explanations on the spot when pressed. However, we accept human-generated rationalizations as providing some lay insight into the mind of the other.

AI rationalization has a number of potential benefits over other explainability techniques: (1) by communicating like humans, rationalizations are naturally accessible and intuitive to humans, especially non-experts (2) humanlike communication between autonomous systems and human operators may afford human factors advantages such as higher degrees of satisfaction, confidence, rapport, and willingness to use autonomous systems; (3) rationalization is fast, sacrificing absolute accuracy for real-time response, appropriate for real-time human-agent collaboration. Should deeper, more accurate explanations or interpretations be necessary, rationalizations may need to be supplemented by other explanation, interpretation, or visualization techniques.

We propose a technique for AI rationalization that treats the generation of explanations as a problem of *translation* between ad-hoc representations of states and actions in an autonomous system's environment and natural language. To do this, we first collect a corpus of natural language utterances from people performing the learning task. We then use these utterances along with state information to train an encoder-decoder neural network to translate between state-action information and natural language.

[*] Harrison and Ehsan equally contributed to this work

To evaluate this system, we explore how AI rationalization can be applied to an agent that plays the game *Frogger*. This environment is notable because conventional learning algorithms, such as reinforcement learning, do not learn to play Frogger like human players, and our target audience would not be expected to understand the specific information about how an agent learns to play this game or why it makes certain decisions during execution. We evaluate our approach by measuring how well it can generate rationalizations that accurately describe the current context of the Frogger environment. We also examine how humans view rationalizations by measuring how satisfying rationalizations are compared to other baseline explanation techniques. The contributions of our paper are as follows:

- We introduce the concept of *AI rationalization* as an approach to explainable AI.
- We describe a technique for generating rationalizations that treats explanation generation as a language translation problem from internal state to natural language.
- We report on an experiment using semi-synthetic data to assess the accuracy of the translation technique.
- We analyze how types of rationalization impact human satisfaction and use these findings to inform design considerations of current and future explainable agents.

Background and Related Work

For a model to be interpretable it must be possible for humans to explain why it generates certain outputs or behaves in a certain way. Inherently, some machine learning techniques produce models that are more interpretable than others. For sequential decision making problems, there is often no clear guidance on what makes a good explanation. For an agent using Q-learning (Watkins and Dayan 1992), for example, explanations of decisions could range from "the action had the highest Q value given this state" to "I have explored numerous possible future state-action trajectories from this point and deemed this action to be the most likely to achieve the highest expected reward according to iterative application of the Bellman update equation."

An alternate approach to creating interpretable machine learning models involves creating separate models of explainability that are often built on top of black box techniques such as neural networks. These approaches, sometimes called *model-agnostic* (Ribeiro, Singh, and Guestrin 2016; Zeiler and Fergus 2014; Yosinski et al. 2015) approaches, allow greater flexibility in model selection since they enable black-box models to become interpretable. Other approaches seek to learn a naturally interpretable model which describes predictions that were made (Krause, Perer, and Ng 2016) or by intelligently modifying model inputs so that resulting models can describe how outputs are affected (Ribeiro, Singh, and Guestrin 2016).

Explainable AI has been explored in the context of ad-hoc techniques for transforming simulation logs to explanations (van Lent, Fisher, and Mancuso 2004), intelligent tutoring systems (Core et al. 2006), transforming AI plans into natural language (van Lent et al. 2005), and translating multiagent communication policies into natural language (Andreas, Dragan, and Klein 2017). Our work differs in that the generated rationalizations do not need to be truly representative of the algorithm's decision-making process. This is a novel way of applying explainable AI techniques to sequential decision-making in stochastic domains.

AI Rationalization

Rationalization is a form of explanation that attempts to justify or explain an action or behavior based on how a human would explain a similar behavior. Whereas explanation implies an accurate account of the underlying decision-making process, AI rationalization seeks to generate explanations that closely resemble those that a human would most likely give were he or she in full control of an agent or robot. We hypothesize that rationalizations will be more accessible to humans that lack the significant amount of background knowledge necessary to interpret explanations and that the use of rationalizations will result in a greater sense of trust or satisfaction on the part of the user. While Rationalizations generated by an autonomous or semi-autonomous system need not accurately reflect the true decision-making process underlying the agent system, they must still give some amount of insight into what the agent is doing.

Our approach for translating representations of states and actions to natural language consists of two general steps. First, we must create a training corpus of natural language and state-action pairs. Second, we use this corpus to train an encoder-decoder network to translate the state-action information to natural language (workflow in Figure 1).

Training Corpus

Our technique requires a training corpus that consists of state-action pairs annotated with natural language explanations. To create this corpus, we ask people to complete the agent task's in a virtual environment and "think aloud" as they complete the task. We record the visited states and performed actions along with the natural language utterances of critical states and actions. This method of corpus creation ensures that the annotations gathered are associated with specific states and actions. In essence we create parallel corpora, one of which contains state representations and actions, the other containing natural language utterances.

The precise representation of states and actions in the autonomous system does not matter as long as they can be converted to strings in a consistent fashion. Our approach emphasizes that it should not matter how the state representation is structured and the human operator should not need to know how to interpret it.

Translation from Internal Representation to Natural Language

We use encoder-decoder networks to translate between complex state and action information and natural language rationalizations. Encoder-decoder networks, which have primarily been used in machine translation and dialogue systems, are a generative architecture comprised of two component networks that learn how to translate an input sequence

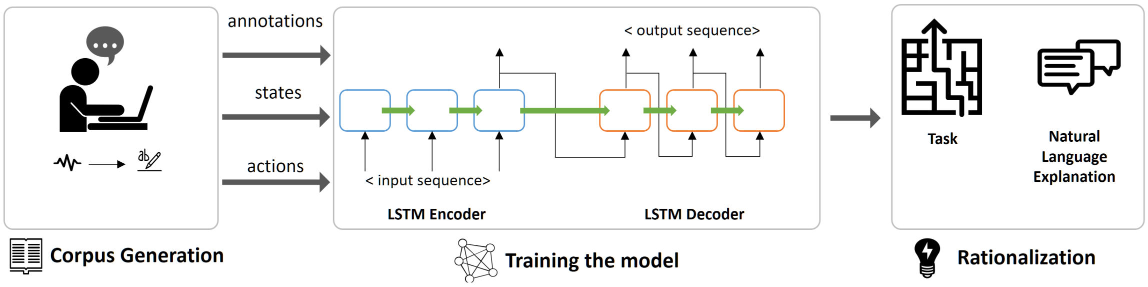


Figure 1: The workflow for our proposed neural machine translation approach to rationalization generation.

$X = (x_1, ..., x_T)$ into an output sequence $Y = (y_1, ..., y_{T'})$. The first component network, the encoder, is a recurrent neural network (RNN) that learns to encode the input vector X into a fixed length context vector v. This vector is then used as input into the second component network, the decoder, which is a RNN that learns how to iteratively decode this vector into the target output Y. We specifically use an encoder-decoder network with an added attention mechanism (Luong, Pham, and Manning 2015).

Experiments

In this work, we test the following two hypotheses:

1. Encoder-Decoder networks can accurately generate rationalizations that fit the current situational context of the learning environment and
2. Humans will find rationalizations more satisfying than other forms of explainability

To test these hypotheses, we perform two evaluations in an implementation of the popular arcade game, *Frogger*. We chose Frogger as an experimental domain because computer games have been demonstrated to be good stepping stones toward real-world stochastic environments (Laird and van Lent 2001; Mnih et al. 2015) and because Frogger is fast-paced, has a reasonably rich state space, and yet can be learned optimally without too much trouble.

Rationalization Generation Study Methodology

Evaluating natural language generation is challenging; utterances can be “correct” even if they do not exactly match known utterances from a testing dataset. To facilitate the assessment of rationalizations generated by our technique, we devised a technique whereby semi-synthetic natural language was paired against state-action representations internal to an autonomous system. The semi-synthetic language was produced by observing humans “thinking out loud” while performing a task and then creating grammar that reproduced and generalized the utterances (described below). This enables us to use the grammar to evaluate the accuracy of our system since we can compare the rationalizations produced by our system to the most likely rule that would have generated that utterance in the grammar. Similar approaches involving the use of semi-synthetic corpora have been adopted in scenarios, such as text understanding (Weston et al. 2015), where ground truth is necessary to evaluate the system.

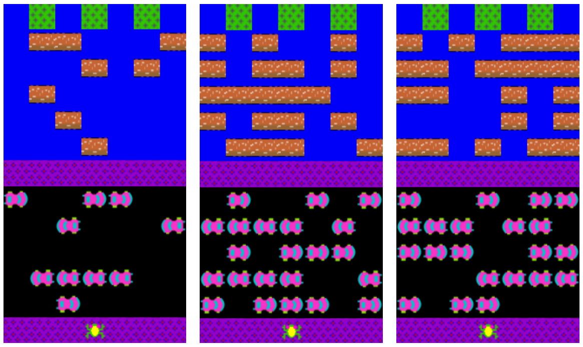


Figure 2: Testing and training maps made with 25% obstacles (left), 50% obstacles (center), and 75% obstacles (right).

We conducted the experiments by generating rationalizations for states and actions in a custom implementation of the game *Frogger*. In this environment, the agent must navigate from the bottom of the map to the top while avoiding obstacles in the environment. The actions available to the agent in this environment are movement actions in the four cardinal directions and action for standing still.

We evaluate our rationalization technique against two baselines. The first baseline, the *random* baseline, randomly selects any sentence that can be generated by the testing grammar as a rationalization. The second baseline, the *majority vote* baseline, always selects sentences associated with the rule that is most commonly used to generate rationalizations on a given map.

Below we will discuss the process for creating the grammar, our training/test sets, and the results of this evaluation in more detail.

Grammar Creation In order to translate between state information and natural language, we first need ground truth rationalizations that can be associated explicitly with state and action information. To generate this information, we used crowdsourcing to gather a set of gameplay videos of 12 human participants from 3 continents playing *Frogger* while engaging in a think-aloud protocol, following the work by (Dorst and Cross 2001; Fonteyn, Kuipers, and Grobe 1993).

After players completed the game, they uploaded their gameplay video to an online speech transcription service and assigned their own utterances to specific actions. This layer of self-validation in the data collection process facilitates the robustness of the data. This process produced 225 action-rationalization trace pairs of gameplay.

We then used these action-rationalization annotations to construct a grammar for generating synthetic sentences, grounded in natural language. This grammar uses a set of rules based on in-game behavior of the Frogger agent to generate rationalizations that resemble the crowdsourced data gathered previously. Since the grammar contains the rules that govern when certain rationalizations are generated, it allows us to compare automatically generated rationalizations against a ground-truth that one would not normally have if the entire training corpus was crowdsourced.

Training and Test Set Generation Since we use a grammar to produce ground truth rationalizations, one can interpret the role of the encoder-decoder network as learning to reproduce the grammar. In order to train the network to do this, we use the grammar to generate rationalizations for each state in the environment. The rules that the grammar uses to generate rationalizations are based on a combination of the world state and the action taken. Specifically, the grammar uses the following triple to determine which rationalizations to generate: (s_1, a, s_2). Here, s_1 is the initial state, a is the action performed in s_1, and s_2 is the resulting state of the world after action a is executed. States s_1 and s_2 consist of the (x, y) coordinates of the agent and the current layout of grid environment. We use the grammar to generate a rationalization for each possible (s_1, a, s_2) triple in the environment and then group these examples according to their associated grammar rules. For evaluation, we take 20% of the examples in each of these clusters and set them aside for testing. This ensures that the testing set contains a representative sample of the parent population while still containing example triples associated with each rule in the grammar.

To aid in training we duplicate the remaining training examples until the training set contains 1000 examples per grammar rule and then inject noise into these training samples in order to help avoid overfitting. Recall that the input to the encoder-decoder network is a triple of the form (s_1, a, s_2) where s_1 and s_2 are states. To inject noise, we randomly select 30% of the rows in this map representation for both s_1 and s_2 and redact them by replacing them with a dummy value.

To evaluate how our technique for rationalization performs under different environmental conditions, we developed three different maps. The first map was randomly generated by filling 25% of the bottom with car obstacles and filling 25% of the top with log platforms. The second map was 50% cars/logs and the third map was 75% cars/logs (see Figure 2). For the remainder of the paper, we refer to these maps as the *25% map*, the *50% map*, and the *75% map* respectively. We also ensured that it was possible to complete each of these maps to act as a loose control on map quality.

Training and Testing the Network The parallel corpus of state-action representations and natural language are used to train an encoder-decoder neural translation algorithm based on (Luong, Pham, and Manning 2015). We use a 2-layered encoder-decoder network with attention using long short-term memory (LSTM) nodes with a hidden node size of 300. We train the network for 50 epochs and then use it to generate rationalizations for each triple in the testing set.

To evaluate the accuracy of the encoder-decoder network, we need to have a way to associate the sentence generated by our model with a rule that exists in our grammar. The generative nature of encoder-decoder networks makes this difficult as its output may accurately describe the world state, but not completely align with the test example's output. To determine the rule most likely to be associated with the generated output, we use BLEU score (Papineni et al. 2002) to calculate sentence similarity between the sentence generated by our predictive model with each sentence that can be generated by the grammar and record the sentence that achieves the highest score. We then identify which rule in the grammar could generate this sentence and use that to calculate accuracy. If this rule matches the rule that was used to produce the test sentence then we say that it was a match.

Accuracy is defined as the percentage of the predictions that matched their associated test example. We discard any predicted sentence with a BLEU score below 0.7 when compared to the set of all generated sentences. This threshold is put in place to ensure that low quality rationalizations in terms of language syntax do not get erroneously matched to rules in the grammar.

It is possible for a generated sentence to be associated with more than one rule in the grammar if, for example, multiple rules achieve the same, highest BLEU score. If the rule that generated the testing sentence matches at least one of the rules associated with the generated sentence, then we count this as a match.

Rationalization Generation Results The results of our experiments validating our first hypothesis can be found in Table . As can be seen in the table, the encoder-decoder network was able to consistently outperform both the random baseline and majority baseline models. Comparing the maps to each other, the encoder-decoder network produced the highest accuracy when generating rationalizations for the 75% map, followed by the 25% map and the 50% map respectively. To evaluate the significance of the observed differences between these models, we ran a chi-squared test between the models produced by the encoder-decoder network and random predictor as well as between the encoder-decoder network models and the majority classifier. Each difference was deemed to be statistically significant ($p < 0.05$) across all three maps.

Rationalization Generation Discussion The models produced by the encoder-decoder network significantly outperformed the baseline models in terms of accuracy percentage. This means that this network was able to better learn when it was appropriate to generate certain rationalizations when compared to the random and majority baseline models. Given the nature of our test set as well, this gives evidence to the claim that these models can generalize to unseen states as well. While it is not surprising that encoder-decoder net-

Table 1: Accuracy values for Frogger environments with different obstacle densities. Accuracy values for sentences produced by the encoder-decoder network (full) significantly outperform those generated by a random model and a majority classifier as determined by a chi-square test.

Map	Full	Random	Majority vote
25% obstacles	**0.777**	0.00	0.168
50% obstacles	**0.687**	0.00	0.204
75% obstacles	**0.80**	0.00	0.178

works were able to outperform these baselines, the margin of difference between these models is worth noting. The performances of both the random and majority classifiers are a testament to the complexity of this problem.

These results give strong support to our claim that our technique for creating AI rationalizations using neural machine translation can accurately produce rationalizations that are appropriate to a given situation.

Rationalization Satisfaction Study Methodology

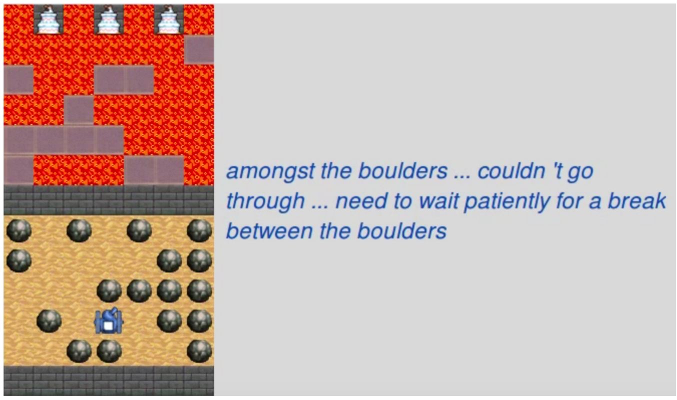
The results of our previous study indicate that our technique is effective at producing appropriate rationalizations. This evaluation is meant to validate our second hypothesis that humans would find rationalizations more satisfying than other types of explanation for sequential decision making problems. To do this, we asked people to rank and justify their relative satisfaction with explanations generated by three agents (described below) as each performs the same task in identical ways, only differing in the way they express themselves. The three agents are:

- *The rationalizing robot*, uses our neural translation approach to generate explanations.
- *The action-declaring robot*, states its action without any justification. For instance, it states "I will move right".
- *The numerical robot*, simply outputs utility values with no natural language rationalizations.

We will discuss our human subjects protocol and experimental results below.

Participants Fifty-three adults (age range = 22 − 64 years, M = 34.1, SD = 9.38) were recruited from Amazon Mechanical Turk (AMT) through a management service called TurkPrime (Litman, Robinson, and Abberbock 2017). Twenty-one percent of the participants were women, and only three countries were reported when the participants were asked what country they reside in. Of these, 91% of people reported that they live in the United States.

Procedure After reading a brief description of our study and consenting to participate, participants were introduced to a hypothetical high-stakes scenario. In this scenario, the participant must remain inside a protective dome and rely on autonomous agents to retrieve food packages necessary for survival. The environment is essentially a "re-skinned" version of Frogger (see figure 3) that is contextually appropriate for the high-stakes hypothetical scenario. To avoid effects of preconceived notions, we did not use the agents' descriptive names in the study; we introduced the agents as "Robot A" for the *rationalizing* robot, "Robot B" for the *action-declaring* robot, and "Robot C" for the *numerical* robot.



amongst the boulders ... couldn 't go through ... need to wait patiently for a break between the boulders

Figure 3: The *rationalizing robot* navigating the modified Frogger environment

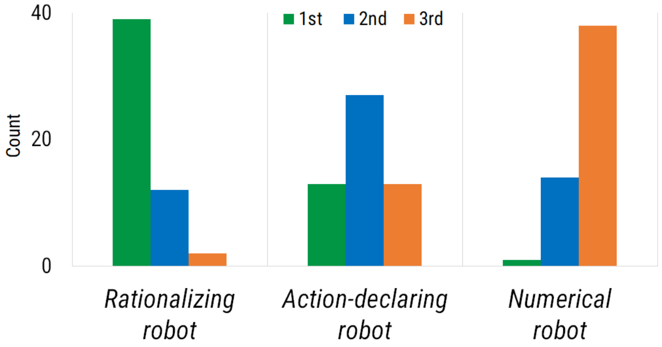


Figure 4: Count of 1st, 2nd, and 3rd place ratings given to each robot. The rationalization robot received the most 1st place, the action-declaring robot received the most 2nd place, and the numeric robot received the most 3rd place ratings.

Next, the participants watched a series of six videos in two groups of three: three depicting the agents succeeding and three showing them failing. Participants were quasi-randomly assigned to one of the 12 possible presentation orderings, such that each ordering was designed to have the same number of participants. After watching the videos, participants were asked to rank their satisfaction with the expressions given by each of the three agents and to justify their choices in their own words.

Satisfaction Results and Analysis Figure 4 shows that the *rationalizing robot* (Robot A) received the most 1st place ratings, the *action-declaring robot* (Robot B) received the most 2nd place ratings, and the *numerical robot* (Robot C) received the most 3rd place ratings. To determine whether any of these differences in satisfaction ratings were significant, we conducted a non-parametric Friedman test of differences among repeated measures. This yielded a Chi-square value of 45.481, which was significant ($p < 0.001$).

To determine which of the ratings differences were significant, we made pairwise comparisons between the agents, using the Wilcoxon-Nemenyi-McDonald-Thompson test (Hollander, Wolfe, and Chicken 2013). All three comparisons yielded a significant difference in ratings. The satisfaction ratings for the *rationalization* robot were significantly higher than those for both the *action-declaring* robot ($p = 0.0059$) as well as the *numerical* robot ($p < 0.001$). Furthermore, the ratings for the *action-declaring* robot were significantly higher than those for the *numeric* robot ($p < 0.001$).

We also analyzed the justifications that participants provided for their rankings using approaches inspired by thematic analysis (Aronson 1995) and grounded theory (Strauss and Corbin 1994). Starting with an open coding scheme, we developed a set of codes that covered various reasonings behind the ranking of the robots. Using the codes as analytic lenses, we clustered them under emergent themes, which shed light into the dimensions of satisfaction. Through an iterative process performed until consensus was reached, we distilled the most relevant themes into insights that can be used to understand the "whys" behind satisfaction of explanations. In our discussion of these responses, we refer to participants using the following abbreviation: P1 is used to refer to participant 1, P2 is used to refer to participant 2, etc.

Findings and Discussion As we hypothesized, the *rationalizing* agent's explanations were rated higher than were those of the other two agents, implying that rationalization enhances satisfaction over action-declaring, natural language description and over numeric expressions.

In addition to the preference for a natural language substrate, four attributes emerged from our thematic analysis that characterize prototypical satisfactory rationalization: *explanatory power*, *relatability*, *ludic nature*, and *adequate detail*. These same attributes can be used to distinguish the rationalizing robot from the action-declaring robot.

In terms of *explanatory power*, the *rationalizing robot's* ability to explain its actions was the most cited reasons for its superior placement in the satisfaction rankings. Human rationalizations allow us to form a theory of mind for the other (Goldman and others 2012), enabling us to better understand motivations and actions of others. Similarly, the rationalizing robot's ability to show participants "...what it's doing and why" (P6) enabled them to "...get into [the rationalizing robot's] mind" (P17), boosting satisfaction and confidence. Despite using natural language, the *action declaring robot* yielded dissatisfaction. As P38 puts it, "[The action-declaring robot] explained almost nothing...which was disappointing." The explanatory attribute of the rationalizing robot reduces friction of communication and results in improved satisfaction.

With respect to *relatability*, the personality expressed through *rationalizing robot's* explanation allowed participants to relate to it:

> [The rationalizing robot] was relatable. He felt like a friend rather than a robot. I had a connection with [it] that would not be possible with the other 2 robots because of his built-in personality. (P21)

Participants also engaged with the rationalizing robot's *ludic quality*, expressing their appreciation of its perceived playfulness: "[The rationalizing robot] was fun and entertaining. I couldn't wait to see what he would say next!" (P2).

A rationalization yields higher satisfaction if it is *adequately detailed*. The *action-declaring robot*, despite its lack of explainability, received some positive comments. People who preferred the action-declaring robot over the rationalizing robot claimed that "[the rationalizing robot] talks too much" (P47), the action-declaring robot is "nice and simple" (P48), and that they "would like to experience a combination of [the action-declaring robot] and [the rationalizing robot]" (P41). Context permitting, there is a need to balance level of detail with information overload.

These findings also align with our proposed benefits of AI Rationalization, especially in terms accessible explanations that are intuitive to the non-expert. We also observed how the human-centered communication style facilitates higher degrees of rapport. The insights not only help evaluate the quality of responses generated by our system, but also sheds light into design considerations that can be used to build the next generation of explainable agents.

Future Work

Our next step is to build on our current work and investigate hypotheses about how types of rationalizations impact human preferences of AI agents in terms of confidence, perceived intelligence, tolerance to failure, etc. To address these questions, it will be necessary to conduct experiments similar to the one described above. It will be interesting to see how inaccurate rationalizations can be before feelings of confidence and rapport are significantly affected. Our experimental methodology can be adapted to inject increasingly more error into the rationalizations and understand human preferences.

Conclusions

AI rationalization provides a new lens through which we can explore the realms of Explainable AI. As society and AI integrates further, we envision the increase in human operators who will want to know *why* an agent does *what* it does in an intuitive and accessible manner.

We have shown that creating rationalizations using neural machine translation techniques produces rationalizations with accuracies above baselines. We have also shown that rationalizations produced using this technique were more satisfying than other alternative means of explanation.

Rationalization allows autonomous systems to be relatable and human-like in their decision-making when their internal processes can be non-intuitive. We believe that AI rationalization can be an important step towards the democratization of real-world commercial robotic systems in healthcare, accessibility, personal services, and military teamwork.

References

Andreas, J.; Dragan, A. D.; and Klein, D. 2017. Translating neuralese. *CoRR* abs/1704.06960.

Aronson, J. 1995. A pragmatic view of thematic analysis. *The qualitative report* 2(1):1–3.

Core, M.; Lane, H. C.; van Lent, M.; Gomboc, D.; Solomon, S.; and Rosenberg, M. 2006. Building Explainable Artificial Intelligence Systems. In *Proceedings of the 18th Innovative Applications of Artificial Intelligence Conference*.

Dorst, K., and Cross, N. 2001. Creativity in the design process: co-evolution of problem–solution. *Design studies* 22(5):425–437.

Fonteyn, M. E.; Kuipers, B.; and Grobe, S. J. 1993. A description of think aloud method and protocol analysis. *Qualitative Health Research* 3(4):430–441.

Goldman, A. I., et al. 2012. Theory of mind. *The Oxford handbook of philosophy of cognitive science* 402–424.

Hollander, M.; Wolfe, D. A.; and Chicken, E. 2013. *Nonparametric statistical methods*. John Wiley & Sons.

Krause, J.; Perer, A.; and Ng, K. 2016. Interacting with predictions: Visual inspection of black-box machine learning models. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 5686–5697. ACM.

Laird, J., and van Lent, M. 2001. Human-level ais killer application: Interactive computer games. *AI Magazine* 22(2):15–25.

Litman, L.; Robinson, J.; and Abberbock, T. 2017. Turkprime. com: A versatile crowdsourcing data acquisition platform for the behavioral sciences. *Behavior research methods* 49(2):433–442.

Luong, M.-T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 1412–1421. Lisbon, Portugal: Association for Computational Linguistics.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.; Veness, J.; Bellemare, M.; Graves, A.; Riedmiller, M.; Fidjeland, A.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529533.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, 311–318. Association for Computational Linguistics.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144. ACM.

Strauss, A., and Corbin, J. 1994. Grounded theory methodology. *Handbook of qualitative research* 17:273–85.

van Lent, M.; ; Carpenter, P.; McAlinden, R.; and Brobst, P. 2005. Increasing replayability with deliberative and reactive planning. In *1st Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 135–140.

van Lent, M.; Fisher, W.; and Mancuso, M. 2004. An explainable artificial intelligence system for small-unit tactical behavior. In *Proceedings of the 16th conference on Innovative Applications of Artifical Intelligence*.

Watkins, C., and Dayan, P. 1992. Q-learning. *Machine Learning* 8(3-4):279292.

Weston, J.; Bordes, A.; Chopra, S.; Rush, A. M.; van Merriënboer, B.; Joulin, A.; and Mikolov, T. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*.

Yosinski, J.; Clune, J.; Fuchs, T.; and Lipson, H. 2015. Understanding neural networks through deep visualization. In *In ICML Workshop on Deep Learning*. Citeseer.

Zeiler, M. D., and Fergus, R. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, 818–833. Springer.