# Natural Language Specification of Reinforcement Learning Policies through Differentiable Decision Trees

Pradyumna Tambwekar pradyumna.tambwekar@gatech.edu
Andrew Silva andrew.silva@gatech.edu
Nakul Gopalan ngopalan@gatech.edu
Matthew Gombolay matthew.gombolay@cc.gatech.edu



***Abstract*— Human-AI policy specification is a novel procedure we define in which humans can collaboratively warm-start a robot's reinforcement learning policy. This procedure is comprised of two steps; (1) Policy Specification, i.e. humans specifying the behavior they would like their companion robot to accomplish, and (2) Policy Optimization, i.e. the robot applying reinforcement learning to improve the initial policy. Existing approaches to enabling collaborative policy specification are often unintelligible black-box methods, and are not catered towards making the autonomous system accessible to a novice end-user. In this paper, we develop a novel collaborative framework to allow humans to initialize and interpret an autonomous agent's behavior. Through our framework, we enable humans to specify an initial behavior model via unstructured, natural language (NL), which we convert to lexical decision trees. Next, we leverage these translated specifications, to warm-start reinforcement learning and allow the agent to further optimize these potentially suboptimal policies. Our approach warm-starts an RL agent by utilizing non-expert natural language specifications without incurring the additional domain exploration costs. We validate our approach by showing that our model is able to produce >80% translation accuracy, and that policies initialized by a human can match the performance of relevant RL baselines in two domains.**

## I. INTRODUCTION

Significant progress has been made in recent years towards developing collaborative human-AI techniques that allow humans to specify a robot's desired behavior or *policy*, a mapping from the state of the world to actions the autonomous agent or robot should take [1]–[3]. However, the proliferation of such human-machine interaction hinges on the development of accessible and interpretable modes for interacting with these autonomous systems. Yet, while humans can provide helpful guiding behaviors for policy specifications, humans have finite cognitive capabilities [4] and may only be able to provide good but supobtimal policy specifications [5]. As such, these interactive agents need to be capable of autonomously improving upon the user's initial policy specification, e.g. via Reinforcement Learning (RL). In this paper, we present a novel collaborative technique that enables humans to (1) specify intelligible policies from unstructured natural language, (2) optimize these specified policies using RL.

To facilitate more accessible human-AI interactions, recent work has advocated for a shift from "model-centered" approaches towards deliberate design of "human-centered" systems focused on human-interactions [6], [7]. Collaborative policy-learning techniques require a similar degree of human-centricity to efficiently integrate autonomous systems or robots in society [8]. However, many contemporary approaches to policy specification do not effectively cater to novice end-users and are unable to provide a means of interpreting an autonomous system's behavior [9]–[11].

Natural language provides an accessible means of interfacing with an autonomous agent or a robot for a novice user [12], [13]. Prior work demonstrated increased user satisfaction when natural language is used as the interface [14]. Experienced users can leverage their expertise to quickly learn how to program robots through a complex interface; however, novice users may struggle or be demotivated [15]. Therefore, we propose a methodology by which novice users can specify their desired behavior through simple and unstructured english language sentences, thus catering to the needs of a more diverse set of users. Policy acquisition through natural language is a well-studied area of research in recent years. Prior work can be broadly divided into either symbolic or connectionist approaches [16]. However, given the lack of interpretability in connectionist methods [17]–[19] and the lack of fine-tuning in symbolic methods [20]–[22], it is difficult to specify policies that are comprehensible to a human trainer and can improve over time.

To address these issues, we propose a Human-AI collaborative policy synthesis architecture that bridges the benefits of both connectionist and symbolic approaches. Our framework consists of a (1) novel deep learning framework, called HAN2Tree, which learns to translate user-generated language descriptions of policies to lexical decision trees (symbolic), and a (2) Differentiable Decision Tree (DDT) [23] model to represent and optimize the human-specified policy with policy gradients (connectionist) given a user-defined task completion signal (i.e., a reward function). Unlike standard deep learning models, DDTs can be discretized after training into intelligible decision tree policies for users to inspect [24]. We empirically demonstrate that optimized policies initialized via our natural language translation technique outperforms or matches the performance of relevant baselines across two domains by utilizing language specifications from novice users. Our contributions are as follows:

1) We present a data collection protocol alongside the largest known dataset mapping natural language instructions from humans to lexical decision trees (400 policy specifications).

All authors were affiliated with the School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA, 30332, while conducting this research

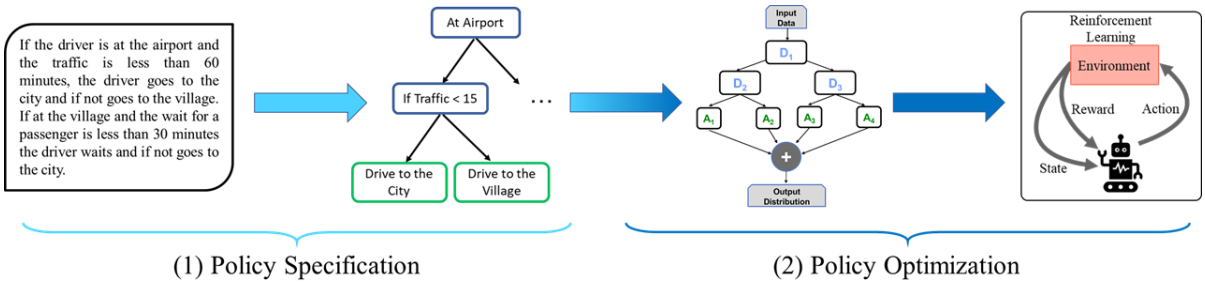


Fig. 1. In our Human-AI collaborative policy synthesis approach, we first convert policy descriptions to lexical decision trees. Each decision tree is encoded as a differentiable decision tree to initialize the RL policy followed by proximal policy optimization to optimize the policy.

2) We develop a novel machine learning architecture to parse natural language instructions into lexical decision trees (HAN2Tree).
3) We show that our method outperforms relevant baselines and obtains a translation accuracies of 86.30% and 80.38% across two domains and demonstrate that these translated trees can successfully warm-start RL.

## II. Related Work

We provide an overview of research in the areas of instruction following and interpretable ML.

**Language to Policies –** Traditional language-based policy specification involves translating natural language to a predicate-based language for planning [25]–[28]. For example, the sentence "move to the left" could map to the predicate $move(a) \wedge dir(a, left)$. This process involves a high degree of feature engineering as both the grammar and the formalizing of the predicate specification require expert design. Some deep learning-based approaches seek to alleviate the dependency on a specified grammar [20], [22]. However, these approaches still require hand-engineering in the form of expert-specified predicates. Similar approaches also consider mapping language directly into an agent's policy [17], [29], [30] or adapt these methodologies to multi-modal systems via language and image conditioned imitation learning [31], [32]. These approaches condition an agent's policy via a combined learned embedding of the state and language instruction. We differentiate ourselves from these approaches by providing a means of human-centered policy specification that is simulatable and accessible, while also allowing for gradient-based policy improvement. In this context, simulatability references the model's ability to be simulated by a human [33].

**Interpretable ML –** Prior work has defined interpretability as "the ability to explain or to present in an understandable way to humans" [34]. One such approach is to visualize the intermediate outputs of a neural network [35]–[37]. These intermediate outputs are usually in the form of post hoc visualizations of intermediate gradients or feature maps to elicit the decision making process of a deep neural network. However, there is ongoing debate over whether these latent representations in a high-dimensional space actually correspond to the meanings assigned to them [38], [39]. Other forms of interpretability include providing post-hoc explanations to provide a rationale for the black-box decision making process of a model [40]–[43]. Popular policy explanation methods include model-based methods, such as constrastive explanations [44], [45], to consolidate competing hypotheses regarding a model's decision making process, or plan-based methods, such as policy summarization [46], [47], which provide a means of summarizing the key details of a learned policy. However, these post-hoc processes are often non-trivial, and the explanations generated may not represent the conceptual model of the machine.

Recent works on interpretable RL have developed architectures which non-experts can utilize to specify and interpret RL-policies through an inherently interpretable design, e.g. decision trees [48], [49]. Such approaches function as "white-box" methods, wherein the explanations provided, and interpretable capabilities come from the transparent design of the architecture itself. However, such approaches, when applied to policy specification, require humans to manually specify all propositional rules corresponding to an entire decision tree, which may be place a high cognitive burden on end-users in complex real-world domains. Combining the compositional structure of decision trees with natural language initialization is more conducive to accessible policy specification. While language may not always be preferred to initializing trees directly, it is important to cater to a variety of end-users, some of whom may prefer language.

**Filling the Gap –** The existing scope of prior work lacks accessibility for non-experts or does not allow for optimization of sub-optimal specifications. Overcoming these limitations, our approach takes steps towards democratizing interactive-AI systems through a transparent, simulatable methodology where end-users can warm-start the agent and interpret the final policy, without expert knowledge or intensive training procedures to learn the domain.

## III. Preliminaries

**Differentiable Decision Trees (DDT) –** Initially introduced for classification and regression [23] and later extended to RL [49], DDTs are parameterized decision trees that can be optimized through backpropagation. In a DDT, the $n^{th}$ node of the tree, $D_n$, contains a set of weights, $\vec{p}_n \in P$, and comparator values, $c_n \in C$. At each decision node, the input state, $X$, is combined with the weights and comparator values and passed through a sigmoid, $\sigma$, approximating reasoning of a decision tree, $D_n = \sigma[\Gamma(\vec{p}_n^T * \vec{X} - c_n)]$, where $\Gamma$ is a scaling constant which throttles the

decision making threshold. Every leaf node, $\vec{l_i} \in L$, contains probabilities (i.e. $l_{i,a} \in [0,1]$ such that $\sum_{a=1}^{|A|} l_{i,a} = 1$) for each output action, $a \in A$, and a path from the root of the tree to its position. The action probabilities in $\vec{l_i}$ are weighted by the path probability of reaching $\vec{l_i}$, which is determined by the output of all decision nodes in the path. The weighted probabilities from all leaves are summed to produce the final output, which is a distribution across all actions, serving as an agent's policy, $\Pi$.

**Language Modelling** - Recurrent neural networks (RNN) are often employed to encode sequences with temporal dependencies [50], such as language. A specific configuration of RNNs, sequence-to-sequence networks (Seq2Seq), have been utilized to great success for language tasks like machine translation, dialogue generation, semantic parsing, etc. [51]. Seq2Seq networks are comprised of an encoder, which generates a fixed size representation of the input, and a decoder, which sequentially models the probability of the next word in the sequence, $p(x_i|X_{j<i}, h)$. Attention was proposed as a means of improving the model's capability to remember long-term dependencies [52], [53]. These approaches produce alignment scores between the words in the input with respect to each word in the output, to produce more contextually grounded predictions. Recent work further expanded on attention-based networks by building sequence encoders built entirely with attention, i.e. Transformers, such that the entire text input is encoded in parallel, via self-attention [54]. Transformers such as BERT [55] and GPT [56] are now widely being deployed to solve language applications that deal with large-scale datasets.

**Hierarchical Attention Network** - The Hierarchical Attention Network (HAN) is a specific RNN architecture, that was shown to be successful for encoding larger language sequences. The HAN consists of two RNN layers. The first layer takes in embeddings, $w_{it}$, for the $t^{th}$ word in the $i^{th}$ sentence in the input and applies self-attention to create a sentence embedding, $s_i$ (Equations 1-3).

$$s_i = \sum_j \alpha_{jt} h_{ij} \tag{1}$$

$$\alpha_{it} = \frac{\exp(h_{it}{}^\intercal W_w)}{\sum_j \exp(h_{ij}{}^\intercal W_w)} \tag{2}$$

$$h_{it} = GRU(h_{it-1}, w_{i:k<t}) \tag{3}$$

Here, $h_{it}$ is the hidden vector corresponding to word $t$ for sentence $i$. $W_w$ represents a weight vector for the self-attention layer and $\alpha_{it}$ corresponds to the importance weight for the hidden state of word $t$ in sentence $i$. The embedding for sentence $i$ is calculated by a weighted combination of the hidden states for every word in the sentence. The second layer of RNNs re-applies hierarchical attention (Equations 1-3) on sentence embeddings to produce the final hidden vector ($H$) for the language description

**Sequence to Tree model -** A relevant model to this paper is the sequence to tree (Seq2Tree) network [57]. Seq2Tree incorporates a special *non-terminal* token to decode a tree structure. Each time the model predicts a non-terminal token, the RNN cell is re-conditioned on the non-terminal token as well as the current hidden vector of the decoder, $h_d$. For example, if the model predicts a non-terminal $y_4$, after three tokens $< y_1, y_2, y_3 >$, the RNN cell is re-conditioned on these tokens and the model begins generating tokens $< y_5, \ldots, y_n >$ using this new RNN cell. A recurrent neural network is trained to maximize the likelihood of predicting the target sequence $a$ ($< \ y_1 \ldots y_n \ >$), given a hidden vector from the encoder $h_e$. For a Seq2Seq network, the likelihood is $p(a|h_e) = p(y_1, \ldots, y_n|h_e)$, as all tokens are predicted sequencially. In the case of this Seq2Tree example, where the a *non-terminal* is predicted at $y_4$, the likelihood is $p(a|h_e) = p(y_1, y_2, y_3, y_4|h_e)p(y_5 \ldots y_n|h_d)$. The decoding process of a Seq2Tree network terminates when all non-terminal sequences have led to an end-of-sequence token.

## IV. Policy Specification and Optimization

Human language can be verbose and unstructured. A successful approach to encoding human-specified policies needs to be theoretically capable of handling document-sized inputs, and identify the individual parts of the instructions pertaining to relevant components of the behavior. For the task of *Policy Specification*, we develop an architecture, HAN2Tree, that leverages the Hierarchical Attention Network as an encoder and extends the *Tree Decoder* proposed in the Seq2Tree paper (see Figure 2).

The ability of the HAN encoder to explicitly learn word-level and sentence-level dependencies allows us to better identify which parts of a disorganized language description are relevant to the policy. A language description is encoded into a vector, $H$, and provided as an input to the *Tree Decoder*. Unlike that of Seq2Tree [57], in our *Tree Decoder*, each non-terminal prediction creates two separate branches to facilitate a decision tree-like structure (see Figure 2). At each decoding step in the *Tree Decoder*, we reapply attention by learning the alignment weights of the hidden vector of the decoder with respect to the hidden vector corresponding to each sentence in the input. By extending the decoder from the Seq2Tree model, each branch of our decoder is able to specifically focus on information from its predecessors while using relevant information from the natural language input via attention.

We define the target language as the set of all possible decision nodes or leaves for a given domain. For our experimental domains (Section V), we specify a dictionary of all possible predicates a priori. Translating from natural language into a structured decision tree of these predicates, we are able to optimize the translated policy by encoding it as a DDT, i.e for *Policy Optimization*. We create a mapping between each lexical predicate to a set of weights, $p_n$ and comparator values, $c_n$, as described in Section III, to initialize the DDT through a lexical tree generated by our network. This intermediate decision tree representation also provides an intelligible modality for a user to visualize their specified policy, should they want to modify or re-specify the policy prior to optimization (see Figure 4). We can then apply proximal policy optimization [58] to improve upon the initial policy specification. Through our approach of

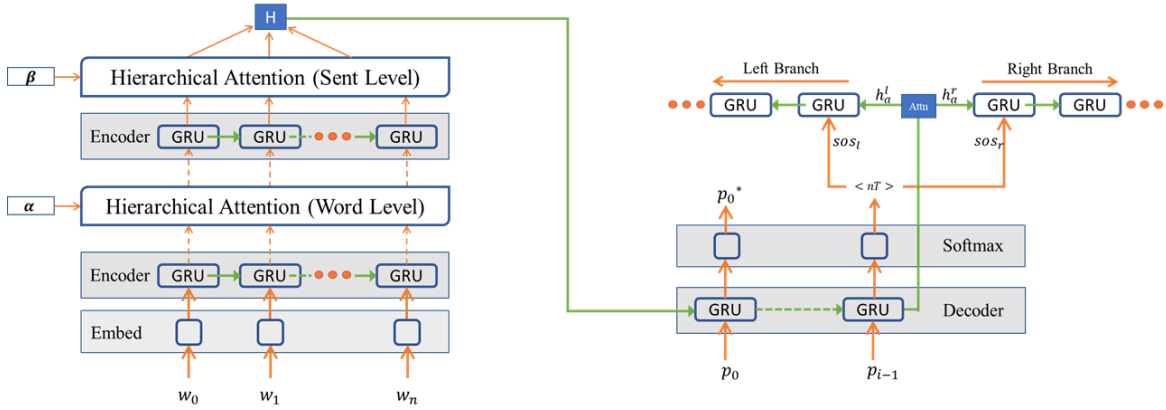


Fig. 2. This figure depicts our HAN2Tree architecture. The left side of the image represents the HAN encoder and the right side of the image represents the Tree decoder. Each sentence in the language description, consisting of n words ($w_0 - w_n$), is input into the HAN encoder. The encoder is comprised of two layers of GRU cells [51], or Gated recurrent units, which are a type of RNN-cell. The encodings for each word in the sentence are then passed into the first word-level attention layer to generate an embedding for each sentence (the attention weights for this layer are represented by $\alpha$). Next, the sentence embeddings pass through another layer of GRU cells, and a second sentence-level attention layer (the attention scores for this layer are represented by $\beta$) to generate a final encoding for the entire input ($H$). This vector is then passed to the Tree Decoder to generate the decision tree. At each step, the GRU cell predicts the next predicate in the tree ($p_i^*$). When a non-terminal token is predicted by the GRU cell, two new branches are created. Decoding stops when all branches predict end of sequence tokens or the maximum depth is reached.

translating natural language into neural network structure and parameters, our HAN2Tree framework facilitates a crucial extension to prior work [49] by accommodating the needs of a larger set of end-user needs, i.e. specification via language.

## V. Experimental Domains

We chose the *taxi* and *highway* domain, which are analogous to sub-tasks within autonomous driving. Autonomous driving is of keen interest in the robotics community [59].

**Taxi Domain –** We adapt the Taxi domain [60] as our first domain. Our domain consists of three locations: the airport, village and city. Passengers are always available at the city, however they may encounter traffic. Whereas at the village, there will be no traffic, but there may be a wait for passengers. The state space consists of the taxi's location, the traffic towards the city, and the village wait time. Actions consists of driving to a destination or waiting for a passenger.

**Highway Domain –** The highway domain was initially proposed by [61] for apprenticeship learning. The highway consists of three lanes, with the traffic all moving in the same direction [62]. The state space is comprised of the [$x$, $y$, $\dot{x}$, $\dot{y}$] vectors for the four closest cars to the agent. ($x$, $y$) corresponds to the position of the car and ($\dot{x}$, $\dot{y}$) represents the velocity of the car in the $x$ and $y$ directions. The agent is rewarded for safely traversing the highway at a high velocity and is given a negative reward for crashing.

## VI. Data Collection

To collect our dataset, we employed Mechanical Turk to crowdsource a novel supervised learning dataset for policy specification. The objectives of our data collection were to: (1) Collect free-form natural language which accurately describes policies that lead to plausible behavior; and (2) Facilitate a varied set of policies specified to ensure that our model is not biased towards specific strategies.

To collect the requisite data, we built a Qualtrics survey [63] and deployed it on Mechanical Turk under a protocol approved by an Institutional Review Board (IRB). We did not collect demographics information for our study. Any participant between the ages of 18 - 65 from an English speaking country was eligible to participate in our study. However, we had no way of enforcing these constraints through Mechanical Turk. Participants did not interact with the domain; rather, users received pictures of the domain (Figure 3). Our interface contained a binary tree of depth four consisting of fifteen empty text-fields (Figure 3). Participants were asked to fill in the tree using a collection of predicates to specify their desired policy. After creating a tree, participants were instructed to submit a natural language description of their specified tree. Collecting language after trees was a deliberate design choice in order to elicit language descriptions that were relevant to the participant-specified tree. Each submitted response was carefully evaluated according to a pre-defined rubric, and only the data points where the instructions described the policy specification were included in our final dataset. Our rubric included checks for whether the majority of the decisions in the tree-policy were references in the language descriptions. We also checked for whether information included in the description was copied from external sources, parts of the study itself, or was completely irrelevant to the policy specified, e.g. "Trees are great, I liked working with trees. I enjoyed this task..." Minor edits were made to the submitted data based on correctness and grammar. Our study collected 400 policy specifications (Text + Tree).

### A. Dataset Preparation

We augmented our dataset by applying synonym replacement and back-translation on the policy descriptions. Back-translation is a common method in language augmentation which leverages machine translation models to translate a

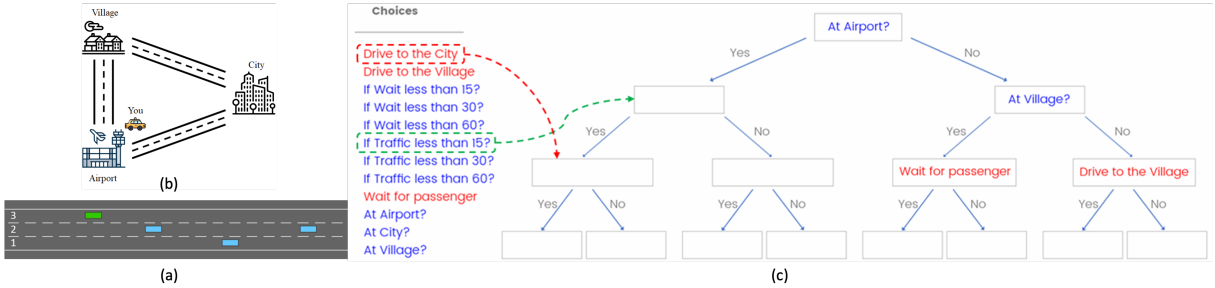


Fig. 3. Figures (a) and (b) show the highway and taxi domains. Figure (c) provides a depiction of the interface we developed to collect decision trees corresponding to natural language descriptions of policies. Turkers could drag and drop options from a list of possible actions/decisions to generate a tree.

TABLE I

MEANS (STANDARD DEVIATIONS) FOR TREE AND TOKEN-WISE ACCURACY WITH 5-FOLD CROSS-VALIDATION. WE REPORT THE AVERAGE 5-FOLD ACCURACY ACROSS THREE 5-FOLD RUNS TO BETTER COMPARE S2T AND H2T, SINCE THE RESULTS OF THESE MODELS WERE VERY SIMILAR.

| | Taxi | | Highway | |
|---|---|---|---|---|
| | Tree Acc | Token-wise Acc | Tree Acc | Token-wise Acc |
| Seq2Seq | 76.54% (0.33) | 90.35% (0.11) | 65.11% (0.96) | 88.12% (0.11) |
| Seq2Tree | 86.04% (0.64) | 94.83% (0.23) | 80.11% (0.27) | 91.58% (0.23) |
| Seq2Tree (BERT Enhanced) | 81.84% (1.26) | 93.07% (0.50) | 77.17% (0.65) | 90.38% (0.67) |
| HAN2Tree (ours) | **86.30% (0.55)** | **95.23% (0.12)** | **80.38% (0.48)** | **92.83% (0.36)** |

sentence to a different language and then translate it back to english. This serves as a means of syntactically changing a sentence while retaining the same semantic meaning. After augmentation, our entire dataset totalled 978 and 998 examples for the Taxi and Highway domains, respectively. Language is highly variable modality, in that two potential users could describe the same behavior in completely different ways. Augmenting our data adds some of this variation to our corpus and makes our model more robust to real-world language. Our source and target vocabulary size amounted to 1283 and 20, respectively, for the Taxi domain and 1162 and 20, respectively, for the Highway domain. The source vocabulary size represents the total number of words utilized in the language descriptions, and the target vocabulary represents the number of action/decision predicates among the trees in the dataset. We applied a 70/30 split on our augmented dataset to create our training and validation sets. While training, words with a frequency of less than 5 were replaced by an unknown token. [1]

## VII. RESULTS

In this section, we empirically validate the advantages of our approach to warm-starting policy optimization with natural language-based policy initialization. First, we show that our approach (HAN2Tree) generates lexical decision trees from language with high accuracy, outperforming relevant baselines (Section VII-A). Second, we demonstrate that we can bootstrap policies through natural language to outperform reinforcement learning baselines (Section VII-B).

[1]Similar language-to-structure datasets [57], [64], [65], typically include language sequences which are almost $1/10^{th}$ the length of those of our datasets, as our language inputs describe entire policies rather than a single command (GeoQuery - 7.56, ATIS - 10.97, Scholar - 6.69, Taxi - 81.37, Highway - 92.03). To the best of our knowledge, no pre-existing dataset is comparable to ours, in terms of size of samples or correspondence of data.

### A. Policy Specification from Language

We hypothesize that (1) no related network will be able to outperform our HAN2Tree architecture (2) training models from scratch is more suitable to our specialized task on a small dataset when compared to leveraging pretrained embeddings on large-scale internet corpora. We employ K-fold cross validation accuracy as our evaluative measure. The effectiveness of our model depends on how accurately it is able to generate decision trees from language, therefore we employed a standard classification measure utilized in prior work. To evaluate our Policy Specification method (HAN2Tree), we employ the following baselines:

1) Seq2Seq: We trained a Seq2Seq network with attention [52] to generate flattened representations of trees.
2) Seq2Tree: An extension of the architecture presented by [57], with a binary Tree Decoder.
3) Seq2Tree (BERT enhanced): The Seq2Tree baseline augmented with pretrained BERT [55] embeddings.

We include a "BERT enhanced" baseline to leverage large-scale pretraining for our encoder, allowing us to see what benefits may be gained by equipping our network with this prior knowledge. We chose to leverage BERT as a feature encoder as prior work has shown that a fused-embedding structure is more effective means of incorporating BERT [66]. Therefore we applied the methodology used by the ELMO architecture [67] to incorporate BERT embeddings into the encoder of the Seq2Tree baseline. We report the 5-fold cross validation accuracy for each of our baselines (Table II). Tree Accuracy is the percentage of trees that exactly matched the target tree, and the token-wise accuracy reports the per-token accuracy of the model. Our best-performing approach achieved a mean tree accuracy of 86.30% (0.55) in translating natural language to decision trees in the Taxi domain. We do note that the Seq2Tree

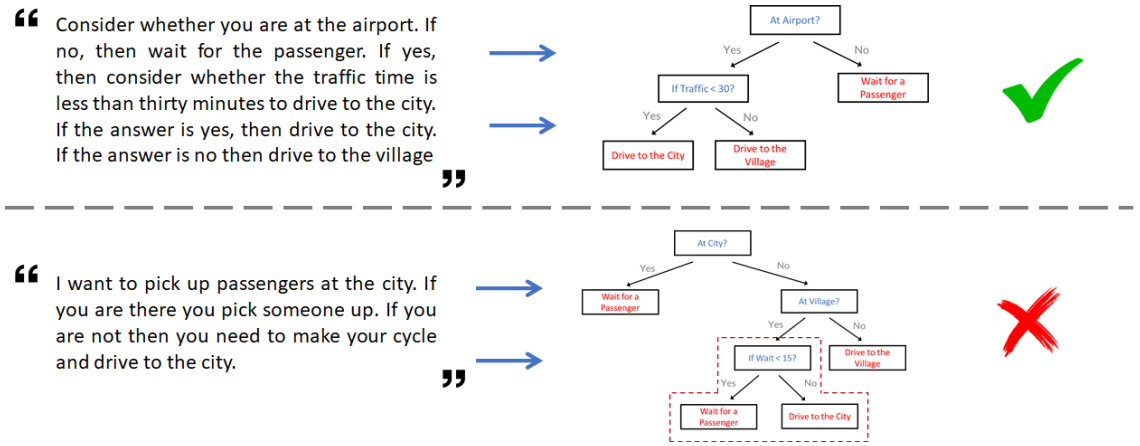


Fig. 4. Two examples of trees correctly and incorrectly parsed by our system. The red box corresponds to the incorrect part of the tree predicted. The model replaced "Drive to the City", with a longer expression which first checked the wait time before driving to the city.

approach achieves a comparable performance to HAN2Tree in both domains. However, our approach, remains a better fit for real-world applications, due to its capacity to more effectively process large text inputs. Despite attention, and updated RNN architectures, RNN models still struggle to retain long-term dependencies for larger input sequences. The HAN encoder explicitly separates words and sentences in order to split up the input into meaningful segments to mitigate some of these issues. Particularly in instances with free-form language descriptions of policies, which could feasibly be comprised of $\sim$1000 words, it is important that the network has the ability to effectively segment the input and model the dependencies of each segment.

The performance of the BERT-enhanced baseline is unable to match either the Seq2Tree or HAN2Tree approaches. We hypothesize that due to the size of our dataset, adapting the information learnt from BERT's large-scale pretraining methodology is more challenging than learning the specific information required for this task from scratch. We acknowledge that the benefits of using a BERT enhanced structure, or the entire pretrained BERT transformer as an encoder, would increase in situations where data for this problems is more abundantly available. However, we leave this point to future work given the challenge of creating a large enough dataset for such a trend to possibly become evident. Sample outputs from our approach are shown in Figure 4.

### B. Policy Optimization through RL

In this section, we show that leveraging language to specify policies as DDTs is a viable method for policy learning. We hypothesize that (1) utilizing natural language initializations from non-expert participants does not inhibit policy learning for DDTs, and that (2) policies initialized through language, via lexical decision trees, can be sufficiently optimized to match or outperform those without language initialization. We compare our approach, a DDT initialized by natural language, with the following baselines:

1) NN (PPO): A neural network (NN) with three fully-connected layers, a baseline used in prior work [49].

TABLE II

THIS TABLE DISPLAYS THE MEDIANS AND STANDARD ERRORS FOR THE INITIAL AND MAXIMUM ROLLING REWARDS FOR DDTS ACROSS TRAINING RUNS. THE WINDOW-SIZE FOR BOTH THE INITIAL AND MAXIMUM ROLLING REWARD WAS 100.

| | Taxi | Highway |
|---|---|---|
| NN Initial | 0.8 $\pm$ 0.43 | -3.79 $\pm$ 0.03 |
| Random Initial | 3.45 $\pm$ 0.28 | -1.49 $\pm$ 0.18 |
| Natural Language Initial (Ours) | **4.04** $\pm$ **0.32** | **-1.28** $\pm$ **0.27** |
| NN Best | 8.22 $\pm$ 0.04 | -0.36 $\pm$ 0.30 |
| Random Best | 8.26 $\pm$ 0.12 | 9.81 $\pm$ 0.43 |
| Natural Language Best (Ours) | **8.79** $\pm$ **0.17** | **10.00** $\pm$ **0.41** |

2) Random DDT: A randomly initialized DDT initialized with eight leaves adopted from prior work [49].

In Table VII-B, we depict the initial and maximum rolling rewards for each baseline. For the NL baseline, we report the average of the initial and maximum rolling reward across the best 5 trees generated through natural language initializations out of a selection of 10-15 trees. The rolling reward was computed across 100 episodes. We find that the best NL model achieves a higher average in terms of the best performing model as well as the initial model compared to the random and FC baselines. The Random DDT baseline also outperforms the NN baseline, which is likely due to helpful inductive bias seen in prior work [49]. The average initial performance was also found to be higher for DDTs initialized by language rather than randomly initialized DDTs. It is interesting to note that the relative initial performance with respect to the random baseline is greater within the taxi domain (+0.6) rather than the highway domain (+0.2). The drop in initial performance between the taxi and highway domain, indicates that the loss in translation accuracy affects the quality of the policy initializations. In future work, we hope to expand our approach to a larger dataset in order to overcome this translation cost. However, by successfully initializing DDTs through natural language, and matching the performance of prior work, we facilitate the first such methodology, combining symbolic and connectionist con-

cepts, where humans can program and visualize their desired policies through free-form natural language.

## VIII. Limitations and Future Work

In this work, we assume that specifying policies via natural language is preferable to directly specifying policies as decision trees through a graphical user interface. While this assumption is supported by prior work [68], it would be interesting to see if this assumption bears out in practice. In future work, we hope to study which modality, between natural language and decision trees, is more suitable for allowing non-experts to specify robot policies.

Another important limitation pertains to the size and scope of our dataset. In our data-collection protocol, participants were limited to specifying trees of depth four to reduce their cognitive load during the study. Based on analysis from pilot studies, trees of depth four were ascertained to be the deepest trees needed to describe the majority of behaviors possible for our domains. However, trees of depth four may be insufficient for other domains. In such cases, one would have to restructure the data-collection UI to collect variable size decision trees, but our approach would still be applicable. With respect to the dataset, we also acknowledge the presence of an inductive bias pertaining to the method of data collection. Since we collected language after participants submitted their trees, participants were more likely to produce language that reflected the tree structure. It might be possible that our approach will not perform as well on language instructions collected "in the wild." However, we maintain that the building blocks presented in this approach will be crucial for future work towards developing a simulatable model for in-the-wild policies.

## IX. Conclusion

In this paper, we bridge relevant symbolic and connectionist methods, developing a human-centered, interpretable framework for policy specification through natural language, policy improvement via reinforcement learning. Furthermore, we showcase a data collection methodology that can be used to collect decision trees, of any size, and the natural language descriptions corresponding to these trees for any given domain. Utilizing this protocol, we crowd-sourced the largest known corpus mapping unstructured, free-form natural language instructions to lexical decision trees. Our novel machine learning architecture, called HAN2Tree, was the first approach capable of generating lexical decision trees from language while outperforming a model that leveraged pretrained embeddings. We demonstrate the utility of using language specifications from novice users by showing that our approach outperforms or matches relevant baselines without natural language initialization. We hope that this work promotes a greater emphasis on building accessible machine learning systems which can cater to the needs of the diverse sets of users they may interact with.

## X. Acknowledgements

This work was supported by the Office of Naval Research under N00014-19-1-2076, the National Science Foundation under IIS-2112633 and FMRG-2229260, and a gift to the Georgia Tech Research Foundation by Konica Minolta.

## References

[1] L. Chen, R. R. Paleja, and M. C. Gombolay, "Learning from suboptimal demonstration via self-supervised reward regression," in *4th Conference on Robot Learning, CoRL 2020, 16-18 November 2020, Virtual Event / Cambridge, MA, USA*, ser. Proceedings of Machine Learning Research, J. Kober, F. Ramos, and C. J. Tomlin, Eds., vol. 155. PMLR, 2020, pp. 1262–1277. [Online]. Available: https://proceedings.mlr.press/v155/chen21b.html

[2] S. Chernova and A. L. Thomaz, "Robot learning from human teachers," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 8, no. 3, pp. 1–121, 2014.

[3] A. J. Shah, P. Kamath, S. Li, and J. A. Shah, "Bayesian inference of temporal task specifications from demonstrations," 2018.

[4] R. Selten, "Bounded rationality," *Journal of Institutional and Theoretical Economics (JITE)/Zeitschrift für die gesamte Staatswissenschaft*, vol. 146, no. 4, pp. 649–658, 1990.

[5] M. Kaiser, H. Friedrich, and R. Dillmann, "Obtaining good performance from a bad teacher," in *Programming by Demonstration vs. Learning from Examples Workshop at ML*, vol. 95, 1995.

[6] S. Amershi, M. Cakmak, W. B. Knox, and T. Kulesza, "Power to the people: The role of humans in interactive machine learning," *Ai Magazine*, vol. 35, no. 4, pp. 105–120, 2014.

[7] U. Ehsan and M. O. Riedl, "Human-centered explainable ai: Towards a reflective sociotechnical approach," *arXiv preprint arXiv:2002.01092*, 2020.

[8] A. Sciutti, M. Mara, V. Tagliasco, and G. Sandini, "Humanizing human-robot interaction: On the importance of mutual understanding," *IEEE Technology and Society Magazine*, vol. 37, no. 1, pp. 22–29, 2018.

[9] R. T. Icarte, T. Klassen, R. Valenzano, and S. McIlraith, "Using reward machines for high-level task specification and decomposition in reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2107–2116.

[10] P. Nilsson, S. Haesaert, R. Thakker, K. Otsu, C.-I. Vasile, A. Agha, R. Murray, and A. Ames, "Toward specification-guided active mars exploration for cooperative robot teams," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[11] R. Toro Icarte, T. Q. Klassen, R. Valenzano, and S. A. McIlraith, "Teaching multiple tasks to an rl agent using ltl," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2018, pp. 452–461.

[12] N. G. Buchina, P. Sterkenburg, T. Lourens, and E. I. Barakova, "Natural language interface for programming sensory-enabled scenarios for human-robot interaction," in *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2019, pp. 1–8.

[13] R. Liu and X. Zhang, "A review of methodologies for natural-language-facilitated human–robot cooperation," *International Journal of Advanced Robotic Systems*, vol. 16, no. 3, p. 1729881419851402, 2019.

[14] H. A. Napier, R. R. Batsell, N. S. Guadango, and D. M. Lane, "Impact of a restricted natural language interface on ease of learning and productivity," *Communications of the ACM*, vol. 32, no. 10, pp. 1190–1198, 1989.

[15] P. J. Hayes, "The utility of natural language interfaces (panel session)," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1985, p. 19.

[16] P. Smolensky, "Connectionist ai, symbolic ai, and the brain," *Artificial Intelligence Review*, vol. 1, no. 2, pp. 95–109, 1987.

[17] V. Blukis, Y. Terme, E. Niklasson, R. A. Knepper, and Y. Artzi, "Learning to map natural language instructions to physical quadcopter control using simulated flight," *arXiv preprint arXiv:1910.09664*, 2019.

[18] D. Misra, A. Bennett, V. Blukis, E. Niklasson, M. Shatkhin, and Y. Artzi, "Mapping instructions to actions in 3d environments with visual goal prediction," *arXiv preprint arXiv:1809.00786*, 2018.

[19] J. Andreas, D. Klein, and S. Levine, "Modular multitask reinforcement learning with policy sketches," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 166–175.

[20] D. Arumugam, S. Karamcheti, N. Gopalan, L. L. S. Wong, and S. Tellex, "Accurately and efficiently interpreting human-robot instructions of varying granularities," in *Robotics: Science and Systems XIII, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, July 12-16, 2017*, 2017. [Online]. Available: http://www.roboticsproceedings.org/rss13/p56.html

[21] A. Boteanu, J. Arkin, S. Patki, T. Howard, and H. Kress-Gazit, "Robot-initiated specification repair through grounded language interaction," *arXiv preprint arXiv:1710.01417*, 2017.

[22] N. Gopalan, D. Arumugam, L. Wong, and S. Tellex, "Sequence-to-Sequence Language Grounding of Non-Markovian Task Specifications," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[23] A. Suárez and J. F. Lutsko, "Globally optimal fuzzy decision trees for classification and regression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 12, pp. 1297–1311, 1999.

[24] A. Silva, M. Gombolay, T. Killian, I. Jimenez, and S.-H. Son, "Optimization methods for interpretable differentiable decision trees applied to reinforcement learning," ser. Proceedings of Machine Learning Research, S. Chiappa and R. Calandra, Eds., vol. 108. Online: PMLR, 26–28 Aug 2020, pp. 1855–1865. [Online]. Available: http://proceedings.mlr.press/v108/silva20a.html

[25] H. Kress-Gazit and G. E. Fainekos, "Translating structured English to robot controllers," *Advanced Robotics*, vol. 22, pp. 1343–1359, 2008.

[26] C. Matuszek, E. Herbst, L. Zettlemoyer, and D. Fox, "Learning to parse natural language commands to a robot control system," in *Experimental Robotics*. Springer International Publishing, 2013, pp. 403–415.

[27] S. Tellex, T. Kollar, S. Dickerson, M. Walter, A. Banerjee, S. Teller, and N. Roy, "Understanding natural language commands for robotic navigation and mobile manipulation," in *Proceedings of the National Conference on Artificial Intelligence*, 2011.

[28] J. Thomason, A. Padmakumar, J. Sinapov, N. Walker, Y. Jiang, H. Yedidsion, J. Hart, P. Stone, and R. J. Mooney, "Improving grounded natural language understanding through human-robot dialog," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6934–6941.

[29] D. Bahdanau, F. Hill, J. Leike, E. Hughes, A. Hosseini, P. Kohli, and E. Grefenstette, "Learning to understand goal specifications by modelling reward," *arXiv preprint arXiv:1806.01946*, 2018.

[30] V. Blukis, N. Brukhim, A. Bennett, R. A. Knepper, and Y. Artzi, "Following high-level navigation instructions on a simulated quadcopter with imitation learning," in *Proceedings of the Robotics: Science and Systems Conference*, 2018.

[31] C. Lynch and P. Sermanet, "Grounding language in play," *arXiv preprint arXiv:2005.07648*, 2020.

[32] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. B. Amor, "Language-conditioned imitation learning for robot manipulation tasks," *arXiv preprint arXiv:2010.12083*, 2020.

[33] V. Belle and I. Papantonis, "Principles and practice of explainable machine learning," *Frontiers in big Data*, p. 39, 2021.

[34] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.

[35] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

[36] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," *arXiv preprint arXiv:1506.06579*, 2015.

[37] T. Zahavy, N. Ben-Zrihem, and S. Mannor, "Graying the black box: Understanding dqns," in *International Conference on Machine Learning*, 2016, pp. 1899–1908.

[38] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing*, vol. 73, pp. 1–15, 2018.

[39] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[40] C. Burns, J. Thomason, and W. Tansey, "Interpreting black box models via hypothesis testing," in *Proceedings of the 2020 ACM-IMS on Foundations of Data Science Conference*, 2020, pp. 47–57.

[41] N. Frosst and G. Hinton, "Distilling a neural network into a soft decision tree," *arXiv preprint arXiv:1711.09784*, 2017.

[42] Z. Juozapaitis, A. Koul, A. Fern, M. Erwig, and F. Doshi-Velez, "Explainable reinforcement learning via reward decomposition," in *IJCAI/ECAI Workshop on Explainable Artificial Intelligence*, 2019.

[43] L. Sanneman and J. Shah, "Explaining reward functions to humans for better human-robot collaboration," *arXiv preprint arXiv:2110.04192*, 2021.

[44] J. van der Waa, J. van Diggelen, K. v. d. Bosch, and M. Neerincx, "Contrastive explanations for reinforcement learning in terms of expected consequences," *arXiv preprint arXiv:1807.08706*, 2018.

[45] J. Hoffmann and D. Magazzeni, "Explainable ai planning (xaip): overview and the case of contrastive explanation," *Reasoning Web. Explainable Artificial Intelligence*, pp. 277–282, 2019.

[46] O. Amir, F. Doshi-Velez, and D. Sarne, "Summarizing agent strategies," *Autonomous Agents and Multi-Agent Systems*, vol. 33, no. 5, pp. 628–644, 2019.

[47] I. Lage, D. Lifschitz, F. Doshi-Velez, and O. Amir, "Exploring computational user models for agent policy summarization," in *IJCAI: proceedings of the conference*, vol. 28. NIH Public Access, 2019, p. 1401.

[48] K. D. Humbird, J. L. Peterson, and R. G. McClarren, "Deep neural network initialization with decision trees," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 5, pp. 1286–1295, 2018.

[49] A. Silva and M. Gombolay, "Neural-encoding human experts' domain knowledge to warm start reinforcement learning," 2020.

[50] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[51] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[52] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[53] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[54] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[55] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[56] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[57] L. Dong and M. Lapata, "Language to logical form with neural attention," *arXiv preprint arXiv:1601.01280*, 2016.

[58] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[59] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[60] T. G. Dietterich, "Hierarchical reinforcement learning with the maxq value function decomposition," *Journal of artificial intelligence research*, vol. 13, pp. 227–303, 2000.

[61] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 1.

[62] E. Leurent, "An environment for autonomous driving decision-making," https://github.com/eleurent/highway-env, 2018.

[63] Qualtrics, "Qualtrics," https://www.qualtrics.com, 2019.

[64] R. Jia and P. Liang, "Data recombination for neural semantic parsing," *arXiv preprint arXiv:1606.03622*, 2016.

[65] S. Iyer, I. Konstas, A. Cheung, J. Krishnamurthy, and L. Zettlemoyer, "Learning a neural semantic parser from user feedback," *arXiv preprint arXiv:1704.08760*, 2017.

[66] J. Zhu, Y. Xia, L. Wu, D. He, T. Qin, W. Zhou, H. Li, and T.-Y. Liu, "Incorporating bert into neural machine translation," *arXiv preprint arXiv:2002.06823*, 2020.

[67] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[68] Z. C. Lipton, "The mythos of model interpretability," *Queue*, vol. 16, no. 3, pp. 31–57, 2018.