Crowd grounding: finding semantic and behavioral alignment through human robot interaction.

David Matthews, Josh Bongard
University of Vermont
{dmatthe1, jbongard}@uvm.edu

Abstract

Word embeddings have triggered great advances in natural language processing for non-embodied systems such as scene describers. Embeddings may similarly advance natural language understanding in robots, as long as those robots preserve the semantic structure of an embedding corpus in their actions. That is, a robot must act similarly when it hears 'jump' or 'hop' and differently when it hears 'crouch' or 'launch'. This could help a robot learn language because it would immediately obey an unknown word such as 'hop' if it had been trained to obey 'jump'. However, ensuring such alignment between semantic and behavioral structure is currently an open problem. In previous work we showed that the choice of a robot's mechanical structure can facilitate or obstruct a machine learning algorithm's ability to induce semantic and behavioral alignment. That work however required the investigator to create a loss function for each natural language command, including those for which formal definitions are elusive, such as 'be interesting'. A more scalable approach is to bypass loss functions altogether by inviting non-experts to supply their own commands and reward robots that obey them. Here we found that more semantic and behavioral alignment existed among robots reinforced under popular commands than among robots reinforced under less popular commands. This suggests the crowd either chose alignment-inducing commands and/or preferred robots that acted similarly under similar commands. This may pave the way to scalable human-robot interaction by avoiding loss function construction and increasing the probability of zero-shot obedience to previously unheard commands.

Introduction

Directing robots using natural language commands could greatly scale robotics by enabling large numbers of non-experts to safely teach and guide them.

However, how best to enable embodied machines to understand natural language remains an open problem. Recently, automated methods for embedding semantic relationships across natural language into a vector space (Mikolov et al., 2013) have led to great advances in non-embodied natural language processing such as translation (Zou et al., 2013) and image description (Karpathy and Fei-Fei, 2015). This suggests that similar advances may be achieved in robotics. But, despite acknowledgements that embodiment is an important aspect of natural language processing (Bisk et al., 2020) and, before word embeddings were invented, it was shown how language symbols could be grounded in robots' sensorimotor experiences (e.g. Steels (2008)), little effort has yet been exerted to connect word embeddings and robots.

Attempting such a connection raises several challenges. For example, if word embeddings are supplied to a robot, it is not clear that a robot will be able to preserve the semantic structure latent in an embedding space in its behavioral space. That is, there is no guarantee that a robot trained to obey one command will act similarly—and thus likely obey—a semantically-similar command. Here we show that if a crowd of non experts are recruited to collectively train robots, they tend to favor commands that lead to such 'aligned' robots.

Action grounding. The simplest approach for training robots to obey natural language is to ground that language in action (Selfridge and Vannoy, 1986; Matuszek et al., 2013), commonly referred to as action grounding. First, a set of command / obedient action pairs is used to train a model that, if supplied with one of the commands, outputs as closely as possible the corresponding obedient action sequence. A trained model is then deployed such that, if supplied with a novel command, it generates a novel action sequence obedient to that command. This simple method requires the investigator, or an objective function, to generate obedient actions.

Reward grounding. More recently, an alternative method has been proposed which overcomes many of the limitations of action grounding. In reward grounding (Arumugam et al., 2018), a model is trained on command / reward function pairs such that, if a trained model is supplied with a novel natural language command, it generates a novel reward function. If a robot's control policy is trained against this predicted reward function, it is likely to generate actions that obey that command. Although this approach requires a reward function for each training command, inverse reinforcement learning (Hadfield-Menell et al., 2016) can be

employed to learn reward functions from human demonstrations for commands that lack investigator-supplied reward functions.

Crowd grounding. However, there are many potential commands that may elude formal encapsulation in an objective function, such as 'be interesting'. Also, some commands may be difficult for humans to demonstrate. Walking may be demonstrable to a humanoid robot, but demonstrating rolling to a round robot may be difficult, as may 'grasping' to a robot with an electromagnetic manipulator.

Thus, in previous work (Anetsberger and Bongard, 2016) we demonstrated a method we refer to as 'crowd grounding'. Through a website, casual human participants proposed natural language commands for the robots to obey, and they provided positive reinforcement to those robots that exhibit increasing obedience to those commands. We showed that, even in the absence of direct reward, casual human participants provide reinforcement sufficiently accurate to enable simple robots to successfully ground simple natural language commands.

In a related study we (Matthews et al., 2019) found, as did Thomason et al. (2019), that robots can immediately obey embeddings of commands they have never heard before (e.g. 'halt'), if they were previously trained on embeddings of their synonyms (e.g. 'stop'). However, unlike Thomason et al. (2019), in our work we also showed that this ability could be frustrated or enhanced, depending on the choice of the robot's embodiment.

This suggests the following hypothesis: can participants indirectly find robots with body plans that best align semantic and behavioral similarity? We tested this by exposing participants to large numbers of robots whose neural control policies and body plans were altered over time by an interactive evolutionary algorithm. We found that robots reinforced under popular commands tended to have more alignment than those reinforced under less popular commands, providing preliminary evidence supporting this hypothesis.

Methods

An evolutionary algorithm was constructed and run on a local server such that it continuously evaluated virtual robots in a physical simulator. The resulting video was streamed continuously to a channel on Twitch.tv, a streaming and chat service. Participants arriving at the channel could influence the evolution of the robots by typing in chat messages, which were processed by a chatbot (Fig. 1)[1].

Website. From 3:46pm ET on May 29th, 2019 until 12:50 am ET on March 13th, 2020, members of the Twitch.tv community could visit twitch.tv/twitchplaysrobotics and there observe pairs of virtual robots behaving in various simulated environments. Twitch Plays Robotics recruited participants through the Twitch Plays group which is a collection of Twitch streams that consist of crowd sourced game-play. Twitch Plays Robotics also recruited participants through reddit.com/r/artificial. Since Twitch streams are public, it is likely that some participants discovered our experiment through other means.

Participants were shown what command the two robots just 'heard'. They were asked to indicate, in chat, which of the two robots was more obedient to that command by typing in the first letter of the more obedient robot's color.

Commands. At the outset of the experiment, the set of commands C was seeded with two investigator-formulated commands, c_0 ='moving' and c_1 ='being interesting'. The infinitive was employed because commands were inserted into the participant-facing query

''Which is better at *[c_i]?''.

These two commands were chosen because the first is short and motoric and, from previous work Bongard and Anetsberger (2016); Mahoor et al. (2017), was predicted to be groundable in action by the robots. The second subjective command was chosen because it is unclear how to formally specify it in a fitness function.

As the experiment ran, participants could add their own commands to C by typing one or more words preceded by an asterisk. As explained below, robots are supplied with word2vec embeddings of commands. So, if any of the words in a candidate command c_i were not recognized by Google's word2vec pre-trained network, the command was rejected and the proposing participant was informed. Otherwise, c_i was added to C.

For each pair of robots selected for presentation to and evaluation by the participants, a command was chosen randomly from C_t, the set of all unique commands proposed by the crowd since experiment start to the time t of the current evaluation. Selection was biased to favor selection of commands that the crowd preferred. This was accomplished by selecting commands using the probability distribution $\{c_{1t}/c_{Nt}, c_{2t}/c_{Nt}, \ldots, c_{|C_t|t}/c_{Nt}\}$, where c_{it} denotes the number of times that command c_i was proposed in chat from experiment start to time t and $c_{Nt} = \sum_i^{C_t} c_{it}$ denotes the number of command proposals in chat before t.

Reinforcement. At any time, the evolutionary algorithm hosts 10 robots. Each robot is assigned a unique color to ease identification and recognition of particular robots by participants. If any single-character chat response belonged to the set {[r]ed, [g]reen, [b]lue, [y]ellow, [o]range, [p]urple, [w]hite , [j]ade , [c]yan , [s]ilver'} the most recent evaluation in which that robot was retrieved. This method for assigning reinforcement to a robot ensures that, despite differing lag times between when the evaluation was conducted on the

[1]All code available at: https://github.com/davidmatthews1uvm/2020-ALIFE

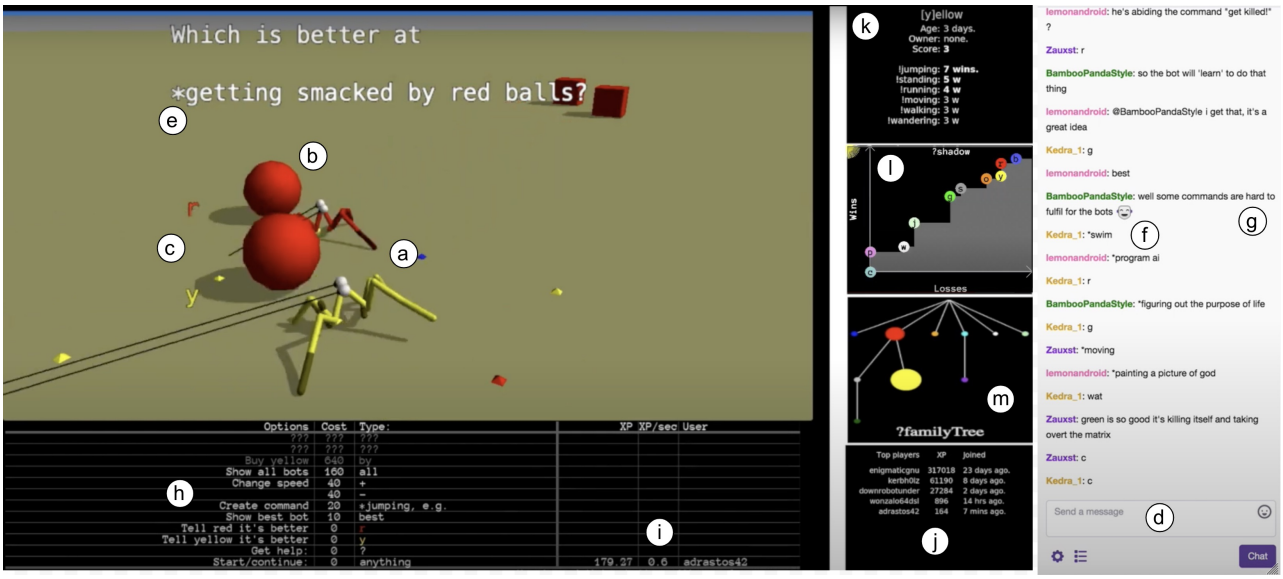

Figure 1: **The participants' view.** Participants were shown pairs of robots (**a**) in different simulated environments (**b**) and asked to type the color of the more obedient robot (**c**) into chat (**d**), given the current command (**e**). Participants could also propose new commands in chat (**f**) and talk to each other (**g**). Participants could 'unlock' new functions (**h**) with earned points (**i**) and see their score relative to others (**j**). Information about one of the two robots is also displayed (**k**), along with two different views of the evolving robot population (**l,m**). Video from the first hour of deployment: **youtu.be/0tHmdHbUbZY**.

local server (t_0) and when participant i observed the evaluation on their device (t_i), the reinforcement signal is assigned to the correct robot. To further minimize the probability of incorrect assignment, three evaluations had to elapse before a robot was eligible for presentation again.

Evaluations always contain two robots: the robot with that color is assumed to have been positively reinforced by the participant, and the other robot is assumed to have been negatively reinforced. Each such chat response is thus processed by incrementing y_i by 1 (the total positive reinforcements of robot i under all commands and all participants) and incrementing n_j by 1 (the total negative reinforcements of robot j). Whenever a robot i is created by the evolutionary algorithm, $y_i = 0$ and $n_i = 0$.

Robot morphology. Each robot is defined as an n-ary tree: each tree segment points to information describing one cylindrical body part and each tree node points to information about c, one degree-of-freedom rotational hinge joints, each of which connects one of the c child segments to its parent segment. Parent/child and sibling cylinder pairs are allowed to interpenetrate, but collisions between other cylinder pairs are detected and resolved.

When a robot is to be evaluated, its tree is duplicated, and all of the x-components describing the positions of the body parts and joints are negated. This creates robots bilaterally symmetric about the x axis (Fig. 1a). Two eyes are placed at the location of the root tree node to help the participant infer that the robot can 'see' (the rays emanating from the eyes) and to establish a local, robot-centric coordinate frame with which to anchor deictic commands such as 'move forward' or 'turn to your left'.

A binary touch sensor is placed in each of a robot's n cylinders, and a proprioceptive sensor is placed on each of the $n-1$ hinge joints. Two, four-component ray sensors emanate from the robot's eyes, each of which returns a maximum distance if the ray does not collide with any objects or the ground plane, and the length of the beam to the nearest object it collides with otherwise. The second, third and fourth components report the red, green, and blue component's of the color of the hit object, and return all zeros if no objects are hit by the ray. Finally, a single auditory sensor neuron is added to each robot. Its function is described below. This yields a total of $n + (n-1) + 9 = 2n + 8$ sensor values that are fed into the neural network controller.

Robot control. The neural controller contains $n-1$ motor neurons which provide desired torques to the $n-1$ hinge joints. Each sensor neuron is connected to each of five hidden neurons, each hidden neuron is recurrently connected to every other hidden neuron as well as itself, and each hidden neuron is connected to each of $n-1$ motor neurons. This yields $5(2n+8) + 5 \times 5 + 5(n-1)$ synapses per robot, whose weights are evolved as described below.

Word embeddings. Just prior to evaluation, the neural networks for each of the two robots about to be simulated are created. The natural language command associated with

that evaluation is translated into its word embedding vector, the first element in the vector is supplied to each of the two networks' auditory neurons, and the hidden neurons in both networks are updated according to that robot's synaptic weights. The second element in the embedding is fed in next, and the hidden neurons in both networks are again updated. This process continues until the entire embedding vector has been digested by both networks. If the command contains additional words, those are also translated into embedding vectors and passed through the two auditory neurons as well. Each of the two robots' body plans are then constructed and deployed to the simulator, along with the control policy and its primed hidden layer. Each robot's resulting behavior is thus a function of the embedding vector it just 'heard' before deployment and the sensor values it experiences during deployment.

The robot controllers used in this paper are only able to retain a small subset of the information contained in the 300 element embeddings. Different weight sets dictate which parts are retained; evolutionary search may, in theory, select for robots that retain the subset of the embeddings that best enable them to elicit positive crowd reinforcement. Without knowing *a priori* what can safely be compressed out, applying traditional dimensional compression algorithms will likely result in loss of useful information and the preservation of unneeded information. For example, city-capital relationships may be preserved even if they are not needed.

Evolutionary algorithm. Following (Sims, 1994; Cheney et al., 2018) (among many others), we constructed an evolutionary algorithm that simultaneously evolves simulated robot body plans along with neural control policies. However, no fitness function was employed: the participants exert selection pressure on the population.

At the outset of the experiment 10 virtual robots with randomly-generated body plans and neural control policies were created, as described above. Every 30 seconds of wall clock time, two of the 10 robots were randomly chosen and shown simultaneously with a randomly-chosen command as explained above. A bi-objective optimization method was employed such that robots were evolved to maximize obedience (y_i, the first objective) and minimize disobedience (n_i, the second objective). Although more sophisticated algorithms exist, a bi-objective algorithm was chosen as a simple method to maintain diversity within the population.

Immediately after an evaluation, any robot j that became dominated by robot i ($y_j > y_i$ and $n_j < n_i$) was deleted. Note that given the asynchronous nature of reinforcement, the most recent evaluation may not have involved robots i or j. If deletion did occur, with 0.5 probability, a new robot with a randomly-generated body plan and neural control policy was created. Otherwise, a randomly-modified copy of robot i was created. In both cases the new robot was assigned the deleted robot's color.

If r_i is the mutated copy of robot r_j, with 0.5 probability r_i's body plan was mutated; otherwise, its control policy was. Morphological mutation was effected by mutating each node in the right subtree of the tree describing the robot's body plan with $1/n$ probability, where n denotes the number of nodes in the right subtree. If node i was mutated, the orientation of its encoded cylinder was altered by re-drawing θ_i and ϕ_i from $[-\pi, \pi]$ using a uniform random distribution.

Control mutation was effected by mutating neurons with probability 0.5, and mutating synapses otherwise. If neurons were targeted for mutation, sensor- (probability $\frac{1}{3}$), hidden- (probability $\frac{1}{3}$) or motor neurons (probability $\frac{1}{3}$) were mutated. For each neuron class, a single neuron in that class was chosen using a uniform distribution. The decay rate for the selected neuron i was randomly changed by re-drawing a random value for τ_i from $[0.1, 0.5]$ using a uniform distribution. Synaptic mutation involved resetting a random synapse to a random value drawn uniformly from $[-1, 1]$.

The environments. When a pair of robots was chosen for evaluation, they were simulated at very different positions in the virtual environment to ensure that they neither collided with one another or 'saw' each other with their ray sensors. However, they were drawn next to one another to aid participant comparison. In addition, each robot could be exposed to immobile or moving objects. At the outset of each simulation, one of the active environments was chosen at random. Initially there were two environments — one containing a stationary cube and the other containing a sphere rolling towards the robots. To incentivize participation, participants could gain points through reinforcing robots and could 'buy' additional environments if they so choose. If an environment was unlocked, it was temporarily added to the list of active environments. A total of 10 environments could be unlocked. Such unlocking occurred very infrequently, so it was unlikely to have had an impact on the results. The role of the environment in the ability of a robot to align semantic and behavioral structure will be investigated in future work.

Results

Deployment. The deployment lasted over 9 months starting on May 29, 2019 and ending on the March 13, 2020. During this time 1,058 different anonymous users visited the Twitch.tv stream and entered something into chat; 827 of them reinforced at least one robot. Over two million robot evaluations were shown of which about 1.2% (28,409) received at least one crowd reinforcement. Collectively the users proposed 641 commands, of which 400 were single word commands. The commands varied widely from motoric commands like 'walk' to abstract commands like 'math', however the most popular commands were all motoric ('walking', 'jumping', 'running', 'clapping', and 'rolling'). Most participants only supplied one reinforcement, while one participant supplied 3963 of them

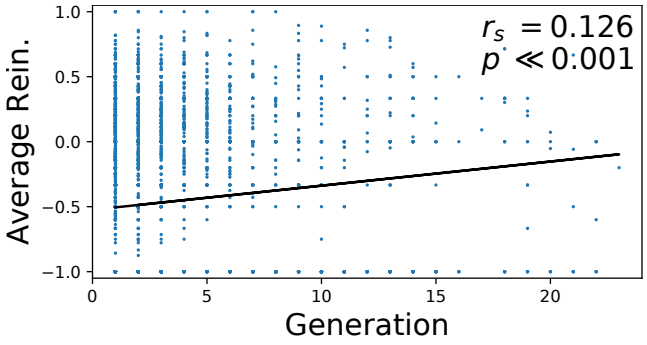

Figure 2: **Evidence of Evolution.** Each point corresponds to a robot. On the Y axis is the average reinforcement that it has received across all of its evaluations. On the X axis is the age of the robot. The Spearman rank correlation coefficient is reported as r_s. There is a very weak positive correlation between the age of a robot and its average reinforcement.

(mean/median reinforcements per user = 38/6). A total of 8345 robots received at least one reinforcement signal.

Evolution. To determine if evolution is occurring and thus better than random search, we ask whether robots with older genotypes attract more positive reinforcement than new robots with random genotypes. Since we employ a steady state evolutionary algorithm here, the age of a robot is measured as the number of generations that its lineage has survived in the population since its founding random member was injected into the population. A newly-injected random robot has age one and children are assigned the age of their parent plus one. We found that there is a slight positive relationship between the age of a robot and the reinforcements that it receives. We can rule out this occurring due to survivorship bias because there is no guarantee that a robot of generation 5, for instance, will attract more positive reinforcements than a robot of generation 1. Average or normalized reinforcement is calculated as: $\frac{y_i - n_i}{y_i + n_i}$, where y_i and n_i are the total number of positive and negative reinforcements that robot i received respectively. Figure 2 displays a scatter plot of the age of robots and their average reinforcements. There is a very weak positive correlation as measured by Spearman's rank correlation coefficient. This weak fit is highly influenced by the fact that many new robots — both random and children — first receive negative reinforcements and quickly die. About half of the data in this scatter plot lie at $y = -1$ and $y = 0$ (receiving only 1 or 2 reinforcements prior to replacement).

To counter the problems caused by unbalanced data, we compare the distribution of reinforcements for old and young robots in Figure 3 through the use of inverse empirical cumulative distribution functions (ECDF). Robots are grouped based on age and an inverse ECDF of each age group is plotted. Almost 60% of robots aged 1 die

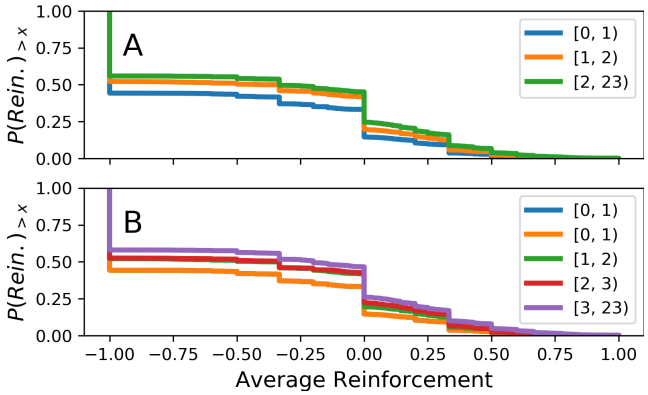

Figure 3: **Evidence of Evolution.** Each line charts the fraction of robots that have an average reinforcement signal greater than x. The robots which first receive negative feedback quickly die. This is why there are steep vertical lines at $x = -1$ and $x = 0$. In **A** and **B**, the population of robots is divided into 3, and 5 groups respectively based on their age (bin edges shown in legend). In both A and B the older robots (e.g. purple, red, green) are above the younger robots (e.g. blue, orange). This means that the older robots are achieving higher reinforcements than the younger ones.

with unanimous negative reinforcements, however for robots aged 3 to 23, only 40% of the robots die with this reinforcement. Here, the older age groups are above the younger age groups (e.g. purple, red, green above orange, blue). This means that older robots are achieving higher average reinforcements overall.

This provides some evidence that the evolutionary algorithm performed better than random search. The lack of clear evolution may be due to the problem of catastrophic interference whereby the robots are being trained on too many different tasks for them to learn them all. If this were occurring, we might expect the robots to specialize and only perform a few tasks well. Figure 4 shows that this is occurring. Each column represents the performance of a given robot at various commands. Most columns only have a small amount of dark green and thus most robots are only good at a few commands. This suggests that they are specializing to a few commands and performing the remaining commands mediocre or poorly.

Semantic and Behavioral Alignment. A robot which successfully aligns semantics with action should behave similarly and differently when issued semantically similar and semantic different commands, respectively. Video youtu.be/rYEbRBzEtBI illustrates, anecdotally, a robot with such an alignment: it exhibits similar movement when issued 'walking' and 'moving', and less movement when issued 'stretching' or 'standing'.

In order to quantify how well a robot aligns semantic meaning with behavior, we need to measure how similar

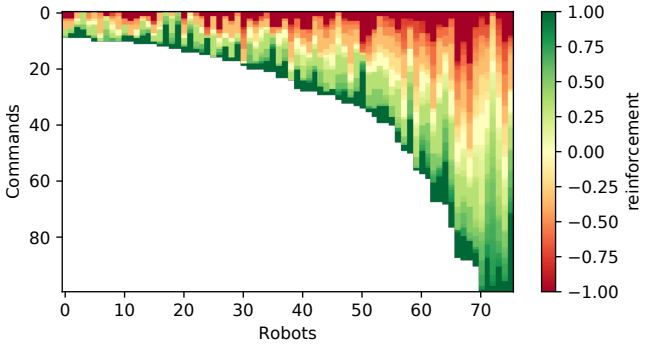

Figure 4: **Robot Specialization.** Each column represents reinforcements that a robot R_i received under qualifying commands $C_a,\ C_b,\ ...,\ C_{a+n}$, sorted from R_i's worst command at the top to its best command at the bottom. Robots are sorted from left to right by the number of qualifying commands that they have. A qualifying command is any command where the number of reinforcements $n_{i,j}$ R_i received under command C_j is at least 3. Robots are excluded if they have less than 10 qualifying commands.

two commands are semantically and how similar the resulting behaviors are. To measure the semantic distance between two commands we employ cosine similarity. Since commands are vectors from a Word2Vec vector space, we can measure the distance between two commands simply by measuring the distance between them. Although we allow users to propose multi word commands, we exclude them from this analysis since it is not clear how to measure the semantic distance between two commands of different length.

$\frac{A \cdot B}{\|\|A\|\|\|\|B\|\|}$	cmd A	rank	cmd B	rank
0.13	swim	43	swimming	84
0.16	dancing	5	dance	51
0.23	play	21	playing	77
0.25	dodge	132	dodging	169
0.26	tearing	401	ripping	438
1.11	speak	286	snagging	371
1.12	crouching	54	launch	411
1.13	handling	120	promenading	467
1.15	sawing	107	launch	411
1.17	disagreeing	191	launch	411

Table 1: The five most and least semantically similar commands (according to word2vec absolute cosine similarity) out of the top 300 most reinforced single-word commands.

Table 1 reports the top 5 most and least similar commands out of the 300 most popular single-word commands. As we would expect, commands with similar meaning have low semantic distance and commands with different meanings have higher semantic distance.

Although measuring semantic distance is fairly simple, behavioral distance is much more difficult to quantify. One way to compute behavioral distance would be to compare sensor time series data generated by a robot when issued two different commands. This data requires subjective choices however, such as which sensors to employ. To avoid this introduction of bias, we instead employ the crowd.

We have shown in previous work that a robot trained on a command will tend to behave correctly when issued a previously unheard synonym of that command (Matthews et al., 2019). This suggests that robots trained on embeddings spontaneously obey behaviorally similar commands. It also suggests a method for using crowd reinforcement to stand in as a proxy for behavioral distance, thus avoiding biased, investigator-defined behavioral distance metrics.

Consider two robots a_1 and b_1 where the crowd indicates that a_1 is better than b_1 at command 'dance'. If for many other robot pairings (a_2, b_2), (a_3, b_3), $(\ldots)$, whenever a_i beats b_i at 'dance', a_i also beats b_i at 'dancing', then one could argue that 'dance' and 'dancing' are behaviorally similar. This rests on the observation that robots are unlikely to be good at many different commands, as reported in Fig. 4. Conversely, if knowing that a beats b at 'crouching' does not predict that a beats b at 'launch', then 'crouching' and 'launch' could be said to be behaviorally distant.

Using this intuition, we formally define estimated behavioral distance as follows: We filter the data set by

$$b_{ij} \quad = \quad 1 - \left(\sum_{i \in C} \sum_{j \in C} \sum_{p_k^{(i)}, p_k^{(j)} \in S_{ij}} \delta_{p_k^{(i)}, p_k^{(j)}} \right) / \|P_{ij}\|$$

where b_{ij} denotes the estimated behavioral distance between robots reinforced under commands i and j; C denotes the set of all commands issued to the robots; $S_{ij} = \{(s^i_{m_1,n_1}, s^j_{m_1,n_1}), \ldots, (s^i_{m_x,n_x}, s^j_{m_x,n_x})\}$ is the set of simulation pairs in which command i was sent to robots m_y and n_y in the first simulation ($s^i_{m_y,n_y}$), and command j was sent to the same two robots in the second simulation ($s^j_{m_y,n_y}$); $p_k^{(i)}$ and $p_k^{(j)}$ denote the preferences drawn from the first and second simulation in the kth pair drawn from S_{ij} respectively; $p_k^{(i)}, p_k^{(j)} \in S_{ij}$ denote all unique preference pairs drawn from S_{ij}; $\|P_{ij}\|$ denotes the number of all such unique preference pairs; $p_k^{(i)}$ is set to one (or minus one) if the user who indicated this preference indicated they thought robot m obeyed command i better (or worse) than robot n, respectively; and $\delta_{p_k^{(i)}, p_k^{(j)}} = 1$ if the same participant (or two different participants) thought robot m obeyed commands i and j better than robot n in the both simulations, or robot m obeyed both commands worse than robot n, and $\delta_{p_k^{(i)}, p_k^{(j)}} = 0$ otherwise. Taken together, $b_{ij} = 1$ if whenever the same two robots were reinforced under commands i and j, there was disagreement as to their relative obedience, and $b_{ij} = 0$ when everyone agreed, for all robot

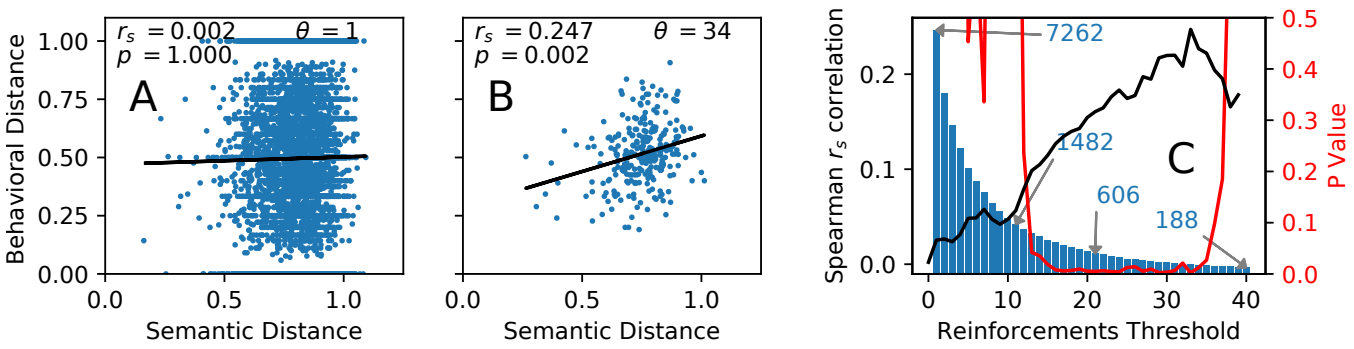

Figure 5: Evidence of alignment between semantic and behavioral distance in crowd-trained robots. **A**: Each point corresponds to a pair of natural language commands provided by the crowd. The semantic distance of a command pair is computed as the cosine distance between their two word embeddings. The behavioral distance (b_{ij}) is estimated as how often users indicated the same two robots acted differently when exposed to a command pair. **B**: The same comparison, but command pairs were restricted to only those that received sufficient data to estimate their behavioral similarities (using θ) **C**: In black from left to right, the Spearman rank correlation coefficient r_s for increasing values of θ. In red, the Bonferroni corrected p values are reported. In blue, the number of command pairs passing the threshold are reported.

pairs exposed to the same two commands, that one obeyed both commands better than the other.

Over all command pairs i and j in C, we found no correlation between semantic distance and behavioral distance (Fig. 5A). This is likely due to the fact that for most command pairs, there is just one $p_k^{(i)}$ and one $p_k^{(j)}$ collected. This likely introduces much noise into the calculation of b_{ij}. To clean the signal, we filtered the data set several more times, each time using a unique value of a threshold parameter θ, where θ dictates that we only consider simulations in which each simulation in S_{ij} received at least θ reinforcements. As we incrementally increased θ from 1 to 40, we found significant positive correlations appeared for a wide range of θ values above 12 (Fig. 5C). The strongest positive correlation appeared at $\theta = 34$ (Fig. 5B).

Because we found in previous work that the physical structure of a robot may help or hinder the ability of a search method to ground natural language in a robot's behavior (Matthews et al., 2019), we investigated here whether semantic and behavioral alignment was uniform across all the robots generated, or whether alignment was more pronounced for some body plans than others. To do so, we calculated alignment separately for the six largest, genetically distinct lineages, where the root of a lineage was a randomly-generated robot (Fig. 6). To do so, we restricted S_{ij} to include only those simulations that simulated pairs of robots from the same lineage being considered. We also used $\theta = 1$. As can be seen, due to how new robots were randomly generated and how morphological mutations within a lineage were applied, robots within a lineage tend to have similar body plans, but body plans tend to differ widely across lineages. We found that only one of the six lineages (clade 5) had significant semantic and behavioral alignment.

Discussion

Taken together, these results suggest that a crowd of non-experts can not only discover robots obedient to natural language commands, but also robots that obey similar commands in similar ways. Further, Fig. 5C suggests that robots are more aligned under more popular commands (those commands issued enough to collect more than θ reinforcement pairs) than they are to the less popular commands. There are several explanations for this apparent relationship between command popularity and alignment, none of which have been confirmed yet. First, it may simply be due to experimental artefacts, such as the fact that there is a lot of noise in the alignment calculation for rarely-reinforced commands. Second, it may be that the participants notice, or unconsciously favor, robots that act similarly when issued certain sets of similar commands, ensuring those commands become popular. Or, participants may instinctually formulate and supply alignment-induced commands: if the same command is typed in more often (perhaps by many different participants), it is send to robot pairs more often, leading to those commands becoming more popular.

Additionally, Fig. 6 suggests that robot alignment is influenced by morphology, since body plans are similar within a clade (robots within insets) and different across clades (robots across insets), and one clade exhibited alignment while five others did not. Intuitively, the link between morphology and alignment could be explained by physical complexity: a simple robot may not be able to exhibit enough diverse behaviors to form a meaningful alignment; conversely, a complex morphology with nonlinearities may exhibit rapid behavioral divergence when issued semantically similar commands, also frustrating alignment.

The apparent relation between morphology and alignment could also be an artifact of experimental design: large clades

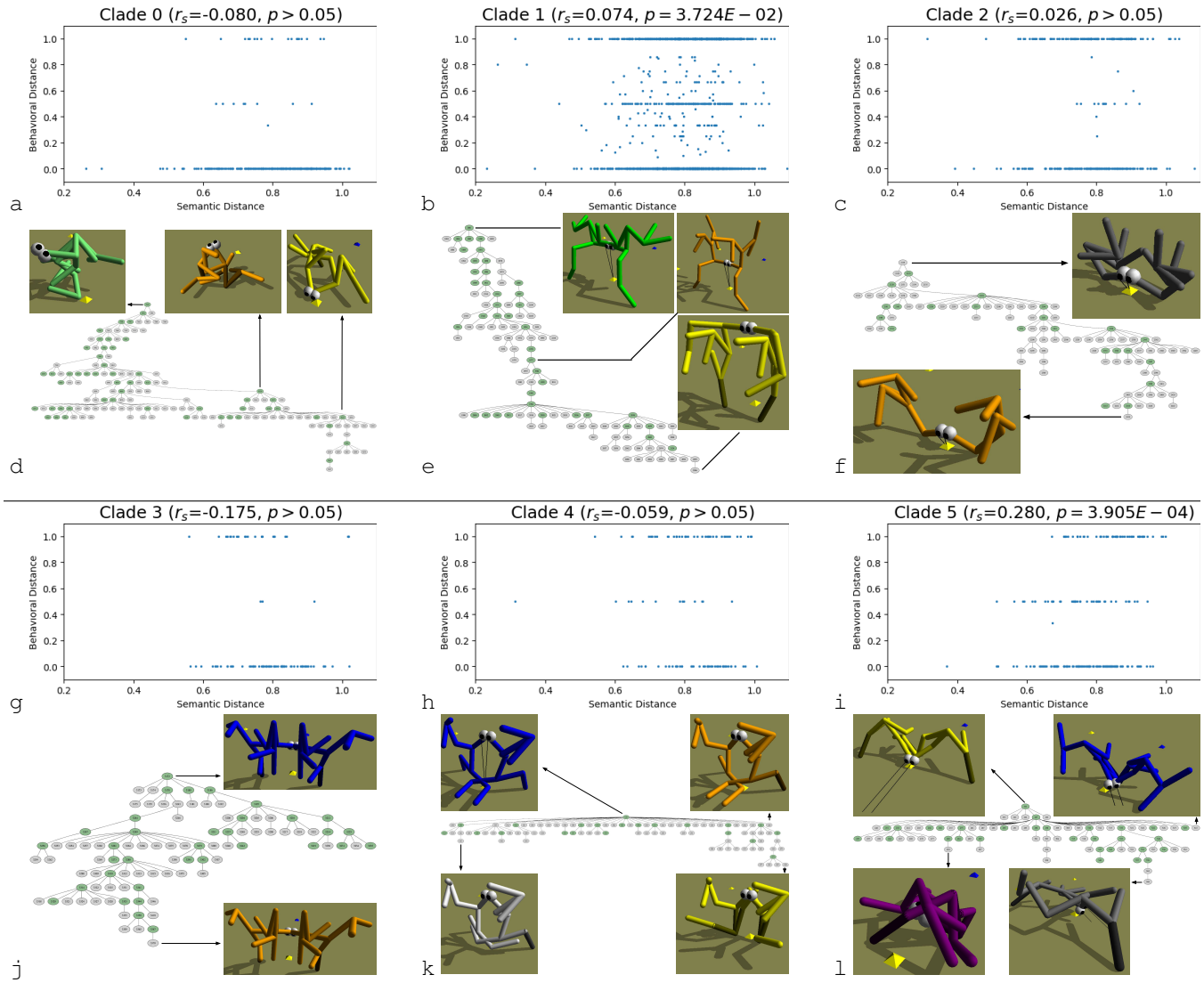

Figure 6: **The six most abundant clades and their semantic/behavioral alignments.** (**d-f,j-l**): The complete lineages of all six clades organized in order of decreasing abundance from (d) to (l), with a few robots drawn from each. Green nodes correspond to robots that collected more positive than negative reinforcements; grey nodes denote those that received more negative than positive reinforcement, or no reinforcement at all. (**a-c,g-i**): The potential semantic / behavioral alignment within each clade. Each point represents the semantic and behavioral distances for a pair of commands issued to robots in that clade. The Spearman rank correlation coefficient was calculated for each clade, and reported as r_s. The Bonferroni-corrected p values for each coefficient are also reported.

tended not to form in the population at the same time. This temporal separation means that different clades get reinforced by different users, and a clade early on in the experiment may be exposed to a different set of commands than one later on in the experiment. Further, different users may be more or less honest, causing other confounding effects.

In future work we will seek such explanations, as well as reduce the breadth of commands sent to the robots to minimize catastrophic forgetting. This, and other algorithmic changes, are expected to increase the likelihood of enabling the crowd to evolve more generally obedient robots. In the long term, we believe such democratization of robotics may yield autonomous machines that resist catastrophic forgetting through semantic and behavioral alignment, resist perverse instantiation by forgoing objective functions, and resist bias by having been evolved by diverse human trainers.

Acknowledgements

This work was supported by NSF awards EFRI-1830870 and OAC-1827314, and DARPA contract HR0011-18-2-0022.

References

Anetsberger, J. and Bongard, J. (2016). Robots can ground crowd-proposed symbols by forming theories of group mind. *Proceedings of the Artificial Life Conference*, pages 684–692.

Arumugam, D., Karamcheti, S., Gopalan, N., Wong, L. L., and Tellex, S. (2018). Accurately and efficiently interpreting human-robot instructions of varying granularities. In *Robotics: Science and Systems*.

Bisk, Y., Holtzman, A., Thomason, J., Andreas, J., Bengio, Y., Chai, J., Lapata, M., Lazaridou, A., May, J., Nisnevich, A., Pinto, N., and Turian, J. (2020). Experience grounds language. *arXiv preprint arXiv:2004.10151*.

Bongard, J. and Anetsberger, J. (2016). Robots can ground crowd-proposed symbols by forming theories of group mind. In *Proceedings of the Artificial Life Conference 2016 13*, pages 684–691. MIT Press.

Cheney, N., Bongard, J., SunSpiral, V., and Lipson, H. (2018). Scalable co-optimization of morphology and control in embodied machines. *Journal of The Royal Society Interface*, 15(143):20170937.

Hadfield-Menell, D., Russell, S. J., Abbeel, P., and Dragan, A. (2016). Cooperative inverse reinforcement learning. In *Advances in neural information processing systems*, pages 3909–3917.

Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

Mahoor, Z., Felag, J., and Bongard, J. (2017). Morphology dictates a robot's ability to ground crowd-proposed language. *arXiv preprint arXiv:1712.05881*.

Matthews, D., Kriegman, S., Cappelle, C., and Bongard, J. (2019). Word2vec to behavior: morphology facilitates the grounding of language in machines. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Matuszek, C., Herbst, E., Zettlemoyer, L., and Fox, D. (2013). Learning to parse natural language commands to a robot control system. In *Experimental Robotics*, pages 403–415. Springer.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Selfridge, M. and Vannoy, W. (1986). A natural language interface to a robot assembly system. *IEEE Journal on Robotics and Automation*, 2(3):167–171.

Sims, K. (1994). Evolving 3D morphology and behaviour by competition. *Artificial Life IV*, pages 28–39.

Steels, L. (2008). The symbol grounding problem has been solved. So what's next? In *Symbols and embodiment: Debates on meaning and cognition*, pages 223–244. Oxford University Press USA.

Thomason, J., Padmakumar, A., Sinapov, J., Walker, N., Jiang, Y., Yedidsion, H., Hart, J., Stone, P., and Mooney, R. J. (2019). Improving grounded natural language understanding through human-robot dialog. *arXiv preprint arXiv:1903.00122*.

Zou, W. Y., Socher, R., Cer, D., and Manning, C. D. (2013). Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398.