Active Learning of Reward Dynamics from Hierarchical Queries

Chandrayee Basu[1], Erdem Bıyık[2], Zhixun He[1], Mukesh Singhal[1], Dorsa Sadigh[2,3]

***Abstract*— Enabling robots to act according to human preferences across diverse environments is a crucial task, extensively studied by both roboticists and machine learning researchers. To achieve it, human preferences are often encoded by a reward function which the robot optimizes for. This reward function is generally static in the sense that it does not vary with time or the interactions. Unfortunately, such static reward functions do not always adequately capture human preferences, especially, in non-stationary environments: Human preferences change in response to the emergent behaviors of the other agents in the environment. In this work, we propose learning *reward dynamics* that can adapt in non-stationary environments with several interacting agents. We define reward dynamics as a tuple of *reward functions*, one for each mode of interaction, and *mode-utility functions* governing transitions between the modes. Reward dynamics thereby encodes not only different human preferences but also how the preferences change. Our contribution is in the way we adapt preference-based learning into a *hierarchical* approach that aims at learning not only reward functions but also how they evolve based on interactions. We derive a probabilistic observation model of how people will respond to the hierarchical queries. Our algorithm leverages this model to actively select hierarchical queries that will maximize the volume removed from a continuous hypothesis space of reward dynamics. We empirically demonstrate reward dynamics can match human preferences accurately.**

I. Introduction

One of the most important challenges in robotics is to enable robots to act and perform tasks in a way that matches with human preferences. Since specifying all possible scenarios is very impractical for most of the real-world systems, machine learning techniques are widely employed. Two common approaches are to learn a reward function that implicitly gives a policy for the robot, or to learn directly the policy itself. While the latter approach has been successfully applied for several tasks [1]–[4], the former one is often preferable as it outputs an interpretable reward function.

Inverse reinforcement learning (IRL) is one such method that is widely employed for learning reward functions using expert demonstrations [5]–[8]. However, acquiring expert demonstrations might be hard when the system of interest has high degrees of freedom [9], or when the demonstrations significantly differ from actual human preferences [10].

Preference-based learning has emerged as a successful alternative to IRL for learning reward functions for robotic systems that uses other forms of human guidance, such as pairwise comparison queries, when demonstrations are not available [11]–[19]. Further studies combined IRL with preference-based learning [20].

Like IRL, preference-based learning assumes a static reward function, which is not expressive enough to match human preferences in all environments. Real world is often non-stationary due to environment complexity or changes in objectives in the environment. Surrounding agents continuously change their behavior which in turn requires the robot to adapt to these changes. For example, in driving people continuously adapt their reward functions in response to traffic complexity and behavior of other drivers. It is quite common for us to get impatient behind a slow driver and make drastic maneuvers different from our usual driving style. Here we may weigh efficiency more than collision avoidance than we usually do.

As an important class of non-stationary environments, human-robot and robot-robot adaptation have recently attracted much attention, where the aim is to ensure robots adapt to their changing environments and other agents [21], [22]. In contrast, our goal is to learn the reward functions that dynamically change depending on the interactions between the agents and the environment. We augment preference-based learning to recover such multi-modal reward functions.

Prior works have also theoretically investigated how to perform preference-based learning for multi-modal reward functions [23]–[25]. Specifically in [25], it was shown that pairwise comparisons can be used to learn only unimodal reward functions, or as we call, static reward functions. In this paper, by modeling and learning the transitions between the modes, we relax the problem and empirically show that we are able to learn bimodal rewards.

In our work, in addition to learning the reward function for each mode, we are also interested in how they change in non-stationary environments. Modeling behaviors in such environments is a well-studied problem especially for driving. For example, [26] characterizes driving styles based on sensor data using deep learning. In a more related paper [27], the authors modeled the drivers with a latent state space which can affect their driving behavior. While they stated these latent states might change over time, both of these works made the assumption that latent states remain unchanged over the trajectories of interest, so they did not address changing behaviors. In [28], the authors modeled the latent states of the drivers using Hidden Markov Models (HMM) where they also allow adaptation. However, they did not specifically learn reward functions, and they focused on identifying the maneuvers the drivers will perform from

[1] UC Merced, School of Engineering, Electrical Engineering and Computer Science. 5200 North Lake Road, Merced, CA 95343.

[2] Stanford University, School of Engineering, Electrical Engineering. 350 Serra Mall, Stanford, CA 94305.

[3] Stanford University, School of Engineering, Computer Science. 353 Serra Mall, Stanford, CA 94305.

Emails: {cbasu2,zhe5,msinghal}@ucmerced.edu, ebiyik@stanford.edu, dorsa@cs.stanford.edu

a predefined database. With a similar objective, [29] used HMM for latent state estimation for human-robot interaction.

In this paper, we propose to learn an expressive representation of preferences in non-stationary scenarios, where interactions and adaptations better reflect the real-world conditions. We assume that the non-stationary scenarios arise from changing behaviors of other agents interacting with our system, which in turn affect human preferences. We formalize reward dynamics which encodes not only different human preferences but also *how* they change.

> *Our insight is that reward dynamics matches human preferences more accurately in a wide range of scenarios than a static reward function.*

We actively select comparison queries from a database, similar to [12], [17], to learn a probability distribution over *reward dynamics*: a mixture of static reward functions representing different moods and a set of parameters for the transitions between the moods. We tailor comparison queries to capture longer term interactions between the robot and the surrounding agents, and develop a probabilistic model of user responses for any number of static reward functions and the transitions between them.

In this paper, we make the following contributions:

Reward Dynamics: User preferences may change based on the behaviors of other agents in the environment. We encode the momentary human preference by a static reward function and assume at any point of time the human has an internal *preference mode* (mood) which dictates what reward function the human will optimize next. We introduce the notion of *reward dynamics* as a tuple of reward functions and parameters governing transitions between those.

Hierarchical Queries: We formalize *hierarchical queries* as a sequence of pairwise comparison questions, each of which we call a *sub-query*. The sub-queries sequentially follow each other so that the user moods are reflected into their choices.

Active Query Selection: We provide an algorithm that actively selects informative hierarchical queries in order to efficiently learn reward dynamics through interactions with the users. We evaluate our algorithm on an autonomous driving example. We show in simulations that we can efficiently learn changes in preferences when preferences indeed vary based on interactions with different environments.

II. Problem Statement

Let us denote the agent in the environment of interest (e.g. driving scenarios) that should match human preferences as $\mathcal{H}$ (ego car), and the other agents in the environment $\mathcal{E} = \{\mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_P\}$. These agents can act differently at different times. For example, in case of driving, some cars aggressively swerve through the traffic and others may follow a more cooperative strategy allowing other cars to merge smoothly. Prior works on autonomous driving [30]–[37] assume $\mathcal{H}$ should follow the same reward function over time in both of the above scenarios. We argue user preferences may vary in response to the changing behaviors of the environment agents in both driving and potentially other multi-agent environments. Our goal is to learn an expressive reward function corresponding to these dynamic preferences.

We model the environment as a fully-observable dynamical system. For driving, the continuous state of the system $x \in X$ includes the positions and the velocities of $\mathcal{H}$ and $\mathcal{E}$. The state of the system changes based on actions of all the agents through a function f.

$$x^{t+1} = f(x^t, u_H^t, u_E^t) \tag{1}$$

where u_E are the actions of $\mathcal{E}$, which affect the reward function and in turn the actions u_H of $\mathcal{H}$. We define a finite trajectory $\xi \in \Xi$ as a sequence of continuous state-action pairs $\xi = (x^0, u_H^0, u_E^0, \ldots, x^T, u_H^T, u_E^T)$ over a finite horizon T, and Ξ is the set of all feasible trajectories that satisfy the dynamics of the system.

Our goal is to learn human preferences for how $\mathcal{H}$ should behave in the presence of different environment agents $\mathcal{E}$. We learn this reward function by making hierarchical comparison queries to the users.

III. Hierarchical Comparison Queries

Prior works have learned static reward functions by asking people to compare between two different trajectories of robots. There, each query is a pair of short videos that demonstrate two trajectories of the system [11], [12], [16]. Such short trajectories do not capture the nuances of interaction in a non-stationary multi-agent system. As an example, in a 1-step comparison query in Fig. 1(a), an environment agent $\mathcal{E}$ (white car) aggressively merged in front of $\mathcal{H}$ (orange car). One option is for our user to slow down (optimize a cooperative reward function). This sudden slow down may have frustrated the user, causing a mode change. So in a similar situation later (in another query) the user prefers a trajectory optimal with respect to a competitive reward function and prevents $\mathcal{E}$ from merging in front. This change in preference manifests as noise in 1-step preference-based learning approaches (see the Fig. 1(b)). However, we would like to learn a composite reward function that not only captures both of these preferences but also *how* they have changed in such non-stationary environment.

To do so, we allow the users to change their preferences within the same query. We present each query q as a sequence of several sub-queries. Each sub-query q_i in the sequence is a continuation from the final state of the trajectory of the previous sub-query q_{i-1}. This allows us to learn how the behavior of other interacting agents in one sub-query affects user preference in the next sub-query. We assume that the users' next immediate *preference mode* depends only on their current experience. We, therefore, reset their preference mode at the beginning of each query with a demonstration, which we denote as q_0. After q_0, each sub-query is a pairwise comparison $q^A, q^B \in \Xi$. q_1^A and q_1^B are both continuations of q_0. In general, for the rest of the sub-queries, q_i^A and q_i^B are continuations from $q_{i-1}^{a_{i-1}}$ where a_{i-1} is the answer for the $(i-1)^{\text{th}}$ sub-query, as shown in Fig. 1(c).

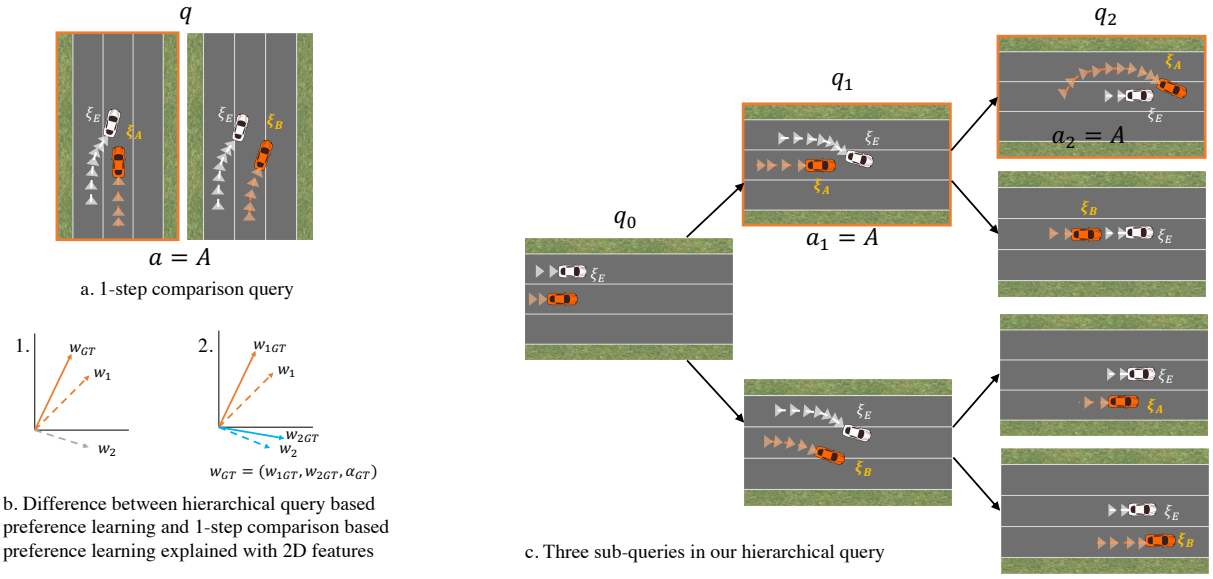

Fig. 1. a. 1-step comparison query. In any two iterations a user with bimodal preference may pick the trajectories optimal with respect to two different true weights w_{1GT} and w_{2GT} (GT stands for ground truth). b. This ambiguity shows up as noise in 1-step comparison based learning where the goal is to learn a single reward function w_{GT} (on the left). In reality the true preference function of the user w_{GT} changes between w_{1GT} and w_{2GT} depending on the environment, α_{GT} governs the transition. Our algorithm learns such a bimodal preference: w_1 close to w_{1GT} and a w_2 close to w_{2GT} (on the right). c. Our proposed hierarchical query consists of 3 sub-queries: q_0 is a context sub-query, q_1 is a comparison between two trajectories, each a different continuation of q_0 and q_2 continues the preferred trajectory from q_1.

IV. Reward Dynamics Model

A. Preliminaries

Throughout the paper, we will use $[n]$ to denote the integer set $\{1, 2, \ldots, n\}$ for $n \in \mathbb{Z}_{>0}$.

We denote the ith sub-query as q_i for $i \in [s] \cup \{0\}$, where the number of sub-queries within one query is s.

We assume there is a finite set of modes M and we enumerate each mode such that $M = \{1, 2, \ldots, k\}$ where k is the total number of modes. M_j is the jth element of the set of modes M. We also assume the mode of the user is stable during each time period, the duration of a sub-query. We denote the mode in the ith sub-query as $m_i \in M$.

Each q_i, except q_0, consists of two trajectories $q_i^A, q_i^B \in \Xi$. The user selects A or B as his/her response to each of these sub-queries. The user's response to q_i is $a_i \in \{A, B\}$. $\bar{a}_i$ denotes the complement of a_i, i.e. $\{a_i\} \cup \{\bar{a}_i\} = \{A, B\}$.

In addition, we assume a features function $\phi : \Xi \to \mathbb{R}^d$ that maps every trajectory to a d-dimensional feature space. This function depends on both $\mathcal{H}$ and $\mathcal{E}$. We assume the d features of the environment $\mathcal{F}$ are known. For example, some representative features for driving are distance to the closest environment car, distance to the road boundaries, the speed and the heading angle of the ego vehicle.

B. Human Preference Model

Reward functions under known modes. We define a user-specific reward function parameterized by the mode of the user, for example, two different reward functions representing calm and rushed driving: $R_{M_j} : \Xi \to \mathbb{R}$ for $j \in [k]$. With linearity assumption as in [11], [12], it is defined as: $R_{M_j}(\xi) = w_{M_j}^\top \phi(\xi)$ where $\xi \in \Xi$ and $w \in \mathbb{R}^{d \times k}$ is a user-specific weight matrix, and w_j is the jth column of w, with each column corresponding to a particular mode for a user. Then, the user response to a sub-query q_i is probabilistic based on Luce's Choice Axiom [38], [39] which is a widely used human decision model in cognitive science as it nicely captures the uncertainty in humans' choices:

$$P(a_i | q_i, m_i, w) = \frac{\exp(R_{m_i}(q_i^{a_i}))}{\exp(R_{m_i}(q_i^{a_i})) + \exp(R_{m_i}(q_i^{\bar{a}_i}))} \quad (2)$$

This models probability of the human making a choice given a sub-query, the humans' mode during that sub-query, and the user-specific preferences.

Prior on mode transitions. We also learn how people change modes. For example, how likely is a person to transition from aggressive driving to defensive driving or vice versa. We assume that a prior $G \in \mathbb{R}^{k \times k}$ over the mode transitions is given by the designer. The matrix G alone represents the natural propensity to transition between different modes and is independent of the sub-queries and the current state of the learning algorithm. For example, some mode transitions are naturally more likely than the others: If we have three modes that correspond to defensive, neutral and offensive moods, then it would be more likely for a defensive user to switch to the neutral mode than to the offensive mode. G captures this prior. Hence, it is constrained to be a proper Markov chain matrix. We note that Markov chains are employed similarly for mood changes by psychiatrists, e.g. [40]. We explain the formulation of the mode transitions next.

Mode transition model. The users change their mode based on what they experienced in the previous sub-query and their previous mode. We define a *Mode-Utility* function to capture this effect of the sub-queries. Specifically, we model the mode transitions as follows: The user has an underlying

mode-utility function that quantifies the previous trajectories. If the user thinks she would have higher utility with mode M_j, then she transitions to M_j. As an example, imagine you are driving in a very calm mood. If someone suddenly cuts in front of you, you would think "if I were aggressive, I could keep a shorter headway with the car in front and the other car would not have been able to cut in front of me", and you also switch to an aggressive mood. It is of course also possible that you keep calm. Hence, the transition should be stochastic.

We model the mode-utility as a function of trajectories: $U_{M_j} : \Xi \to \mathbb{R}$ for $j \in [k]$. Again with linearity assumption, it is defined as: $U_{M_j}(\xi) = \beta_{M_j}^\top \phi(\xi)$ where $\beta \in \mathbb{R}^{d\times k}$ is another user-specific weight matrix and β_j is the jth column of β.

The probability of transitioning from any mode M_{j_0} in sub-query q_{i-1} to any mode M_{j_1} in the next sub-query is given by multiplying the prior G with the likelihood computed using the mode-utility function:

$$\begin{aligned} &P_{j_0 j_1}(q_{i-1}, a_{i-1}, \beta) \\ &\quad := P(m_i = M_{j_1} | m_{i-1} = M_{j_0}, q_{i-1}, a_{i-1}, \beta) \\ &\quad = \frac{1}{Z} \frac{\exp(U_{M_{j_1}}(q_{i-1}^{a_{i-1}}))}{\sum_{m\in M} \exp(U_m(q_{i-1}^{a_{i-1}}))} G_{j_0 j_1} \end{aligned} \tag{3}$$

where Z is the normalization constant. In a completely "neutral case", when the likelihood (softmax) gives equal values for each mode, the transition is solely defined by the prior G. Some examples of G are:

- $G_{j_0 j_1} = 1/k$ for $\forall (j_0, j_1) \in [k]^2$ means that the user may change from any mode to any other mode just based on the previous sub-query with a uniform prior. This is suitable when the modes are categorical, not sequential.
- $G = I$ means the user will not ever change her mode and will remain in her initial mode. Note that the initial mode will also be modeled in a probabilistic way.
- If G is a band matrix, then the user can only change between the modes that are "close". This is suitable for sequential modes.

While our learning model is valid for any feasible G, we will do simplifying assumptions to actively select the hierarchical queries for sample-efficient learning.

Definition IV.1. *Reward dynamics* of a user is a tuple of (w, β), which governs both the user preferences and how they transition with the interactions the user is involved in.

Therefore, our aim is to learn the *reward dynamics* rather than a static reward function.

Initial State. We do not know the initial mode m_0 of the user, which is the active mode during q_0. One simple way is to assume uniform distribution over all modes. However, imagine G is such that transitioning to M_j is very unlikely from any mode. Then, the uniform assumption will not hold, because the user is unlikely to be in mode M_j. Then a better model is the following:

$$P_j := P(m_0 = M_j) = \pi_{M_j}(G) \tag{4}$$

where π_{M_j} denotes the probability of mode M_j in the stationary distribution of the Markov chain G. If there exist several stationary distributions, the designer should pick one of them using domain knowledge.

C. Learning Reward Dynamics

To make the learning of reward dynamics effective and efficient, we should restrict the continuous space of reward dynamics. For that, we make assumptions on the norms of the columns of w and β similar to [11], [12].

There is also the problem of *label switching*. That is, all the probabilities will remain the same if we switch the order of modes both in w and β. Since this can completely disable the learning, we enforce another constraint on the ordering, as mentioned by [23], such that $\beta_{M_1,1} > \beta_{M_2,1} > \cdots > \beta_{M_k,1}$ where $\beta_{M_j,1}$ is the first element of M_j^{th} column of β.

Our goal is to learn a distribution over the *reward dynamics* by making informative queries. We start with a uniform prior over the space of all feasible (w, β). After receiving all the answers to a query q, $(a_1, a_2, \ldots, a_s)$, we perform a Bayesian update:

$$\begin{aligned} &p(w, \beta | a_s, a_{s-1}, \ldots, a_1, q_s, q_{s-1}, \ldots, q_0) \\ &\quad \propto p(a_s, a_{s-1}, \ldots, a_1 | w, \beta, q_s, q_{s-1}, \ldots, q_0) p(w, \beta) \end{aligned} \tag{5}$$

Next we derive the expression for the update function $p(a_s, a_{s-1}, \ldots, a_1 | w, \beta, q_s, q_{s-1}, \ldots, q_0)$ and present some simplifications that we adopted for our implementation.

D. Derivation and Simplifications

In this section, we present how we compute the update function for $p(w, \beta)$. We note q_0 does not receive any response. For the simplicity of notation, we let $q_0^{a_0}$ be the associated trajectory in q_0, so that $P_{j_0 j_1}(q_0, a_0, \beta)$ is well-defined for $\forall (j_0, j_1) \in [k]^2$. We then derive

$$\begin{aligned} &P(a_s, a_{s-1}, \ldots, a_1 | w, \beta, q_s, q_{s-1}, q_0) \\ &= \sum_{(j_0,\ldots,j_s)\in[k]^{s+1}} P_{j_0} P_{j_0 j_1}(q_0, a_0, \beta) \ldots P_{j_{s-1} j_s}(q_{s-1}, a_{s-1}, \beta) \\ &\prod_{l\in[s]} P(a_l | w, q_l, m_l = M_{j_l}) \end{aligned} \tag{6}$$

In our implementation, we restrict ourselves to the cases where $s = 2$. Then, the above equation is simplified as

$$\begin{aligned} &P(a_2, a_1 | w, \beta, q_2, q_1, q_0) \\ &\quad = \sum_{j_0\in[k]} \sum_{j_1\in[k]} \sum_{j_2\in[k]} P_{j_0} P_{j_0 j_1}(q_0, a_0, \beta) P_{j_1 j_2}(q_1, a_1, \beta) \\ &\qquad P(a_1 | w, q_1, m_1 = M_{j_1}) P(a_2 | w, q_2, m_2 = M_{j_2}) \end{aligned} \tag{7}$$

To eliminate the normalization Z from the equation, we assume $G_{j_0 j_1} \in \{0, 1/c_{j_0}\}$ for $\forall (j_0, j_1) \in [k]^2$ where c_{j_0} is an appropriate constant. That is, we assume the model designer will just decide on whether or not it is possible to

move between any two modes and will not assign specific prior probabilities. Then,

$$P_{j_0 j_1}(q_0,a_0,\beta)=\frac{\exp(\beta_{M_{j_1}}^\top \phi(q_0^{a_0}))}{\sum_{j'\in[k]:G_{j_0 j'}=1/c_{j_0}} \exp(\beta_{M_{j'}}^\top \phi(q_0^{a_0}))} \quad (8)$$

If we further assume $k = 2$ and $G_{j_0 j_1} = 1/2$ for $\forall(j_0, j_1) \in [k]^2$, such as the case of cooperative and competitive modes, we also have $P_{j_0} = \frac{1}{2}$, so we can write:

$$P(a_2, a_1|w, \beta, q_2, q_1, q_0) = \sum_{(j_1,j_2)\in\{1,2\}^2} \prod_{i\in\{1,2\}} \frac{\exp(w_{M_{j_i}}^\top \phi(q_i^{a_i}))}{\exp(w_{M_{j_i}}^\top \phi(q_i^{a_i})) + \exp(w_{M_{j_i}}^\top \phi(q_i^{\bar{a}_i}))} \frac{\exp(\beta_{M_{j_i}}^\top \phi(q_{i-1}^{a_{i-1}}))}{\exp(\beta_{M_1}^\top \phi(q_{i-1}^{a_{i-1}})) + \exp(\beta_{M_2}^\top \phi(q_{i-1}^{a_{i-1}}))} \quad (9)$$

This formulation enables us to update $p(w, \beta)$. We are now ready to present our active query selection algorithm that improves data-efficiency.

V. Active Query Selection

In this section, $\mathcal{D}$ denotes all the information about w and β up to the current iteration of interest —we dropped the subscript for simplicity.

Each answer tuple (a_1, a_2) removes some volume from the hypothesis space of (w, β), where, volume removed is given as the difference between the unnormalized posterior distribution over (w, β), and its prior distribution. We have a belief over what the user answers could be. We leverage this probabilistic model to *actively* select a query at each iteration that will maximize the expected volume removal. Formally, we solve the following optimization:

$$(q_0^*, q_1^*, q_2^*) = \arg\max_{q_0,q_1,q_2} \mathbb{E}_{a_1,a_2}\left[\mathbb{E}_{w,\beta}\left[1-p(a_2,a_1|w,\beta,q_2,q_1,q_0)\right]\right] \quad (10)$$

where both expectations are taken given $\mathcal{D}$, q_0, q_1 and q_2. To compute the inner expectation, we sample (w, β) from $p(w, \beta|\mathcal{D})$ using Markov Chain Monte Carlo methods. Unlike previous works in active reward learning [11], [12], our update function is not log-concave. We, therefore, resort to Metropolis-Hastings algorithm. Now, let's say we take M samples. We let $\bar{w}$ and $\bar{\beta}$ represent those samples:

$$(q_0^*, q_1^*, q_2^*) \approxeq \arg\min_{q_0,q_1,q_2} \mathbb{E}_{a_1,a_2}\left[\frac{1}{M}\sum_{\bar{w},\bar{\beta}} p(a_2, a_1|\bar{w}, \bar{\beta}, q_2, q_1, q_0)\right]$$

Formally writing $\mathbb{E}_{a_1,a_2}$, the minimization objective is

$$\sum_{(a_1,a_2)} p(a_2, a_1|q_2, q_1, q_0, \mathcal{D})\frac{1}{M}\sum_{\bar{w},\bar{\beta}} p(a_2, a_1|\bar{w}, \bar{\beta}, q_2, q_1, q_0)$$

where the first sum is over $\{A, B\}^2$. By the law of large numbers, we also have:

$$p(a_2, a_1|q_2, q_1, q_0,\mathcal{D}) = \lim_{M\to\infty}\frac{1}{M}\sum_{\bar{w},\bar{\beta}} p(a_2, a_1|\bar{w}, \bar{\beta}, q_2, q_1, q_0)$$

as $\bar{w}$ and $\bar{\beta}$ are drawn from $p(w, \beta|\mathcal{D})$, independent from q_2, q_1, q_0; and (a_1, a_2) are conditionally independent from $\mathcal{D}$. Then, for large M, we can write the optimization as:

$$\arg\min_{q_0,q_1,q_2} \sum_{(a_1,a_2)\in\{A,B\}^2} \left(\sum_{\bar{w},\bar{\beta}} p(a_2, a_1|\bar{w}, \bar{\beta}, q_2, q_1, q_0)\right)^2 \quad (11)$$

where the probability expression in the objective function is already derived in Section IV.

VI. Simulation Experiments

A. Problem Domain

We focus on learning driving preferences. Each component of the learned *reward dynamics* weighs 5 features for driving that attempt to encode safety and traffic rules: one feature for penalizing closeness to the edges of the road, one for velocity, and three more features of proximity to lane centers, to other cars and alignment with the road, similar to [11][1]. Each sub-query consists of the driving environment and a pair of trajectories of $\mathcal{E}$ and $\mathcal{H}$ whose preferred behavior we are learning. The environment is represented by the trajectory of an environment car and the initial states of $\mathcal{H}$. Our query database consists of 10000 randomly generated hierarchical comparison queries.

B. Dependent Measures

In our implementation we learned $\alpha_1 := \beta_1 - \beta_2$, instead of β, as it has fewer parameters[2]. The same approach generalizes to any k with $\alpha_j := \beta_j - \beta_k$ for $\forall j \in [k-1]$.

We measure the performance of hierarchical preference learning in terms of expected dot product between learned weights and true weight as in [11], [12], separately for each component of (w, α):

$$r = \mathbb{E}\left[\frac{\hat{v}.v^*}{\|\hat{v}\|_2\|v^*\|_2}\right] \quad (12)$$

where $v \in \{w_1, w_2, \alpha_1\}$, $\hat{v}$ and v^* are the estimated and true weights, respectively, and the expectation is taken over the sampled $\hat{v}$ values. Hence, r is a measure of convergence, as its value being close to 1 indicates learned weights are close to the true weights.

C. Experiments with Random Data

We first conduct experiments with completely random and independent sub-queries without the driving environment. We assume we can generate queries in an unconstrained way such that any ϕ-vector is possible, i.e. there is no dynamics constraint in the generation of queries. Here, we simulate *oracle users*: users who are perfectly aware of their true reward dynamics. That is, they always behave (change mode

[1]While traffic rules are not explicitly modeled in our simulation, many of the features can be weighed appropriately to encode them. For example, the feature for velocity measures the deviation from the maximum allowed speed which can be easily adapted to the road type.

[2]Note we cannot do the same trick for w, because while β in Eq. (9) can be simplified to α, there is no such simplification on w.

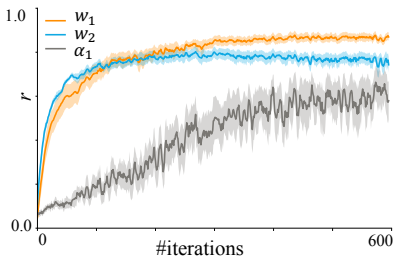

Fig. 2. r value shows that our algorithm converges well for non-driving data with non-active query selection when the simulated user is *oracle*. Here we show an average r over 5 different ground truth *reward dynamics*.

and respond) with respect to the higher probability out of softmax models. In Fig. 2, the average results of 5 different simulated oracle users show convergence of (w_1, w_2, α_1) whose true values were independently drawn from standard normal distribution independently for each entry.

D. Experiments with Driving Data

Active versus non-active query selection. We compare the performance of our active query selection algorithm with a non-active baseline where we uniformly sample the queries from a discrete database of 10000 queries. Here our simulated users are always oracle. We test the following hypothesis: **H1**. *The reward dynamics learned with our active query selection algorithm converges to the true weights faster compared to the non-active baseline.* Our results in Fig. 3 supports this hypothesis by demonstrating that active query selection accelerated the learning of one of the modes (w_2) compared to the non-active baseline.

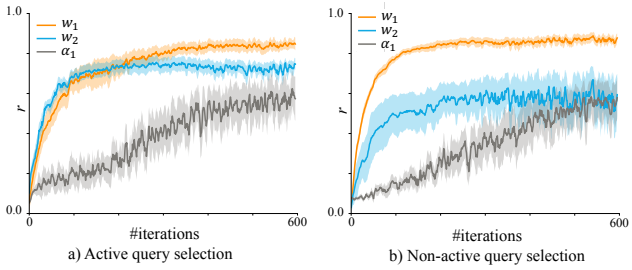

Fig. 3. r values show that our algorithm with active query selection from dataset of 10000 discrete queries (left) can learn reward dynamics faster than non-active query selection (right) when the simulated user is *oracle*. Here we show an average r over 5 different ground truth reward dynamics.

Testing different mode preferences. Next we simulate 5 noisy users, who choose between options A and B with respect to $p(a|w, \beta, q)$. Our algorithm actively selects queries from the same discrete dataset of size 10000 as in the case of oracle users. We first set the following hypothesis: **H2**. *Our algorithm learns the reward dynamics even when the users are noisy.*

We also test the performance of our algorithm for different mode likelihoods, i.e. probability of transitioning to a given mode, $p(m = M_j)$. We manipulated the ground truth reward dynamics to reflect different mode likelihoods. For example, one user might be in one mode 80% of the time while another user has equal chances of being in one of the two modes. Although this might actually affect the priors P_1 and P_2 as we explained in Section IV, we still adopted the derivations based on uniform prior to test the robustness of our framework. Therefore, we test the following hypothesis: **H3**. *Our algorithm learns the weights w that correspond to both modes, and it converges faster for w_{M_j} if $p(m = M_j)$ is higher.*

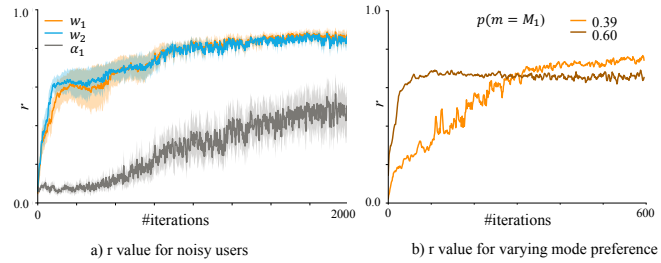

Fig. 4. r value shows even when the users are noisy our algorithm can learn the true *reward dynamics* (left) and that as $p(m = M_1)$ increases, w_1 converges faster (right).

Fig. 4(a) shows that our algorithm was able to learn w_1, w_2 and α_1 even when the users are noisy. We note that in general we learn α_1 slowly and we need more queries for its convergence. The second plot shows that we are able to learn the reward weights w_j of a mode M_j regardless of its likelihood probability being high or low. The same plot also shows the algorithm converges faster for the modes that are visited more often. This is intuitive, as the algorithm is able to gather more information about those modes, even though it does not perfectly know that the user is in the corresponding mood. Hence, **H3** has strong empirical support.

VII. User Study

A. Hypotheses

We test the following hypotheses with the user study: **H1**. *Our algorithm learns weights that can represent the driving behavior of the users.* **H2**. *Some people indeed change preferences depending on the driving behaviors of the interacting agents.*

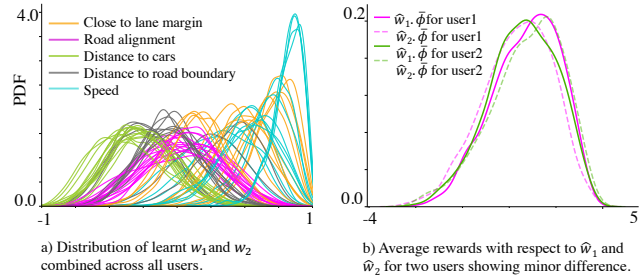

Fig. 5. Distribution of $\hat{w}_1$ and $\hat{w}_2$ across all users for individual features. (a) user preferences vary widely for adherence to lane center and distance to road boundaries, but are very similar for efficiency (speed) and safe driving (collision avoidance). (b) While we did not learn significantly different w_1 and w_2 for individual users, the average reward w.r.t to $\hat{w}_1$ and $\hat{w}_2$ differ slightly for some of our study participants.

B. Study Design

To validate our hypotheses, we collected data from 10 real users in a within-subjects study. We first learn a general reward dynamics $(\hat{w}, \hat{\alpha})$ using 50 hierarchical queries. We then use the posterior distributions over these parameters to jump-start the process for each subject with a reasonable prior that better represents legally correct driving. During

validation, we ask users to provide ratings for trajectories locally optimized with respect to the learned reward dynamics. We compare the expressiveness of the learned weights $(\hat{w}_1, \hat{w}_2)$ against their perturbed versions (w_1^p, w_2^p). We sampled these perturbed versions from Gaussian distributions centered around $(\hat{w}_1, \hat{w}_2)$ and a standard deviation of $0.5\times|\hat{w}_1|$ and $0.5\times|\hat{w}_2|$. While creating perturbed versions of $\hat{w}_1$ and $\hat{w}_2$, as an attempt to ensure legally correct driving, we constrain the weight components for the features that correspond to staying within the road and avoiding collision with cars. We also compare with w_{r} sampled from a Gaussian distribution centered on either $\hat{w}_1$ or $\hat{w}_2$ with a standard deviation of $2 \times |\hat{w}_j|$ with the corresponding mode index j. Each rating question consists of two parts. The first part is similar to q_0 of the learning step, where we show user one trajectory demonstration of $\mathcal{H}$ as an attempt to set their initial mode. In the next part, we show users 5 trajectories continued from the first part, optimal with respect to 5 reward functions: $\hat{w}_1$, $\hat{w}_2$, their perturbed versions w_1^p and w_2^p, and w_{r}. For each of the 5 trajectories, we ask users a 7-point Likert scale rating question: *Indicate your level of agreement with the following statement: I would like to ride this car*.

In **H1** we claim 1) users will give the highest overall rating to the trajectories generated from $\hat{w}_1$ and/or $\hat{w}_2$ most of the time, and 2) if $p(M_j)$ is very high, we expect people to give the highest rating to trajectories generated from corresponding weight $\hat{w}_{M_j}$. To validate the first part, we repeat the same demonstration across several rating queries preserving $\mathcal{E}$'s trajectory alike and changing the trajectory of $\mathcal{H}$, varying between different local optimal with respect to w_1, w_2 and the other weights. We randomize demonstration trajectories across the rating questions. In **H2** we hypothesize that subject to different interactions in the environment, users will sometimes give higher rating to trajectory optimal for $\hat{w}_1$ and sometimes to those optimal for $\hat{w}_2$.

C. Results

Like previous work in this area [11], [12], we found that the users have somewhat similar preferences: proximity to cars has high negative weight and speed has high positive weight showing that people generally prefer safe and efficient driving (see Fig. 5). On the other hand, features that encode staying on the road and alignment with the road vary more. While the general direction of the feature weights is similar between $\hat{w}_1$ and $\hat{w}_2$ for each user, there is some difference in the magnitudes. We computed the percentage difference between average reward with respect to $\hat{w}_1$ and $\hat{w}_2$ as $\frac{\hat{w}_1.\bar{\phi}-\hat{w}_2.\bar{\phi}}{\hat{w}_1.\bar{\phi}}$, where $\bar{\phi}$ is the average feature values for our application. This gave us the percentage difference in the average reward. We found that of all the users the maximum difference is 12% and the minimum difference is 6%. While we also learned α, it becomes relatively unimportant here, as w_1 and w_2 are very close.

As it can be seen in Fig. 6, the users gave the highest scores to the trajectories generated from $\hat{w}_1$ and $\hat{w}_2$ with statistical significance. This suggests an empirical evidence for **H1**. While we also observed that users sometimes gave

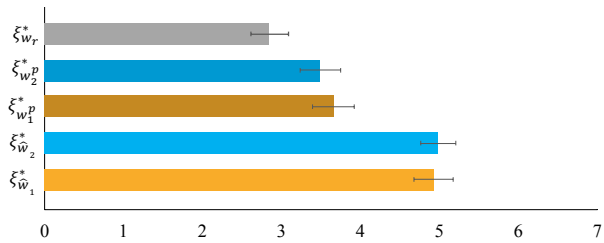

Fig. 6. Most users gave high ratings to the trajectories optimal for $\hat{w}_1$ and $\hat{w}_2$ and low ratings to trajectories optimal for their perturbed versions w_1^p and w_2^p and the lowest rating to the trajectories that were optimal with respect to some random weight w_r.

high ratings to $\hat{w}_1$ and sometimes to $\hat{w}_2$, we have not observed a significant dependence on the modes. This is due to the fact that the learned weights were very close to each other as they represent the legal driving behavior, which is a very small subset of all the reward space. Further, our simulation environment may not be realistic enough to elicit emotions like anger, frustration etc. that cause behavioral changes in different traffic situations [41]–[43]

VIII. Discussion

Summary. We developed a model of how humans change their moods based on the interactions with the environment, as well as how they respond to the choice queries when they are in a particular mood. Using this model, we developed a volume removal-based active learning algorithm to efficiently learn the reward functions and the mode transitions. We demonstrated through simulations and user studies that this framework can efficiently learn dynamic preferences.

Limitations. While the framework is very general, we tested it only for driving environment. In fact, because legal driving is a very small subset of the space of static reward functions, we observed in our experiments that queries focus mostly on learning this small subset, and we would need higher number of queries to recover different modes within this set. More extensive experiments on various environments where all reward functions are sound but the personalization is more important could give more interesting results.

We also used a specific set of parameters $k = 2$, $s = 2$. Other values require further research for theoretical identifiability guarantees with the relaxed problem of multimodal reward learning we presented.

Future Directions. Besides research on identifiability and task environments, we are also investigating how we can incorporate into our framework other methods for latent state inference and improve the expressivity of reward dynamics.

Acknowledgments

This work is partly supported by FLI grant RFP2-000. Toyota Research Institute (TRI) provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity.

References

[1] S. Ross, N. Melik-Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert, "Learning monocular reactive uav control in cluttered natural environments," in *2013 IEEE international conference on robotics and automation*. IEEE, 2013, pp. 1765–1772.

[2] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 4565–4573.

[3] B. C. Stadie, P. Abbeel, and I. Sutskever, "Third-person imitation learning," in *International Conference on Learning Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=B16dGcqlx

[4] J. Song, H. Ren, D. Sadigh, and S. Ermon, "Multi-agent generative adversarial imitation learning," in *Advances in Neural Information Processing Systems*, 2018, pp. 7472–7483.

[5] 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.

[6] A. Y. Ng, S. J. Russell, *et al.*, "Algorithms for inverse reinforcement learning." in *Icml*, vol. 1, 2000, p. 2.

[7] P. Abbeel and A. Y. Ng, "Exploration and apprenticeship learning in reinforcement learning," in *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005, pp. 1–8.

[8] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning." in *Aaai*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[9] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz, "Keyframe-based learning from demonstration," *International Journal of Social Robotics*, vol. 4, no. 4, pp. 343–355, 2012.

[10] C. Basu, Q. Yang, D. Hungerman, M. Sinahal, and A. D. Draqan, "Do you want your autonomous car to drive like you?" in *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 2017, pp. 417–425.

[11] D. Sadigh, A. D. Dragan, S. Sastry, and S. A. Seshia, "Active preference-based learning of reward functions," in *Robotics: Science and Systems (RSS)*, 2017.

[12] E. Bıyık and D. Sadigh, "Batch active preference-based learning of reward functions," in *Conference on Robot Learning*, 2018, pp. 519–528.

[13] E. Bıyık, K. Wang, N. Anari, and D. Sadigh, "Batch active learning using determinantal point processes," *arXiv preprint arXiv:1906.07975*, 2019.

[14] E. Biyik, D. A. Lazar, D. Sadigh, and R. Pedarsani, "The green choice: Learning and influencing human decisions on shared roads," in *Proceedings of the 58th IEEE, Conference on Decision and Control (CDC)*, December 2019.

[15] R. Akrour, M. Schoenauer, and M. Sebag, "April: Active preference learning-based reinforcement learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2012, pp. 116–131.

[16] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," in *Advances in Neural Information Processing Systems*, 2017, pp. 4299–4307.

[17] C. Basu, M. Singhal, and A. D. Dragan, "Learning from richer human guidance: Augmenting comparison-based learning with feature queries," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 2018, pp. 132–140.

[18] D. Braziunas, "Computational approaches to preference elicitation," *Department of Computer Science, University of Toronto, Tech. Rep*, p. 62, 2006.

[19] J. Fürnkranz, E. Hüllermeier, W. Cheng, and S.-H. Park, "Preference-based reinforcement learning: a formal framework and a policy iteration algorithm," *Machine learning*, vol. 89, no. 1-2, pp. 123–156, 2012.

[20] M. Palan, N. C. Landolfi, G. Shevchuk, and D. Sadigh, "Learning reward functions by integrating human demonstrations and preferences," in *Proceedings of Robotics: Science and Systems (RSS)*, June 2019.

[21] S. Nikolaidis, D. Hsu, and S. Srinivasa, "Human-robot mutual adaptation in collaborative tasks: Models and experiments," *The International Journal of Robotics Research*, vol. 36, no. 5-7, pp. 618–634, 2017.

[22] M. Al-Shedivat, T. Bansal, Y. Burda, I. Sutskever, I. Mordatch, and P. Abbeel, "Continuous adaptation via meta-learning in nonstationary and competitive environments," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=Sk2u1g-0-

[23] Z. Zhao, P. Piech, and L. Xia, "Learning mixtures of plackett-luce models," in *International Conference on Machine Learning*, 2016, pp. 2906–2914.

[24] A. Liu and A. Moitra, "Efficiently learning mixtures of mallows models," in *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2018, pp. 627–638.

[25] F. Chierichetti, R. Kumar, and A. Tomkins, "Learning a mixture of two multinomial logits," in *International Conference on Machine Learning*, 2018, pp. 960–968.

[26] W. Dong, J. Li, R. Yao, C. Li, T. Yuan, and L. Wang, "Characterizing driving styles with deep learning," *arXiv preprint arXiv:1607.03611*, 2016.

[27] J. Morton and M. J. Kochenderfer, "Simultaneous policy learning and latent state inference for imitating driver behavior," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–6.

[28] H. Berndt, J. Emmert, and K. Dietmayer, "Continuous driver intention recognition with hidden markov models," in *2008 11th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2008, pp. 1189–1194.

[29] D. Kulic and E. A. Croft, "Affective state estimation for human–robot interaction," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 991–1000, 2007.

[30] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions." in *Robotics: Science and Systems*, vol. 2. Ann Arbor, MI, USA, 2016.

[31] D. Sadigh, S. S. Sastry, S. A. Seshia, and A. Dragan, "Information gathering actions over human internal state," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 66–73.

[32] D. Sadigh, N. Landolfi, S. S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for cars that coordinate with people: leveraging effects on human actions for planning and active information gathering over human internal state," *Autonomous Robots*, vol. 42, no. 7, pp. 1405–1426, 2018.

[33] D. Sadigh, S. S. Sastry, and S. A. Seshia, "Verifying robustness of human-aware autonomous cars," *IFAC-PapersOnLine*, vol. 51, no. 34, pp. 131–138, 2019.

[34] H. Bai, S. Cai, N. Ye, D. Hsu, and W. S. Lee, "Intention-aware online pomdp planning for autonomous driving in a crowd," in *2015 ieee international conference on robotics and automation (icra)*. IEEE, 2015, pp. 454–460.

[35] V. Sezer, T. Bandyopadhyay, D. Rus, E. Frazzoli, and D. Hsu, "Towards autonomous navigation of unsignalized intersections under uncertainty of human driver intent," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 3578–3585.

[36] J. F. Fisac, E. Bronstein, E. Stefansson, D. Sadigh, S. S. Sastry, and A. D. Dragan, "Hierarchical game-theoretic planning for autonomous vehicles," in *International Conference on Robotics and Automation (ICRA)*, May 2019.

[37] E. Stefansson, J. Fisac, D. Sadigh, S. Sastry, and K. H. Johansson, "Human-robot interaction for truck platooning using hierarchical dynamic games," in *European Control Conference (ECC)*, June 2019.

[38] R. D. Luce, *Individual choice behavior: A theoretical analysis*. Courier Corporation, 2012.

[39] R. Holladay, S. Javdani, A. Dragan, and S. Srinivasa, "Active comparison based learning incorporating user uncertainty and noise," in *RSS Workshop on Model Learning for Human-Robot Communication*, 2016.

[40] E. Holmes, M. Bonsall, S. Hales, H. Mitchell, F. Renner, S. Blackwell, P. Watson, G. Goodwin, and M. Di Simplicio, "Applications of time-series analysis to mood fluctuations in bipolar disorder to promote treatment innovation: a case series," *Translational Psychiatry*, vol. 6, no. 1, p. e720, 2016.

[41] D. Shinar, "Aggressive driving: the contribution of the drivers and the situation," *Transportation Research Part F: traffic psychology and behaviour*, vol. 1, no. 2, pp. 137–160, 1998.

[42] T. Zhang, A. H. Chan, and W. Zhang, "Dimensions of driving anger and their relationships with aberrant driving," *Accident Analysis & Prevention*, vol. 81, pp. 124–133, 2015.

[43] A. Lotz, K. Ihme, A. Charnoz, P. Maroudis, I. Dmitriev, and A. Wendemuth, "Recognizing behavioral factors while driving: A real-world multimodal corpus to monitor the drivers affective state," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, 2018.