Reinforcement Learning for Fleet Applications using Coregionalized Gaussian Processes

Timothy Verstraeten
Vrije Universiteit Brussel
tiverstr@vub.be

Ann Nowé
Vrije Universiteit Brussel
anowe@vub.be

ABSTRACT

The general interest in the *Internet of Things* allows control devices and/or machines to connect through cloud-based architectures in order to share information about their status and environment. In many settings, as for example wind farms, similar machines are instantiated to perform the same task, which is called a *fleet*. Exploiting such a formation is especially useful in control settings. Specifically, seamless data sharing between fleet members could greatly improve the sample-efficiency of reinforcement learning techniques. However, in practice these devices, while similar, have small discrepancies due to production errors or degradation, preventing devices to simply aggregate and employ all fleet data. We propose a novel reinforcement learning method that learns to transfer knowledge between similar fleet members, and creates member-specific dynamic models for control. To this end, our algorithm uses Gaussian processes to establish cross-member covariances. We demonstrate our approach on the continuous mountain car setting as a preliminary experiment. Our method significantly outperforms two baseline approaches, namely individual learning and joint learning where all fleet samples are used.

KEYWORDS

reinforcement learning; fleet learning; multi-task Gaussian processes

1 INTRODUCTION

In Reinforcement Learning (RL) [10], the objective of an agent is to optimize its control decisions based on rewards given by the environment it is operating in. One of the greatest challenges in RL is the efficient usage of the gathered experience. State-of-the-art techniques typically require a large amount of experience, which renders the technique ill-suited for real industrial applications.

In this work, our objective is to improve sample efficiency in the context of a *fleet*, i.e., a set of similar devices instantiated to perform the same task (e.g., wind farms and industrial robot arms). To this end, we present a method that aggregates the experience of the different fleet members, in contrast to the experience of a single device. The time is right for such a method, as the general interest in the *Internet of Things* allows fleet members to share data from modern wireless sensors through a cloud-based architecture, providing rapidly a complete view of the problem.

As each fleet member carries out the same task, they are typically designed in the same way. In reality, fleet members can differ slightly in terms of dynamics, for example due to production errors or degradation. Thus, naively aggregating data over all members is insufficient and could prove to be detrimental for the learning process. Therefore, information should only be shared between fleet members that are sufficiently similar.

We propose a novel method for fleet control, where we allow knowledge transfer between similar devices on a dynamics-level without compromising the specificity of an individual's model. More specifically, we create a Bayesian RL method that uses Gaussian processes (GP) to model the fleet members' transition functions and aims to establish correlations between them by using coregionalization.

GPs are Bayesian models known to successfully model complex non-linear surfaces using only a limited amount of data. They have been used before in RL [2, 3, 8] and are popular when high sample-efficiency is required. Coregionalization was originally introduced in geostatistics to generate valid covariance matrices for modeling multivariate data sets [4]. It has later been used in multi-task learning to describe correlations between a set of tasks in the context of GPs [1].

The setting we consider is closely related to Bayesian multi-task RL. [5, 11] propose a hierarchical Bayesian model, where the transitions, rewards and/or values are modeled as samples from a common distribution per task group. The objective is then to cluster the tasks, such that similar ones draw their parameters from the same prior distribution. Thus, shared information is encoded in a single prior per class of tasks. Our method does not use an hierarchical architecture with class priors, but instead focusses on direct pair-wise correlations between members, allowing for more focussed knowledge transfer.

In this preliminary work, we develop a novel reinforcement learning method based on coregionalized Gaussian processes. We start by giving background on RL (Section 2) and GPs (Section 3). We describe the Bayesian fleet transition model and how each member can access its specific predictive statistics (Section 4). Next, we test our method in the continuous mountain car setting and discuss the results (Section 5). Finally, we conclude by discussing future work (Section 6).

2 REINFORCEMENT LEARNING

Consider the Markov decision process (MDP) $\mathcal{M} = (S, A, \mathcal{T}, \gamma, R)$ [7], where S, A are *continuous* state and action spaces, $\mathcal{T}(\mathbf{s}, a)$ is a probability distribution for entering state $\mathbf{s}'$ when executing action a in state $\mathbf{s}$, $R : S \times A \times S \to \mathbb{R}$ is the immediate reward function, and $\gamma \in [0, 1]$ is the discount factor determining the importance of future rewards. Additionally, consider a policy $\pi : S \to A$, which defines the control decision mechanism of an agent.

The long-term reward when following a policy π is defined as a value function V^π. For the given continuous MDP, the value function can be written recursively as

$$V^\pi(\mathbf{s}) = \mathbb{E}_{\mathbf{s}' \sim \mathcal{T}(\mathbf{s}, \pi(\mathbf{s}))} \left[R(\mathbf{s}, \pi(\mathbf{s}), \mathbf{s}') + \gamma V^\pi(\mathbf{s}') \right] \tag{1}$$

which is essentially the sum of all possible future rewards weighted by their probability of occurrence. The goal of an agent is then to find the optimal policy $\pi^* : S \rightarrow A$ which maximizes this expression.

Generally, the value function cannot be analytically computed for continuous state spaces due to the integral in the expected value. However, if we model $\mathcal{T}(\cdot,\cdot)$ as a GP and $R(\cdot,\cdot,\cdot)$ as a Gaussianly shaped reward around a desired goal state, it can be shown that V^π becomes a GP as well [8].[1] We refer the reader to [8] for the analytical expressions of the value function, as well as the policy iteration method for finding the optimal policy π^*. Throughout the paper, we assume a known reward function around a specified goal state, and focus on learning the GP transition model.

3 GAUSSIAN PROCESS TRANSITION MODEL

As explained above, the transition model of an RL agent is defined as a probabilistic distribution $\mathcal{T}(\mathbf{s},\mathbf{a})$ over possible next states, conditioned on the current state-action pair $(\mathbf{s},\mathbf{a})$. For simplicity, we assume that the output state features are independently distributed, thus, $s'_e \sim \mathcal{T}_e(\mathbf{s},\mathbf{a})$ for each state feature e.

Gaussian processes (GPs) [9] are an extension of multivariate normal distributions. Similar to the latter, a GP describes a set of normally distributed random variables that are potentially correlated, meaning that knowledge about one variable gives information about another. However, the difference is that a GP is defined over *arbitrary* sets of *annotated* random variables. In a regression context, these random variables are the outputs of an unknown function, and their annotations are the inputs to that function.

Formally, assuming a zero-mean GP prior and any arbitrary set of inputs X, we can model the associated outputs $\mathbf{y}$ as follows:

$$\mathbf{y} \mid X \sim \mathcal{N}(\mathbf{0}, K) \tag{2}$$

where $K_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$ is the covariance between variables y_i and y_j. When regressing over a training set $(X_{\text{tr}}, \mathbf{y}_{\text{tr}})$, we can compute the posterior statistics of $(\mathbf{y} \mid X, X_{\text{tr}}, \mathbf{y}_{\text{tr}})$ to obtain the predictive outputs $\mathbf{y}$ for inputs X. For the zero-mean GP described in Equation 2, we have

$$\begin{aligned}\mathbb{E}[\mathbf{y} \mid X, X_{\text{tr}}, \mathbf{y}_{\text{tr}}] &= K_{X,X_{\text{tr}}} K^{-1}_{X_{\text{tr}},X_{\text{tr}}} \mathbf{y} \\ \mathbb{V}[\mathbf{y} \mid X, X_{\text{tr}}, \mathbf{y}_{\text{tr}}] &= K_{X,X} + K_{X,X_{\text{tr}}} K^{-1}_{X_{\text{tr}},X_{\text{tr}}} K_{X_{\text{tr}},X}\end{aligned} \tag{3}$$

where $K_{X,X_{tr}}$ contains the pair-wise covariances between sets X and X_{tr} according to the covariance kernel.

The choice of covariance kernel $k(\cdot,\cdot)$ is important, as it defines various characteristics about how the model generalizes from the training set. A popular choice for the covariance kernel is the squared exponential (SE) kernel, defined as:

$$k^{SE}_\theta(\mathbf{x},\mathbf{x}') = \exp\left(-\sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{2l_d^2}\right) \tag{4}$$

where θ contains the hyperparameters l_d, and l_d denotes the length scale along dimension d, which characterizes the smoothness of the unknown function. This kernel has several properties, including continuity, differentiability and stationarity, rendering it a popular choice for generic modeling purposes. We optimize the hyperparameters θ using evidence maximization on the training set [9].

Considering a state transition $(\mathbf{s},\mathbf{a},\mathbf{s}')$, we can approximate the transition model of the RL agent using multiple GPs, where we define the inputs and output respectively as $\mathbf{x} = [\mathbf{s},\mathbf{a}] \in \mathbb{R}^D$ and $y = s'_e \in \mathbb{R}$. Thus, we have one GP for each feature in the output state description.

4 COREGIONALIZATION OVER MULTIPLE TRANSITION MODELS

Fleet members should only share information when they are correlated. Thus for a given state-action, the outputs of the transition models of two members are either close, or there exists a linear transformation between them. Intuitively, the GP's covariance kernel allows for generalization in the regression model by correlating unobserved outputs to observed ones. Coregionalization extends this concept to the outputs of different GPs, suggesting that information from one process can be generalized to another. The main contribution in this work is to use coregionalization to capture similarities between multiple transition models, in order to decide whether knowledge transfer should occur.

Formally, we define a fleet MDP $\mathcal{M}_F = (S, A, T^F_\theta, \gamma, R)$. Comparing to the definition in Section 2, all properties are the same, except that $T^F_\theta = \{\mathcal{T}^{(m)}_\theta\}_{m=1}^M$ is now a set of transition models, one for each fleet member m. Note that we assume the same hyperparameters θ for each model, and thus, the models only differ from one another when regressing on the member-specific training sets.

In order to achieve knowledge transfer between models, we construct a fleet-wide model $\mathcal{T}^F_\theta$. We introduce a covariance kernel that uses a matrix G, describing the correlations between members m and m' [1]:

$$k^F_\theta([\mathbf{x},m],[\mathbf{x}',m']) = k^{SE}_{\theta_{SE}}(\mathbf{x},\mathbf{x}') G_{m,m'} \tag{5}$$

where entry $G_{m,m'}$ denotes the covariance between the output states of members m and m'. In order to make sure G is a valid covariance matrix, we use the decomposition [4]

$$G = \mathbf{w}\mathbf{w}^T + \boldsymbol{\alpha} I, \text{ with } \mathbf{w} \in \mathbb{R}^M, \boldsymbol{\alpha} \in \mathbb{R}^M_{\geq 0} \tag{6}$$

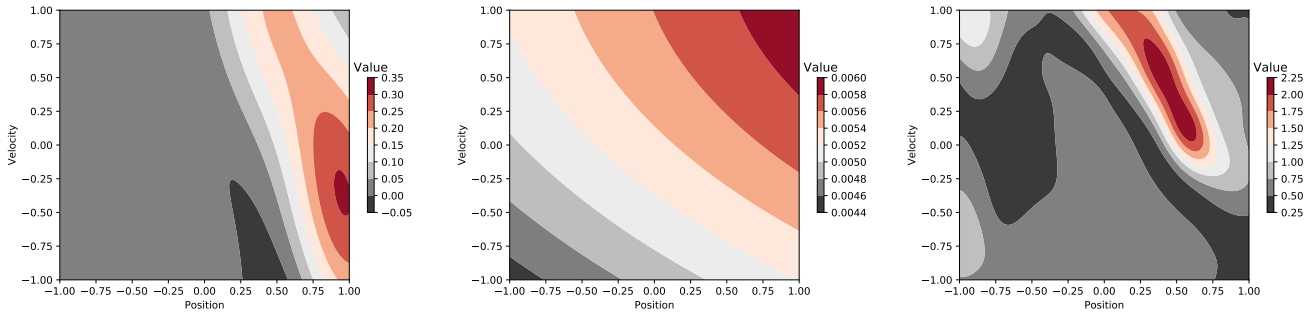
where $\mathbf{w}\mathbf{w}^T$ encodes relationships between members and $\boldsymbol{\alpha}$ contains independent variance terms for each member. θ contains now both the hyperparameters θ_{SE} of the SE kernel, as well as the parameters of G, and can be optimized using the training set $(X^F_{\text{tr}}, \mathbf{y}^F_{\text{tr}})$ of the entire fleet, annotated with the indices of its members.

Using the new fleet covariance kernel, we can describe a single GP jointly over the outputs of all fleet members, given in Equation 2, and compute the posterior statistics according to Equation 3 for regression. Note that even though the new model uses the whole fleet data set, each member can access its own individual model by computing their own posterior statistics for $\mathbf{y}^{(m)} \mid X^{(m)}, X^F_{\text{tr}}, \mathbf{y}^F_{\text{tr}}$. Thus, we can define a transition model $\mathcal{T}^{(m)}_\theta(\mathbf{s},a) = \mathcal{T}^F_\theta(\mathbf{s},a,m)$ for each member m.

5 MOUNTAIN CAR EXPERIMENT

To illustrate our method, we set up a preliminary experiment in the continuous mountain car domain, based on [6]. The car is positioned in a valley and its objective is to reach the top of the right-most hill.

[1] More complicated forms can be used for the reward function (e.g., a GP), without altering the form of V^π.



(a) Single **(b) Joint** **(c) Fleet**

Figure 1: Contour plots of the learned value functions (GPs) during the least performant runs (i.e., with the largest distance from the goal). Each of the scenarios are depicted; single (a), joint (b) and fleet (c) learning.

However, the slope is too steep for the car to simply accelerate to the top. Thus, it has to first drive up the opposite side of the valley, and then accelerate from there to reach the top.

In this problem, a state $\mathbf{s}$ consists of the position and velocity of the car, while an action is a force applied to either side of the car. Both the state features and action are in the range $[-1, 1]$. The start and goal state are respectively given by $\mathbf{s}_{\text{start}} = [-0.5, 0]$ and $\mathbf{s}_{\text{target}} = [0.5, 0]$, i.e., the bottom and top of the hill. We consider the following reward Gaussian function:

$$R(s) = \exp(-||\mathbf{s} - \mathbf{s}_{\text{target}}||^2/0.05^2) \tag{7}$$

where maximum reward is achieved at $\mathbf{s}_{\text{target}}$.

We consider a fleet of three mountain cars: a learner with a car mass of 1.0 kg, and two teachers with masses 1.1 kg and 5.0 kg. For each teacher, we provide a batch of 300 transitions randomly sampled from their environment. We do the same for the learner, but only sample 20 times, resulting in 620 samples total.

The learner cannot sufficiently estimate its transition model with only 20 samples, making it challenging to find the optimal policy. Transferring knowledge from the first teacher will be helpful. However, the last teacher is incapable of successfully climbing the hill due to its high mass, thus sharing samples with it would misrepresent the learner's transition model. Therefore, the objective of the learner is to estimate a sufficiently accurate transition model in order to solve the task, by figuring out the correlations with all teachers and use their knowledge accordingly. Once the transition model is learned, we compute the optimal value function and policy using the policy iteration method presented in [8].

To measure the performance of our algorithms, we run the experiment 100 times for three scenarios. In two scenarios, the learner simply uses the SE kernel described in Equation 4 and learns a model using only its own samples (single) or all fleet samples (joint). This sets the two extrema we compare our method to. In the third scenario, the learner uses all samples, but employs the fleet kernel described in Equation 5.

We focus on how close the learner can get to the goal state. Therefore, we compute the 2-norm of the smallest distance achieved during a run and fit a density curve (see Figure 2). We observe that simply learning from the full data set without the fleet kernel

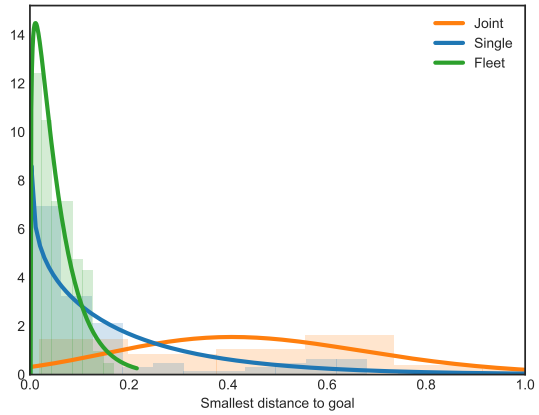


Figure 2: Densities over the smallest distances from the goal achieved during 100 runs, for each of the three scenarios. Both a fitted $\mathcal{X}^2$ distribution and histogram are plotted for each of the three scenarios; single, joint and fleet learning.

rarely leads to reaching the goal. This is due to the heavy teacher being incapable to reach the goal, which is untrue for the learner. Therefore, it is often standing still during runs. Learning from own experience can lead to good results, but is usually insufficient to represent the dynamics well. Due to the uncertainty in the dynamics model, the car is often incapable of finding a suitable policy. The only learner to consistently achieve good results is the one with the fleet kernel, as the learner is able to figure out which teacher is most useful to share data with. On average, the fleet learner achieves a distance of 0.05 from the goal state, while the single and joint learners respectively maintain an expected distance of 0.46 and 0.18. The fleet learner's mean is significantly different from the others with p-values $< 1e{-}4$ on Wilcoxon signed-rank tests.

Interestingly, none of the runs of the fleet learner fall above a minimum distance of 0.2, meaning that the learner at least knows the direction it should move towards, which is not true for the

	L	T1	T2
L	1	0.9996	0.5207
T1	0.9996	1	0.5208
T2	0.5207	0.5208	1

(a) Position

	L	T1	T2
L	1	0.9986	0.3099
T1	0.9986	1	0.3103
T2	0.3099	0.3103	1

(b) Velocity

Table 1: Optimized correlation matrix between the fleet members, i.e., learner (L), teacher 1 (T1) and teacher 2 (T2) for both state dimensions.

other two scenarios. This becomes clear when we examine Figure 1, which shows the resulting value functions during the worst runs in terms of distance from the goal. The fleet learner finds the region of highest reward, which is around s_{target}. The single learner misidentifies this region, and according to the reward scale, it also captures a fairly flat value surface due to its uncertainty. The joint learner just learns to go as right as possible, but expects not much difference in reward according to the scale.

We focus now on the correlation matrices learned by the fleet learner. Given the optimized covariance matrix G from Equation 6, we can compute the correlation matrix

$$\text{corr}(G) = (\text{diag}(G))^{-0.5} G (\text{diag}(G))^{-0.5} \tag{8}$$

The element-wise means of these matrices over all runs are given in Table 1. The learner successfully identifies the first teacher to be similar, while assigning a notably lower correlation value to the last one. However, note that there still exists a moderate amount of correlation between the two agents, especially in the position-dimension, while the joint learner clearly fails to solve the task. As we are using correlations, two members are considered similar if there exists a linear relationship between their outputs, and not only when they are spatially close to one another. The fleet kernel detects this linear relationship and accomplishes to take this into account.

6 CONCLUSION

In this work, we introduced a novel sample-efficient fleet RL method based on Gaussian processes and coregionalization. It detects correlations between a fleet of devices, and manages to transfer knowledge between these devices. We provided a preliminary example of how the method works in a setting where a learner is given data from a similar and dissimilar fleet member. While the extrema of no transfer and single model learning perform poorly, the learner that uses our method manages to solve the task consistently.

In future work, we will compare our method to the Bayesian hierarchical multi-task RL approach [11] in more complex settings. We postulate our method will outperform the hierarchical alternative in fleet settings where members are highly similar, and direct knowledge transfer would prove to be beneficial in contrast to shared prior parameter distributions over subsets of members. Additionally, we will further develop the fleet transition model. Two main issues should be investigated. First, as a learner acquires more experience, information from the other fleet members becomes irrelevant, or will even bias the transition model and therefore decrease performance. This is called negative transfer, and it is important to analyse this phenomenon. However, we expect correlations to diminish on their own, as the model does not compute these statistics on the data directly, but rather optimizes these values w.r.t. the model's accuracy on the observed data. This means that when the model becomes inaccurate when including data from a similar member, it should automatically reduce the correlation with that member. Secondly, it is often the case that new or non-stationary members require more of the fleet's data, while others can sufficiently solve the task with their own acquired experience. However, correlations are bidirectional, causing either negative transfer to occur to the more knowledgeable fleet member when the correlations are high, or insignificant transfer to the less knowledgeable fleet member when they are low. We will explore possible unidirectional relationships to counter this issue.

ACKNOWLEDGMENTS

Timothy Verstraeten was supported by a PhD grant of the FWO (Fonds Wetenschappelijk Onderzoek - Vlaanderen). We thank the anonymous reviewers for their careful reading of our manuscript and their insightful suggestions.

REFERENCES

[1] Edwin V. Bonilla, Kian Ming A. Chai, and Christopher K. I. Williams. 2008. Multi-Task Gaussian Process Prediction. *Advances in Neural Information Processing Systems* 20 (2008), 153–160.
[2] M. P. Deisenroth and C. E. Rasmussen. 2011. PILCO: A Model-based and Data-Efficient Approach to Policy Search. In *Proc. of the International Conference on Machine Learning*.
[3] Yaakov Engel, Shie Mannor, and Ron Meir. 2005. Reinforcement Learning with Gaussian Processes. In *Proc. of the 22nd International Conference on Machine Learning*. ACM Press, 201–208.
[4] Pierre Goovaerts. 1997. Geostatistics for Natural Resource Evaluation. 42 (1997).
[5] Alessandro Lazaric and Mohammad Ghavamzadeh. 2010. Bayesian Multi-Task Reinforcement Learning. In *Proc. of the 27th International Conference on Machine Learning*. Omnipress, 599–606.
[6] A. Moore. 1990. *Efficient Memory-Based Learning for Robot Control.* Ph.D. Dissertation. University of Cambridge.
[7] Martin L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (1st ed.). John Wiley & Sons, Inc.
[8] Carl Edward Rasmussen and Malte Kuss. 2003. Gaussian Processes in Reinforcement Learning. 16 (2003), 751–758.
[9] Carl Edward Rasmussen and Christopher K. I. Williams. 2006. *Gaussian Processes for Machine Learning.* The MIT Press, Cambridge, MA, USA.
[10] Richard S Sutton and Andrew G Barto. 1998. *Reinforcement learning: An introduction.* MIT press Cambridge.
[11] Aaron Wilson, Alan Fern, Soumya Ray, and Prasad Tadepalli. 2007. Multi-task reinforcement learning: A hierarchical Bayesian approach. In *Proc. of the 24th International Conference on Machine Learning*. 1015–1022.