Adapting Neural Models with Sequential Monte Carlo Dropout

Anonymous Author(s)
Affiliation
Address
email

Abstract: The ability to adapt to changing environments and settings is essential for robots acting in dynamic and unstructured environments or working alongside humans with varied abilities or preferences. This work introduces an extremely simple and effective approach to adapting neural models in response to changing settings, without requiring any specialised meta-learning strategies. We first train a standard network using dropout, which is analogous to learning an ensemble of predictive models or distribution over predictions. At run-time, we use a particle filter to maintain a distribution over dropout masks to adapt the neural model to changing settings in an online manner. Experimental results show improved performance in control problems requiring both online and look-ahead prediction, and showcase the interpretability of the inferred masks in a human behaviour modelling task for drone tele-operation.

Keywords: Model Adaptation, Meta-Learning, Online Robot Control and Prediction

1 Introduction

Neural models and policies are now ubiquitous in modern robotics. The prevailing approach to training these follows a two stage process – a large, comprehensive collection of data (often state and action pairs) is used to train a suitable model or policy, which is then frozen and deployed. Unfortunately, this results in models that are unable to adapt to changes in the environment, which is a particular concern in robotics. For example, it would be preferable for a robot dynamics model to handle context dependent kinematic or dynamic properties, or a collaborative robot relying on predictions of human behaviour to adapt to different human abilities or preferences. Many existing adaptive control techniques [1] attempting to tackle this problem rely on carefully considered parametric models, but these may lack the requisite capacity for prediction that is typically associated with neural models. In contrast, meta-learning and adaptive neural control approaches addressing this problem are often quite cumbersome to train and implement.

This paper introduces a simple and effective approach to achieve adaptation for neural network models. First we train a model, $f_{\boldsymbol{\theta}}(\cdot)$ using dropout [2], on data gathered from different settings or tasks. Dropout can be considered analogous to randomly masking neurons in the network, producing an ensemble of predictive models, or a distribution over possible outputs [3]. At run-time, a particle filter is used to update the dropout mask, depending on the predictive quality of the model, so as to condition the network to operate in a given context. This allows for neural models that can easily and rapidly adapt to changes in the environment or behaviour of a human collaborator in online settings.

Results show that this simple-to-implement approach improves upon more complex meta-learning training strategies, and that the resulting masks can be interrogated to obtain additional information about the adaptations required for particular contexts. This paper shows how this information can be leveraged in a human modelling application to answer questions about a given user. In summary, the

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

core contributions of this work are to show that: **(1)** sequential Monte Carlo dropout is an effective technique for online model adaptation in rapidly changing settings, **(2)** the masks inferred by this process provide a context-dependent 'signature' that is of particular value for human behaviour modelling in support of personalised robot decision making.

2 Background and Related Work

Model adaptation: Model adaptation in an online fashion is a core focus of adaptive control [1], and typically involves the design of a suitable observer or estimation scheme [4] to infer parameters of interest. Many of these system identification approaches [5, 6, 7, 8, 9] rely on models conditioned on known physical properties like friction, mass and gravity, and infer these from data. These approaches require knowledge of the parametric form of uncertainty and are only feasible for very specific model structures [10]. Our approach aims to challenge this, by introducing an adaptation mechanism that can be used with more general predictive neural models.

For deep learning, model adaptation is typically studied as a generalisation problem in a meta-learning context. Popular approaches like model agnostic meta learning (MaML) [11] seek to learn network weights that can be easily adapted for different tasks by gradient descent, using specialised training regimes that alternate between task specific training and maintaining a representation that is well suited to multiple tasks. This can be time-consuming, but faster, first order meta-learning approximations such as Reptile [12] have proven relatively effective. The idea of enhancing generalisation by learning well-structured internal representations underpins a number of works such as ProtoNet [13] and Relation networks [14], alongside more recent approaches that rely on multi-task training with an explicit learned auxiliary input [15, 16] that can be adapted online. Our approach can also be viewed through a representation learning lens, in that masks 'prime' the network to produce context dependent outputs. This representation-priming perspective also aligns with recently developed transformer based architectures tackling meta-learning [17, 18, 19]. Here, key embeddings are used to unlock network components required for context specific prediction. The dropout mask used in our work can be considered analogous to a transformer key that is inferred at run time.

Several works targeting online model adaptation have argued that we should ideally update our models and beliefs in response to evidence in a Bayesian fashion. This approach is taken in elastic weight consolidation, which aims to reduce the plasticity of network weights that are important to a task [20]. There is some evidence that MaML can be reinterpreted as a Bayesian hierarchical model [21, 22]. Dropout [2] is a popular regularisation technique for neural network training, and has also been used to represent prediction uncertainty in deep learning [3]. We leverage dropout within a sequential importance sampling framework to provide an explicit Bayesian update rule for mask inference and model adaptation. Like [20], this preserves connections important to a task. As an alternative to back-propagation, evolutionary, sampling-based approaches to training neural models [23, 24, 25] have been explored previously and shown to be effective in a reinforcement learning context [26]. Pathnet [27] uses evolution to find the neural network pathways that should be trained for specific tasks. Particle filtering has also been used to prune neural networks for compression [28]. Our work is similar, but considers the online adaptation case and samples dropout masks, rather than considering all weights and connections.

Human modelling: The dropout mask inferred at run-time using our approach can be considered an embedding or summary of the context-specific adaptation required by the network at a given time step. This is valuable for adaptive human modelling in shared-autonomy and similar collaborative applications, where context-specific masks could help guide the type of assistance offered to an operator. Human behaviour models help robots to understand human preferences, predict future human actions, and adapt to changes in human behaviour. Two general approaches to human behaviour modelling tend to be taken in robotic applications. The first postulates that humans are rational agents whose actions are in accordance with some (un)known intentions and rewards [29]. To account for human variability and diversity in behaviour – in particular deviations from optimal action choices, these models are augmented with computational interpretations of high-level

decision factors such as trust [30], expertise [31], fatigue [32], and adaptability [33], among others. These factors can be inferred in an online manner during interaction, thus allowing robots to adapt their responses to different humans (e.g., [34]). These models are often preferred because of their simplicity and interpretability, however they suffer two main drawbacks. First, they strongly rely on simplistic parametric model structures, which can limit model expressivity and predictive power. In particular, most models assume that a single high-level decision factor is sufficient to capture and explain most of the observed human behaviour (e.g., [34]). Second, to make inference tractable in real-time, these approaches are often limited to simplified state and action space representations, and further assume that the model parameters remain stationary for the duration of the whole interaction (e.g. [33]). Our approach aims to relax these constraints, by introducing an adaptation mechanism that can be used with more general predictive and expressive neural models and that also captures changing behaviour.

In the second family of approaches, collected human behaviour data is used to train machine learning models that predict future behaviour (i.e., imitation learning) [35, 36, 37]. Because of their data-driven nature, and assuming good quality data is provided during training, these models can implicitly capture most of the sources of variability in human behaviour without making explicit assumptions about what those sources are. Thus, these models of human behaviour are no longer limited to single decision factors. However, this increase in model expressivity and predictive power comes at the cost of limited interpretability [38]. Furthermore, these models can fail to generalize and adapt to humans whose actions differ from the training data set [16]. Our approach addresses the interpretability and adaptation limitations of imitation-based human models through the online estimation of dropout masks. As shown in Sec. 5, this mask inference process allows for successful local adaptation in extrapolation (to human users never seen during training). Similarly, we show that contextual information pertaining to high-level factors such as a human's skill level and preferences can be queried through a simple k-nearest mask approach.

3 Method

We consider a standard Markov decision process, with $\mathbf{x}_t$ denoting a state at time t, and R_t denoting the reward received at time t after transitioning from state $\mathbf{x}_{t-1}$ to state $\mathbf{x}_t$, due to control action $\mathbf{u}_t$. Let $f_\theta(\cdot)$ be a neural network with parameters θ. Networks such as these are frequently trained to approximate dynamics models for future state prediction $\mathbf{x}_t \sim f_\theta(\mathbf{x}_{t-1}, \mathbf{u}_t)$, for reward prediction $R_t \sim g_\theta(\mathbf{x}_{t-1}, \mathbf{u}_t)$, or as policies for selecting an action given a state $\mathbf{u}_t \sim \pi_\theta(\mathbf{x}_t)$.

Our goal is to find a way of adapting a pre-trained network to changing settings, in response to some deviation between predicted states, actions or rewards and those actually observed. Let $f_\theta(\cdot|\mathbf{M}_t)$ denote a neural network trained using dropout regularisation. Here $\mathbf{M}_t$ represents a binary mask which suppresses or drops neurons in the network. When training with dropout, neurons are randomly masked during the forward pass. We do not consider any specialised training regime, and simply use multitask training - samples drawn at random from a large training dataset.

We now show how this can be exploited to enable online adaptation. For convenience, we restrict the discussion below to a dynamics model setting, where a neural network is trained to predict a state $\mathbf{x}_t$, given a previous state $\mathbf{x}_{t-1}$ and action $\mathbf{u}_t$. However, it should be noted that the proposed approach can be used to adapt policies, reward predictors and in many other settings.

3.1 Online Mask Adaptation

We use the deviation between predicted $\mathbf{x}_t$ and actual state measurements $\mathbf{z}_t$ to form a Gaussian likelihood,

$$p(\mathbf{z}_t|\mathbf{M}_t) = \mathcal{N}(\mathbf{z}_t|f_\theta(\mathbf{x}_{t-1}, \mathbf{u}_t|\mathbf{M}_t), \mathbf{\Sigma}), \tag{1}$$

where $\mathbf{\Sigma}$ denotes measurement noise. This likelihood measures the chances of making an observation, given a dropout mask. Taking an ensemble view of model training with dropout, we aim to find the mask, and corresponding model, that results in the most accurate prediction in a given set-

ting. More formally, our goal is to determine a posterior distribution over dropout masks $p(\mathbf{M}_t|\mathbf{z}_{1:t})$ conditioned on a series of observations $\mathbf{z}_{1:t}$. To do so, we formulate a sequential update rule

$$p(\mathbf{M}_t|\mathbf{z}_{1:t}) \propto p(\mathbf{z}_t|\mathbf{M}_t)p(\mathbf{M}_t|\mathbf{z}_{1:t-1}), \text{ where} \tag{2}$$

$$p(\mathbf{M}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{M}_t|\mathbf{M}_{t-1})p(\mathbf{M}_{t-1}|\mathbf{z}_{1:t-1})\mathrm{d}\mathbf{M}_{t-1}, \tag{3}$$

using an appropriate transition model $p(\mathbf{M}_t|\mathbf{M}_{t-1})$ for the dropout mask. In this work, we propose to randomly bit-flip some small fraction ($d\,\%$) of the mask for the transition model, $p(\mathbf{M}_t|\mathbf{M}_{t-1})$. This allows for some exploration over potential new masks and adaptation to new settings, while keeping existing masks close to those associated with networks best suited to current settings. Importantly, bit flipping can be efficiently computed using an xor operation.

The integral in (3) is intractable, so we approximate this using importance sampling, reducing the computation of the posterior over masks to a bootstrap particle filter [39]. Here, we approximate the distribution over masks using a set of N weighted particles, $p(\mathbf{M}_t|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{N} w_i \delta(\mathbf{M}_t^i)$, with weights $w_i \propto p(\mathbf{z}_t|\mathbf{M}_t^i)$, particle $\mathbf{M}_t^i$ drawn from the transition distribution $p(\mathbf{M}_t^i|\mathbf{M}_{t-1}^j)$, and Dirac measure $\delta(\cdot)$ at point $(\cdot)$. A resampling step is used after each time measurement to prevent particle degeneracy. This produces a distribution over masks and a corresponding posterior over the predicted state

$$p(\mathbf{M}_t|\mathbf{z}_{1:t}) \approx \frac{1}{N}\sum_{i=1}^{N} \delta(\mathbf{M}_t^i); \tag{4}$$

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{z}_{1:t}) &= \int p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{z}_{1:t}, \mathbf{M}_t)p(\mathbf{M}_t|\mathbf{z}_{1:t})\mathrm{d}\mathbf{M}_t \\ &\approx \frac{1}{N}\sum_{i=1}^{N} \delta\left(f_\theta(\mathbf{x}_{t-1}, \mathbf{u}_t, \mathbf{M}_t^i)\right). \end{aligned} \tag{5}$$

For downstream tasks like control, we are only interested in the adapted prediction model, so we compute a minimum-mean squared error mask estimator,

$$\bar{\mathbf{M}}_t = \frac{1}{N}\sum_{i=1}^{N} \mathbf{M}_t^i, \tag{6}$$

from the mask particles and use this to condition the network f_θ.

Algorithm 1 illustrates the adaptation process, which we term sequential Monte Carlo dropout (SMCD). We first sample N possible masks from the prior mask distribution by bit flipping, and then make a batch prediction of possible states for each mask. We then compute the likelihood for each mask using (1), normalise and re-sample, to obtain an approximate distribution over masks that give predictions close to the observation. Finally, we compute a minimum-mean squared error mask, which can be used to condition the network in downstream tasks. The process repeats recursively, in an online fashion.

Algorithm 1 Sequential Monte Carlo Dropout (SMCD) Model Adaptation

Input: Trained network f_θ, Observation sequence $\mathbf{z}_{1:T}$,
Transition probability d, Measurement uncertainty $\mathbf{\Sigma}$
Output: Mask estimate sequence $\bar{\mathbf{M}}_{1:T}$
Initialise N random masks $\{\mathbf{M}_0^i\}_{i=1\ldots N}$
for $t = 1 \ldots$ T **do**
  for $i = 0 \ldots N$ **do**
    Sample $\mathbf{M}_t^i \sim p(\mathbf{M}_t^i|\mathbf{M}_{t-1}^i)$ by bit-flipping
    Sample $\mathbf{x}_t^i = f_\theta(\mathbf{x}_{t-1}, \mathbf{u}_t|\mathbf{M}_t^i)$
    Evaluate $w^i = \mathcal{N}(\mathbf{z}_t|\mathbf{x}_t^i, \mathbf{\Sigma})$
  end for
  Draw N masks: $\{\mathbf{M}_t^i\}_{i=1\ldots N} \sim \frac{\sum_{i=1}^{N} w_i \delta(\mathbf{M}_t^i)}{\sum_{i=1}^{N} w_i}$
  Compute best mask estimate $\bar{\mathbf{M}}_t = \frac{1}{N}\sum_{i=1}^{N} \mathbf{M}_t^i$
end for

A key benefit of this approach is that it does not change the underlying models learned. The original prediction can be recovered at any time, by simply restoring or resampling the mask. This is in contrast to approaches that continue to train or take gradient steps in response to errors, which may be more capable of continual learning, but are vulnerable to catastrophic forgetting [40, 20].

4 Experiments

We investigate model adaptation using SMCD both in simulation and with real-world human data. In simulation, we use a two-link arm with changing link lengths to show how the proposed model adapts to changes in the system's kinematics and how it performs in comparison to standard meta-learning approaches, both in prediction and control. The real-world data corresponds to a drone tele-operation scenario where operators of varying skill levels completed several landing tasks. This scenario illustrates mask interpretability by showing how different high-level characteristics (landing strategy and an operator's skill level) can be determined by a simple mask comparison process.

4.1 Two-Link Arm Example

The 2-link arm follows the dynamics and forward kinematics:

$$\dot{q_1} = u_1,\ \dot{q_2} = u_2 \tag{7}$$

$$x = l_1 \cos(q_1) + l_2 \cos(q_1 + q_2),\ y = l_1 \sin(q_1) + l_2 \sin(q_1 + q_2). \tag{8}$$

We train neural models $f_\theta(\boldsymbol{q})$ to perform forward kinematics (predicting end-effector position $\mathbf{x} = [x, y]^{\mathrm{T}}$ given joint angles $\boldsymbol{q} = [q_1, q_2]^{\mathrm{T}}$) for variable limb lengths l_1, l_2. Training data is gathered by randomly initialising $q_i \sim U(-\pi, \pi)$, and episodically motor babbling $u_i \sim \mathcal{N}(0, 1)$ for 10 steps. We generate 150 episodes for 1000 tasks, with fixed link lengths $l_i \sim \mathcal{N}(1, 0.3)$ in each task.

The two-link forward kinematics task is used to investigate the adaptation properties of SMCD in two possible use cases, N-step look-ahead prediction after T adaptation steps, and online, on-policy adaptation using a proportional-derivative (PD) control law. For the former, we evaluate prediction error for an unknown manipulator with link lengths ($l_i \sim \mathcal{N}(1, 0.3)$) fixed over a motor babbling episode ($u_i \sim \mathcal{N}(0, 1)$), starting from a randomly sampled angle $q_i \sim U(-\pi, \pi)$. This evaluates "interpolation" ability, and suitability for receding-horizon model predictive control tasks.

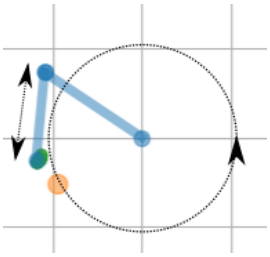


Figure 1: A 2-link arm with sinusoidally varying length l_2 following a moving target.

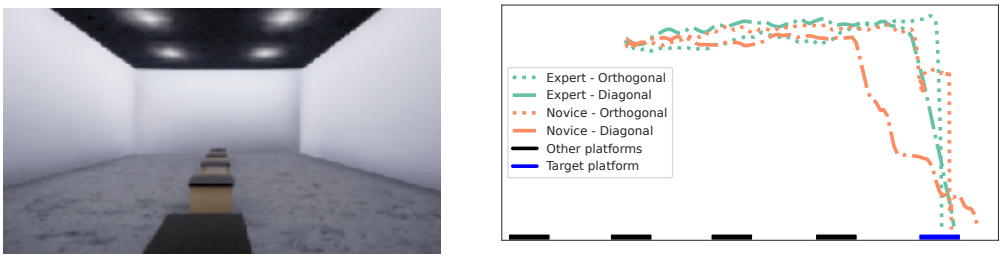
The online control task (Fig. 1) is substantially more challenging. Here, we control the 2-link manipulator to follow a moving target $\mathbf{x}_g$ around a circle of radius $r \sim U(0.3, 1.5)$, with starting angle $\phi \sim U(-\pi, \pi)$, measured relative to the base of a randomly initialised manipulator. We use the PD control law $\mathbf{u} = -K_1 \mathbf{J}^{\dagger}(\mathbf{x} - \mathbf{x}_g) - K_2 \dot{\boldsymbol{q}}$ to follow the moving target ($K_1 = 20, K_2 = 5$). Here, $\mathbf{J}^{\dagger}$ is the Moore-Penrose pseudo-inverse of the Jacobian of the network $f_\theta(\boldsymbol{q})$. In order to test online forward kinematics adaptation, we consider a telescoping second link, with sinusoidally varying length $l_2 = 1 + 0.25 \sin(\pi k/20)$, with timestep $k \in (0, 100)$. This experiment evaluates convergence speed in response to an initial disturbance, alongside continual online adaptation ability. We benchmark against an oracle model, no adaptation and a model trained using Reptile [12], a first order meta-learning approach, and investigate adaptation using both gradient descent and SMCD.

4.2 Drone Landing Example

We use the human data previously introduced in [41][1] for this experiment. This dataset comprises 528 tele-operation trajectories collected for 33 human operators (16 trajectories per operator) with different skill levels in a simulated environment. Here, operators were asked to pilot and land a drone on one of five platforms by providing linear velocity commands through keyboard inputs. As shown in Fig 2a, the operator's view was fixed to a single-point perspective, which resulted in an increase in task complexity the further away the target platform was located. The simulated environment and landing task sequence were the same for all operators.

We train neural models $\boldsymbol{u}_t \sim f_\theta(\boldsymbol{x}_t)$ to predict the operator's action $\boldsymbol{u}_t$ given the current state $\boldsymbol{x}_t$ for variable tasks (i.e., different target platforms), user skill levels, and landing strategies (see Fig. 2b

[1] Secondary analysis of this data was reviewed and approved by an institutional Research Ethics Committee.



(a) Drone landing simulation environment. (b) Examples of tele-operation trajectories (YZ view).

Figure 2: Drone landing data consists of tele-operation trajectories collected in a simulated environment (a). Data includes trajectories executed by operators of varying skill (one novice and one expert) and captures different landing strategies (orthogonal or diagonal) as shown in (b).

for examples of tele-operation trajectories with these characteristics). A tele-operation trajectory $\boldsymbol{\tau} = (\boldsymbol{x}_1, \boldsymbol{u}_1, \ldots, \boldsymbol{x}_T, \boldsymbol{u}_T)$ consists of a sequence of states and actions, where each state $\boldsymbol{x}_t = (\mathbf{p}_t, \mathbf{v}_t, d_t)$ denotes the drone's 3D position $\mathbf{p}_t = (x_t, y_t, z_t)$, linear velocity $\mathbf{v}_t = (\dot{x}_t, \dot{y}_t, \dot{z}_t)$, and distance $d_t = |\mathbf{g}^i - \mathbf{p}_t|_2$ to target platform $\mathbf{g}^i$ at time t; and $\boldsymbol{u}_t = (\dot{x}_t, \dot{y}_t, \dot{z}_t)$ corresponds to the linear velocity commands provided by the human operator. We randomly split the dataset into training (83%), known-operators testing (10%), and unknown-operators testing (7%) sets.

The drone landing example is used to further assess the performance of the proposed approach in look-ahead prediction, and to investigate the interpretability of the masks obtained using SMCD. For look-ahead prediction, we evaluate prediction error on trajectories obtained from known (some trajectories were seen during training) and unknown (all trajectories excluded during training) operators.

For the mask interpretability task, we evaluate whether one high-level characteristic of a trajectory (landing strategy) and one characteristic about the operator (skill level) are captured by the inferred masks[2] (see Fig. 2b for trajectories showcasing these characteristics). Specifically, we measure how the prediction confidence of each target characteristic changes as the number of observed state-action pairs seen during mask inference increases. We use the distance-based confidence score proposed in [42] to evaluate this. Given the mask estimate, $\bar{\mathbf{M}}_{1:n}$, for trajectory $\boldsymbol{\tau}$ obtained after n observations, with $1 \leq n < T$, let $A(\bar{\mathbf{M}}_{1:n}) = \{\bar{\mathbf{M}}^j_{\text{train}}\}_{j=1}^k$ denote the set of k-nearest masks of $\bar{\mathbf{M}}_{1:n}$ in the training set, and let $\{c^j\}_{j=1}^k$ denote the corresponding characteristic labels associated with masks in $A(\bar{\mathbf{M}}_{1:n})$. Following [42], the confidence score $D(\boldsymbol{\tau})$ for the assignment of trajectory $\boldsymbol{\tau}$ to characteristic $\hat{c}$ based on the mask estimate $\bar{\mathbf{M}}_{1:n}$ is defined as:

$$D(\boldsymbol{\tau}) = \frac{\sum_{j=1, c^j=\hat{c}}^k e^{-\|\bar{\mathbf{M}}_{1:n} - \bar{\mathbf{M}}^j_{\text{train}}\|_2}}{\sum_{j=1}^k e^{-\|\bar{\mathbf{M}}_{1:n} - \bar{\mathbf{M}}^j_{\text{train}}\|_2}}. \tag{9}$$

The score $D(\boldsymbol{\tau}) \in (0, 1)$ (the higher the better) is monotonically related to the local density of trajectories in the training set that share similar characteristics and lie in the neighborhood of trajectory $\boldsymbol{\tau}$, as defined by the distance between inferred masks.

5 Experimental Results

Two-link arm: look-ahead prediction error: Fig. 3 shows the root mean square prediction error (RMSE) over 1000 trials, for varied burn in steps over a range of prediction horizons. We compare best performing models[3] trained using multitask pre-training and Reptile meta learning, and then adapted using a range of adaptation methods. We also include an oracle model baseline, where link-length parameters in a known forward kinematics model are inferred directly using a particle filter.

[2]Characteristic definitions and details are provided in the Appendix.

[3]Best performing models were selected following a hyperparameter search - see Appendix.

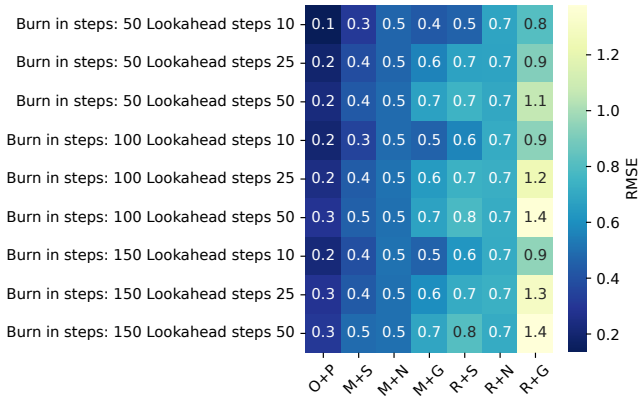


Figure 3: Look-ahead prediction errors. (O: Oracle, M: Multitask, R: Reptile; P: Particle Filter, S: SMCD, G: Gradient descent, N: No adaptation.)

Multitask training with SMCD is most effective, with the exception of the oracle model. Interestingly, adaptation using gradient descent seems to exhibit more local adaptation, as prediction error grows larger with look-ahead distance than when SMCD is applied. Local adaptation seems to provide an accurate model in a specific region of state space, rather than globally adapting to a particular link length. The Reptile model performs surprisingly poorly, particularly given that it was trained for substantially longer (1000 epochs) than the multitask model (200 epochs).

Two-link arm: online adaptation control error: Fig. 4 shows the average control error for the online control task with sinusoidally varying links, across 1000 repetitions. Results are shown for the best performing reptile and multitask-trained models for no adaptation (None), online gradient descent adaptation (GD) and online sampling-based adaptation (SMCD).

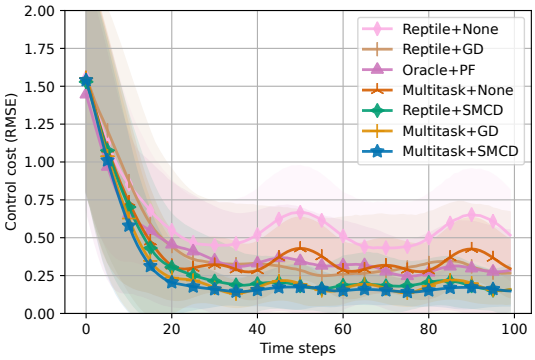


Figure 4: Closed-loop PD control of a 2-link arm following a moving target with a sinusoidally telescoping link (1-sigma shaded traces).

As in the look-ahead case, multitask training with SMCD adaptation is most effective, producing rapid convergence to the moving goal when used for control. Interestingly, particle filtering with a known model (Oracle+PF) struggles to deal with the rapidly changing forward kinematics, but adaptation in mask space or by gradient descent converges much faster.

Drone landing: look-ahead prediction error: Fig. 5 shows the RMSE for held-out tele-operation trajectories. We consider two test sets: known (left) and unknown (right) operators for varied burn in and prediction horizon ratios. We use ratios instead of steps since trajectories are of varying lengths. As with the two-link example, we compare the best performing models trained using multitask pre-training and Reptile meta learning, with and without adaptation. Note that multitask pre-training without adaptation corresponds to a behaviour cloning approach to action prediction.

As expected, the look-ahead prediction error obtained for trajectories executed by known operators is lower than the error rates obtained for the unknown operator trajectories. However, as initially observed in the two-link arm example, this difference in performance grows smaller as the duration of the adaptation and look-ahead prediction horizons increases. This suggests that rather than globally adapting to a particular task instance or operator's tele-operation style, the proposed approach is better at providing accurate models in specific regions of the state space. In terms of overall model performance, we observe that multi-task training with SMCD is most effective, followed by multi-task training with Gradient Descent adaption. Once again, the model trained using Reptile performs poorly even when trained for substantially longer (1000 epochs) than the other models (400 epochs).

Drone landing: mask interpretability: Fig. 6 shows the average confidence score obtained for the prediction of the true skill level of a trajectory's operator (i.e., novice, intermediate, or expert) and true landing strategy (i.e., orthogonal or diagonal) for all trajectories in our testing sets as the number of time steps used during mask inference increases. Overall, we observe relatively good average confidence scores (~ 0.7) for both testing sets. This suggests that the masks inferred using SMCD

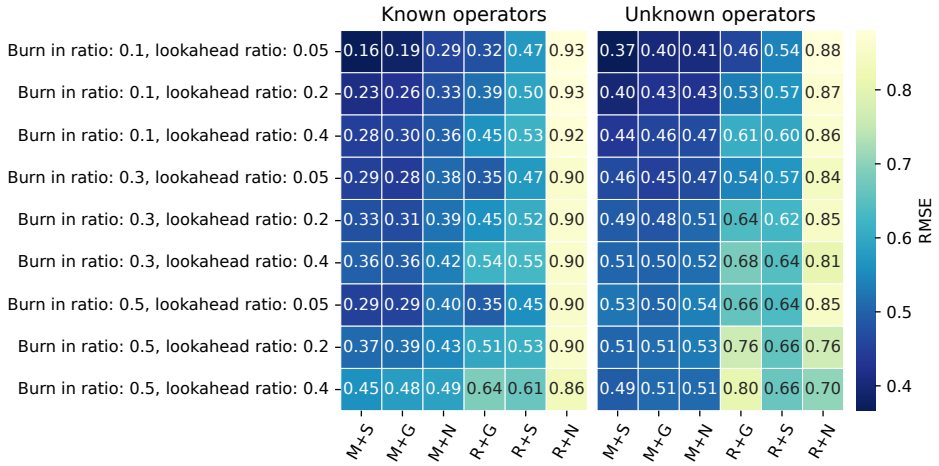


Figure 5: Look-ahead prediction errors for human data, ordered by performance. (M: Multitask, R: Reptile, G: Gradient descent, S: SMCD, N: No adaptation).

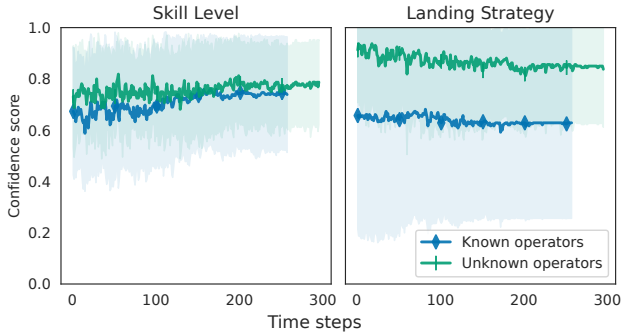


Figure 6: Average confidence score for the correct assignment of the true skill level and landing strategy for all testing trajectories (1-sigma shaded traces). Results are presented for the set of 5-nearest masks.

contain enough global information to allow for the identification of operator skill level and the landing strategy that they would likely employ. Both characteristics can be identified early on in the trajectory. This is particularly important for the landing strategy, which appears to change more towards the trajectory end. Better confidence scores are obtained for trajectories executed by unknown operators than for the known operator testing instances. This might be due to imbalances in the number of instances that belong to each possible characteristic, as visual inspection of trajectories indicated unknown operators exhibited trajectory characteristics more common in the dataset.

6 Limitations

Although neural dynamics modelling holds the potential to scale to high dimensional inputs of varied modalities, this work has only considered adaptation in relatively small scale problems with low dimensional state spaces. Future work should explore the potential of the proposed approach to operate in more realistic control settings from higher dimensional inputs. It should be noted that control with adaptive neural models may lack the stability and safety guarantees associated with conventional modelling. Moreover, although SMCD is fast, and converges rapidly, doing so does require an energy expensive GPU. Users of the neural adaptation schemes proposed here should carefully consider their model size to determine whether sampling or back-propagation is more computationally efficient, given the relative efficacy in the multitask training setting.

7 Conclusion

This work proposed a simple neural network adaption strategy, relying on sequential Monte Carlo adaption of the masks in a dropout regularisation network layer. These masks offer a level of interpretability, capturing local contextual information, as illustrated in a real world human behavioural modelling application. Results show that when paired with multitask training, this simple adaptation strategy outperforms more computationally complex meta learning (Reptile) schemes.

References

[1] P. A. Ioannou and J. Sun. *Robust adaptive control*, volume 1. PTR Prentice-Hall Upper Saddle River, NJ, 1996.

[2] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15 (1):1929–1958, 2014.

[3] Y. Gal and Z. Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR.

[4] E. D. Sontag. *Observers and Dynamic Feedback*, pages 315–345. Springer New York, New York, NY, 1998.

[5] M. Burke. Path-following control of a velocity constrained tracked vehicle incorporating adaptive slip estimation. In *2012 IEEE International Conference on Robotics and Automation*, pages 97–102, 2012.

[6] J. Wu, I. Yildirim, J. J. Lim, B. Freeman, and J. Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[7] M. Asenov, M. Burke, D. Angelov, T. Davchev, K. Subr, and S. Ramamoorthy. Vid2param: Modeling of dynamics parameters from video. *IEEE Robotics and Automation Letters*, 5(2): 414–421, 2019.

[8] M. Jaques, M. Burke, and T. M. Hospedales. Physics-as-inverse-graphics: Unsupervised physical parameter estimation from video. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[9] F. Ramos, R. Possas, and D. Fox. Bayessim: Adaptive domain randomization via probabilistic inference for robotics simulators. In *Proceedings of Robotics: Science and Systems*, FreiburgimBreisgau, Germany, June 2019.

[10] L. Brunke, M. Greeff, A. W. Hall, Z. Yuan, S. Zhou, J. Panerati, and A. P. Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5, 2021.

[11] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[12] A. Nichol and J. Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2(3):4, 2018.

[13] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

[14] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018.

[15] S. Belkhale, R. Li, G. Kahn, R. McAllister, R. Calandra, and S. Levine. Model-based meta-reinforcement learning for flight with suspended payloads. *IEEE Robotics and Automation Letters*, 6(2):1471–1478, 2021.

[16] C. Laidlaw and A. Dragan. The boltzmann policy distribution: Accounting for systematic suboptimality in human models. *arXiv preprint arXiv:2204.10759*, 2022.

[17] S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-Maron, M. Gimenez, Y. Sulsky, J. Kay, J. T. Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

[18] A. Zhmoginov, M. Sandler, and M. Vladymyrov. Hypertransformer: Model generation for supervised and semi-supervised few-shot learning. *arXiv preprint arXiv:2201.04182*, 2022.

[19] A. Gupta, L. Fan, S. Ganguli, and L. Fei-Fei. Metamorph: Learning universal controllers with transformers. *arXiv preprint arXiv:2203.11931*, 2022.

[20] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[21] E. Grant, C. Finn, S. Levine, T. Darrell, and T. Griffiths. Recasting gradient-based meta-learning as hierarchical Bayes. *arXiv preprint arXiv:1801.08930*, 2018.

[22] J. Yoon, T. Kim, O. Dia, S. Kim, Y. Bengio, and S. Ahn. Bayesian model-agnostic meta-learning. *Advances in neural information processing systems*, 31, 2018.

[23] O. Miglino, H. H. Lund, and S. Nolfi. Evolving mobile robots in simulated and real environments. *Artificial life*, 2(4):417–434, 1995.

[24] K. O. Stanley and R. Miikkulainen. Evolving Neural Networks through Augmenting Topologies. *Evolutionary Computation*, 10(2):99–127, 06 2002.

[25] J. Schmidhuber, D. Wierstra, M. Gagliolo, and F. Gomez. Training Recurrent Networks by Evolino. *Neural Computation*, 19(3):757–779, 03 2007.

[26] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

[27] S. Imai, S. Kawai, and H. Nobuhara. Stepwise pathnet: a layer-by-layer knowledge-selection-based transfer learning algorithm. *Scientific Reports*, 10(1):1–14, 2020.

[28] S. Anwar, K. Hwang, and W. Sung. Structured pruning of deep convolutional neural networks. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 13(3):1–18, 2017.

[29] C. L. Baker, R. Saxe, and J. B. Tenenbaum. Action understanding as inverse planning. *Cognition*, 113(3):329–349, 2009.

[30] M. Chen, S. Nikolaidis, H. Soh, D. Hsu, and S. Srinivasa. Trust-aware decision making for human-robot collaboration: Model learning and planning. *ACM Transactions on Human-Robot Interaction (THRI)*, 9(2):1–23, 2020.

[31] P. Carreno-Medrano, S. L. Smith, and D. Kulic. Joint estimation of expertise and reward preferences from human demonstrations. *arXiv preprint arXiv:2011.04118*, 2020.

[32] L. Peternel, N. Tsagarakis, D. Caldwell, and A. Ajoudani. Adaptation of robot physical behaviour to human fatigue in human-robot co-manipulation. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 489–494. IEEE, 2016.

[33] S. Nikolaidis, D. Hsu, and S. Srinivasa. Human-robot mutual adaptation in collaborative tasks: Models and experiments. *The International Journal of Robotics Research*, 36(5-7):618–634, 2017.

[34] D. Fridovich-Keil, A. Bajcsy, J. F. Fisac, S. L. Herbert, S. Wang, A. D. Dragan, and C. J. Tomlin. Confidence-aware motion prediction for real-time collision avoidance. *The International Journal of Robotics Research*, 39(2-3):250–265, 2020.

[35] D. Vogt, S. Stepputtis, S. Grehl, B. Jung, and H. Ben Amor. A system for learning continuous human-robot interactions from human-human demonstrations. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2882–2889, 2017.

[36] B. Wang, E. Adeli, H.-k. Chiu, D.-A. Huang, and J. C. Niebles. Imitation learning for human pose prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7124–7133, 2019.

[37] M. Carroll, R. Shah, M. K. Ho, T. Griffiths, S. Seshia, P. Abbeel, and A. Dragan. On the utility of learning about humans for human-AI coordination. *Advances in neural information processing systems*, 32, 2019.

[38] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[39] A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000.

[40] R. M. French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[41] Anonymous. Details omitted for double-blind review. 2021.

[42] A. Mandelbaum and D. Weinshall. Distance-based confidence score for neural network classifiers. *arXiv preprint arXiv:1709.09844*, 2017.