EXPLORING GENERALISABLE MULTI-TASK REINFORCEMENT LEARNING AGENTS USING TASK SIMILARITY METRICS

A thesis submitted to The University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Science and Engineering

2022

Jonathan Crawford

Department of Computer Science

Contents

Word Count: 43122

List of Tables

List of Figures

List of Abbreviations

ACL Automatic Curriculum Learning

cos Cosine Similarity

DQN Deep Q-Networks

GAN Generative Adversarial Network

KL Kullback–Leibler Divergence

MAML Model Agnostic Meta-Learning

MDP Markov Decision Process

Meta-RL Meta-Reinforcement Learning

MSE Mean Squared Error

MTL Multi-Task Learning

MTRL Multi-Task Reinforcement Learning

POMDP Partially Observable MDP

PPO Proximal Policy Optimisation

RBM Restricted Boltzmann Machine

RL Reinforcement Learning

RNN Recurrent Neural Network

SGD Stochastic Gradient Descent

TL Transfer Learning

TRPO Trust-Region Policy Optimisation

Abstract

EXPLORING GENERALISABLE MULTI-TASK REINFORCEMENT LEARNING AGENTS USING TASK SIMILARITY METRICS
Jonathan Crawford
A thesis submitted to The University of Manchester
for the degree of Doctor of Philosophy, 2022

One of the many issues that has limited the usage of reinforcement learning (RL) in its application to real-world problems, such as robotics, has been its inability to train agents to both learn optimally and adapt to slight changes in the tasks interacted with—this can be described as the agents ability to *generalise* to new experiences. Many of the current state-of-the-art RL methods focus on performing optimally in a single-task setting and therefore suffer from a lack of generalisability, thus limiting their usage in a real-world context. There has been efforts to tackle this using multi-task learning (MTL) approaches, however within RL this has generally focused on specific and limited task adaptations. This results in agents that are unable to generalise to multi-task environments involving problems that are diverse in their properties including rewards, goals, transition dynamics, etc. Therefore, this work focuses on the following aspects: (a) providing a model- and task-agnostic definition of generalisation in RL, which expands on the current definitions that focus on a single-task case [PGK+18] to provide a metric that quantitatively judges an agents ability to deal with adaptations in a set of tasks, (b) analyse how the current state-of-the-art methods attempt to remedy the challenges presented by a MTL setting through the understanding of the similarity of tasks, and, (c) proposes a new method that leverages task similarity to form a curriculum that encourages an agent to generalise effectively in an environment containing tasks with diverse and adaptive characteristics. Across all of this work, we find the importance of task similarity with regard to investigating generalisability, forming adaptive curricula to encourage task relevance that results in greater agent generalisability whilst also understanding the limits of current methods to transfer knowledge across tasks.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

Firstly, I would like to express my thanks to my supervisory team. In particular, my thanks goes to Ke Chen who has offered me invaluable guidance across all aspects of my PhD. It has been an absolute pleasure to have worked with you.

Throughout the course of my PhD, I have met many amazing people, both within academia and outside, that have made the entire experience all the more enjoyable.

Thank you to Dom, Danny, Cameron, Georgiana, Andrew, Alessio and Will for giving me inspiration and enthusiasm. My thanks also goes to Mike for always lending an ear to listen.

I would also like to thank Paul and Adam for their continued friendship, and I look forward to many more adventures to come.

Thank you to the Engineering and Physical Sciences Research Council for funding my research through the Centre for Doctoral Training Grant [EP/1038099/1].

Finally, I would like to say thank you to my family, of which I can't express my gratitude for the support, kindness and encouragement given over such a turbulent period of time.

Chapter 1

Introduction

Learning across several different tasks often presents itself with many different challenges. These can generally be summarised by the following aspects: *what* to share and *how* to share. Once an understanding of the requirements of these is gained, they can be defined for a specific problem and the learning from one task can be shared across to another.

A machine learning paradigm that has found challenges in this regard in the area of dealing with environments with adaptive or multiple tasks is that of reinforcement learning (RL). The issue of a lack of understanding of how to share learning across tasks in an RL problem, an area of machine learning applied to the problem of temporally delayed cumulative rewards, is one of the major limiting factors to its application to a larger domain of real-world problems, such as robotics. Many of the current RL techniques suffer from the inability to share learning across tasks even with minor tasks dissimilarities, such as changes in goal and transition dynamics (the way in which the agent moves in the environment) [CKH+18]. This issue can be summarised as a lack of generalisation to changes from the tasks exposed during training. In order to provide clarity for the rest of the thesis we define generalisation as the following:

Definition 1.1 (Generalisability)**.** The degree to which an agent is able to share useful knowledge across a set of inter-related tasks in a manner that maintains a level of performance.

One of the techniques used to combat the issue generalisation in RL is multi-task learning (MTL) [Car98]; a framework for coordinating the sharing of learnt knowledge across several different tasks. However, although this provides a framework for facilitating sharing, the *how* with regard to sharing, the main challenge is deciding *what* knowledge is specific to a task, to optimise per-task performance, along with *what* can be shared, to maximise overall generalisability, as the decision highly limits the tasks that a method can perform effectively on and the knowledge that can be shared across.

The issues in RL with sharing learning across tasks has resulted in an increased focus on generalisation and the understanding of what a RL agent is learning. This thesis is concerned with providing a further understanding of the current limitations of RL with regard to generalisation, detailing methods of analysing generalisation in a task- and method-agnostic manner, and using techniques to remedy the issue of generalisation by formulating an idea of *what* knowledge should be shared.

1.1 Motivation

There has been recent work on defining and quantifying generalisation in RL, however there is still a lack of understanding on how to make these methods applicable to settings including adaptive and diverse multi-task distributions [PGK+18, CKH+18]. This is an important challenge to overcome due to generalisation limiting the impact of RL algorithms to real-world problems, and instead confining it to more theoretical applications such as games [MKS+13]. In this thesis we focus on leveraging an understanding of task similarity, which has been shown to be essential in defining the scope of agent learning required across tasks [CS05], to provide a method for quantifying generalisation regardless of learning method and task distribution diversity, and facilitate a discussion on the current limitations of RL methods in this area.

Furthermore, due to the lack of useful definitions of generalisability, and approaches to quantify it, there is also limited work in current research to understand how we can improve generalisation in RL agents. An agent that does not have knowledge, whether implicit or explicitly, of how the tasks it interacts with relate to each other will not be able to modify its representation so that it can perform across those tasks and therefore generalise (i.e. it will not understand what features it needs to extract in order to generalise). Therefore, understanding the mechanisms for the gathering of these similarities, and how we can apply them, provides a vital aspect of how we can remedy the lack of generalisability in RL agents when learning on environments containing diverse task distributions.

1.2 Research Questions

This thesis asks the following important question: *How does task similarity affect the generalisability of reinforcement learning agents?*

This question covers a broad array of possible aspects, and therefore is difficult to provide a concise answer to. As such, we provide the following concise questions that can be meaningfully answered within this thesis to allow us to contribute to the current literature on the wider question. Overall, this thesis aims to answer the following specific questions:

- What are the mechanisms currently used to compare the similarity of RL tasks? In particular, what types of task differences and to what degree of adaptation do these mechanisms facilitate knowledge sharing across?
- It has been shown that current RL methods lack generalisability when learning across diverse task distributions. Can an understanding of task similarity be useful in quantifying generalisation to further describe these limitations? What is the impact of task diversity, including adaptation in their properties, on current methods?
- How can we leverage metrics that describe the similarity of tasks an agent interacts with in order to increase the generalisability of a RL agent when learning on a diverse task distribution?

1.3 Contributions

As mentioned in the motivations, current research is lacking in providing an understanding of how to quantify generalisability in RL, in order to gain understanding of the current limitations in methods to further indicate their reasons, along with lacking approaches that aim to encourage generalisation in environments that contain tasks distributions with diverse differences. We provide more detailed summaries of the contributions to each of the research questions mentioned above in the relevant chapters, however the contributions can be described briefly as the following:

- We provide a taxonomy of task similarity metrics which focuses on the source of the information that the task representation is formed from along with the level of abstraction of which the overall metric belongs within. We provide a literature review of current methods within literature, primarily focusing on the application of multi-task and meta-learning applied to reinforcement learning, that make use of task similarity metrics as a mechanisms, whether implicitly or explicitly, to encourage higher performance on environment's containing adaptive and/or different tasks.

- We provide a definition of generalisation for reinforcement learning which makes use of task similarity metrics to quantify the relationship between performance and inter-task relationships to define the generalisability of an agent's task representation with regard to an environment's adaptation. This highlights the importance of task similarity in understanding the task adaptations that an agent is able to generalise across, thus outlining the benefits and drawbacks of a given method.

- We state the considerations that need to be taken to understand how to separate assessing optimal performance, compared with understanding whether a model is able to generalise, and in what instance.

- We conduct an analysis of the effect on both generalisation and performance of a non-uniform task sampling distribution on a gradient-based meta-learning method on a multi-task environment. This presents a combination of gradient-based task similarity metrics to form the non-uniform task sampling distribution, which indicated effective separation of tasks based on the gradient updates of a meta-learning agent.

- We detail an extension to a standard gradient-based meta-learning algorithm, which is also applicable to further gradient-based methods, that uses a gradient-based task similarity metric to encourage curriculum learning through the adaption of a non-uniform task sampling distribution leveraging different metrics about the gradients, focusing on a multi-task meta-reinforcement learning environment.

The contributions mentioned here are the subject matters presented in Chapters 3, 4 and 5.

1.4 Thesis Outline

The remainder of this thesis is organised as follows:

- Chapter 2 provides a background of the topics related to the research details in this thesis, including RL and two of the current methodologies that provide frameworks for the learning of shared knowledge from across a set of tasks, multi-task learning and meta-learning.

- Chapter 3 examines the way that task similarity is used in current RL methods through the introduction of a taxonomy of task representations and their associated task similarity measure, summarised as a task similarity metric, with regard to their source of information and the level of abstraction of their task representation in order to detail how the current methods approach improving generalisation in RL agents.

- Chapter 4 details a definition of generalisation in RL with regard to task similarity in environments containing diverse task distributions, providing a process for analysing both multi-task and meta-learning methods' generalisability, along with providing a discussion of the considerations needed in order to mitigate bias.

- Chapter 5 provides an investigation into whether gradients are a useful task representation in a task similarity metric consisting of two different measures of similarity within this space, and presents a mechanism for using task similarity within a gradient-based meta-reinforcement learning method with the aim of encouraging generalisation through an adaptive task curriculum.

- Chapter 6 contains the thesis' conclusion and details possible directions for future work.

Along with these chapters, the thesis contains three appendices. In Appendix A we present the environments used within the research with regard to their specific details. Both Chapters 4 and 5 have an associated appendix (Appendices B and C) containing supplementary material including experiment hyperparameters and further results—these were added in the appendix as, although they provide further evidence to support the topics presented in this thesis, they would distract from core results presented in the main text of the chapters.

Chapter 2

Background: Reinforcement Learning and The Approaches To Combine Learning

This chapter provides a background on the topics discussed throughout this thesis: firstly, we introduce reinforcement learning which details the machine learning paradigm that this thesis focuses on, in particular mentioning the issues with the paradigm in relation to generalisation; secondly, we introduce approaches that look at facilitating knowledge sharing across task distribution adaptations, including those methods within reinforcement learning itself, multi-task learning and meta-learning.

2.1 Reinforcement Learning

Reinforcement Learning (RL) defines a machine learning framework for agents to learn how to behave through the interaction with an environment, in particular dealing with temporally-delayed tasks. This framework is modelled as a loop, as shown in Figure 2.1, that includes the agent receiving observations, based off the agents current state, and a reward from the environment which the agent then uses to select an action that maximises its future reward. This process of interacting with the environment that a task is within enables the agent to form an optimal policy, a mapping of states to actions, and its associated value function, a representation of the value of a state. This loop is formally modelled as a *Markov Decision Process* (MDP). MDPs are defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$, with state space $\mathcal{S}$, associated action space $\mathcal{A}$, transition dynamics $\mathcal{T}$, reward function $\mathcal{R}$, and the discount on future rewards γ. Through the agents interaction with the MDP, it learns a mapping from states to actions that maximises the expected discounted sum of the rewards. A task is defined by this MDP, describing the goal, detailed as the rewards, of the agent's interaction with the environment's states and the transitions between them.

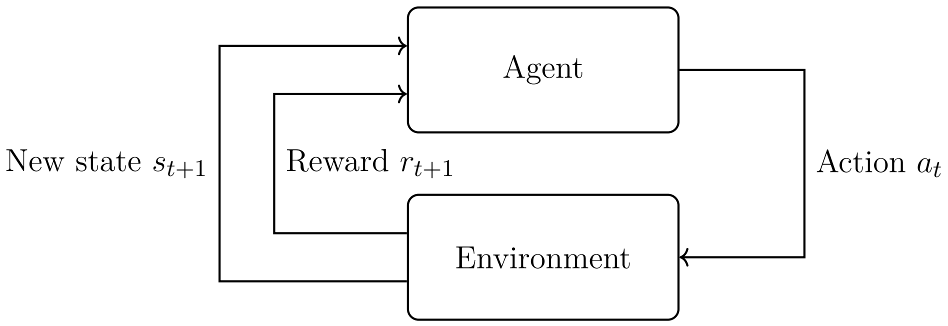


Figure 2.1: Reinforcement learning cycle [SB18].

RL approaches are generally grouped into either model-free or model-based approaches: model-based approaches attempt to learn a model of the environment and use that model's predictions of the next state and reward to calculate optimal actions, whereas model-free approaches learn a policy based on the experiences of interaction with the environment without perfect knowledge of the environments dynamics. Within model-free approaches, we then generally have the following approaches: policy-based methods, which directly learn a policy (a mapping of $\pi : s \rightarrow a$) using the policy gradient method, such as the Proximal Policy Optimisation (PPO) [SWD$^+$17] and Trust-Region Policy Optimisation (TRPO) [SLA$^+$15] approaches; and, value-based methods which instead learn a value function for the trajectories in the MDP and use this value to choose the optimal action, such as the Deep Q-Networks (DQN) method that use deep learning within the Q-learning method introduced in work by [MKS$^+$13]—this work focuses on learning the Q-value, otherwise known as the action-value function, which defines the expected return starting at a state and taking a specific action then following the policy until the end of the episode. Policy-based methods have been found to have better convergence to optimal behaviour properties than value-based learning due to directly optimising for the policy, however value-based approaches have been shown to be more sample efficient. Most current methods are placed either in one of these approaches, however there are also hybrid approaches such as actor-critic methods [KT00] that use information from both value-based and policy-based methods to reduce bias innate within the policy gradient method's gradients estimates.

2.1.1 Reinforcement Learning Adaptation

RL has been classically unable to adapt to changes across the tasks that they have been trained on. These changes are generally seen as adaptations, or differences, in the properties of the MDP (i.e. reward structure, observation space, etc.) that has caused for an agent to be in situations that are classified as out-of-policy from the interactions that have been learnt from during training—however these modifications to the MDP can also occur during training which can cause the agent to not learn optimal behaviour across tasks and their differences. These adaptations can generally be described as a different task that contain modifications of the MDP properties as outlined in the MDP description above. Overall, the differences across tasks define the overall task distribution that the agent is learning on. As such, we define whether a task distribution is diverse by the degree of similarity in the properties of the tasks within the distribution.

There has been recent work in understanding the reasons for the lack of generalisability in RL. In particular, it has become commonly accepted that the learnt model's lack of generalisability, to both slightly and highly dissimilar tasks, is due to extreme overfitting [CKH$^+$18, TFR$^+$17]—this describes the case of learning task specific features in a manner that the learnt representation is not shareable. RL has shown to overfit to many aspects of a task including the observation space [SJT$^+$20]. We generally see that agents are able to memorise an optimal path in a task but not extrapolate learning that can be shared across different forms of task adaptations [FTF$^+$20]—this difference in understanding the memory systems required for optimal performance, especially with regard to generalisation, is a vital area of research that is currently lacking in current RL literature. Some of the issues that cause a lack of generalisability are summarised as the following: an agent is trained, generally with algorithms designed around MDPs, however this formulation innately trains the agent with the view that the environment it is trained on gives complete information about the environments that it may encounter in the future, which is an incorrect assumption—this is an aspect discussed by [GRK$^+$21]. The adaptation in RL is generally considered with regards to the continual learning case, where an agent needs to continually adapt to new experiences as defined within the MDP—overall, the core issues are when we modify the MDP too far outside of policy the performance highly degrades. The main approaches used in this thesis are model-free, as it is a section of RL algorithms that have been found to be highly affected by overfitting, and therefore has especially suffered in environments that include tasks where there MDP is adapted to create a new task [ZVMB18].

A large step forward for the paradigm in addressing the issue of RL methods dealing with adaptation within the task distributions was with the introduction of deep learning [GBC16], a machine learning method that uses a hierarchical approach of composing representations in increasing complexity and abstraction to form complex function between an input and output. One of the initial approaches for this was in being used as the Q-value model representations instead of q-tables [MKS$^+$13], whilst many of the current techniques now make use of deep learning for function approximation. This has enabled for an agent to have some form of generalisability outside of experiences encountered during training. However, the inclusion of deep learning in RL has meant that the generalisability issues classically found within deep learning approaches [HHS17], have also needed to be considered with regard to their application to RL problems. There has been work on understanding generalisation in deep learning approaches with regard to supervised learning, such as work by [VPL20] who look at the bounds on generalisation performance. There has also been research into remedying the issues present within the method, such as work by [PZG17] who look at reducing the catastrophic forgetting that deep learning approaches are prone to. However, the particular usage of deep learning techniques in RL, as function approximation of functions that are constantly dealing with domain shift depending on the agent's current environment interaction, means that we must understand generalisation for the particular RL use cases—whereas currently, most of the current literature in deep learning with regards to generalisation look at problems involving classification.

2.1.1.1 Adaptive Environments

Along with the methods within RL focused on addressing the issue of generalisation, another vital aspect is having the environments available for the testing and analysing of these methods. Many of the classic RL benchmark environments focused on non-adaptive tasks (i.e. task distributions with tasks containing the same MDP), such as Arcade Learning Environment (ALE) [BNVB13], whereas recent work has looked at addressing this such as the ProcGen [CHHS19] which includes several games with procedurally generated levels that adapt several of the MDP properties and form a task distribution containing several variants of the same task. This shift in focus is vital for understanding the current limitations of methods in RL with regard to generalisability. We discuss this further in Section 4.2, where we provide more detail on the relevance of adaptive multi-task environments to the issue of generalisation in RL.

2.2 Learning Across Multiple Tasks

The standard RL formulation can be seen as a lifelong learning [KSCP18] approach, which should be able to continually adapt to new experiences in a given environment. However, due to the difficulties that RL has had in dealing with adaptations in the task environments as defined by the task's MDP, as outlined in Section 2.1.1, there has been research into how the paradigm can encourage the learning of several tasks within the framework itself, along with using techniques used in other machine learning paradigms.

In this section we look at the relevant approaches that have been used to encourage RL agents to be able to learn across diverse task distributions. These approaches can be summarised as the following: approaches that make modifications to the core RL formulation, as detailed in Section2.1, to allow for the agent to learn an adaptive representation of the task (e.g. a policy that is able to adapt); multi-task learning approaches focused on learning multiple tasks at once; and finally, meta-learning approaches dealing with the learning of an agent that is able to adapt quickly to new related but unseen tasks.

2.2.1 Reinforcement Learning Approach

Within RL, to remedy the issue of overfitting to specific tasks, we have seen various approaches that are generally characterised as one of the following: a modification of the RL problem formulation, or an alteration to the environment that the agent interacts with and is then used in the standard RL formulation.

Although many of the methods within RL make use of the MDP formulation, several methods have attempted to remedy the issues of adaptation by modifying this formulation to encourage greater sharing of knowledge across particular adaptations—this is commonly done by factoring out particular parts of the task representation to extract certain features that relate to certain forms of task adaptation. An example of this is the use of successor features [BDM$^+$17, MWB18, KSGG16] which decouples the dynamics from rewards, resulting in both a reward prediction model and a successor map that models the expected future occupancy from a given state, in the task representations. This is done by using the discounted features of state-actions pairs, in order to generate a policy from the features rather than the specific moves. This approach has been shown to result in an agent that allows for changes in reward functions in the task distribution, but depends on the same transition dynamics function. This method was then extended to model features [LL18] which further separates the representation to allow for adaptation in both transition and reward functions but requires behavioural equivalence in observation states. There has also been work in options, a closed-loop policy for taking an action over a number of epochs [KB07, PP$^+$99, SPS99], which also deals with changes in reward structure but requires the rest of the MDP to be consistent. Work by [GRK$^+$21] attempt to formulate the problem of learning across different tasks as a partially observable MDP (POMDP), which causes the agent to have uncertainty in the tasks that it may see in future interactions providing a bias to learning that mitigates overfitting—this is in comparison where the interactions in a standard MDP assumes complete information from the interaction. The decoupling of the representation space of the dynamics and reward functions has shown to increase transfer in dynamics and reward changes [ZSP18].

Another approach for encouraging performance in environments containing adaptive tasks is that of directly modifying the task that the agent is exposed to (i.e modifying the properties of the MDP). This can be done by decomposing the environment into fragments that form the task—for example, work by [FD02] look at decomposing tasks into MDPs that contain multiple goals, which allows for hierarchical transfer of related sub-goals. This is related to work on changing the representation of rewards (reward shaping) to learn intermediate rewards, which has shown that learning of novel tasks can be improved [KB06]. By extracting certain aspects of an environments representation we can reduce the amount of extraneous data that we are using to learn, and thus reduce overfitting. As such, another approach to increase generalisation is to remove task-specific extraneous features and noise from the environment in order to unify the environments. As show by [HPR$^+$17], by creating a disentangled representation of the observations we can improve the transfer of learning across tasks, in particular focusing on observation adaptations. Converse to this, another approach is that of data augmentation, where random noise is added to the results of the agents interaction with the environment to reduce the signal of task-specific features. This has been conducted through both the domain randomisation of the observation space [LLSL20] and by combining the observations of multiple training environments through a regularisation term [WKSF20], thus artificially adding adaptation to the environment and reducing the overfitting of agent's to a particular task. Overall, this illustrates that an issue with current techniques is how the representation of the environment affects the learning of a policy, currently causing overfitting to particular properties of the MDP.

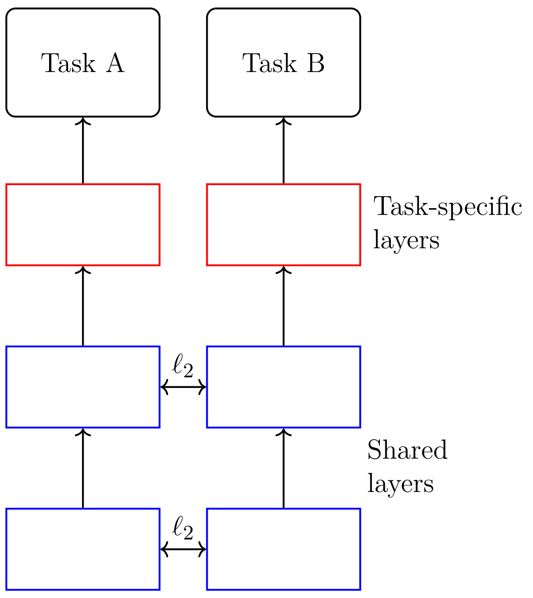
In summary, the paradigm of RL contains different approaches for dealing with a task distribution that deals with many forms of task diversity and adaptation. However, there still lacks methods for a generalised approach to performing optimally across several forms of task adaptation simultaneously. For example, successor features are able to manage with adaptations in reward structure, due to the decoupling of reward and transition dynamics, but struggle with adaptations in the observation space in particular over large adaptations in the task distribution. In general, the adaptations evident in solutions primarily looking at modifying the RL formulation, focus on decoupling certain aspects of the task representation, but suffer with finding abstract task differences outside of the MDP defined representation (i.e. the extrapolation of ideas) as discussed in Section 2.1.1. This results in methods that are limited to the types of environment they can be perform effectively on, and thus results in the reduction in possible applications for the RL framework.

2.2.2 Multi-Task Learning

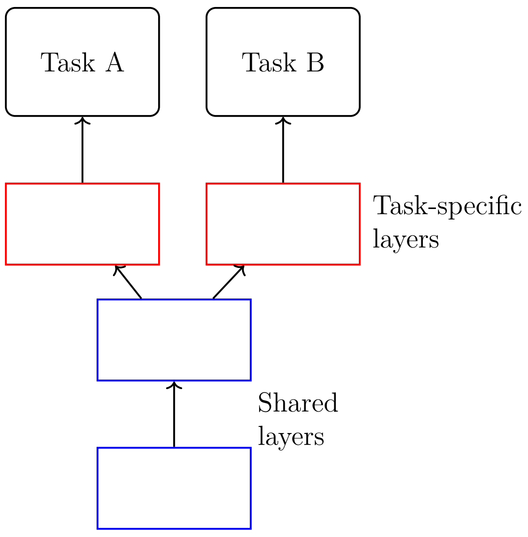
Multi-task learning (MTL) is a human-inspired approach for the sharing of knowledge between multiple tasks in order to improve overall learning effectiveness [Car98]. For a multi-tasking agent their overall aim is to solve a fixed set of tasks optimally by sharing the related knowledge of tasks to positively aid in the learning of higher-order representations [ZY17]—this is analogous to learning across a set of different sports that each share similar movements. This paradigm compares to other approaches incorporating the learning of multiple tasks, including transfer learning TL [TS09] that focuses on transferring knowledge from a source task to a target task whereas MTL aims to improve the learning across all tasks—there is however a convergence between these two in the asymmetric MTL case which defines the transfer learning case. Overall, MTL aims to share task-specific understanding from across the training signals of inter-related tasks in order to increase the performance, compared to single-task performance, and generalisability of an agent on a set of tasks.

This paradigm aims to learn across several different tasks and benefit from this through regularisation of either shared parameters or the diversity in learnt representations [ZY17]. As such, the main approaches to MTL, considering in particular deep learning approaches due to the application to modern RL approaches, can either be categorised into either feature-based or parameter-based. The feature-based approaches attempt to learn a shared feature representation that is constructed from common features across the tasks, whereas parameter-based approaches use the model's parameters relating to a task in order to aid the learning of parameters in another task—this is done through cross-model regularisation (e.g. ℓ_2 norm) to bias the learning process. The focus of research in this thesis is within the application of deep learning approaches within MTL applied to RL. Within MTL using deep learning there are two main approaches that exist: soft parameter sharing and hard parameter sharing [Rud17, Cra20]—these are both illustrated in Figure 2.2. In hard parameter sharing, the hidden layers are shared between tasks with task specific output layers, which greatly reduces overfitting, but generally fails when trained on dissimilar tasks due to destructive learning—for MTL to be successful the tasks must be related within some task representation in order to facilitate the joining of learning across tasks, however with greater diversity in tasks, and more tasks to learn across, the representation must facilitate sharing in order to not negatively impact performance [MPRP16]. In soft parameter sharing, each task has its own model, but network layers are cross-stitched in order to share learning—this greatly reduces the impact of negative transfer due to having separate task representations, however is highly reliant on an appropriate regularisation term to share appropriate features.

Building upon the approaches within MTL in its general application to machine learning paradigms, when we consider it's application to RL in particular we describe this as multi-task RL (MTRL). MTRL attempts a learn a single



(a) Soft Parameter Sharing. Using a regulariser (e.g. ℓ_2) between task specific models' encourage similar parameters.



(b) Hard Parameter Sharing. Using the same model parameters across all tasks with task specific output layers.

Figure 2.2: Multi-Task Learning (MTL) deep learning sharing mechanisms.

policy $\pi(a|s)$, which is generally conditioned on a task identifier. The aim is to maximise the mean expected discounted return across all of the tasks within the task distribution $p(\mathcal{T})$, given by $\mathbb{E}_{\mathcal{T}\sim p(\mathcal{T})}[\mathbb{E}_{\pi}[\sum_{t=0}^{T}\gamma^t R_t(s_t, a_t)]]$ where $R_t(s_t, a_t)$ is the reward function for a given action a in a state t within an collection of trajectories T sampled from a task $\mathcal{T}$. In this case there is no explicit distinction between training and testing with the aim to learn optimally across a single set of tasks without consideration of sharing to new tasks, however the curriculum used during training will influence learning—this is discussed in Chapter 5.

There are several challenges found within MTL, in particular related to understanding *which* tasks, and *when*, to share across due to the effectiveness of MTL being highly dependent on being able to identify related features across tasks, and how to effectively build upon learning. Several works have been introduced in order to apply MTL to RL [VVM20] that include: policy distillation (network compression) approaches [TBC$^+$17, AKKL18, PBS15], which use a comparative loss function in order to learn the difference between the task representations in order to update a joint compressed representation; hierarchical approaches [SXS17, SOL19, OSLK17], which decompose the task into hierarchical structures consisting of sub-tasks, defined by their relation to parent and child nodes depending on their level of abstraction, that can be shared across tasks; and, shared task representations [HPR$^+$17] approaches, which learn a representation composed of sampled features of the tasks in order extrapolate shared task understanding. Across all of the these approaches we find that, just as with the standard RL algorithms, they depend on the amount of diversity within the task distribution, however, in general, they don't focus on specific types of adaptation. Although the application of MTRL approaches have been shown to be effective there remains the tendency to train dedicated networks for each task remains when there is a need to attain single-task performance. A large reason for this is the noise that is added when learning across tasks due to learnt knowledge not being shared effectively across tasks (i.e. a useful task representation is not extracted during training to facilitate large task differences)—this can lead to instances of negative transfer, where a task negatively affects the overall performance that can be attained on a new task. As such, attempts at quantifying the types and form of adaptations has been made. An example of this is task mappings which form a mapping between each task's learnt knowledge [TS09] due to the high dependence on the level of domain shift (the degree of adaptation) in certain approaches—we discuss this issue, encapsulated in task similarity metrics, in Chapter 3.

Overall, the application of MTL learning approaches in RL have attempted to remedy the limited ability of RL to transfer across a broad spectrum of task distribution adaptations and diversity. However, these approaches make the assumption that the tasks within the training set are related and therefore beneficial to be learnt together [KGS11]. We see that within these approaches, the ability for an agent to learn an shared optimal representation across tasks is highly dependent on the types of tasks that they are attempting to solve—there is yet to be a method that produces an agent that is agnostic to the tasks that they are trained on and is able to perform over diverse task representation spaces, resulting in the requirement of having an understanding of task similarity to validate whether they will be effective in the environment.

2.2.3 Meta-Learning

Meta-learning is an approach based on the idea of 'learning-to-learn' [TP98] which aims to optimise an agent to adapt quickly to new, but related, tasks using a small support set of data—this is similar to zero-shot learning approaches where the aim is to perform well during the testing of an agent on tasks not seen during training. Meta-learning is primarily focused on learning a representation that is able to optimally adapt to new tasks as fast as possible, and is a multi-level optimisation problem which includes a meta-objective, such as learning the inter-relatedness of tasks. This paradigm has been shown to be useful in remedying the issues found within conventional machine learning paradigms including: the information bottleneck, the requirement of many approaches to require large amounts of data to perform well; and, generalisation [HAMS20].

Meta-learning approaches are generally defined as either metric-based, model-based or gradient(optimisation)-based [Van18]. Metric-based approaches aim to learn a distance function over input data, e.g. Siamese networks [BGP21] and relation networks [SYZ$^+$18]. These approaches are aligned with the idea of metric learning [KB19] that focuses on learning a representation function that maps from the input into an embedding space which includes a similarity function that preserves similarity whilst increasing separation in the embedding space. Model-based approaches use a model specifically designed for the fast updating of model parameters with few training steps. An example of this is the RL^2 [WKNT$^+$16] method that uses recurrent neural networks (RNNs), a type of neural network that deals with sequential data, to input past interactions into the network in order to gain an understanding of, and map between, the dynamics between states, action and rewards, and thus increase adaptability. Optimisation-based approaches use backpropagation, a technique for updating the parameters of the neural network, based on a loss function on the output of the network, in a manner that allows for the model's parameters to be adjusted to facilitate faster learning from small sets of data. Some of the most well-known gradient-based approaches include Model-Agnostic Meta Learning (MAML) [FAL17] and Reptile [NAS18] which both work by sampling trajectories from a small support set of tasks from the training task set, and perform an update to the parameters based on the loss from each of the tasks interactions.

Meta-reinforcement learning Meta-RL aims to use learning from previous training tasks in order to learn a policy $\pi(a|s)$ that can adapt quickly to new previously unseen test tasks drawn from the same task distribution $p(\mathcal{T})$. Approaches within meta-RL are generally focused on remedying the issues found within RL methods themselves, including the exploration vs. exploitation issue, task acquisition. With regard to the balancing of exploration and exploitation, there have been attempts to control for the effect of overfitting to the train distribution in work by [ZSI$^+$19] who incorporate task uncertainty into action selection by conditioning the policy by the posterior distribution over a new task, represented by a variational auto-encoder—an issue with this method, however is that it has a strong dependency on the both the training and test tasks being sampled from the same task distribution. Following this, the issue of task acquisition, deciding which tasks are relevant to a optimise a learners representation, has seen recent, but limited, work in order to tackle the issue of generalisability in RL agents, such as [EGIL18] who look at meta-learning a controller for the selection of appropriate skills. This area has be investigated as many of the current meta-RL methods require for tasks to be sampled from the same distribution, across both train and test phases [GEFL20]—this is the focus of Chapter 5.

Overall, although meta-learning has provided a framework for the learning of representations that are able to adapt to new and unseen tasks, in a RL setting they still heavily depend on an appropriate training task distribution to facilitate the learning of relevant shared features, as further discussed in Chapter 5—an issue that is also found in both RL and MTRL methods.

Chapter 3

Understanding the Similarity of Reinforcement Learning Tasks

3.1 Introduction

Due to the inability of RL methods to learn across certain forms of task adaptations and diversity, as detailed in Section 2.1.1, it is therefore important to understand in what way the tasks we want to learn across are similar. This will allow for an understanding of whether the method of learning being used is appropriate for training an agent in this task distribution, and provide a possible mechanism to facilitate shared learning. Many of the standard RL methods, along with the approaches in MTL and meta-learning, depend on similarity in certain properties of MDPs across tasks in order for knowledge sharing to be performed effectively. However, many of the methods don't use similarity directly in order to aid learning, with regards to learning the degree of similarity and in what regard, and they instead make the assumption that they share properties within the MDP. The learning and use of similarity between tasks is encapsulated in the area of task similarity metrics.

It is important to analyse the ways that task similarity can be measured and used. In particular, given an environment, we need to form a representation of the environment and the tasks within it that is conducive to being compared to each other. In order for us to properly attain this we require that the task similarity metric, both the task representation and the measure of similarity within in it, to capture the relevant information required by the method to enact the intended behaviour. This is split into two parts, the measure of similarity used within task representation must be valid within the environment (i.e. the task representation must be formed of properties of a task that can cause for a separation in the tasks by a task similarity metric, which is linked to the idea of the *identity of indiscernibles*), and the properties extracted must provide the corresponding information that the agent requires to exhibit the intended behaviour. A task similarity metric that follows these principles can therefore allow methods to leverage whether and what part of a task is useful for being used to encourage positive performance in an agent on a given environment.

Within current literature there are several different methods for forming a task similarity metric within different RL settings, each using different task representations and measures of similarity in that space. All of these methods can be grouped together by what information they use, how they perform the comparison, and where they can be applied. Many of the current usages of task similarity metrics are found within the transfer learning setting, due to the current approaches requiring to learn amongst similar tasks. There have been several attempts at forming taxonomies to facilitate the discussion of the applicability of current methods in literature including that of [VGF21] who propose a taxonomy based on similarity measures formed directly from the MDPs in a RL setting. One of the main issues of this taxonomy is that it solely focuses on the direct use of the MDPs, whereas [CS05, FPP04] have shown that this results in issues resulting in the methods being only applicable to certain forms of transfer, thus limiting the applicability of the method. This highlights the importance of understanding the relevance of the task representation, in which has been investigated in other machine learning paradigms [DR19], to its applicability of facilitating the sharing of knowledge across different forms of task distribution adaptation and diversity. This task representation then dictates the forms of similarity measures we can use, and the types of separation we see between tasks, thus impacting the types of tasks that should be learnt together [SZC$^+$20], and the way they can be used [TS11]. As such, it is vital to understand what task representations and similarity measures are currently used in literature with the overall aim of understanding whether and how we can form a metric that effectively produces task similarity across as many types of MDP adaptations as possible.

The contributions in this chapter are as follows:

- We provide a taxonomy of task similarity metrics, comprising both the source of information for their task representations and the measure of similarity in this space, with regard to the level of abstraction that they are defined in.

- By using the taxonomy we provide a review of the current approaches in RL which use a measure of task similarity metric to enhance the ability of the agent to sharing knowledge across tasks. We show that many of the current approaches focus on using low-level environment-based task similarities, which limit the overall applicability of the task similarity to specific types of task changes, as defined by the task representation, limited in their diversity.

The remainder of the chapter is organised as follows. Section 3.2 provides a taxonomy of task similarity metrics with regard their level of abstraction and the source of information for used in the task representation, where we also provide a comprehensive list of the methods in the current literature grouped within the relevant categories as defined using the task similarity taxonomy. In Section 3.3 we provide a discussion of the overall state of methods within current literature, along with mentioning some of the important aspects of the domain. Finally, Section 3.4 provides some concluding remarks of the chapter.

3.2 Taxonomy

In this section, we provide a taxonomy for classifying the current approaches of task similarity metrics and their usages in RL. This will allow us to see the areas that have been investigated, and the gaps in current literature. In order to frame task similarity metrics for the taxonomy, we provide a definition of task similarity as the following:

Definition 3.1 (Task Similarity Metric)**.** A task similarity metric is defined as the combination of both a task representation r and distance measure d which states the similarity of a collection of tasks. Given three tasks, all defined in the same task representation space r, T_1^r, T_2^r and T_3^r, the distance function $d(T_1^r,T_2^r) \to [0,1]$ allows us to state that if $d(T_1^r,T_2^r) < d(T_1^r,T_3^r)$ then T_2^r is more similar to T_1^r than T_3^r. If $d(T_1^r,T_2^r) = 0$ then T_1^r and T_2^r are the same task.

From this definition we can state that in order to form a similarity metric we require both a consistent *task representation*, r, that all tasks can be defined within, and a function d that calculates the distance between each of the tasks within the shared task representation space—this acts as a *measure of similarity*. The form and type of task representation that all of the tasks within a distribution are defined in will dictate the applicability of the measure of distance between the tasks within that space. A tasks' similarity metric is defined by both the representation that the agent has of the task and the mechanism used to compare tasks within this representation space. As mentioned by [CS05], an optimal task similarity metric should have the following characteristics: (a) should provide approximate value to the degree of learning between tasks (e.g. this could be related to the performance of the agent which can be defined as reward, generalisability, etc.), (b) should provide a partial ordering of tasks, i.e. better to learn closer tasks first, and (c) it should be computable outside of the traditional train/test cycle.

This similarity metric allows us to order the tasks within a representation space by the similarity that they have—a properly formed similarity metric will be symmetric, have the triangle inequality, non-negative and tasks defined within this space that have the same properties as defined by the representation will have a distance of 0.

A description of a task similarity metric is detailed in the following example. In this example we are attempting to learn optimally on an environment consisting of a pendulum, defined by both a length and mass, which an agent acts upon with torque, the action, around the fulcrum to get the pendulum upright—this details the classical pendulum task, further detailed in Appendix A. In this case, an appropriate task representation can be formed from the parameterisation of the function defining the behaviour of the (i.e. the length and mass), in which can be adapted to form a collection of tasks (i.e. the task distribution). An appropriate measure of performance from this example could be the distance from the upright position at the end of the episode.

It has been found that the level of abstraction within a task representation heavily affects the applicability of the task representation to being useful [HSB$^+$19]—the appropriate level of abstraction is heavily dependent on the application—this is one of the fundamental ideas behind hierarchical representations which have found successful application in this regard [SOL19, JGMF19]. The level of abstraction can be described as the amount of higher order structure being learnt, and is often attributed to higher-order learning and memory systems (i.e. memorisation, extrapolation, etc.) being encoded, such as causal understanding [KSB17], or forming analogies between visual features [SWT18] through the extraction of underlying structure from the task representation.

In the context of the example, with a task distribution that includes various pendulum lengths and masses, the pendulum's transition dynamics will alter across task MDPs. This will therefore result in the agent requiring to learn how the torque affects movement irrespective of the pendulums mass and length—this can be called an extrapolation of higher order understanding. This is in comparison with instances of task distributions with the same length and mass, where a memorisation of specific torque at certain states would reproduce optimal performance—an example of lower-order task understanding which is not generalisable.

		Experience Origin	
		Environment-based	Behaviour-based
Level of Abstraction	High-level	• Underlying structure of MDP properties	• Underlying structure of agent behaviour
	Low-level	• MDP properties (e.g. observations)	• Gradients • Policy • Q-Values

Figure 3.1: Task similarity metric taxonomy.

Therefore, due to the importance of understanding the application of a task similarity metric, the taxonomy that we propose in order to group the task similarity metrics and their usages, as provided in Figure 3.1, focuses on both the source of information that is used to form the task representation and the levels of abstraction that exist in the representation. This is as they have been found to be innately linked to the level of generalisability that we can encourage in the learner [HSB+19]. As the applicability of the task similarity metric, in particular the task representation itself, is highly dependent on the level of abstraction of the task representation, we see this is an important factor to be mentioned when detailing the task similarity metric. We define the level of abstraction as the degree of data extrapolated from the learners interaction with the environment. The intention of the level of abstraction is to represent the different forms of similarity including the forming of analogies between tasks or agents, which facilitate classically harder forms of generalisation across more types of task difference and diverse task distributions [FLBPP18b] due to the higher-order forms of memory systems leveraged.

For each of the levels of abstraction, the taxonomy divides the methods into two different groupings representing different trends in current methodologies: environment-based task similarity metrics, which use a task representation and similarity measure focusing of the task structure, and behaviour-based task similarity metrics which use a similarity measure within a representation formed from the agents representation of optimal behaviour learnt from interacting with the environment. The representation space defines the particular properties of the tasks that we are focusing on (i.e. the task representation may be focusing on a particular property of an MDP). There are instances where a methods' task similarity metric can be defined as both environment-based and behaviour-based. In this case, we assert that it is a behaviour-based approach as this will define the applicability of the metric to certain scenarios as we discuss in Section 3.3.2.

In context of the example provided above, we can describe the different levels of abstractions by what and how we utilise certain forms of information from the environment and the learner itself. A low-level task representation that has already been mentioned is that of the task's parameterisation values. This is stated as low-level as it is information gathered directly from the environment that doesn't describe any underlying structure of the task. In order to gain higher-order task representations we instead need to learn an underlying structure from one or a combination of low-level representations. An example that would be applicable to the pendulum task is the factorisation of the Q-value function, as is done in successor representations [KSGG16], into transition dynamics and reward spaces to learn underlying MDP properties.

All of the methods and their task similarity metric usages outlined in Section 2.1.1 all aim to focus on specific differences in the environment's tasks. This has the consequence that comparing the methods with regards to their performance is not possible, as they each target specific aspects of the environment with regard to gaining performance in different ways. Instead, our intention here is to provide a detailed survey of the areas that have been looked at, with regard to the types of task adaptation and diversity that have been attempted to generalise across, and the mechanisms that they use to do this.

The following sections provide a detailed description of each part of the taxonomy stated in Figure 3.1, providing a breakdown of the main categories, and how we define the methods of task similarity within each, being a combination of both the task representation and the similarity measure used within. Along with the defining of the taxonomy of task similarity metrics (we also provide methods that include examples of task representations that could be used within task similarity metric), in the following sections we provide a comprehensive list of the current methods in literature that use certain task similarity metrics within a task representation space, placing them within the defined categorisations as detailed in the taxonomy in Section 3.2—in particular, these focus on the methods relevant to the work carried out in Chapters 4 and 5. This includes an intended type of difference in task that they are intending to transfer across. Along with this, we provide additional information on the target of the task similarity metric, for example optimal performance, generalisation, etc, and detail the resultant behaviour with regard to the level of learning (i.e. memorisation, interpolation, extrapolation). Some of these methods are only examples of methods using a unified task representation space, with respect to their position within the taxonomy, and do not include a task similarity metric—these instead present avenues for where task similarity could be used to encourage particular targets.

3.2.1 Behaviour-based

Behaviour-based task similarity metrics use representations of tasks that are formed by encoding properties of an agent's learning process and behaviour within the representation, along with a measure of similarity within this space. This includes representations such as the policy, π, for how an agent behaves in the environment and, with relation to deep learning approaches, how the modelling and updating of these models directly can be used. The policy is defined within this category as, although it is part of the MDP formulation it is a representation that is learnt by the agent for how to interact within the environment rather than a piece of information gained directly for it—as discussed in Chapter 2 there have been many approaches that have attempted to factorise the policy representation to allow for sharing knowledge across certain types of adaptation.

Due to the task similarity being innately linked to both the learning approach and performance of the specific agent, as these are generally learnt as part of the learning process for training an agent, this does limit the applicability of the task similarity metric to other method usages, such as a shared view of task similarity across methods. This is as the properties of the learning process, such as the agent's architecture, mechanism for learning, and how they use performance to guide their learning, will form a task representation that is optimised for the particular method—we discuss this further in Section 3.3.2.

Behaviour-based task similarity metrics provide a means of leveraging the particular learners understanding of how to behave within the environment, which can therefore take advantage of the agent's particular requirements for the tasks similarities with regard to their performance. The following sections define the methods included in both low and high levels of abstraction in behaviour-based task similarity metrics within this taxonomy and the current methods in literature that are classified within them.

3.2.1.1 Low-level abstraction

Low-level behaviour-based task similarity metric consists of a representation of the tasks that are gained *directly* from the learners' representation of how to behave within in a task—these approaches use measures of similarity directly within these spaces. These include policies or action-value functions, depending on whether they are value-based or policy-based RL approaches, along with how they are modelled (e.g. gradient updates in a deep neural network). We formally define this as the following:

Definition 3.2. A low-level behaviour-based task similarity metric is defined by a representation space r that is directly from the model's representation of learning, including but not exclusively the policy π_t for a task t, and for deep learning approaches the parameters of the model, θ, or process for updating the model, and use a distance function, d, within that space.

Method	**Task similarity metric**	**Usage**
Auxiliary losses with gradient similarity [DCJ$^+$18]	Cosine similarity of policy spaces	Used as a weighting of auxiliary task policies
Policy similarity [CS05]	Finds the number of states with identical policies	Clustering of tasks for task selection
Q-value similarity measures [CS05, CPS03]	MSE of Q-value spaces	Clustering of tasks for task selection
Self-organising maps of Q-values [KB18]	Cosine similarity of Q-value spaces	Use of task similarity to choose similar value functions for use in exploration strategies
Train task selection using policy-based terms [GL21]	Two terms: KL divergence of policies over states, and difference in entropy between learning phases	Training task selection for optimising relevance of task distribution to encourage test task generalisation

Table 3.1: Behaviour-based low-level abstraction task similarity metrics with their associated usage.

The current approaches that are in literature that can be characterised as using a low-level behaviour-based task similarity metric are shown in Table 3.1.

Firstly, many of the approaches directly use the agent's policy as the task representation in the task similarity metric, where either the measure of similarity or their usage of it differs. We see this usage in work by [CS05] who provide a set of task similarity metrics including policy overlap which measures the number of states that have identical policies across the tasks—this *hard* form (i.e. strict matching of policies) of similarity however doesn't cater for the closeness of policies, and requires for the learning of good estimates of the function before being effective. Work by [GL21] combine the following policy-based terms to find similarity: the average KL divergence between the respective policies over the states of the tasks (between task a and task b), along with the difference in entropy over the states with respect to the on-policy distribution before learning. These are used as evaluation criteria on the validation set in a meta-learning setting in order to select the most relevant training tasks to form a generalisable agent for the validation set—this work showed that a selection of a subset of tasks can be more useful in forming a generalisable agent for a particular validation task than training on all available training tasks, hence limiting negative transfer.

Secondly, Q-values are also widely used as a task representation. These approaches require for value-function based approaches, such as Q-learning, or hybrid approaches, such as actor-critic, for access to the Q-function. We see this in work by [CS05, CPS03] who detail the usage of the mean squared error MSE, or their euclidean distance, of Q-values at corresponding states and actions as a similarity metric. These are used for the clustering tasks to provide a set of inter-related tasks to generate a task set which encourages optimal transfer during training. One consideration for this approach is that it requires for the tasks to have a good representation of the Q-value, which impacts the initial effectiveness of the similarity metric during the early stages of training. Another work that uses Q-values is that of [KB18] who take the cosine similarity of Q-value to choose similar value functions in the selection of tasks during training. It was found in this work that the cosine similarity measure resulted in faster learning, however they found that the particular method only performed well in instances where tasks only differed in reward structure.

Finally, a method that uses the architecture and learning process as a representation of tasks is that of [DCJ+18] who use the cosine similarity of gradients learnt across different tasks as an adaptive weight for task relevance within an auxiliary loss. This approach was used as part of a policy distillation method to impact the influence of the teacher policy on the learning of the student. Results indicated that this term acted a useful bias for limiting negative transfer whilst also encouraging positive transfer by increasing task relevance. However, this method only is applicable on methods that learn distinct gradient updates for each task.

Overall, the low-level behaviour-based task similarity metrics showed effective usage at facilitating the sharing of knowledge across task adaptations, in particular most of the cases were not limited in the type of adaptation in the task distribution. However, they were limited in the degree of change that they were able to share learning across, as they were dependent on finding sufficient matches across representations, for example within the state space. This means that the applicability of the method is highly dependent on the encoded behaviour being an appropriate representation of the tasks optimal behaviour in order to be a useful comparative space.

3.2.1.2 High-level abstraction

A high-level behaviour-based task similarity metric consists of both a task representation that embeds the properties of the agent's learning representation and/or process of learning this representation in order to extrapolate the underlying structure of the properties with regard to the agent's behaviour, and a measure of distance within this space. These approaches shouldn't directly use a measure of distance within the agent's behavioural representation (e.g. policy space, or Q-value), and instead should learn a the underlying structure of this space using these representations that describe the underpinnings of how the agent should optimally behave in the task. We formally define this as the following:

Definition 3.3. A high-level behaviour-based task similarity metric is defined by a learnt representation space r that is an embedding space of properties from the agent's learning, such as a policy π, that captures underlying structure from this space, and uses a distance function, d, within that space.

Method	**Task similarity metric**	**Usage**
Policies at states for local and long-term similarity [AMCB21]	Computes a metric embedding based on the ℓ_1 difference in actions at a state, and the Wasserstein distance between policies	Formed using contrastive learning where metric can be used as an auxiliary loss
Value-function latent embedding [ZY20]	Use smallest non-zero eigenvectors of factorisation of Q-value functions from different tasks	Directly transfer Q-values from source to target policy
Task descriptor and policy coupled dictionary [IRE16]	Coupled dictionary of task descriptor (from environment) and latent policy space	Use of task descriptor linked to nearest policy in coupled dictionary to select optimal policy for subtask and to optimise relevance.

Table 3.2: Behaviour-based high-level abstraction task similarity metrics with their associated usage.

The current approaches in literature that can be characterised as using a high-level behaviour-based task similarity metric are shown in Table 3.2. In the approaches found in this category we see that they are mainly focused on extracting and learning underlying structure from the model representations discussed in the low-level behaviour-based task similarity metrics.

Work by [ZY20] attempts to find structure within the value function to find similarity across tasks by factorising the Q-value functions learnt from tasks and taking the smallest non-zero eigenvectors as the measure of similarity. These are then used transfer learning to the target policy using an ϵ-greedy action policy selection between the source and target task rather than direct transfer to mitigate negative transfer from possible task dissimilarity. Results indicated that this task similarity metric provided a useful signal in the policy selection of various from of task adaptations that aided learning, and therefore performance, throughout the training phase. A drawback of this approach is that it is only applicable to value-based methods.

Following this, work by [AMCB21] look at forming similarities amongst different tasks policies at commonly shared state spaces which combines two terms looking at local policy (behaviour) similarity and long-term policy difference to form an embedding space. This work presented empirical results that suggested behavioural structure can be shared across MDPs, and thus increase generalisation across various types of environment adaptations.

We also see work using task descriptors as a means of identifying tasks that encode properties of the task in order to be used to link to certain tasks. Work by [IRE16] encapsulate task similarity within task descriptors formed from task trajectories, consisting of an embedding space of state-action pairs, to create mappings between policies. This method uses a hybrid approach using both policy similarity, using a latent embedding of the policy space, and task descriptors based on environment interaction with in order to learn coupled dictionaries where the two forms of similarity are jointly used by balancing their contribution—this results in a task descriptor that is able to produce the attributed policy space. Results indicate that this approach produces agents that are able to learn aspects of a task that facilities positive transfer. Also, this approach allows for optimal policies to be matched through only task descriptors, thus being suitable in zero-shot applications, however it is highly dependent on the policy representations being suitable representations of the entire tasks and therefore an agent must be sufficiently trained on the old tasks to facilitate the learning of similarity and therefore effective sharing.

Overall, the high-level behaviour-based task similarity metrics that are in current literature have shown that as they find high-level abstractions and underlying structures of the agent's representation of the environment, they have shown to be able to form use a task similarity metric that encourages the successful sharing of knowledge with regards to generalisation and overall asymptotic performance in diverse environments and not be constrained by the form of adaptation.

3.2.2 Environment-based

Environment-based task similarity metrics use task representations formed from interaction with the environment irrespective of the learners specific representation of task behaviour and the process for learning it, and a measure of similarity within this space—these generally follow with the recommendation of [CS05] of being pre-processed in terms of access with the learning of the metric happening before the training of the agent that leverages it. This information is mainly gained from interaction with the environment, which, as discussed in Section 2.1, is generally modelled as an MDP, consisting of such properties including observations ($\mathcal{S}$), rewards ($\mathcal{R}$), etc. For example, an environment can consist of visual input that the agent learns from, where an environment-based task similarity metric will only use the visual information from this, rather than a policy learnt from the interaction with the environment. Environment-based task similarity metrics allow for the agent's own representation to not affect the learnt task distances by taking a representative sampling of the environment—we discuss this further in Section 3.3.2.

The remainder of this section provides a definition for both low- and high-level environment-based task similarity metrics and the methods in current literature that are classified within these.

3.2.2.1 Low-level abstraction

Low-level environment-based task similarity metrics include a representation that is formed from the base task properties, either from the parameterisation of the tasks or the interaction of the agent with the tasks' within the environment (e.g parameter space, raw MDP information such as observations and rewards), using a representative sample of the entire environment, and a measure of distance in this space. In particular, these values are only accessed through interaction with an environment and should not be associated with a particular learners approach or representation of the task. We formally define this as the following:

Definition 3.4. A low-level environment-based task similarity metric is defined by a representation space, r, that is a feature space consisting of properties directly from interaction the MDP, such as an observation ($\mathcal{S}$) or rewards ($\mathcal{R}$), and uses a distance function, d, within that space.

Method	**Task similarity metric**	**Usage**
Returns similarity [CS05]	MSE of returns from state and action	Clustering of tasks
State-based similarity [FPP04, FCPP12]	Use of bisimulation to estimate difference between states in an MDP (based on rewards and state probability)	Aggregation of MDPs by state
State-action trajectories [NL19]	Jensen-Shannon divergence of state-action transition dynamics	Used to analyse the SEAPoT [NLL17] algorithm with regards to the effect of task similarity
Action-reward sequence similarity [GPA07]	A ratio of common action-reward sequences from each task against the number of action-reward sequences from a policy	Use the updates from a states Q-value to update the Q-values of similar states
State-action similarity [TSL07]	Handcrafted distances between features in state and actions	Metric provides a mapping across Q-values to facilitate sharing
Reward and transition dynamics similarity [SGWA16]	Maximum value from the actions in a state based on the difference in rewards across states and the Kantorovich distance between transition dynamics	Initialisation of value function based on the similarity of state-action pairs
Policy clustering using Expectation-Maximisation [ARW21]	Performance difference, based on rewards gained on episodes	Used for clustering of tasks for selection of nearest tasks
Probabilistic policy reuse [FV06]	Reward difference across policy usage	Follows an probabilistic selection policy based on the reward gained from using the policy, selecting from a pool of policies

Table 3.3: Environment-based low-level abstraction task similarity metrics with their associated usage.

The current approaches that are in literature that can be characterised as using a low-level environment-based task similarity metric are shown in Table 3.3. We can see that all of the methods directly use value from the MDP.

The work in current literature using base MDP properties include work by [CS05] who have shown the use of the MSE of immediate returns from state and action trajectory spaces to be a feasible similarity metric. However, although they found that it can describe transfer similarity, it suffers in not being able to sufficiently describe adaptations in the tasks if a full representation of the task is not sampled from (i.e. if an agent doesn't complete a task, then the reward differences are not a sufficient representation of the task). Reward difference is also used in work by [FV06] who use reward gain when using a policy as a probabilistic selection process from a pool of policies. Results show that the biased selection of policies using the reward difference as a signal produces policies that learn the structure of the training tasks as particular policies are able to be effectively reused across different tasks. Further from this, [ARW21] use an Expectation-Maximisation process based on the performance of an agent on a specific task in an agent with different policies for each task. This allows for control over the impact of tasks on learning. This work also highlights the issues in the assumption of uniform similarity in task relevance. In this approach, tasks are clustered based on the reward difference of the agent when using different policies, where results have shown that this aids in a reduction in negative transfer. Methods that also incorporate other features than just reward include work by [FPP04, FCPP12] who investigate the approaches for state aggregation in finite MDPs, in particular investigating the use of bisimulation to estimate difference between states in an MDP based on rewards and state probability. This approach does allow for the aggregation of states, however it requires for a tabular state space and perfect information about unseen task dynamics and rewards [AMCB21]. This work is built upon by [SGWA16] who introduces a similarity metric based on a combination of the reward difference at a state and Kantorovich distance of a combination of state and transition dynamics between tasks. They also build upon this by applying the Hausdorff distance metric to the result which allows for a mapping between the MDPs in which the distance is then used for model initialisation. Results from this indicated that this similarity metric provided a useful measure for analysing the effect of similarity in transfer learning and mitigated negative transfer by limiting the impact of dissimilar tasks. However, it requires for MDPs to be homogeneous, therefore limiting applicability to certain multi-task learning settings. Furthermore, [GPA07] define task similarity as the ratio between common state's history of action-reward sequence across tasks against the source tasks state-action trajectories—results showed that it allows for quicker convergence to the optimal Q-values. One of the limitations of this method is that it requires a sufficient history of state-action trajectories in a particular policy to be shared across tasks. Following this, we also find methods that don't use reward, and therefore a signal of performance. This includes work by [NL19] who look at the similarity of the state-action distributions using the Jensen-Shannon divergence. This methods is however limited in that it facilities the analysing of a method that is only able to manage with changes in transition dynamics. Work by [TSL07] looks at forming mappings across tasks between task states and actions to share Q-values across tasks, where results showed that this enabled for faster convergence to optimal Q-values. However, this approach requires hand-crafted mappings for specific tasks, focusing on environments with adaptive state and action spaces, and therefore only provides an indication of the benefits of using this task similarity metric over these types of adaptations.

Overall, the low-level environment-based task similarity metrics cover most of the standard MDP properties for the task representations and provide several different measures of similarity within this space. These approaches have shown to be able to share learning across many different types of task adaptation. However, the type of adaptation that they are able to maintain performance across, and therefore generalise across, is heavily dependent on the information that the method uses to form its task similarity metric. It also requires a level of similarity within tasks to facilitate the gathering of a representative sample of the chosen environment property across each task within similar spaces, thus limiting the scope of adaptation that they can perform well in.

3.2.2.2 High-level abstraction

High-level environment-based task similarity metrics consist of both a task representation that is formed learning the underlying structure of the environment's properties, such as object relatedness, learnt from input from the interaction from the tasks, including observations ($\mathcal{S}$) and rewards ($\mathcal{R}$), and a measure of distance within this space. This approach builds upon using raw features of the environment, e.g. MDP properties, visual input, etc. and attempts learn the underlying structure of the environmental information. We formally define this as the following:

Definition 3.5. A high-level environment-based task similarity metric is defined as using environment-based properties, primarily those gathered directly from the MDP, such as states ($\mathcal{S}$), or rewards ($\mathcal{R}$), to capture underlying structure from this space, and uses a distance function, d, within that space.

The current approaches that are in literature that can be characterised as using a high-level environment-based task similarity metric are shown in Table 3.4. We see that the methods here either focus on forming task similarity metrics

Method	**Task similarity metric**	**Usage**
Observation latent representation [HPR+17]	Implicit similarity in observation space	Learns a factorised latent representation of observation to enable for domain shift
Reconstruction of MDP properties [AET+14]	Reconstruction error (euclidean distance) of RBM trained using state and actions	Selection of similar tasks for transfer learning
Visual analogies [SWT18]	Implicit similarity from visual features (attention operators and dilation filter for feature extraction) in GAN	GAN used as a transformer for the observations space of other (i.e. domain shift in order to force both tasks to be the same)
Meta-World Models [WLW18]	Implicit similarity in state and action trajectories	Feature space of shared dynamics that can be used in RL methods, for example as a meta-learnt space
Invariant feature spaces [GDL+17]	Time-based similarity comparing number of steps to a particular state	Used as an additional signal in the reward space
Abstract Representations [FLBPP18a]	Latent embedding space of states' transitions and rewards. Each state encountered is the encoded within the lower dimensional space with the transition model of the environment based on the agents learning	Implicit environment similarity through embedding experience in the lower-dimensional space
Latent observation similarity [MKNT21]	Difference between observations in the latent embedding space of sampled observations	Similarity of observations in used in a contrastive learning approach to form embedding space

Table 3.4: Environment-based high-level abstraction task similarity metrics with their associated usage.

constructed from similarity measures in abstract representations, or attempt to unify the space across tasks using abstract representations.

We find methods that leverage an understanding of task similarity that attempt to capture the latent structure of environment-based representations. Work by [AET+14] look at using MDP properties within a restricted Boltzmann machine (RBM) in order to form a similarity through the reconstruction of tasks from their sampled state and actions. This is then used in task selection, with results showing that the error in reconstruction from the latent embedding of these properties shows that distance is negatively correlated with performance—this approach requires for the MDPs to be within the same domain with relation to dimensionality. Work by [GDL+17], attempt to improve sharing in a transfer learning setting by learning invariant feature spaces, formed from a time-based similarity metric which assumes that similar tasks will reach the same task at the same time—they do further optimise this using expectation maximisation to take into account the sensitivities of this representation. This work specifically allows for the transferring of knowledge on tasks where an agent's transfer dynamics alter—in this case the robotic arms have different numbers of joints. This indicates that the invariant feature space is able to form "analogies" across tasks involving changes to the transition dynamics (e.g. changes in the robots morphology), but does not facilitate sharing across other forms of task adaptation.

Although not directly using an explicit form of similarity, we also see approaches that find a shared space that effectively minimises similarity across tasks in this space—we mention them here as approaches that implicitly use task similarity. The approaches that form a shared representation of task and use this to transfer the task into a shared space includes work by [HPR^{+}17] who form latent embeddings from the observation space, which is used instead of the raw observations by agent—this attempts to remove unwanted information from the observation, and only information related to completing the task, as such reducing the task-specific noise the agent must separate and learn from. This doesn't explicitly use a measure of similarity, however it does aim to implicitly find a representation that causes all tasks to be represented the same. Results showed that this allowed for large adaptations in the observations space, however it didn't fix the issue of transferring across tasks with other task adaptations, in particular reward shifts. This is similar to work by [SWT18] who form visual analogies through the extraction of features using attention operators and dilation filters to extract higher order information about tasks. This information is used to form an embedding using generative adversarial networks GANs, a generative learning approach, to generate an embedding for the observations that extracts visual features. This approach also relies on similarity in reward and action spaces, but shows effective generalisation over large observation shifts. Also, work by [WLW18] look at forming a shared representation of task dynamics based on state and actions. They use trajectories in a visual unit to form a latent vector of the environment, and use this in a memory unit which uses an action to reconstruct the next state from the latent vector—through comparison of the actual next state and the reconstructed state, results showed on the Atari Pong game showed that this representation forced agent to learn the ability to visually self-recognise (i.e. see where they are when considering a mirrored task). This work is shown to work on tasks involving adaptations in state and transitions dynamics, however any adaptation in the reward space dramatically affects performance. Along with these, work by [FLBPP18a] form a abstract representation of the environment where they also incorporate a loss term that is conditioned on the cosine similarity of actions taken from a state against the state in the embedded abstract representation. Results showed that this allows for the representation to facilitate generalisation across the state space. Further from this, work by [MKNT21] has shown that by using a latent embedding of the observation space the information relating to the similarity of tasks within this space allow for the learning of a policy that can transfer across tasks with different observation spaces.

Overall, the high-level environment-based task similarity metrics showed that they were able to learn the structure of a task without regard with how an agent learns within it (i.e. without directly using the agent's representation of the task). As was the case with the low-level metrics, certain approaches still depended on similarity within certain aspects of the task (e.g reward and action spaces), however they learnt a representation that was able to transfer across large task diversity, such as the mirroring of task (i.e. changing overall observation spaces)—this illustrated that high-order memory skills were able to be learnt using these metrics that were invariant to the particular adaptations of the MDP property.

3.3 Discussion

The taxonomy described in Figure 3.1 provides a framework for defining the type of task similarity metric that a method uses in order to understand their applicability to certain types of environments defined by the differences between tasks and diversity in the task distributions contained. As such, we see that both behaviour-based and environment-based task similarity metrics have extensively examined the usage of a large amount of the applicable information that is available. We find that a broad spectrum of both task representations and task distance measures, and their overall usages, have been investigated within current literature. Across all of the approaches discussed in this chapter we see that there is predominant work in environment-based approaches, in particular using low-level information by directly leveraging distance measures on the observation and action space. Although these approaches have shown to be useful in facilitating the learning of a representation that is able to transfer understanding across task adaptations, we see that they are more limited in the types of adaptation they can share across and the degree of this adaptation. We see by incorporating behaviour-based approaches and higher-level understanding you can, respectively, remedy these issues, with them showing greater ability to generalise across over multiple types of adaptation, and to a higher degree—although, it is to be noted that these do have in themselves have limitations in their usage and applicability. The reason for this is that the behaviour-based approaches are able to leverage the requirements of the learner with regard to task similarity in order to encourage tasks more similar their current representation. This also doesn't limit the type of adaptation due to the overall task properties of the MDP being encapsulated within the intended behaviour. This topic is discussed further in Chapter 5.

Overall, we see that across the methods detailed above, there has been a recent increase in focusing on finding task similarity metrics that guide an agent to learn a task representation that allows it to maintain performance when tested on tasks that include MDP properties that are adapted from the tasks encountered during training. This compares to traditional methods that have been found to memorise optimal paths, and therefore suffer with slight changes from those tasks that were learnt from. However, many of the experiments analysing the effect of incorporating task similarity metrics into the method only look at performance against current baselines instead of understanding the effect in relation to generalisation. As mentioned in Section 2.1.1, this limits the amount of understanding with regards to effectiveness of transfer when considering overall agent generalisability—we discuss this topic further in Chapter 4.

In the rest of this section we provide a discussion of important aspects of the methods in the taxonomy detailed in Figure 3.1 that need to be considered in their usage, including the learning of abstract concepts and the approaches with regard to off and on-line usage of task similarity, and the impacts of these, found in the methods in current literature.

3.3.1 Learning of Abstract Concepts

We have seen in the literature, with many of the methods categorised using the taxonomy stated above, that the level of abstraction of the task similarity metric doesn't constrain the emergence of higher-order memory systems. Even in the low-level abstraction, both performance and behaviour-based, we do see instances where higher-order memory systems are learnt by those with lower-level abstractions. However, in these instances the environments used generally show lower levels of adaptation in terms of the degree of adaptation they can perform across. Although it is claimed by [HSB$^+$19] that in order facilitate the learning of higher-order memory systems on tasks distributions with diverse task adaptations it requires a higher-level of abstraction of agent representation of the tasks, we've seen across many of the performance and behaviour-based approaches the agents are able to form higher-order memory systems (i.e. memorisation, interpolation and extrapolation). Instead, the difference is in the environment task distribution diversity, in terms of type of diversity and scope, to capture the relevant generalisable features that can be transferred across tasks. We claim therefore, that the higher-order representations aren't more useful for forming higher-order memory systems in all situations, and instead they are more adept at causing these systems to be formed over more diverse task adaptations and differences.

Overall, there still remains the issue of forming a method that is able to control for this level of abstraction depending on the requirements of the task distribution, where current methods still are limited to the types of adaptation they can effectively transfer knowledge across. This however has relevance to the idea that there is no best task similarity metric for all situations [CS05]. This aspect of understanding generalisation across tasks with large and diverse adaptations is discussed further in Chapter 4.

3.3.2 Off vs. On-line Usage

As described by [CS05], the preferred condition for the reuse of task similarity metrics is when they are formed from experience calculated off-line from the agent. Generally, we find that the methods that use performance-based task representations in their task similarity metric are on-line methods (i.e. learnt whilst they are being used), whilst environment-based methods are learnt in both off- and on-line situations.

Following from this, we state that the outcome of an on-line behaviour-based metric is likely to not be as applicable to other methods due to it's dependence on the architecture of the agent's model. This will constrain the form and representation that the agent has of the tasks in the environment—this is represented as the policy (π), and therefore the agent's type of representation (i.e. deep learning gradients) will impact the sampling of tasks in the environment, and the policy space that the agent will explore. The collection of performance-based task representations, and therefore their defined similarities, will heavily bias the overall representation formed (i.e. if a learner is using a task similarity metric to influence the interactions that they have with the environment or task the representation of this environment will be innately biased by the performance of the agent and therefore the task similarity metric itself), hence why [CS05] recommends that a task similarity metric is pre-computed. This is mainly a factor when dealing with behaviour-based task similarity metrics due to how the model being innately connected to how the learnt task similarity metric is used. This is in comparison with environment-based task similarity metrics where the impact of this bias will be lessened as long as we are finding a representative sampling of the tasks and associated trajectories.

Finally, comparing the environment-based and behaviour-based approaches we note that the environment-based task similarity metrics provide reliable information during learning immediately, given small enough adaptation space, however behaviour-based approaches would generally need a *warm-up* period in order to be fully applicable to aid in learning—this is a topic mentioned by [CS05] in relation to the effectiveness of Q-values compared to reward similarity. Overall, this impacts the reusability of the metric as to when the learnt metric can be used during the training phase.

3.4 Summary

In this chapter, we have introduced a taxonomy for the categorisation of task similarity metrics used in RL. This extends current literature by focusing both on the source of the information used to form the task representation, described as either environment- or behaviour-based, and the level of abstraction that is learnt using this information, either high- or low-level abstraction. This allows for a larger collection of methods to be included not only looking at direct usages of MDPs but also higher-order representations of properties including the learners representation of optimal behaviour. By providing this taxonomy we have detailed a means of comparing the methods that use task similarity metrics with regard to their ability to facilitate shared learning across certain forms of task difference and diversity.

We have highlighted the importance of fully understanding the limitations of certain forms of task similarity metric to ensure their correct application on an environment's task distribution. In regards to the task representation, we have shown that the source of information, whether environment- or behaviour-based, will highly affect the types of task differences that they facilitate shared learning across. We have also shown that the level of abstraction of the task similarity metric highly affects the amount of diversity within the environment's task distribution that can be learnt across. Further work is needed to understand the possible approaches and the impact of using high-level abstractions in both environment- and behaviour-based approaches in order to allow for the sharing of knowledge across task distributions with high diversity.

Finally, by investigating current methods that use task similarity metrics, this work provides details of the areas of current literature that require further investigation to understand the effect of types of task adaptation and degree of diversity on the effectiveness of the task similarity metrics in facilitating shared learning. As discussed in Chapter 2, many of the approaches in RL, along with meta-RL and MTRL, all require an understanding of task similarity to ensure applicability to certain environments. As such, this work supports the overall aim of illustrating the importance of this domain in order to encourage further research in understanding how task similarity metrics can further describe the requirements of an agent, and thus encourage generalisation within an environment containing a diverse and adaptive task distribution.

Chapter 4

A Multi-Task Definition of Generalisation in Reinforcement Learning

4.1 Introduction

A lack of generalisability is one of the main issues limiting RL to many real-world applications, as was discussed in Section 2.1.1. It has been seen that many of the current single-task RL methods are highly affected by even slight changes in the task and environment, with one of the main reasons being due to the agents learning how to repeat and memorise optimal actions rather than extrapolate understanding from the sample interactions [CKH$^+$18]. In particular it has been shown that one reason for the lack of generalisability is current methods tendency for catastrophic overfitting to the training task distribution [ZVMB18]. The remedying of this issue consists of two factors that need to be considered: (a) a detailed approach of how to assess whether the agent is able to generalise, and (b), an understanding of the methods available that will enable for a RL agent to generalise to different types of task distribution adaptation and diversity. There have been several attempts at remedying the issues of a lack of generalisation in RL methods, however we still lack approaches for quantifying the generalisability of an agent, along with understanding of why RL agents suffer from poor generalisation, and how to find a general method of fixing it [FYF$^+$16].

Although many of the approaches that aim to specifically remedy generalisability have been shown to work on particular forms of task adaptation, there is also evidence that in certain conditions, classical approaches have been able to generalise more effectively [PGK+18]—this highlights the need to understand how generalisation works and why we are currently limited in certain forms of task distribution diversity. The issue of quantifying the generalisability of an agent's model in RL has become an area of vital importance to allow for the framework to become more widely applicable. It has been shown that current methods for evaluating RL method's performance generally use the overall rewards per episode that the agent gets in a train/test split, an approach of which is generally found within supervised learning, produces performance measures that are artificial and do not actually evaluate the generalisability of the agent [WLT+18].

One of the major issues with the evaluation approaches that are currently provided is that they mainly focus on the single task case. This results in them lacking the ability to describe generalisability with regards to task distributions containing diverse and multiple types of adaptations, defined by differences in MDP spaces such as state $\mathcal{S}$ and reward $\mathcal{R}$ space, thus impacting their required policies [CKH+18]. In the multi-task learning case it is vital to produce measures that are able to perform across tasks to ensure that a shared understanding of how to interact with task differences is optimally learnt. The divide between single-task and multi-task is described by the added challenge of balancing the optimal reward from a single-task whilst maximising the mean reward across a set of inter-related tasks [VVM20], where the similarity of tasks within the distribution will highly affect the degree of challenge in performing this. This issue represents an important challenge in enabling RL to become applicable in real-world problems due to the continually changing nature of real-world situations [TFR+17].

The contributions in this chapter are as follows:

- We provide a discussion of current methods to quantify generalisation in RL. We show that current methods of generalisation are constrained in their application and can't universally be used irrespective of model and task distribution form.

- We detail a definition for generalisation with regard to task distance and performance that is both model- and task-agnostic, thus allowing for the comparison of any method on tasks distributions with any type of adaptation and diversity of task.

- We list the process for analysing the generalisability of an agent with regard to the type of task difference as defined by the task similarity metric, using the definition of generalisation. We also provide the considerations needed when performing analysis using the provided definition. This highlights the importance of understanding the similarity of tasks within the task distribution in being able to analyse the generalisability of a method in any given environment's task distribution.

- We provide empirical results from an analysis of current state-of-the-art methods in both meta-learning and multi-task learning applied to RL using the generalisation definition. We find that methods are able to mitigate the impact of increasing task distance in regard to their performance across different task distributions, however, providing evidence to the findings of Chapter 3, the applicability of the task similarity metric is vital to understand in order to effectively quantify an agent's generalisability.

This chapter focuses on discovering approaches to increase the generalisability of RL models in both MTL and meta-RL frameworks. In Section 4.2 we investigate the current definitions of generalisations and their limitations. In Section 4.3 we provide a breakdown of the new definition, in terms of how we define it with regards to performance and task similarity, and then provide a process for conducting the analysis with regards to the definition. In Sections 4.4, 4.5 and 4.6 we provide the experimental setup and results of conducting analysis using the generalisation definition in order provide an understanding of state-of-the-art approaches in meta-RL and MTL with regards to generalisation. Finally, Section 4.7 contains the concluding remarks of the chapter.

4.2 Current Generalisation Definitions

Traditional methods of assessing transfer performance within RL generally focus on reward differences across train/test splits. These include such approaches as jumpstart (i.e. the amount of performance an agent can start with on a new task), overall asymptotic performance (i.e. increase in performance compared to learning without transfer) and rate of learning increase (i.e. the time to a threshold performance) [TS09]. Another common approach is to make use of the idea of underfitting (i.e. not learning features that are found to be particular for a specific task, therefore generalisable, but not learning specific task features) and overfitting (learning only features associated to task, therefore not generalisable), which is often used in supervised learning [ZVMB18]—this is generally performed using reward difference methods. However, nearly all of the measures that make use of reward differences attempt to quantify generalisation without consideration of the difference in the tasks within the environment's task distribution. This makes an incorrect assumption that the performance of the agent is sufficient to fully describe the type of learning that the agent has been exposed to, in particular the difference between train and test tasks.

As such, due to the issues stated above with using only reward as the measure of generalisation in RL methods, recent progress has been made on defining new measures of generalisation. These have focused on either understanding how the agent has interacted with the environment or forming environments that force the agent to interact with the environment in a manner that forces generalisability—in this case the standard performance measures are more valid as the assumption about the learning of the agent holds greater validity.

A method that introduces new measures defining the generalisability of an agent with regard to the type of task understanding it has is that of [WLT$^+$18]. In this work, they focus on looking at quantifying whether the agent is memorising, interpolating or extrapolating learning through measurements of the value function with regard to the approximated model and its interaction with the environment. This approach is however only applicable to value-based approaches as it requires the Q-value to analyse the difference between expected and actual Q-value when interacting with the environment in off-policy and unreachable states from the learnt function. By introducing these measures of agent understanding based on the agent's own representation, this allows for the measure to take advantage of task understanding at particular states, thus limiting the impact of constant train/test splits during training. This work represents a step forward in assessing agent generalisability, however, it is the only method that we found at the time of writing that provides strict measures of generalisation.

Another aspect that has seen an increase in research in is the environments that are being used as benchmarks for experimenting new methods on, as mentioned in Section 2.1.1. We see in many of the traditional environments, such as the Arcade Learning Environment (ALE) [BNVB13] the provides a large set of arcade games, that although they include many diverse tasks with regard to goals and state spaces, they are constant in terms of their adaptations (i.e. the goal of each task remains constant). Another issue in current benchmark environments is that we generally also see that environments are using the same task distribution for both training and testing phases. This results in the environment not being suitable to test for adaptability, as agents are prone to overfit to train task distributions [ZVMB18]. As such, this has driven work in creating environments that remedy the issues presented by traditional environments. An environment that attempts to remedy the issues of ALE in particular is that of ProcGen [CHHS19] which provide a set of procedurally generated game environments. This environment was created with aim of providing an environment that could be used to both test current applications, and investigate whether adding variance in the training data would lead to higher generalisability within agent's representations, as is similar to work in transfer from simulation to real-world robotic tasks [TFR$^+$17]. This ensures that the agent is learning how to extrapolate understanding of the environment from interacting with it rather than just learning how to repeat the same movement. This is similar to work by [PGK$^+$18] who adapt the parameters controlling the environment whenever the environment is reset to force the agent into out-of-policy situations and thus force the agent to demonstrate higher-order memory systems than just memorisation of states. Another environment that attempts to remedy the issue of consistent train-test splits is that of Meta-World [YQH$^+$19]. This is a wrapper around several of the MuJoCo [TET12] robotic manipulation tasks that provides a means of sampling tasks, within both the MTL and meta-RL settings, containing diversity between train and test task distributions, ensuring that overfitting to tasks is assessed. We also found environments that focus on specific MDP property adaptations across tasks in the task task distribution. This includes work by [WWGT18] who provide an environment consisting of 3D houses with differences in the observations space, along with variations in colours, objects, and layout. Across all of these environments, it is evident that a definition of generalisation requires to incorporate the understanding of how tasks within a distribution are different with regard to both the diversity of task and the way each task may adapt.

Although there has been progress in defining a definition of generalisation in RL which is useful in providing understanding of what the agent is learning, we find that these methods are not applicable to cases where an environment contains a task distribution with high diversity and task adaptability. This is because they don't use an understanding of task similarity to understand how the agent's behaviour across tasks links. This means that even though the approaches that form adaptive environments do allow for reward to be used more effectively as a measure of generalisability in these cases, they don't provide information over *why* an agent is either performing in the manner that it is on an set of tasks which, as shown by [WLT$^+$18], is vital to assessing generalisation.

Overall, we find that the current approaches to defining the generalisability of an RL agent are limited in the following ways:

- **Task type**: Application to diverse task distributions, in particular to those that are out-of-policy, for the analysis of shared task understanding. A large variety of task diversity and adaptation exist, which is not covered in all current benchmark environments.

- **Method type**: Non-specific in application to type of method used. For example, work by [WLT$^+$18] is only applicable to value-based RL methods. It would be optimal to have a definition of generalisation that is both agnostic to the type of RL model and learning approach.

Due to these issues, the remainder of this chapter introduces and details a definition of generalisation that is agnostic to both task and method type and can be used to analyse a method's generalisability on environments dealing with diverse task distributions. In particular we include cases where it is not straight forward to define how the tasks differ without interaction with the environment, in a MTL, where an agent learns the optimal performance across a set of tasks, along with frameworks with a train/test task split, such as a meta-RL. After providing the definition, we detail examples of it's usage to illustrate its usefulness in quantifying generalisability in relation to the type and diversity of task adaptation the agent is learning on. Finally, we also include a discussion of the considerations needed when performing analysis using this definition.

4.3 Task Similarity-Based Generalisation Definition

In this section we provide the definition of generalisation that will facilitate a greater understanding of agent generalisability in regard to dealing with diverse

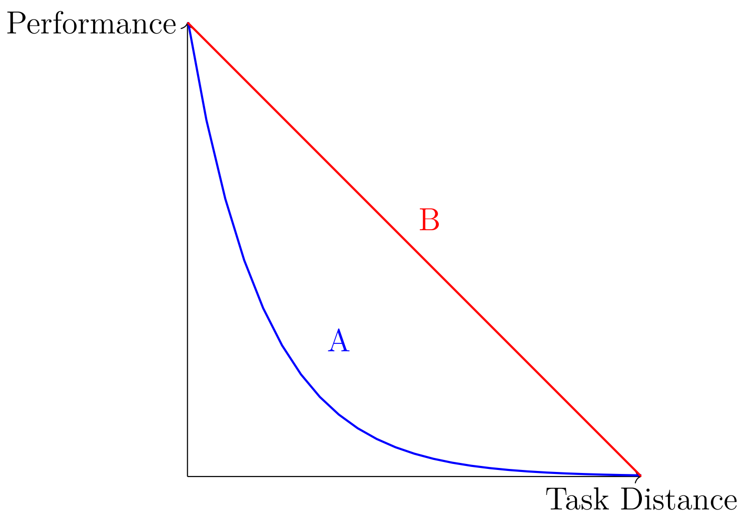


Figure 4.1: Relationship between *task distance* and *performance*. Method A and B represent two possible relationship forms.

and adaptive task distributions, including in MTL and meta-RL settings where current methods are unable to perform within as detailed in Section 4.2. As part of providing the definition, we state the steps required for gathering the required data, and the considerations needed in order to properly analyse the results in order to reduce performance bias.

4.3.1 Definition

The definition of generalisation that we propose is agnostic to the type of method and tasks used and is based on the relationship between *task similarity* and an agent's *performance* across these tasks. The definition of generalisation is as follows:

Definition 4.1 (Generalisation)**.** The generalisability of an agent is defined as the integral of the function between an environment's task distribution, over a defined domain of *task distances*, learnt from a task similarity metric, and an agent's *performance* on these tasks. The performance of the agent should reduce as the distance between tasks increases (i.e. a negative correlation between performance and distance). A generalisable agent maximises the integral of this function.

The definition of generalisation, described in Definition 4.1, details a function, as shown in Figure 4.1, that states how an agent's performance changes over different task distances, defined by a task similarity metric. This is able to provide a quantifiable measure of how the agent is able to generalise it's representation to changes in the task distribution by assessing how the differences in the tasks within a particular environment, defined as the task distribution's similarity as stated in Definition 3.1, affects the performance of the agent—this illustrates a diminishing performance as the distance between tasks increases. This provides a method of defining generalisation whilst being agnostic to how the learning method has defined the task, and instead only requiring the distances between tasks and a measure of performance. We define these terms as the following:

- **Task distance**: the task distance is a measure of how far away a task, encapsulated within a task similarity metric including the task representation, is from another tasks representation within an environment's task distribution, as is discussed in Chapter 3.

- **Performance**: the performance is a measure of the agent's ability to behave optimally on a task, or set of tasks. In regard to RL, this is often defined as the total expected returns an agent receives in an episode.

Using this definition we can state that if an agent is able to maximise the integral of the function defined by its performance across a set of tasks sampled from a task distribution with increasing task distance, then the agent is generalisable. This allows us to understand the degree in which an agent is able to mitigate any performance degradation over the entire domain of task changes in the task distribution. For example, within Figure 4.1 we see that the function that defines Method A's performance across a changing task distance has a dramatically lower integral than that of Method B's. We can therefore state that Method B is more generalisable than Method A over this particular domain of task similarities in the task distribution. By taking the integral of the function we are able to quantifiably measure the generalisability of the agent across the entire domain, however analysis can also be aided through other properties of the function. These include the start and end performance, which details overall performance degradation over the task distance, along with the gradient of the function at particular task similarities to understand the rate of performance degradation at a particular distance. For example, although both Method A and B start and end with the same performance within the task distance domain, the rate of performance degradation over the first half of the domain has resulted in the generalisability over the entire domain to be dramatically lower than method B.

Overall, this comparison between performance and task distance places several requirements on both the assessed method and the environment that the tasks are being sampled from in order to ensure that the process for performing the analysis accurately represents the generalisability of the method.

4.3.1.1 Definition Requirements

The definition of generalisation therefore requires three components, and places requirements on the environment that we are testing on, as follows:

- **Environment's task sampling process**: The environment must allow for a set of tasks to be sampled from it. This can either be through adaptations in a single-task (e.g. CoinRun [CKH+18]) or an adaptive MTL environment (e.g. Meta-World [YQH+19]).

- **Task similarity metric**: We must be able to have a shared task representation across the tasks within the environment's task distribution that allows us to compare the tasks, as discussed in Chapter 3. The task similarity metric, the combination of task representation and measure for distance between tasks within this space, also follows the recommendations stated by [CS05] over the optimal conditions of a task similarity metric: it should provide approximate value to the degree of learning between tasks, (b) should provide a partial ordering of tasks (i.e. better to learn closer tasks first), and (c) it should be computable outside of the traditional train/test cycle. A task similarity metric can also utilise a similarity function. In this case the relationship between itself and performance would be inverse.

- **Measure of performance**: The value that we attribute to performance will affect the particular aspect of what we are analysing. The most common measure of performance used in RL is that of returns (i.e. rewards, $\mathcal{R}$), however other measurements are available. These include performance measures that have seen effort in being used to describe generalisation, such as regret [ORW16].

The relationship described by these two must show a negative correlation (or a positive correlation if a task similarity function is used), otherwise the combination of the above factors are not seen as providing a valid measure of generalisation for the model in this environment. This should give indication as to whether the task similarity metric is suitable for describing the relationship between tasks. This is due to the intuition that as tasks diverge from the tasks that were trained on, then the performance should degrade—this notion of decreasing performance against increasing task distance was also investigated by [AET^{+}14].

4.3.2 Definition Process

The process for gathering the data required for using the definition of generalisability defined in Definition 4.1 requires us to gather the performance of an agent across an environment's task distribution. Therefore, we detail the stages for using this definition as the following:

1. Form a task distribution which includes a set of tasks that can be placed within a shared task representation space.
2. For each task pairing (in the case of a train/test setting, we take a train task set and a test task set) taken from the distribution:
   - 2.1. Calculate the **task distance** across the task pairings using an appropriate task similarity metric.
   - 2.2. Train and test the agent on this task pairing to ascertain its **performance**.

By following these steps we will have a collection of task distances against the agent's performance on the associated task groupings, which we can then used to quantify their generalisability by the relationship between the two terms as in Figure 4.1.

An example of a possible task distribution is provided in Figure 4.2 which details a standard train/test setting, for example within a meta-RL setting. Both of the train sets and the test set contain a collection of tasks, each with their task representation defined by the parameters x_1 and x_2, that are spread across the task distribution. We can then calculate the task distance within this task representation space—in the provided example we take the euclidean distance between the centroids of the train and test task sets to form D_1 between Train$_1$ and Test, and D_2 between Train$_2$ and Test. From this, we would then train and test an agent using a particular method on both of these pairings. This

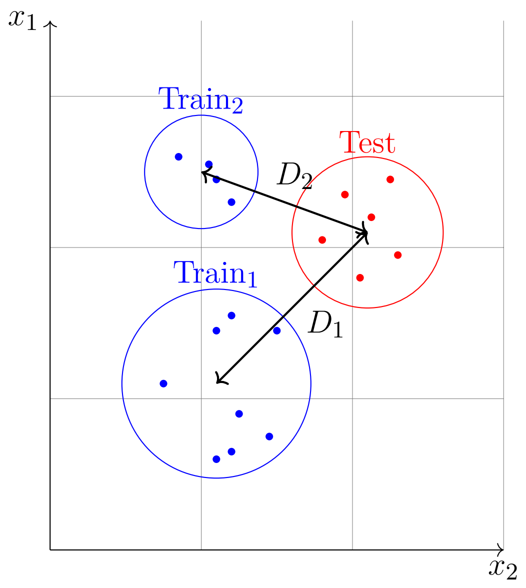


Figure 4.2: Groupings of train/test sets from task distribution in task representation space defined by two variables x_1 and x_2, and their task distances D_1 and D_2.

would form a graph comparing task distance and performance similar to that of Figure 4.1—the overall analysis is aided by an increased number of pairings, and therefore more comparisons of task distance and performance.

As detailed above, in a setting consisting of both a train and test set of data, the task distance is calculated across the train and test sets—there are many methods for doing this such as clustering [HLK18]. However, in a MTL setting we instead need to calculate the task distance as the inter-task distance across each of the tasks within the task set itself. For example, if we take the task distributions of Train$_1$ and Train$_2$ from Figure 4.2 as separate MTL task sets, we can see that both of these have different task distances within each of the task sets—this is signified by the size of the circle which is the clustering size of the task sets. As such, we train our MTL methods on each of these task sets, and then can compare the performance against the inter-task-based task distance. It should be noted that for both of these approaches, the task selection from the task distribution, with regard to number and location, should be considered with regard to how this would affect performance—we discuss this further in Section 4.3.2.1.

It is to be noted that the definition of generalisability does not impose any particular task representation or similarity metric. Instead, an appropriate choice for each of the components must be made by the designer based on the chosen environment and aims of the analysis. Possible options for these components are described in Chapter 3.

4.3.2.1 Performance Normalisation

When selecting tasks from within the task distribution we need to understand the effect that this could have with regard to performance. This process is looking in particular at generalisation across tasks and should therefore limit the affect of innate performance difference in the task distribution's domain. If we were to just compare the difference of performance between two agent's trained across two different task groupings within a task distribution's domain we make the following assumption about this domain with regard to performance:

Assumption: To judge generalisation from train to a *static* test set with regard to task similarity we assume that all optimal train task performances are the same, and only the distances between them should differ. This would also assume that rewards across tasks are the same.

This assumption does not always hold, and therefore we will need to correct for it. We can do this, in a train/test split case, for example with a meta-learning setting, by normalising the test performance by the training performance, i.e. train performance/test performance. If the test set was not static, we would also need to normalise the difference in the test set results by the optimal performance on the individual tasks. With the experiments in this chapter we will focus on a *static* test set. However, should these experiments be expanded to environments where the test tasks are different between different train/test task pairings, it should be noted that normalisation across the test task difficulty would be required to ensure only generalisation is measured and not difficulty innately within the task distribution for where the test tasks are sampled from.

Assumption: We assume that the optimal performance of each of the tasks when learnt on its own is the same.

Due to this assumption, with regard to dealing with only a training set, such as the case in MTL where we want to learn optimally across a set of tasks, we would therefore need to normalise by an optimal performance on each separate task in the task distribution, i.e. single task performance/train performance. This would mean that we are solely assessing generalisation rather than any innate performance difference in the tasks across the task distribution.

For both of these assumptions on both MTL and train/test setups such as meta-RL, it should be noted that the difficulty surface in a task distribution will be model dependent and should be calculated separately—although there may be specific tasks that are considered difficult, certain models may find the task difficult in certain contexts which could impact the performance measure used in comparison, which would result in a different difficulty surface.

4.4 Experimental Setup

In order to demonstrate the multi-task generalisation definition, and provide an analysis of the performance of the current state-of-the-art methods in relation to their ability to generalise with task distribution adaptations and diversity, we provide experiments across environments that differ in the type and degree of their task differences, along with methods in both the MTL and meta-RL settings.

4.4.1 Environments

For use in the experiment, we use environments that have a range of differences in their task distributions. This allows for the definition to be illustrated as to its applicability to quantify generalisation in environments consisting of a range of diversities in the task distribution.

4.4.1.1 Adapted Environment Case: Pendulum

The Pendulum task is a classical adapted continuous action single-task RL problem, where the agent attempts to balance a pendulum of a certain length and mass in the upright position. This environment is adapted in these experiments in order to form a task distribution through the parameterisation of values for both the length and mass of the pendulum. This allows for varying tasks to be generated which differ in the tasks transition dynamics when referring to the tasks MDP. This task was chosen as it represents a simple adaptable environment that can generate different task instances across a task distribution. This was chosen over other environments that provide the same function, such as CoinRun [CKH^{+}18], however using the adapted Pendulum task gave me full control of accessing environment parameters and how the tasks are adapted. This environment is further discussed in Appendix A.

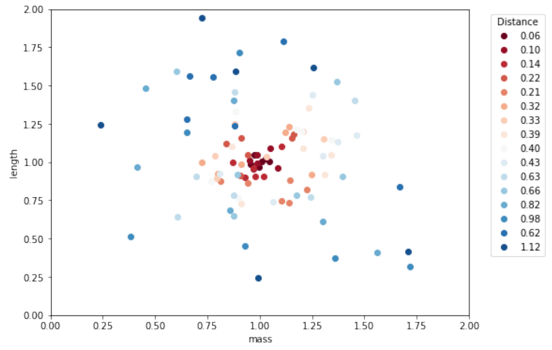


Figure 4.3: RL Pendulum task sampling distribution across the domain for a MTL setting. Task distributions parameterised by pendulum length and mass, with varying inter-task distances as defined by the cosine distance between each task's initialisation parameters within the train set in the parameter-space.

The MTL setting's environment consists of 16 task groupings, where each group contains 6 train tasks, as highlighted in Figure 4.3. These are spaced across the entire task distribution domain with varying inter-task task distance. The tasks samplings were chosen as random sampling with the centroids of all task collections located in the centre which allows for the sampling of various task groups with varying inter-task distances across the entire domain—this is an artificial sampling, however will be sufficient provide indication over efficacy of process, and agent generalisability. Due to the different positioning of tasks within the domain, it will be vital to understand whether certain areas of the domain are more difficult than others—this is previously discussed in Section 4.3.2.1.

As can be seen in Figure 4.4, we are sampling from several different bounds across the initialisation parameter-based domain space. Each of the crosses represent a task within a batch, uniformly sampled from within a bounds forcing a centroid within that area of the parameter space. Each of these are then used within the train and test phases. The meta-RL setting's environment consists of 16 task groupings, where each group contains 6 train tasks and 4 test tasks—the 4 test tasks, highlighted in the bottom left corner of the figure, are maintained as the same tasks across all 16 train/test task pairings as to simplify the comparison of performance so it isn't biased by task difficulty and ensures only generalisation is assessed.

The task similarity metric used across both the MTL and meta-RL task subsets represents the euclidean *distance* between the mean positions of each of the tasks within each group (i.e. the euclidean distance between the centroids of each

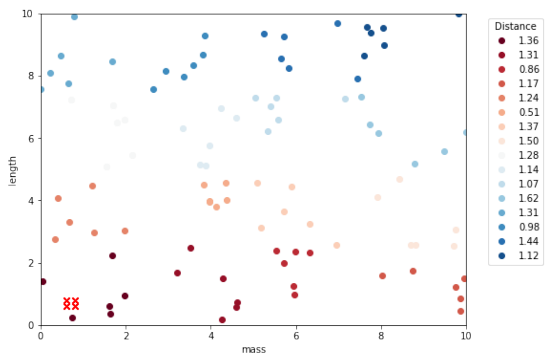


Figure 4.4: RL Pendulum task sampling distribution across the domain for a meta-RL setting. Task distributions parameterised by pendulum length and mass, with varying inter-task distances as defined by the cosine distance between each task's initialisation parameters. Several different train and test distributions within an overall joint task definition, each task defined by the bounds within the overall definition. A pairwise test/train dataset is formed (the same train distribution on each) will be used against each of the test distributions with each having a different distance between train and test. The confidence intervals are formed from final loss on several different test runs, with those test task distributions.

task subsets)—we should therefore note that the relationship described in Figure 4.1 should be a negative correlation. This was chosen as it provides a simple metric that can also be useful for visual investigation. This approach however does treat the weightings of both the parameters, mass and length, as equal factors in the task similarity metric, which will therefore be interesting to analyse during the experiments whether they have equal impact on the final performance and generalisation—as previously stated, an optimal task similarity metric would weight these according to their affect on generalisability.

The complete list of task subsets, and their associated task similarity values, for the Pendulum environment, for both MTL and meta-RL settings are outlined within Appendix B.1.1.

4.4.1.2 Multi-Task Case: Meta-World ML10

The Meta-World environment [YQH$^+$19] is a wrapper around several of the MuJoCo [TET12] tasks, focusing on a diverse set of robotic arm manipulation tasks, that provides a set of different train/test combinations for both meta-RL and MTRL settings—these are further detailed within Appendix A. Here we use the version 2 (v2) of the tasks which fixes several issues that were present within the version 1 tasks. For the Meta-World environment we are limited to the number of tasks that are available, therefore to produce varying task distributions with different overall inter-task similarities we select varying subsets of the tasks and calculate the inter-task similarity using an appropriate measure. It is important to note that we will be using task collections of the same size—if they were of differing sizes this could result in a bias towards performance being aided not only by closer task sets, the topic of discussion in this chapter, but also a reduction in the number of tasks to learn from as in principle it is easier to transfer to a smaller set of tasks.

As discussed in Chapter 3, there are various task representations and metrics that can be used in this case, however here we limit ourselves to a metric that must be accessed before training. Therefore, for generating a set of train tasks we sample 15 episodes on each task and task the cosine distance (i.e. $1 -$ Cosine Similarity) between the observations of each task. This forms a task distance matrix providing a means of comparing the tasks. We then take a random sub sample of tasks within the possible train tasks. With regard to test tasks, we keep the same 5 test tasks, and only adapt which train tasks—this simplifies the analysis and will give. A possible extension would be to also modify the test task set, however we don't believe this will provide any further information with regard to how generalisable the agent formed on this environment will be.

Tasks	**Task Distance**
	Train
reach-v2, door-open-v2, peg-insert-side-v2	0.257
door-open-v2, drawer-open-v2, drawer-close-v2	0.269
pick-place-v2, drawer-close-v2, button-press-topdown-v2	0.230
pick-place-v2, peg-insert-side-v2, window-close-v2	0.215
pick-place-v2, drawer-open-v2, button-press-topdown-v2	0.264
push-v2, pick-place-v2, drawer-close-v2	0.197
drawer-open-v2, peg-insert-side-v2, window-open-v2	0.257
reach-v2, pick-place-v2, window-close-v2	0.214

Table 4.1: Meta-World MT10 — Multi-Task Learning task subsets.

As shown in Figure 4.1, we have 8 different task groupings, each consisting of 3 tasks, taken from the overall task distribution, to be learnt on, each with a varying inter-task similarity—this amount of tasks provides a good representation of the entire possible sampling population. Each of these task groupings were selected as a random subset of the overall possible tasks in the MT10 environment task collection. This allows us to analyse how the degree of task similarity (i.e. the difference in overall task distance across the set of tasks within the distribution) of the tasks within the distribution will affect generalisation and overall performance when dealing with learning across a set of tasks without having to generalise to a new set of tasks.

Tasks	**Task Distance**	
	Train	Test
reach-v2, push-v2, peg-insert-side-v2, basketball-v2	0.277	0.222
pick-place-v2, drawer-close-v2, peg-insert-side-v2, sweep-v2	0.262	0.234
reach-v2, drawer-close-v2, button-press-topdown-v2, peg-insert-side-v2	0.277	0.252
push-v2, door-open-v2, button-press-topdown-v2, basketball-v2	0.281	0.247
push-v2, pick-place-v2, peg-insert-side-v2, basketball-v2	0.264	0.205
push-v2, door-open-v2, button-press-topdown-v2, peg-insert-side-v2	0.280	0.223
reach-v2, button-press-topdown-v2, window-open-v2, sweep-v2	0.284	0.258
reach-v2, push-v2, door-open-v2, peg-insert-side-v2	0.280	0.217

Table 4.2: Meta-World ML10 — Meta-Learning task subsets.

As can be seen in Table 4.2, we have 8 task groupings each consisting of 4 tasks randomly sampled from the entire train set in the Meta-World ML10 environment. For the test tasks we use all the test tasks available within ML10 (sweep-into-v2, door-close-v2, drawer-open-v2, lever-pull-v2 and shelf-place-v2) across all provided train task combinations. We provide the train task distance, the inter-task distance amongst the train tasks selected, and test task distance which is the inter-task distance between each train task to the 5 test tasks.

The task distance for these tasks is defined as the mean pairwise cosine distance (stated here as $1 - \text{CosineSimilarity}$) between batches of observations between each tasks. These batches of observations for each task are sampled prior to the training of the models in the following experiments using a random policy for action selection. Due to using a distance function, we should see a negative correlation in the relationship between distance and performance (i.e. performance should decrease as distance increases), as per Figure 4.1. We use observations as the task representation here as due to the types of tasks we have, similar tasks should behave similar observations states—one downside of this is that it doesn't provide an innate measure of task learning difference as it doesn't encode task intent, only similarity in state position [CS05]. This metric was chosen as it is a commonly used method for defining the distance between two vectors.

With regards to reward structures in the v2 environment's tasks, we note that all Meta-World tasks have reward scaled between 0 and 10 (a value of 10 signifies a successful state for the agent) [1]—this therefore means that sampling across the different tasks, with regard to the task subsets, will have the same reward scaling and therefore should be comparable for difficulty by comparing reward during training. Another performance metric that we can use here is the task success rate.

The complete list of task subsets, and their associated task similarity values, for the Meta-World environment, for both the multi-task (MT10) and meta-learning (ML10) settings, are outlined within Appendix B.1.2.

4.4.2 Models

The definition of generalisation encompasses any situation that includes a diverse set of tasks within the task distribution used, whether this solely analysing the generalisation within a training procedure, or across a train/test split. As such, here we provide methods from within both MTL and meta-RL settings.

4.4.2.1 Multi-Task Learning Approach

For the MTRL approach we use policy distillation [RCG$^+$15, GVK19] [2]. Policy distillation comprises of a set of pre-trained teacher agents that are each trained on one of the tasks within the multi-task task distribution, resulting in a set

[1]This is mentioned in an issue in the Meta-World project code repository: https://github.com/rlworkgroup/metaworld/pull/312.

[2]The implementation used within the following experiments for policy distillation is an modification of code from `https://github.com/Mee321/policy-distillation`. This implementation uses Trust-Region Policy Optimisation (TRPO)[SLA$^+$15] for training teacher networks and Kullback-Leibler (KL) divergence for calculating the difference between student and teacher states for the loss function.

of *experts* performing optimally on each of the tasks. These agents' models are then distilled, using a regularisation term (commonly the Kullback-Leibler (KL) divergence between both policies actions in the respective task states, however other approaches such as MSE have been investigated [KOK+21]), comparing the model and a student model in order adjust the students' parameters closer to that of the teachers' model to train the student to have a representation contributed to by all teacher networks in order to learn optimal behaviour across all tasks. This is performed by sampling episodes from the environment, and sequentially adapting the students' parameters based on the performance from each of the teachers and student on the episodes from the teacher-associated task.

Algorithm 1 Multi-task learning with policy distillation [RCG+15]

```
Require: $T_{1...t}$: collection of $t$ teacher networks $T$
Require: randomly initialise $S_\theta$ student network
1: while not done do
2:     for all $T_t$ do
3:         Generate $D^T$ using $T_t$
4:         Run policy $\pi_S$ in $D^T$
5:         Optimise $L_{KL}$ wrt. $S_\theta$ on student actions $a_{S_t}$ from $D_T$ and $a_{T_t}$
6:     end for
7: end while
```

Algorithm 1 details the policy distillation method which can be summarised as the following: for a defined number of learning epochs we iterate over the set of pre-trained teacher networks, with model's optimised for each task $\mathcal{T}_t$, generate a set of state-action trajectories sampled from the teachers interaction with $\mathcal{T}_t$. The student network S_θ is then run on these task states to form a set of student actions a_S. We then use both the teacher actions a_{T_t} and a_S in a KL loss L_{KL} to minimise the difference in the actions between teacher and student. This allows for the parameters of the student network to be iteratively shifted towards each of tasks optimal representations (the teacher networks).

This method was chosen due to it being a method that learns each of the tasks optimally, in a single-task approach, and then distils into a single *student* agent. This allows us to highlight how altering the diversity of the task distributions will affect how well the teachers are able to transfer their learning to the student, and therefore highlight the generalisability of an agent's distilled overall task representation with regard to the task similarity across train tasks.

4.4.2.2 Meta-Learning Approach

For the meta-RL approach we use the Model-Agnostic Meta-Learning (MAML) [FAL17] method[3]. MAML is a gradient-based meta-learning approach that aims to learn a model that can quickly adapt to new tasks. There has been many extensions to this method that attempt to improve on it's ability to do this such as e-MAML [SYH+19] which look at improving episodic sampling distribution with regard to the exploration vs. exploitation trade-off—this is in comparison to the research presented in Chapter 5 which investigates the task-level sampling distribution. We provide the algorithm and further details for this method in Section 5.2.1.

This method was chosen as it is one of the grounding methods in gradient-based meta-learning. This provides a useful means of providing an insight into how the method reacts with regard to generalisation in a diverse multi-task meta-learning setting.

4.5 Results

The following are the results evaluating the uses and efficacy of the generalisation definition, as defined in Section 4.3.1, on the MTL and meta-RL methods, each of which are tested on both of the environments previously stated. The hyperparameters used throughout the following experiments for the training and testing of the agent are provided in Section B.2.

4.5.1 Adapted Environment Case: Pendulum

Here we provide the results for both the MTRL and meta-RL settings using the Pendulum environment, as outlined in Section 4.4.1. As detailed, this environment focuses on changes in transition dynamics, through the adaptation of both the mass and arm length of the pendulum, whilst maintaining the reward function—the task distance function is defined as the euclidean distance between the terms in the parameter-space.

[3]Implementation modified from [Del18].

4.5.1.1 Multi-task Learning Setting

In order to gain an understanding of how task distance affects MTL approaches, using the policy distillation method within these experiments, we sample task collections across the entire domain, as detailed in Section 4.4.1. As can be seen in Figure 4.3 we sample across the entire domain, generally centred on the centre of the domain, however increasing the task distance.

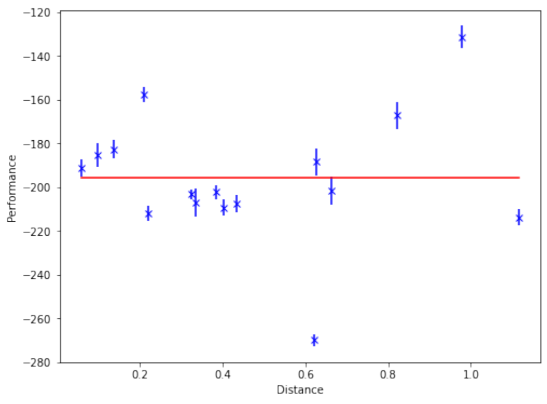


Figure 4.5: Performance (reward) vs. distance (task distributions parameterised by pendulum length and weight, similarity defined by inter-task euclidean distance). Generalisability experiment using a RL Pendulum task with Policy Distillation implementation [RCG+15] with tasks, sampled uniformly across bounds (as defined in Figure 4.3). Performance is defined as the mean reward of last 5 epochs, with error bars representing the standard deviation of 3 repetitions.

By training different agents, using the policy distillation method, across all of the collections tasks we produce a comparison, as shown in Figure 4.5, between performance, defined here as the weighted mean average reward of the last 10 episodes across all of the tasks within the task set, and distance, the mean euclidean distance across all tasks within each task collection. We can see that the performance stays constant as distance between tasks increases, however there is a large amount of noise and therefore no strong correlation. This goes against the intuition of how an agent's performance should degrade with increasing task distance. Following from the discussion on performance normalisation, this could be due to task difficulty, therefore it is important to see the effect that task difficulty is having on the generalisability of the agent.

We can see from Figure 4.6 that we don't have constant performance surface across the task distributions sampled from the domain. This indicates that certain tasks within this domain are inherently hard for the method—we see that the task

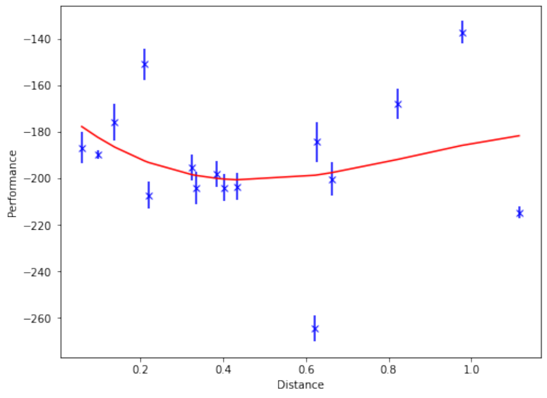


Figure 4.6: Performance (reward) vs. task distance for the expert models (single-task trained models which represents optimal performance for the agent on that particular task) across the Pendulum task distribution parameterised by pendulum length and weight. Performance is the mean reward of last 5 epochs, with error bars representing the standard deviation of 3 repetitions..

groupings with less inter-task distance have relatively similar mean performance, whereas when the distance increases we see tasks with higher mean performance. When we correct for this, by normalising the performance results by the single-task performance (provided by the teacher agents), we should therefore produce a non-constant correlation between performance and task distance.

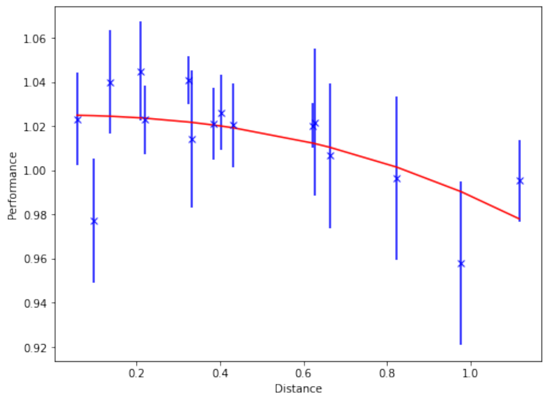


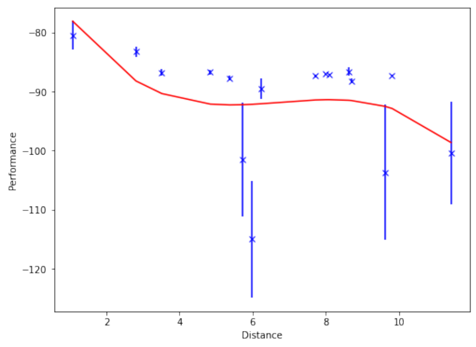
Figure 4.7: Normalised performance (reward / single-task reward) vs. distance (task distributions parameterised by pendulum length and weight, similarity defined by inter-task euclidean distance). Generalisability experiment using the RL Pendulum task with the policy distillation implementation [RCG^{+}15] with tasks, sampled uniformly across bounds (as defined in Figure 4.3). Performance is defined as the mean reward of last 5 epochs, with error bars representing the standard deviation of 3 repetitions.

The results from performing the normalisation are shown in Figure 4.7. This shows that when the difficult of the tasks is accounted the Policy distillation method forms a generalisible agent with regard to the Pendulum environment in this domain, due to a fairly flat slope in the fitted line approximating the relationship between performance and task distance—there is only a slight reduction in performance, from 1.02 down to 0.98 over the entire domain. We see that the relationship between performance and task distance is in line with the expected curve outlined in Figure 4.1, thus validating the task similarity metrics ability to provide an effective means of defining similarity, and showing that the method is able to generalise across this adaptation. This shows that even when the tasks become distant from each other, with regard to their initialisation parameters of mass and length of the pendulum, effecting the transition dynamics of the environment's MDP, the distilled agent is able to maintain near optimal performance, as defined by the teacher performance, when the degree of separation between tasks increases, with only a small degradation in performance. What we also can see is that the distilled agent performs better than the mean of the expert agent's on the task groupings with minimal distance between them which indicates it is sharing knowledge which is useful across the other tasks.

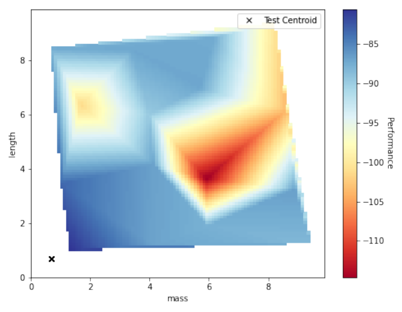
4.5.1.2 Meta-Reinforcement Learning Setting

Here we present the results for the meta-RL setting using MAML on the task subsets of the Pendulum environment. By using the process previously discussed we gather the performance of an agent, measured here as the weighted mean average returns from the final 5 epochs on both the train and test phases of the MAML method, across the different train and test pairs, which results in the following plot of task distance, euclidean distance between centroids of task subsets, against performance.

Comparing performance and task distance, as can be seen in Figure 4.8, there is a reduction in performance with task distance, however it is evident that the reduction is not smooth and consistent as is highlighted by Figure 4.8b. We see a reduction, approximately 10 reward on average, in performance up to a distance 5, which then becomes constant until a distance of 10, where performance the begins to degrade again. This could be an indication of an inappropriate task representation with regard to performance, or as mentioned in Section 4.3.2.1, it could be that the train distribution performance is not uniform in difficulty.



(a) Train-to-Test distance vs. Test Performance



(b) Test Performance across task distribution

Figure 4.8: Generalisability experiment using a RL Pendulum task with the MAML implementation [FAL17]. Task distributions parameterised by pendulum length and weight, sampled uniformly across bounds, with euclidean distance within parameter-space as task distance metric across train/test task distributions. Performance, the mean returns of last 5 epochs. Each point represents a train/test run of meta-learning on a defined train/test task distribution taken from the parameter-space.

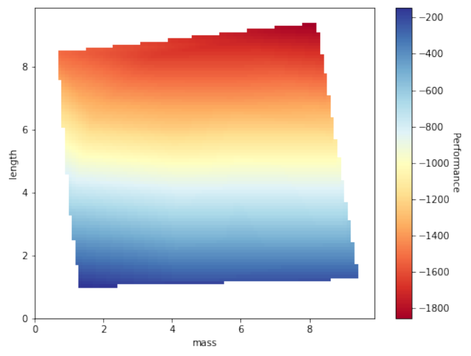
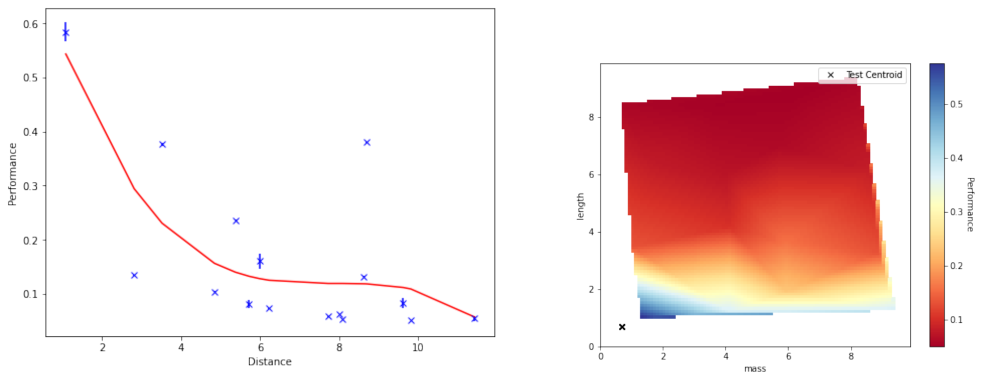


Figure 4.9: MAML performance (reward) across the Pendulum task distribution parameterised by pendulum length and weight. Performance, the mean reward of last 5 epochs, degrades as distance decreases.

Looking at the train performance across the task distribution, as shown in Figure 4.9, the train task distribution does not have a uniform difficulty over the task distribution. In particular, the length of the pendulum has a very large impact on the performance of the agent, where a larger length has a much greater impact on performance than an increasing mass. Therefore, as previously discussed, we need to perform normalisation on the testing performance.



(a) Normalised test performance vs. Task distance

(b) Performance across task distribution

Figure 4.10: Normalised performance (reward / train reward) against task distance on the RL Pendulum task with MAML implementation [FAL17]. Task distributions parameterised by pendulum length and weight, sampled uniformly across bounds, with euclidean distance within parameter-space as task distance metric across train/test task distributions. Performance, the mean returns of last 5 epochs. Each point represents a train/test run of meta-learning on a defined train/test task distribution taken from the parameter-space.

Having performed performance normalisation, the results of which are shown in Figure 4.10, we can see that the task distribution surface has a much smoother and consistent reduction in performance as task similarity decreases—this is in line with Figure 4.1 with regard to an exponential reduction in performance. In particular, we can see that the performance degrades rapidly over the tasks closer to the that of the test tasks and then reduces as the tasks gets further away. There is however still a period of constant performance as task distance increases between a distance of 6 and 10. We also see that, from looking at the performance across the task distribution, the agent is able to maintain performance for longer with changes in the mass, in comparison with length—when they are both increased away from the testing location we see a much larger degradation in performance. Overall, we see that MAML is unable to generalise well over small adaptations away from the train distribution, with the performance degrading quickly, however, although performance at higher distances is low, it is able to maintain a degree of performance after this drop. In particular, it is able to generalise better over mass changes compared to length as indicated by Figure 4.10b.

4.5.2 Multi-Task RL: Meta-World

This section details the results for both the MTL and meta-RL settings using the Meta-World environment (on the ML10 and MT10 environment task subsets, respectively, each with differing inter-task distances based on the cosine distance between task subsets), as outlined in Section 4.4.1. As previously mentioned, the Meta-World environment contains a set of MuJoCo tasks, focusing on robotic manipulation, in a diverse multi-task setting.

4.5.2.1 Multi-Task Learning Setting

Using the MTL method of policy distillation, as described in Section 4.4.2, we provide a set of task distributions from the Meta-World MT10 environment with varying inter-task distance and analyse how each of these affects performance, in this case the reward, with regard to the task distance.

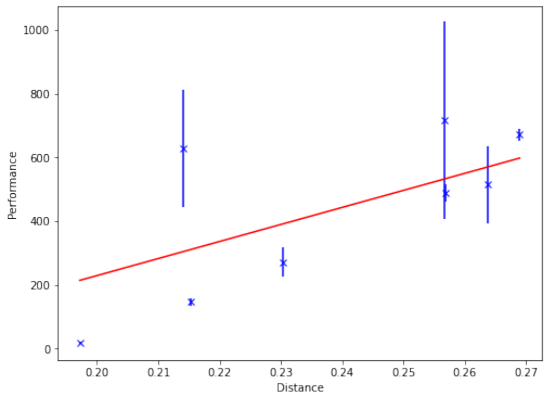


Figure 4.11: Performance (mean reward of last 5 epochs, with standard deviation error bars for 3 repetitions) vs. task distance (task distributions selected from cosine distance in observation space across sampled episodes within each task in the environment). Generalisability experiment using the Meta-World MT10 task collection with policy distillation implementation [RCG+15] with tasks as defined in Figure B.3.

When we compare task distance against performance on the task groupings formed from the MT10 environment tasks, as show in Figure 4.11 we see that as task distance increases, performance also increases, providing a positive linear correlation. This is the inverse of the expected behaviour as defined in Figure 4.1 with a linear, but positively correlated, relationship being shown when using the observation space-based task similarity measure, instead of a negative correlation with regard to distance metrics. Within the task subsets used, we see that over a relatively small increase in similarity, approximately an increase of 0.07, we see a large increase in performance, with mean reward increasing by 400 reward—it is relevant that this represents a strong relationship as we can then infer that the observation space, when used in MTL setting with the policy distillation method, may represent the inverse relationship. Another possible reason for this is that the relationship is due to the difficulty of tasks within each task distribution subset.

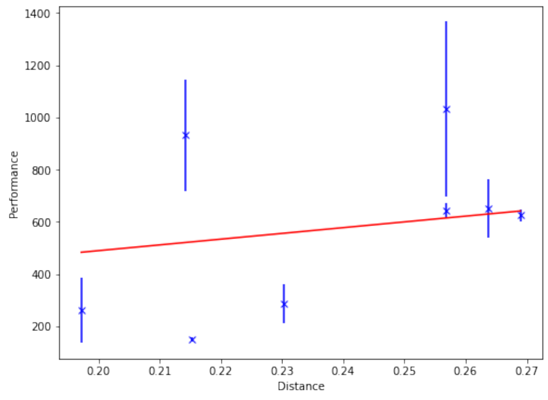


Figure 4.12: Performance (mean reward of last 5 epochs, with standard deviation error bars for 3 repetitions) vs. task distance for the expert models (single-task trained models which represents optimal performance for the agent on that particular task) on the Meta-World MT10 task collection with policy distillation implementation [RCG+15] with tasks, sampled uniformly across bounds (as defined in Figure B.3).

As can be seen in Figure 4.12, the overall task difficulty across the task distributions formed to analyse the generalisation of the policy distillation method is fairly flat, but does show variation across distance. The relationship between performance and task similarity shows that the tasks in the task groupings that are more spread out, i.e. have a higher distance between each other in the task representation space, contain easier tasks on average. This provides useful information about the environment with regard to how the observations pace is related to the difficulty of the task—it should be noted that this may be biased by the learning mechanism in this case, being TRPO, however only to a small degree as they are learnt separately in this method with the performances representing the mean of single-task performance. This therefore confirms that this is a factor in why we are showing the positive correlation, however in order to understand whether this was the only factor we need to view the train relationship with normalised task difficulty.

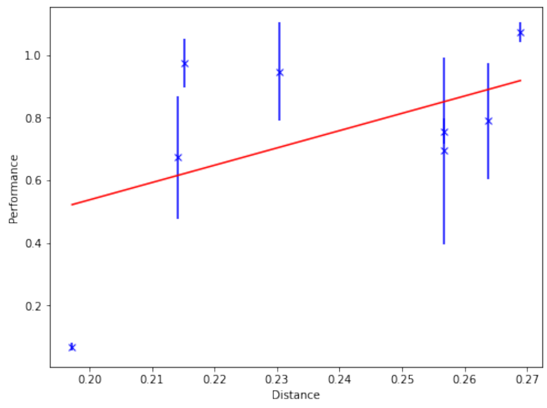


Figure 4.13: Performance (mean reward of last 5 epochs, with standard deviation error bars for 3 repetitions) vs. task distance (task distributions selected from task similarity in observation space across sampled episodes within each task in the environment). Generalisability experiment using the Meta-World MT10 task collection with policy distillation implementation [RCG+15] with tasks, sampled uniformly across bounds (as defined in Figure B.3).

After accounting for non-constant task difficulty across the task distribution, as shown in Figure 4.13, we still see this positive relationship between performance and task distance, instead of a negative relationship when considering distance metrics, where performance is increasing as the task similarity increases. The fact that there is a strong relationship however, we do gain useful understanding about the task similarity metric and the environment itself. This indicates that using episodic observations is an effective metric for comparing the performance with regard to generalisation in the Meta-World MT10 environment, however in the case of policy distillation, the relationship is inverse. Although the observation-based task representation doesn't directly encode performance, there does seem to be a relationship in the Meta-world environment where further task observations result in closer behaviour needed on with policy distillation. With regard to performance compared to optimal performance, we see that policy distillation is able to train a student model that has near optimal performance when task distance is high, in relation to the teacher models performance, and only reducing gradually until a large drop to ~10% at a distance of 0.2—this point may represent noise in the data, however it may indicate a transition phase for the generalisation of the method.

4.5.2.2 Meta-Reinforcement Learning Setting

Using the MAML method as described in Section 4.4.2, we use the task subsets from the on the Meta-world ML10 environments, as defined in Section 4.4.1, designed to provide an indication of how performance is affected by task distance within a meta-RL setting. After running the experiments on each of the train/test task grouping pairs, we can then analyse the performance, the mean success rate, as a percentile, of the last 5 epochs on the train set and adaptation steps on the test set, across all tasks against their task distance, as defined by the cosine distance-based metric on the observation space—the success rate is used in this case as it was found that the rewards across tasks remained fairly constant across the task distribution (the reason for this is due to the reward signals from the environment being distance based from the target rather than based on overall success, and is related to the scaling previously mentioned), whereas success rates were highly affected by task distance.

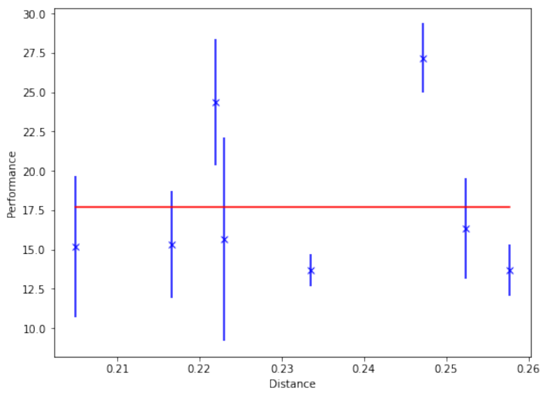


Figure 4.14: Test performance (success) against train-to-test task similarity using the Meta-World ML10 environment with MAML implementation [FAL17]. Task distributions generated as random sampling of 4 train tasks from overall environment distribution with task distance defined as the cosine between sampled observations across train/test task distributions. Performance is the mean success rate of the last 5 epochs, with error bars showing standard deviation for 3 repetitions. Each point represents a train/test run of meta-learning on a defined train/test task distribution. Fitted polynomial regression line to indicate relationship.

The results of running the MAML method on the task subsets of the ML10 environment are provided in Figure 4.14. We can see that the relationship produced is a constant performance across the entire task distribution's distances. Overall, most of the task groupings perform with a success rate of ∼15%, with two outliers having ∼25% success rate. We see that the relationship shown does not indicate any strong correlation, as no reduction in performance occurs over an increase in the task distance, as such not conforming with the expected curve as detailed in Figure 4.1. One of the main reasons for this could be that the MAML method is able to generalise effectively over the observation space based task similarity metric. Another possibility is that the result may be the effect of training difficulty on the generalisability of the agent on this environment's task distribution, as stated in Section 4.3.2.1.

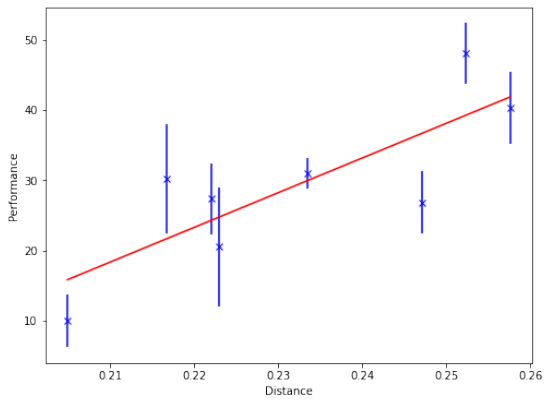


Figure 4.15: Train performance (success) against train-to-test task distance using the Meta-World ML10 environment with MAML implementation [FAL17]. Task distributions generated as random sampling of 4 train tasks from overall environment distribution with task distance defined as the cosine distance between sampled observations across train/test task distributions. Performance is the mean success rate of the last 5 epochs, with error bars showing standard deviation for 3 repetitions. Each point represents a train run of meta-learning on a defined train task distribution. Fitted polynomial regression line to indicate relationship.

We can see from Figure 4.15 that the task grouping difficulty, measured as the performance of each task grouping, is noisy and increases as the task distance increases—this graph represents the train performance when compared against the train to test task differences. This indicates that the task groupings do not provide a fair representation of the relationship shown in Figure 4.14, as certain task groupings include tasks that are inherently harder than others to be successful on, the performance measure we use here, and as such we need to normalise the test performance by the train performance, as discussed in Section 4.3.2.1.

The results of performing normalisation, due to the positive correlation found between train performance and test similarity which indicated a non-uniform task

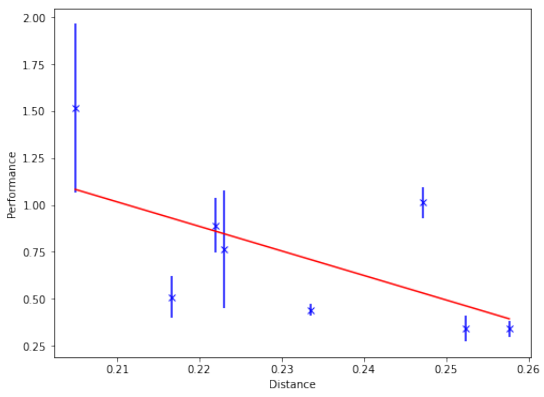


Figure 4.16: Normalised test performance (test / train success) against train-to-test task similarity using the Meta-World ML10 environment with MAML implementation [FAL17]. Task distributions generated as random sampling of 4 train tasks from overall environment distribution with task similarity defined as the cosine similarity between sampled observations across train/test task distributions. Performance is the mean success rate of the last 5 epochs, with error bars showing standard deviation for 3 repetitions. Each point represents a train/test run of meta-learning on a defined train/test task distribution. Fitted polynomial regression line to indicate relationship.

difficulty during training, are provided in Figure 4.16. As we can see, due to us normalising the test performance against by the positive correlation found in the train performance when compared with test distances, the resulting normalised relationship shows a negative correlation between test performance and task distance. This therefore exhibits the expected behaviour as defined in Figure 4.1 with a linear, but negatively correlated, relationship being shown when using the observation space-based task distance measure. With regard to performance relative to training, we see that when the test task grouping is close to the train grouping we are able to attain higher performance on the test tasks than the train tasks, however this rapidly drops to between ∼75% and ∼50% success rate as task distance decreases, but then flattens out. Overall, over this task distribution's domain, we can therefore say that the model is able to generalise well over this environments task changes, in particular focusing on the observation space adaptation.

4.6 Discussion

What we can see from the results is that across all of the environments and their associated task similarity metrics (across all of the experiments conducted here we instead use a task distance measure, which is the inverse of task similarity), we can critically judge the generalisability of the methods used with regard to a form of adaptation. The results indicate that with a suitable task representation and similarity metric, we can critically judge the generalisability of a RL model on any environment as long as you have three factors (a) you can adapt the environment to form a task distribution, (b) the task distribution forms a set of tasks that share a space that allows for a measure of similarity to be accessed, and (c) there is a measure of performance on each set of task collections. With this, we can form a task similarity against performance graph where the gradient of diminishing performance can be compared to understand how well the model generalises and copes with changes in the environment—this can be made use of in cases of adaptations within the same task (i.e. the Pendulum environment), or on more diverse MTL settings (i.e. the Meta-World environment). As discussed in Chapter 3, the task representation and the metric we use to compare tasks within this representation has a dramatic affect on the type of overall similarity we are able to gather. In the case of these experiments we used low-level environment-based task similarity metrics on both environments, with Pendulum represented in the parameter space, and Meta-World in episodic observations from the methods MDP formulation, however we look at using different performance measures including returns and success rates.

When we look the generalisability of the models, both MAML in a meta-RL setting and policy distillation in a MTL setting, we see both are effected differently when tested on adapted and diverse task environments. Results showed that policy distillation was able to generalise across the Meta-World environment's task distribution with regard to the observation space adaptations (this is with consideration that we found the inverse relationship between similarity and performance as expected from the observation-based similarity measure), and performed well across the entire Pendulum parameter-based domain, in particular in the mass adaptations, with a slow reduction in performance when tasks became more diverse in their similarity. Looking in regard to MAML, on the Pendulum environment the agent struggled to adapt over mass and length changes when tasks were more diverse with a rapid reduction in performance when tasks became dissimilar, and on the Meta-World environment, it was able to generalise and maintain good performance with changing observation similarity.

As highlighted from the results, through using this process of analysing the generalisability of an agent, not only can we derive understanding about the methods, but also about the environments themselves. In particular, in the Meta-World environment we chose the task representation space as the observations from a batch of episodes for each of the tasks. We found that policy distillation, the MTL method, showed a negative correlation between train to test task similarity and test performance. This is in comparison in the case of MAML, the meta-learning method, where the relationship was that as the task similarity increases (i.e. the similarity between the batches of observations from each of the tasks increases) the performance increases. This contrast in behaviour exhibited by the two methods using the same measure of task similarity indicates the importance of understanding the task similarity metrics in relation to the specific method. One of the main reasons that should be considered for why this is happening is that the observation space is a fairly naive space for use in a task similarity metric as it doesn't encode any performance information [CS05]—other more complex metrics can be used which may model the inter-relatedness of task more precisely [AET^{+}14], as discussed in Chapter 3. Also, this is an interesting difference from the expectations, and can be caused by either the MAML method dealing with adapting across tasks with different observation spaces more effectively, or the task themselves sharing similarities in behaviour when the observations are different. This highlights the importance of understanding the effect that the selection of task representation has on the performance against task similarity comparison in regard to the what a valid form this relationship should have, and what it means if it is not in that form.

4.6.1 Considerations

Throughout the experiments we encountered differences in the relationships between task distance and performance, that indicated an agents ability to generalise over a task distribution. However, it is vital to understand the possible factors that can affect the relationship that are not related to the generalisability of an agent and instead, the possible biases within the aspects of the process itself. We list them as the following:

- **Understanding task distribution difficulty within defined bounds**: When following the process defined above to form a value of generalisation for a method, you can't assume that the task distribution will be uniform in the tasks sampled difficulties. As part of the process of analysing generalisation using the described process we are required to sample tasks across the task distribution and the sub-performance of the task will affect the final relationship, defined as train performance in meta-RL and single-task performance in MTL. Although it can be useful to analyse how a method deals with performance differences in the sub tasks, as this could provide information over whether this could discourage overfitting, we must consider the impact of this for a fair analysis of generalisation. This can be accounted for by using normalisation of performance with regard to the difficulty of the tasks within the task distribution so that we can gather clear results with regard to only generalisation and not optimal performance, as outlined in Section 4.3.2.1. This is in particular a consideration that needs to be taken into account when selecting an environment that is compatible with this definition, with regard to how it adapts the environment.

- **Performance measure**: the measure of performance will dictate the expected form of the relationship. For example we want to minimise regret (the difference between the reward of the optimal step and the reward of the action the agent actually took), but maximise reward.

4.6.2 Uses

There are two directions available to having further knowledge about generalisation in RL: the first is understanding the current limitations of methods with regard to the types of environments they are able to perform well on, and leveraging this understanding to improve methods generalisability and ability to perform in environments that include diverse task distributions. Both of these can take advantage of the understanding between tasks relationships and the affect on performance through using it within different mechanisms.

4.6.2.1 Comparing methods

One of the main advantages of the definition is that we can use it to compare different agents, learnt from different methods, generalisability with regard to a specific form of task adaptation. Making use of the relationship describing the comparison of performance against rewards can be an invaluable method of quantifying the generalisability of a RL method on a given environment, and therefore their overall ability to generalise. In order to do this however, it is vital for the following to be maintained across all methods:

- **Same task similarity metric**: All tasks must use the same task similarity metric, meaning all methods represent the task distribution within the same space (e.g. episodic observations) along with using the same measure of distance, thus providing across method consistency of representation.
- **Same performance measure**: The performance measure must be the same, with regard to the measures in order for a fair comparison to be made (e.g. reward). Although we calculate a normalised performance measure, using measures such as regret and reward would provide a misleading comparison due to their target for optimisation—we aim to maximise reward whereas we minimise regret in general.

We have to consider that one, or both, of the chosen options for these two factors may not applicable to a certain measure (i.e. as previously mentioned there must be a negative correlation between performance and task distance for it to be a valid combination)—this is a drawback to this method when using it for comparison of methods, and requires appropriate task similarity metrics across methods.

Once we have results across a variety of methods on a given environment, we can then compare them on a graph of performance against task distance. An example of which is shown in Figure 4.17.

When we have a collection of methods comparing performance against task similarity, as mentioned in Section 4.3.1 we can measure the difference in the integral of the functions between performance and task distance. We can also analyse both the slope and the start and end values of performance. For example, if we assume that the lines in Figure 4.17 represent different methods, A and B, we can see that although Method A has lower optimal performance when task similarity is high, compared to Method B, as the gradient across most of the domain is flatter, the integral of the Method A's function is higher than that of Method B. Therefore, we can say that Method A generalises better than Method B with more diverse task distributions, even though it's optimal behaviour on

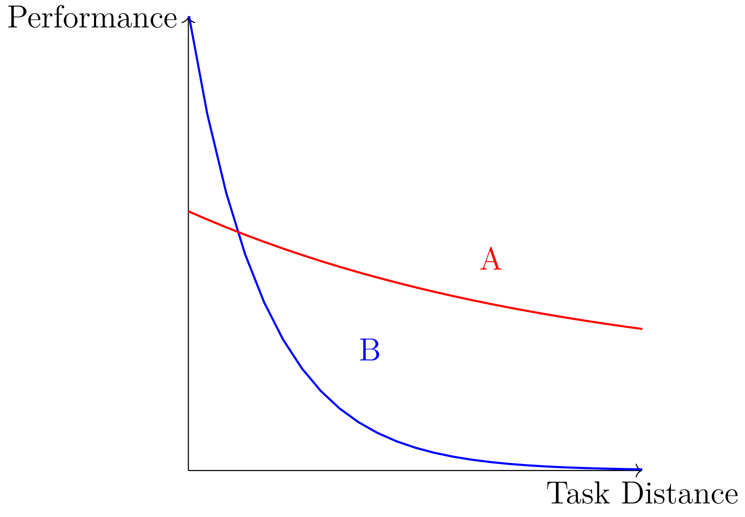


Figure 4.17: Comparison of methods on graph of *task distance* vs. *performance*.

the environment with no adaptations is lower. We perform this comparison on two meta-learning methods in Chapter 5.

Along with the comparison between different methods using the same task similarity metric, another usage is for analysing the different types of task adaptation in regard to the effectiveness of generalising across diverse task distributions. By incorporating different task similarity metrics, such as tracking both the observation space and reward space similarity, we can compare how the method is able to generalise across these two distinct measures of adaptation within the environment. For example, if we take the lines in Figure 4.17 as different task distance measures calculated from different task similarity metrics on the same method, we can perform the same analysis as described above. This provides a useful mechanism for investigating the benefits and limitations of a particular methods with regard to a particular form of adaptation, especially when comparing against these different forms of similarity.

4.6.2.2 Improving Generalisation

Along with using the definition for quantifying a methods generalisability, an understanding of how an agent should behave with regard to how they deal with diverse task representations in terms of their performance offers a learning target that can be leveraged by the agent to encourage generalisation. We propose that this relationship would be useful as a learning target or regularisation factor in a loss function to bias the learner we could encourage the learner to increase its generalisability rather than just optimal performance. As such, this would attempt to maximise the integral of the function defined by the agents' performance against task similarity. For example, we see work by [DCJ$^+$18] who use an auxiliary loss using cosine similarity of gradients which is intended to force the learner to adjust for task similarity, whereas using the generalisation definition with regard to the gradient of task similarity degradation would directly encourage generalisation.

4.7 Summary

This chapter has contributed a general approach, and its associated considerations needed, for quantifying the generalisability of a RL agent in both a MTL and meta-RL settings when dealing with environments containing diverse differences in the task representation. In comparison with other methods looking at generalisability in RL, this approach allows for any method to be analysed using only a task similarity metric, defined by both the task representation and the measure of distance between tasks in the task distribution, and a value of performance from the agent on those task sets. It does however require that the environment that we are sampling our training and test tasks from allows for the tasks to be adapted, including in offering subsets of several different diverse tasks, along with including adaptations of the MDP itself. By only defining these requirements on the analysis of we open up the possibility of analysing a larger collection of RL method's ability to generalise. The definition presented here provides a quantifiable measure of generalisation with regard to a type of adaptation, taken as the integral of the performance and task distance function, along with the start and end performance across the task distance domain. It also provides a understanding of *why* a method is able to generalise, by linking performance to a task similarity metric which, depending on the level of abstraction, can provide information over the skills shared across tasks, as detailed in Chapter 3. Comparing the results across generalisation definitions would support the generalisability of a method, however the reason behind using each definition depends on the requirements that you have in terms of the method and environment, and the aspect of generalisation that is being investigated. This does however show the difficulty of defining a generic approach for analysing and fixing generalisation in RL, and highlights the importance of using multiple approaches to uncover both whether and why a method is generalising.

One vital aspect that RL needs to focus on is to provide environments where we can form adaptations in the tasks, following on from the work of [YQH+19, CHHS19], as we need to be testing any new method with regards to not only their performance but also generalisation to ensure that the paradigm can find use within real-world applications where adaptation in the environment will most definitely exist, as discussed in Section 2.1.1. Many of the current benchmarks, and the associated environments, don't provide train and test environments that sufficiently facilitate analysis of task distribution shifts which are vital to allowing RL methods to increase their generalisability. There has recently been important steps towards fixing this, as discussed in Section 4.2, with the introduction of environments dealing with different forms of task distribution adaptations and train to test domain shifts. However, to facilitate the assessing of all forms of RL methods with regard to their generalisations we need to expand the environments to cover all the possible applicable problem areas.

One of the main focuses of future research would be in expanding this method as to enable it to be used to gain a further understanding of how to correctly use a task similarity metric to leverage the relationship between task distance and performance to improve a given method. As mentioned in the Section 4.6.2.2, there are different usages of this definition that would help in gaining further understanding of generalisation in RL and aid in remedying it. As such, another focus of future research would be to use this approach on methods within both MTL and meta-RL, to ensure that the generalisability of RL agents is taken into account, as it still remains one of the largest challenges to the paradigm being used in real-world applications.

Chapter 5

Gradient-Based Task Similarity Generated Curricula in Multi-Task Meta-Reinforcement Learning

5.1 Introduction

The gradient-based meta-learning method, Model-Agnostic Meta-Learning (MAML) [FAL17], uses a reference task distribution $p(\mathcal{T})$ over the set of available tasks during training for the selection of tasks to sample from. Most implementations use a uniform distribution across these tasks [MDR$^+$20]. This approach makes the assumption that all tasks can be equally useful in training an adaptive and generalisable agent, as all tasks will be sampled at the same rate during training. The field of curriculum learning [BLCW09] has suggested that this is not the case, and that a targeted curriculum of tasks can encourage certain attributes of the agent depending on the available environment and the target of the curriculum. Curriculum-based learning focuses on defining a structure to what data, and its order, the agent is exposed to over the course of learning. This can have many aims from increasing optimal performance and, to the more relevant here, combating overfitting in order to improve overall agent generalisability. There have been several uses of curricula across machine learning, providing a means of controlling the way that an agent learns through the data distributions that they are exposed to [SLK$^+$17], which has illustrated the usefulness of a non-uniform task distribution.

However, there is still a large amount of open questions related to applying non-uniform task sampling distributions in RL in order to form curricula [NPL+20]. More specifically, we find there is limited research looking at this in a meta-learning framework when applied to adaptive RL environments. Within current literature, work by [MDR+20] is one of the only works, at the time of writing, that we can find on the topic that looks at non-uniform task sampling distributions to form curricula in meta-RL. This work looks at using active domain randomisation (ADR), which uses reward difference between pre- and post-adaptation to act as a guide for task difficulty, to form a task distribution that focuses on the difficulty of tasks. Although this work shows promising results in improving performance, rewards do not present a complete metric (i.e. useful for describing all relevant task similarities) for guiding an increase in generalisability of an agent, as detailed in Section 4.2. Another approach related to the use of non-uniform task distributions is that of [JHE+19] who use a visual task representation in order to meta-learn an optimal task sampling distribution using an expectation-maximisation setup. This work indicates that forming an adaptive curriculum provides control of both generalisation and overall transfer performance. This work looks at forming a process for meta-learning a task distribution, whereas the work discussed in this chapter focuses on using a task representation already learnt during the standard meta-learning pipeline. Following these, we also note work by [GL21] who have looked at the selection of relevant train tasks with regard to test tasks, focusing on the pre-selection of training tasks that optimises for performance using the policies learnt on the train tasks and used in testing—this is similar to work by [GEFL20] who look at task selection based on a mutual-information objective. This work highlighted the importance of task relevance during training with regard to a performance target, as defined by the task selection criteria. Finally, work by [WLCS19] attempts to evolve an environment through adaptations according to a learning target to modify the task distribution, whereas work in this chapter is focused on the selection of tasks from a pre-defined set of tasks. Across all of these approaches, we find that the relevance of task distributions with regard to generalisability is often overlooked.

In this chapter we build upon previous work looking at non-uniform task sampling distributions through the use of task similarity metrics which have been shown to illicit behaviour that encourages generalisation across different forms of task adaptation, as discussed in Chapter 3. This is motivated by the following aspects: firstly, we introduce two gradient-based task similarity metrics, which we aim will provide an insight into the applicability of a gradient-based task representation in forming useful separation between tasks during training. Secondly, we aim to further understand the impact of a non-uniform task sampling distribution on the generalisability of a meta-learning agent. Finally, by allowing the agent to update the task sampling distribution during the training phase, we aim to highlight how evolving the curriculum throughout learning can adapt to the requirements of the agent at that time, and that current approaches of forming curricula, constructed prior to learning, limit the ability of an agent to learn effectively.

The contributions provided in this chapter are as follows:

- We provide two gradient-based terms, gradient change and co-task similarity, that are shown to able to capture useful task similarity information with regard to how tasks relate to each other in an agent's model parameter-space and form effective separation in the task representation.

- By exploring the use of a non-uniform task sampling distribution defined by gradient-based similarity within the MAML method, we show that both the gradient change and co-task similarity terms can be used to form a non-uniform task distribution that encourages increased performance in terms of adaptation to new tasks.

- We compare the non-uniform task sampling method against the baseline use of a uniform task sampling distribution within the MAML method. This provides further evidence that a non-uniform task sampling distribution can increase meta-learning performance over using a uniform task sampling distribution, hence adding further evidence to support the importance of task relevance during training.

- We introduce a mechanism for the creation of a non-uniform task sampling distribution within the MAML method that combines the gradient change and co-task similarity metrics in an equation that facilitates automatic curriculum learning in order to encourage generalisability across a diverse task distribution. This highlights the importance of updating the tasks used to learn according to the current state of the agent. It also indicated that by evolving the gradient-based non-uniform task sampling distribution we are able to improve agent generalisability, in this case with regard to observation adaptations, within a meta-learning setting on an environment containing a diverse task distribution.

Overall, this chapter is therefore split up into two main sections: in Section 5.2 we look at what information we can use from a meta-learning method to form a task similarity metric, which we then investigate as to how it affects the task sampling distribution in meta-learning in order to analyse the validity of its usage in forming non-uniform task sampling distributions. Section 5.5 makes use of the results from the analysis of task similarity metrics on task sampling distributions to form an adaptive curriculum using the suggested information in a task similarity metric that encourages an improvement in the generalisability of MAML. In Section 5.6 we provide a discussion of the results across both of these sections related to the usage of a gradient-based non-uniform task sampling distribution. Finally, Section 5.7 provides some concluding remarks for the chapter.

5.2 Gradient-based Task-Sampling for Meta-Reinforcement Learning

As discussed in Chapter 3, an optimal task similarity metric encodes information about the approximate value of learning between two tasks [CS05]. One of the main approaches in meta-learning is that of gradient-based meta-learning. This uses task-associated parameter updates, defined by the gradients, to update a shared model in order to form a representation that can adapt quickly to new and unseen tasks. As such, we propose using gradients as our task-specific representation, as this directly encodes the value of learning required for the current task. We therefore detail the following hypothesis:

Hypothesis 1. *Gradients from a meta-learning agent attained during training will produce task similarity metrics which are representative of the similarity between tasks within a diverse task distribution, and therefore will be useful in forming curricula, through adapting a non-uniform task sampling distribution, for that meta-learning agent, in this case MAML, to improve generalisability when compared with a uniform task sampling distribution.*

This hypothesis will therefore be investigated in this section by analysing methods that use gradients from a meta-learning method, in which is MAML [FAL17], in relation to their effectiveness in defining similarity between tasks in an environment to form a low-level behaviour-based tasks similarity metric, by analysing the tasks that are chosen when used as weights in a task sampling distribution and the affect that this has on performance. In Section 5.5 we shall then introduce a method that forms a non-uniform task sampling distribution, in which its weights are calculated from gradient-based task similarity. Using this we will analyse if it encourages tasks that, at certain points during the training cycle, will aid in training the agent to generalise across diverse and adaptive task distributions, using the definition described in Chapter 4.

5.2.1 Algorithm

The algorithm that we present here is an extension of the standard Model-Agnostic Meta-Learning (MAML) method [FAL17]. This algorithm is a general-purpose optimisation method that is model and task-agnostic, in that it is designed to work with any model or task, that learns through gradient descent. As mentioned in Section 2.2.3, it's designed to encourage fast adaptation to new and unseen tasks, which entails needing only a few interactions with the a new task and therefore few gradient updates, to quickly learn optimal behaviour on the new task.

This attempts to optimise the following the standard RL algorithm:

$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = -\mathbb{E}_{\mathbf{x}_t, \mathbf{a}_t \sim f_\phi, q_{\mathcal{T}_i}} \left[\sum_{t=1}^{H} R_i(\mathbf{x}_t, \mathbf{a}_t) \right] \tag{5.1}$$

where the RL loss is the expected returns, R, over a horizon H, when we take actions, $\mathbf{a}_t$, sampled by the model's policy, f_ϕ, within a specific task, $\mathcal{T}_i$.

The MAML method detailed in Algorithm 2 consists of two loops: the inner loop which updates the agent's model using trajectories, $\mathcal{D}$, sampled from a task, $\mathcal{T}_i$, and updates the model's parameters, θ_i, to θ_i' across all of the tasks ($\mathcal{T}_{i..T}$) sampled from the task sampling distribution p. This attempts to optimise model parameters for each of the tasks, thus forming task-specific models. The outer loop then, with only a few newly sampled trajectories from each of

Algorithm 2 MAML for Reinforcement Learning using Gradient-based Task Similarity

```
Require: $p(\mathcal{T})$: distribution over tasks
Require: $\alpha$, $\beta$: step size hyperparameters
 1: Randomly initialise $\theta$
 2: while not done do
 3:     Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
 4:     for all $\mathcal{T}_i$ do
 5:         Sample $K$ trajectories $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{a}_1, ...\mathbf{x}_H)\}$ using $f_\theta$ in $\mathcal{T}_i$
 6:         Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ using $\mathcal{D}$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation 5.1
 7:         Compute adapted parameters with gradient descent: $\theta'_i = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$

 8:         Sample trajectories $\mathcal{D}'_i = \{(\mathbf{x}_1, \mathbf{a}_1, ...\mathbf{x}_H)\}$ using $f_{\theta'_i}$ in $\mathcal{T}_i$
 9:     end for
10:     Update $p(\mathcal{T})$ using gradients from each $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ in Equation 5.2
11:     Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_{1...i} \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i})$ using each $\mathcal{D}'_i$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation 5.1
12: end while
```

the updated per task models $\mathcal{D}'_i$, optimises for the objective detailed on Line 11 across all tasks—these are referred to as meta-parameters that attempt to find a representation that facilitates fast adaptation to new tasks.

As indicated by Figure 5.1, during the inner-loop the agent uses it's current representation θ to produce a set of trajectories within the associated task, $\mathcal{T}_i$, that are used in the meta-objective, detailed on Line 6, to the compute adapted parameters across each of these tasks. The gradients from each of these task interactions provide an indication of the location within the parameter-space which contains the optimal representation for the agent. This describes the parameters that would make the agent behave within the task's environment optimally according to the RL objective, as described in Equation 5.1, to maximise the expected discounted reward. As such, these gradients provide a guide for how the agent needs to behave within an environment with regard to performance as defined by its parameter representation. This is a low-level behaviour-based task similarity metric, according to the taxonomy detailed in Section 3.2, due to it being innately linked to the behavioural representation of the agent.

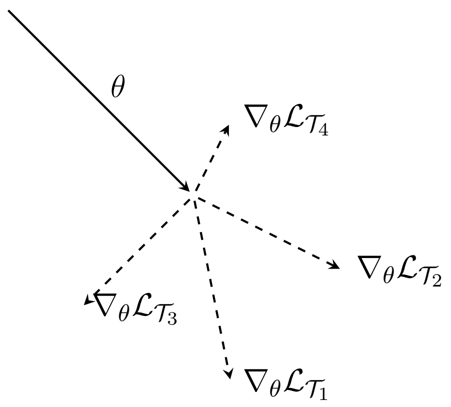

θ
$\nabla_\theta \mathcal{L}_{\mathcal{T}_4}$
$\nabla_\theta \mathcal{L}_{\mathcal{T}_2}$
$\nabla_\theta \mathcal{L}_{\mathcal{T}_3}$
$\nabla_\theta \mathcal{L}_{\mathcal{T}_1}$

Figure 5.1: Diagram detailing inner-loop parameter updates for the Model Agnostic Meta-Learning (MAML) [FAL17] method.

5.2.1.1 Extension

We extend this base MAML algorithm by adapting the task sampling distribution based on the gradients, $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$, taken from each of the task associated inner loops (Line 6)—we identify the gradients of a task by a task identifier. This takes advantage of the gradient representation of an agent on each of the tasks as a guide to how the agent needs to behave to be optimal within an environment, as it describes the learning process of the agent with regards to it's performance on the task distribution. We therefore state that we can leverage this information and compare it across task interactions to understand how tasks are similar with regards to per-task parameter space based behaviour. The gradients for the updating of parameters of a neural network have been shown to provide useful information in describing the similarity of tasks [DCJ+18], along with being shown that they can be useful in describing situations of positive and negative transfer [YKG+20]. Overall, this combines the optimisation-based approach of MAML with a metric-based task similarity learning approach as discussed in Section 2.2.3. The MAML method defines that a reference task distribution $p(\mathcal{T})$ over the set of tasks is used for the sampling a subset of tasks at each epoch as detailed on Line 3 from Algorithm 2. As previously mentioned, many of the current implementations of MAML utilise a uniform distribution for task selection [MDR+20]. However, in the extension the inner-loop task gradients are used to form a task similarity metric, describing the relationship between the tasks in the environment's task distribution by the gradients of the parameter updates from each task, as distribution weights to form a non-uniform task sampling distribution. Overall, the exact modifications to the base algorithm are listed as follows:

- **Line 3**: We sample from $p(\mathcal{T})$, which is now a non-uniform task sampling distribution with weights defined by the inner-loop task gradients as a means of stating each tasks similarity to the current representation and other sampled task's gradients.
- **Line 10**: The task sampling distribution $p(\mathcal{T})$ is updated using Equation 5.2 using the gradients $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ taken from each task associated model update within the inner-loop—this encodes the agent's optimal performance for each of the tasks in the environment's task distribution.

As mentioned above, the task sampling distribution is updated on each outer-loop iteration using the gradients of the per-task inner-loop. The equation that we use to calculate the new task weights for the task sampling distribution update is detailed as the following:

$$\begin{aligned} \text{Softmax}(-(g * \text{GradientChange}(\nabla_\theta \mathcal{L}_{\mathcal{T}_{i..N}}(f_\theta)) \\ + c * (1 - \text{CoTaskSimilarity}(\nabla_\theta \mathcal{L}_{\mathcal{T}_{i..N}}(f_\theta)))/\tau) \end{aligned} \tag{5.2}$$

where g ∈ [-1, 0, 1] and c ∈ [-1, 0, 1]

Equation 5.2 calculates the softmax of two different terms: the *gradient change* and *co-task similarity* terms are both expressed by the gradients of each tasks model updates, $\nabla_\theta \mathcal{L}_{\mathcal{T}_{i..N}}(f_\theta)$ where N is the total number of tasks in the task distribution. Using the softmax, with a temperature, τ, to control the degree of separation between the weights, allows us to calculate weights for the task distribution. Each of these terms have a hyperparameter, g and c, which controls the affect that each of these terms have. Depending on the term, these will either encourage closer or further tasks, based on their gradients and therefore whether the current model that represents the optimal performance model on the new task should be sampled more or less. These hyperparameters perform as switch values (therefore the possible values being -1, 0 or 1) which allows the term to either not be used (value of 0), encourage more similar tasks to be sampled more (value of 1) or less similar tasks to be sampled more (value of -1). This equation adds the two terms together, which means that each of the terms have an equal weighting in the adapting of the task sampling distribution weights—it will therefore be vital to standardise the two terms when combined to ensure that they have the same weighting. Having the equation be an addition of the two terms will be useful for analysing the affect that each of these terms has on modifying the task sampling distribution, as we can control, using the hyperparameters, the affect that each term has. We also add a negation of the result of the two terms being added together. The effect of doing this is that the default behaviour of this equation, when both g and c are set to 1, is to encourage similar tasks to be chosen first. The motivation behind this is that the encouraging the sampling of similar tasks has been found to be highly useful in increasing performance within curriculum learning methods [BLCW09]. However, one of the main issues in RL is that task distributions generally only focus on similar tasks, and therefore methods are designed to be performance on these single-task cases. As such, we include the possibility to inverse this, as a hyperparameter in the implementation, as we want to also gain further understanding of the affect of forcing distant tasks to be sampled in order to understand the impact on the agent's performance when this happens.

5.2.1.2 Gradient Change

The first term in Equation 5.2 is that of gradient change. One aspect of gradients that we can capture with regard to similarity is how far away a task is wanting to move the current representation based on the experience the current model has had on the task. Therefore, in terms of capturing this within gradients, the intuition behind this is defined as the following:

Intuition: How far the "optimal" model for the task is from the current model parameters.

We capture the intuition behind gradient change in the following equation:

$$\text{GradientChange}(\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)) = \|\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)\|_2 \tag{5.3}$$

Equation 5.3 states the equation for gradient change which is defined as the ℓ_2 norm of the per task gradients taken from Line 6 of Algorithm 2. This provides a measure of distance between the current model parameters and the model parameters that are seen by the method as optimal for the task. In terms of understanding how the values represent task similarity, this therefore means that a low value of gradient change indicates the task is close to the current model parameters, whilst higher values show the representations are dissimilar—as such this is a distance metric.

5.2.1.3 Co-task Similarity

The second factor in Equation 5.8 is that of co-task similarity. Another aspect of similarity in gradients within a vector space is that of the angle between two tasks' gradients (i.e. the direction within the parameter-space in which the optimal representation for that task is located). Therefore, with regard to how we can find a similarity between tasks' gradients, we have the following intuition for this term:

Intuition: Are the directions of the gradient vectors of a task in the same direction as the other tasks or do the directions conflict with each other.

We capture the intuition behind gradient change in the following equation:

$$\text{CoTaskSimilarity}(\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)) = \frac{1}{N}\sum_{n=1}^{N} \cos\left(\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta), \nabla_\theta \mathcal{L}_{\mathcal{T}_n}(f_\theta)\right) \tag{5.4}$$

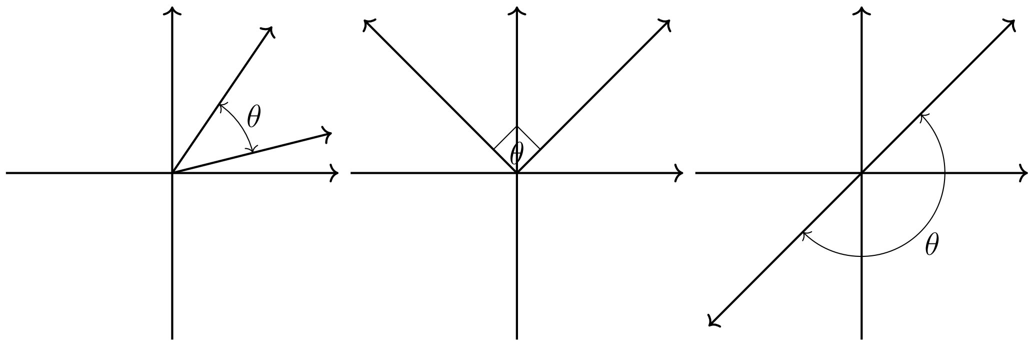
where

$$\cos(\boldsymbol{x},\boldsymbol{y}) = \frac{\boldsymbol{x}\cdot\boldsymbol{y}}{||\boldsymbol{x}||\cdot||\boldsymbol{y}||} \tag{5.5}$$

Equation 5.4 details the per task co-task similarity which is defined as the mean cosine similarity, defined here as cos cos, between each of the tasks' gradients. We use cosine similarity as it provides a similarity based on the angular distance between two vectors—here we don't use the inter-task distance between vectors as this is partially encoded within the gradient change term.

As shown by Figure 5.2, when comparing vectors with cosine similarity their relationship is generally classified by three different states: similar (when the value is close to 1), orthogonal (value close to 0) and opposite (the value is close to -1). Using this, we are able to define whether the directions of gradients for each of the task associated model parameter updates are similar or dissimilar.

It should be noted that cosine similarity is not considered a proper distance



(a) Similar gradients (Value ≈ 1). (b) Orthogonal gradients (Value = 0). (c) Opposite gradients (Value = -1).

Figure 5.2: Cosine similarity categorisation.

function as it does not have the triangle inequality—due to cosine similarity not having a linear relationship when you graph the similarity mappings against $x = cos(\theta)$ where θ is the angle between two vectors, when you have small degrees of change between vectors they can be seen as more similar than they should (this is a lack of precision when you have a small angle between two vectors). As such, we can instead define the metric as angular similarity, stated as

$$1 - (\arccos(\text{CosineSimilarity})/\pi) \tag{5.6}$$

This results in the co-task similarity metric being defined between 0 and 1. In the ablation study detailed later we define the co-task similarity as $1 - \text{CosineSimilarity}$, which therefore states that this is a distance measure where 0 signifies similarity, as although it is not a proper distance function in the negative space we attain similar results as using angular similarity, and therefore the co-task similarity results are bounded between 0 and 2. We also use just cosine similarity in the adapted equation as well as it produced similar results as using the angular similarity but it was more computationally efficient.

5.2.1.4 Forming a Curriculum

The aim of the two terms within the overall task similarity metric, gradient change and co-task similarity, are to form a useful representation of the similarity between the tasks using only the gradients from the model's updates when trained on trajectories from a particular task. We can then use this to form an curriculum by using the results of Equation 5.2 as the weights for a probability distribution over available tasks. This curriculum should be able to decide the tasks that are relevant for improving, or are detrimental to, the performance of the model, depending on performance target. Considering this, we can use the task similarity to encourage the following curriculum-based task sampling targets:

- **Encourage tasks that are close to/far from the current representation**: this is controlled by the gradient change term. By targeting tasks that are close to the current representation we can focus on similar tasks, which should have a similar optimal representation. When the tasks are furthest from the current representation this means more interactions with the environment is required to move the current model parameters closer to that optimal.
- **Encourage similar/dissimilar tasks to each other**: this is controlled by the co-task similarity term. By learning similar tasks together should lead to adaptation to performing optimally on those close tasks—this could however lead to overfitting on a subset of tasks. Whereas, focusing on the tasks that are dissimilar to each other can ensure we learn across the entire task distribution, but can therefore cause underfitting if the tasks are too far away from each other.
- **Combination of the above**: by combining both of the previous curriculum targets, using both the gradient change and co-task similarity terms, we can target both of the behaviours at the same time weighted by the hyperparameters. This can offer a balance between the two terms' trade-offs.

These task similarity-driven curricula offer flexibility over what tasks are targeted in sampling. This provides a means of encouraging generalisation in an agent's overall task representation by balancing overfitting and underfitting, as curriculum learning has been shown to be able to do [BLCW09], with each term offering a trade-off between these two depending on the value of the hyperparameter. Each of the terms provide a way of managing either overfitting or underfitting, with each term providing a mechanism of managing either effects in different ways, whilst the combination provides a way of balancing the two.

The hyperparameter combinations with associated intended curricula, related to the likelihood of overfitting or underfitting, that are included within Equation 5.2, are detailed in Table 5.1.

g	c	**Curriculum Description**
0	0	N/A — no preference on similarity
0	1	Encourage tasks most *similar* to each other
0	-1	Encourage tasks most *dissimilar* to each other
1	0	Encourage tasks most *similar* to the current model
1	1	Encourage tasks most *similar* to the current model and *similar* to each other
1	-1	Encourage tasks most *similar* to the current model and the most *dissimilar* to each other
-1	0	Encourage tasks most *dissimilar* to the current model
-1	1	Encourage tasks most *dissimilar* to the current model and *similar* to each other
-1	-1	Encourage tasks most *dissimilar* to the current model and *dissimilar* to each other

Table 5.1: Gradient-Based Task Similarity Hyperparameter combinations of g and c.

5.2.1.5 Other Details

With the addition of Equation 5.2 to the MAML method, there are specific aspects of this equation that need to be controlled for. Along with these, we also attempted to control for some of the fundamental noise found within RL gradients [NIA+18], in particular in model-free methods, to form a more consistent task similarity representation and therefore a more stable task sampling distribution—this was also found when using the cosine similarity of gradients in work by [DCJ+18].

As such, we detail the mechanisms that we investigated as followed:

- **Experience replay**: We use an experience replay buffer [ZS18] in order to reduce noise across several epochs of training. Across the past n epochs we store the task similarity for each task (this is the combination of the two terms without the softmax)—we save the task similarities rather than the gradients for computational efficiency. The task similarity at a particular epoch is then calculated as the discounted weighted arithmetic mean of the task stored in the buffer. Due to the task similarity rather than the gradients being stored, when we use a replay buffer we have to consider the consistency of metric spaces across epochs. As such, when we perform normalisation on the metrics, as we don't know the bounds of the metrics (in particular the gradient change metric) we therefore use the batch approximation of the bounds for normalisation. Overall, this allowed the task similarity metric defined in Equation 5.2 to produce task sampling distributions that produced separation between tasks with reduced noise.

- **Accounting for non-sampled tasks**: During the sampling of tasks, detailed as Line 3 in Algorithm 2, there will be epochs where some of the tasks are not sampled from—this happens without the extension, but due to us generating a non-uniform task sampling distribution, this is exacerbated. As such, to account for no gradients for a specific task in the iteration we take the mean task similarity across all tasks. This is in particular important when we use a memory-less approach to the calculation of task weights, as this imitates the fact that we currently have no information on this task and therefore can't bias the sampling for this task. However, when using experience replay, we still take the mean but we are able to use task similarities from previous iterations to inform the method of the appropriate weighting for the task.
- **Hyperparameter schedules**: Whilst looking at the affect of using the hyperparameters as switches for each of the terms, we also investigated using a hyperparameter schedule—for example a decay over the course of the training cycle (i.e. in terms of a task sampling curriculum initially the metric would encourage similar tasks, to then by the end of training encourage dissimilar tasks). This allows for the targets of the metrics curriculum, as outline in Section 5.2.1.4, to be adapted during training. The results of the ablation study has been limited to constant hyperparameter values as this allows for a clearer indication of the impact and effectiveness of each of the terms. However, it may be possible to mix curriculum targets by varying hyperparameters values throughout training.
- **Epsilon-greedy task distribution selection**: We use an epsilon-greedy [SB18] task sampling distribution selection policy for choosing between the uniform task distribution and the task similarity-based non-uniform task distribution—this means we are able to trade-off between exploration of tasks out of curriculum and those within the curriculum. We select the task similarity-based task distribution with a probability of $1 - \epsilon$ and the

uniform distribution with a probability of ϵ. The main intention of this is to ensure that when we have hard softmax probabilities, a temperature close to zero which causes sampling to predominantly select from only a small subset of tasks, by using the epsilon-greedy selection policy we can limit overfitting to those tasks and ensure other tasks are sampled as well.

Throughout the following results section, we detail which particular mechanisms are used during the experiments.

5.3 Experimental Setup

The following experiments are split into an two different aspects: the first set of experiments focus on performing an ablation study looking at whether the different terms, using task gradients, create a representation that is able to be used to discriminate tasks by a measure of similarity and therefore be used to form the task sampling-based curricula described in Section 5.2.1.4; and, the second experiment details a hyperparameter-free modification of Equation 5.2, as presented in Section 5.5, which performs automatic curriculum learning using a gradient-based task similarity metric.

Throughout these experiments we include a baseline that uses a uniform task sampling distribution in the standard MAML [FAL17] method, which is used as a comparison against the added extension detailed in Section 5.2.1 which detailed the mechanism for the updating of a non-uniform task sampling distribution. In the experimental results we label the baseline as MAML and the particular configuration of the extension with the relevant values. The MAML implementation used in these experiments is a modification and extension on the code repository provided by [Del18], which uses the Trust Region Policy Optimisation (TRPO) [SLA+15] method for the RL loss—this is an on-policy RL algorithm that updates the gradients by the largest step size possible limited by a constraint based on the difference, using the KL divergence, between the new and old policies conditioned on states visited by the old policy.

The hyperparameter values across all the ablation study variants remained the same apart from the hyperparameters controlling the gradient change and co-task similarity terms in Equation 5.2 as they are the central terms to be analysed across the ablation study. We also modified the softmax temperature, τ, depending on the size of dataset (ML10 contains 10 train tasks and ML45 consists of 45 train tasks). To ensure separation of task sampling weights we harden the softmax method by setting the temperature according to the following heuristic:

$$\tau = \frac{1}{\text{Number Of Train Tasks}} \tag{5.7}$$

This heuristic was used as when the number of tasks increased the means of each of the task similarities tended towards the mean across all of task similarity. This meant that the task sampling distribution tended towards a uniform distribution and didn't form a sufficient separation to analyse task sampling differences. This is an issue partially caused from using the mean of task similarity values to calculate the co-task similarity. We did investigate other methods of calculating co-task similarity, including taking the max across the similarities, however as detailed in the discussion it didn't alleviate this issue. The above heuristic provided an effective means of dealing with this effect.

The full list of hyperparameters used throughout the experiments in this chapter, for both the ablation study and the adapted curriculum method experiments are provided in Appendix C.

5.3.1 Environment

To analyse the effect of a non-uniform task distribution on performance we use the Meta-world environment [YQH+19]. This is a simulated benchmark environment intended to investigate both meta-RL and MTL which offers diverse task distributions that include tasks that are focused on robotic manipulation, taken from the MuJoCo environment [TET12], dealing with continuous action-spaces. Here, we will be using both the ML10 and ML45 environments which provides appropriate task distribution size differences to form an analysis of whether the impact of using a non-uniform task distribution is affected by the number of tasks in the environment—ML10 includes 10 train tasks and 5 test tasks, whilst ML45 has 45 train tasks and 5 test tasks. We are using the updated v2 version of the environment which fixes issues found in v1 such as MDP consistency and reward structures. Each episode of a task is limited to 150 steps taken by a method—this was a design choice by the developers of the Meta-World environment to simplify benchmarking. This infers that if the agent hasn't succeeding the task by then, then it has failed.

This task environment was chosen as it provides a benchmark environment that contains a diverse selection of environment—RL currently has an issue with a limited range of environments that are focused on analysing generalisation in RL agents, beyond the scope of slight adaptations in a single-task, where Meta-World has been designed with tasks that are intended to be a challenge with regards to learning various inter-related skills. Further details of the overall benchmarking environment, including task names and groupings for both the ML10 and ML45 environments are detailed in Appendix A.

5.4 Results: Ablation Study

The following section details an ablation study that investigates the effectiveness and overall affect of both terms of Equation 5.2 in the forming a non-uniform task sampling distribution in a meta-learning framework on a multi-task environment. We also look at whether we are able to successfully use the representation to form the curricula as outlined in Section 5.2.1.4, using the combinations of hyperparameters detailed in Table 5.1. The ablation study is split up into three sections: firstly, we analyse results of using only gradient change term; secondly, we analyse the affect of only using the co-task similarity term; and, finally, we investigate the results of combining both of the terms within a combined task similarity metric. All of these sections look at the different behaviour exhibited when different hyperparameters values are used, investigating whether it recreates the curricula as outlined in Section 5.2.1.4, and analyses whether this behaviour is conducive to encouraging greater train and test performance by reducing overfitting and underfitting.

5.4.1 Gradient Change

Here, we provide the results of investigating the impact of using the gradient change term in a task similarity metric. This details the results from both the train and test phases of the meta-RL framework, and analyses whether the gradient change term in Equation 5.2 produces a representation suitable to discern task similarity. We investigate the affect of the resulting task sampling distribution in relation to overall sampling frequency and the performance of the agent when sampling changes to being non-uniform compared to the standard uniform task sampling distribution. The gradient change term focuses on the similarity of each task, in terms of the gradients of the model when the loss is calculated from trajectories from that task, to the current representation. The values for the gradient change hyperparameter, g, are provided as their switch values: 0, 1 and -1. The first of which, $g = 0$, acts as the baseline for comparison against a uniform distribution—this is labelled as MAML in the results. -1 gives preference in sampling for further tasks from the current model parameters to be sampled more, and 1 encourages closer tasks to the current representation to be sampled more—only constant hyperparameters are chosen. We set the hyperparameter for the co-task similarity term to 0 so this term is not used here.

Figure 5.3 shows the values for gradient change that were calculated during training and the corresponding task sampling weights using the result of Equation 5.2. Overall, we see that there is clear separation in the values for gradient change, which indicates that the method is traversing the model's parameter space differently. This difference in gradient change is therefore also forcing different curricula to be formed in the sampling weights. We can see that the agent with a gradient change hyperparameter of $g = -1$ sees the biggest change in drawer-close-v2, button-press-topdown-v2 and basketball-v2, whilst we see high sampling in drawer-close-v2. When we then compare this to when a value of $g = 1$ is used, this has the highest gradient change values with the door-open-v2 and reach-v2 tasks, whilst the lowest changes happen on the drawer-close-v2 task. The resulting task similarity weights sees basketball-v2 highly samples, and drawer-close-v2 sampled the least. We can see here that there are indications that the sampling schedules, the curricula, are seeing opposite behaviour between when $g = -1$ and $g = 1$. The sampling weights are used as part of a probability distribution and therefore doesn't directly provide sampling rates, however, it does indicate task preference.

We can see from Figure 5.4 that the task similarity weights discussed above significantly affect the overall task frequency across the entire training phase, with the overall number of times each task was sampled being far from the baseline (i.e. uniform distribution across tasks)—we provide more detailed results on how the sampling rates change over the entire training phase in Appendix C.2.1. For both of the hyperparameter cases, $g = -1$ and $g = 1$, we see a large separation between the groupings of certain tasks that have been sampled more than others, along with those that have been sampled less; high sampling for the reach-v2 and drawer-close-v2 tasks when $g = -1$, and button-press-v2 and peg-insert-side-v2 when $g = 1$—in relation to the inverse behaviour we see that the tasks most

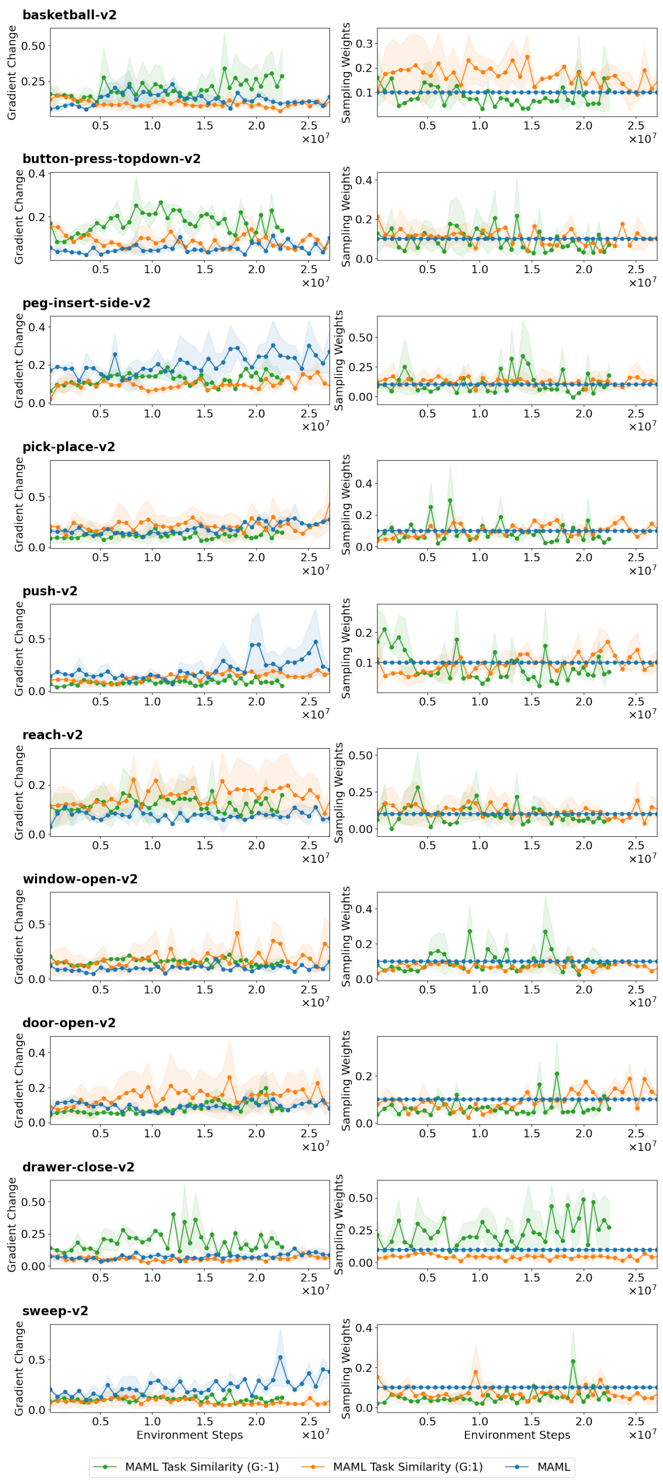


Figure 5.3: Gradient change and task sampling weights during training on the Meta-world ML10 environment [YQH+19] using gradient change to update the task sampling distribution, looking at different values of g, the hyperparameter that controls the influence of gradient change on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

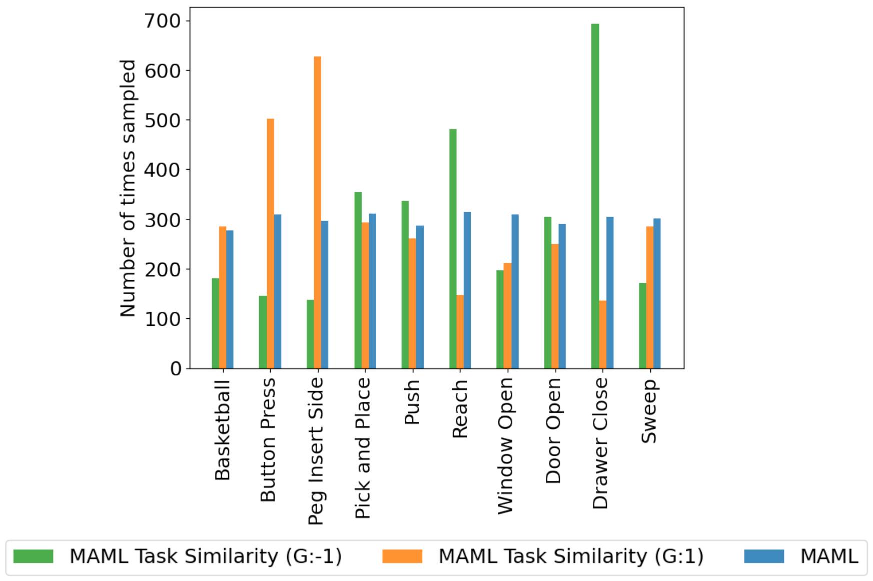
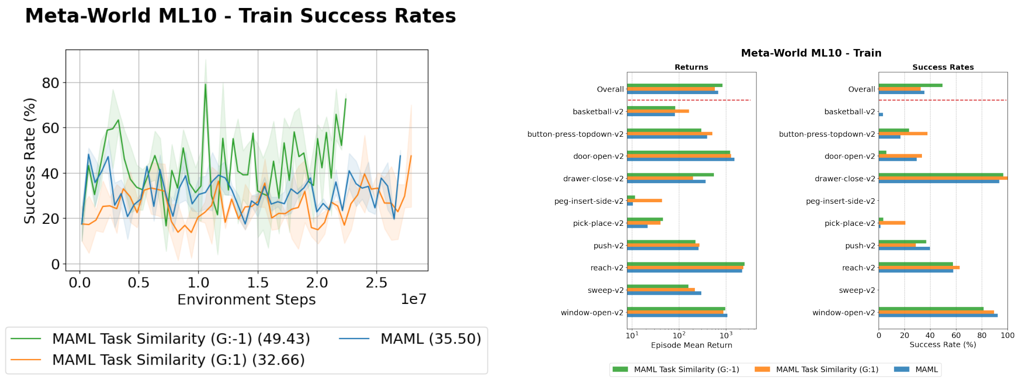


Figure 5.4: Overall task occurrences during training on the Meta-world ML10 environment [YQH+19], looking at different values of g, the hyperparameter that controls the influence of gradient change on the overall task sampling weights.

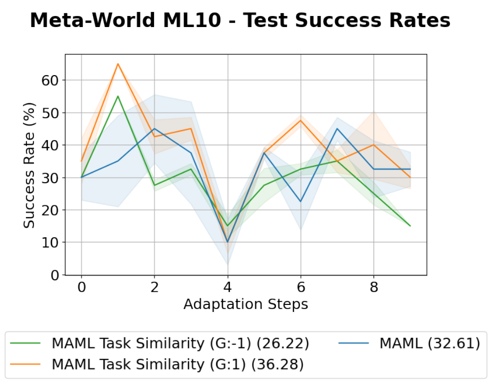
sampled by one hyperparameter are among those least sampled by the other and visa versa—this indicates that the hyperparameter switches are able to control the curriculum that is generated with regards to a switch to either state, as detailed in Section 5.2.1.4.



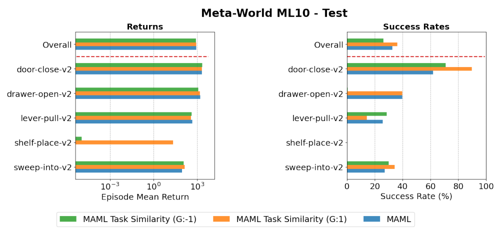
(a) Overall train success rate. (b) Per task returns and success rates.

Figure 5.5: Returns and success rates during training on the Meta-world ML10 environment [YQH+19] using gradient change to update the task sampling distribution, looking at different values of g, the hyperparameter that controls the influence of gradient change on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

Following from looking at the similarity weights and how they have impacted task sampling, we can also analyse what impact this has on performance. We find that, as highlighted by the per task success rates in Figure 5.5b, when using gradient change to generate the task sampling weights during training the success rates vary greatly between a g value of -1 and 1. When $g = -1$, which means we are sampling dissimilar tasks more, the success rate is 15% higher than that of the baseline MAML method—in particular we see that most of the performance is gained from only a few tasks with high success rate (drawer-close-v2, reach-v2, push-v2 and window-open-v2). This is in comparison to when $g = 1$, when we are encouraging the sampling of similar tasks, the performance is similar to that of baseline MAML, but we see increased performance on other tasks (door-open-v2 and pick-place-v2). This would intimate that we are seeing a difference in the sampling of these tasks. It should be noted that the overall training performance is weighted by the tasks that are sampled—for example, the overall train performance when $g = -1$ is weighted by the fact that drawer-close-v2 and reach-v2 have the highest success rates and are also sampled the most in this configuration.



(a) Overall train success rate.



(b) Per task returns and success rates.

Figure 5.6: Returns and success rates during training on the Meta-world ML10 environment [YQH+19] using gradient change to update the task sampling distribution, looking at different values of g, the hyperparameter that controls the influence of gradient change on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

After the training phase we then test the models adaptation to new tasks, provided by the Meta-World ML10 test set. Compared with the train performance, we see from Figure 5.6 that it is evident that both models using gradient change to adapt the task sampling distribution are overfitting to a subset of tasks, and therefore the performance is less than that of the baseline. In particular, we see that when $g = -1$ the agent is only able to perform on par with the other variants in door-close-v2, lever-pull-v2 and sweep-into-v2, but has 0% success rate on the drawer-open-v2 task compared with ∼30% and ∼45% on $g = 1$ and $g = 0$ respectively. In relation to adapting to unseen tasks, we can see that the both methods that use gradient change produce non-zero episodic mean returns on the shelf-place-v2 tasks whereas when using a uniform distribution produced 0 returns—this does not translate to success, which indicates the difficulty of the task with regard to adapting to it from the training set.

g	**Train Success**	**Num. Samples**		**Sampling StdDev**	**Test Success**
		Top	Bottom		
0	35.50	—	—	11.4	32.61
1	32.66	button-press -topdown-v2 peg-insert-side-v2	reach-v2 drawer-close-v2	145.02	36.28
-1	49.43	reach-v1 push-v1	button-press -topdown-v2 peg-insert-side-v2	165.6	26.22

Table 5.2: Comparison of gradient change, g, on task sampling rates in relation to tasks sampled, top and bottom 20th percentile of samples, and success rates on Meta-world ML10 environment [YQH+19] tasks. Standard deviation from 3 repetitions of each configuration.

We summarise the results of gradient change in Table 5.2 which highlights the comparison of the uses of gradient change in relation to task selection and the resultant performance. The highest train success was found when using a gradient change hyperparameter value of -1, however this produced the lowest test success—we saw a 23% drop in performance from train to test success rates. We can see that when $g = -1$ the spread of sampling signified by the sampling standard deviation of 165.6 is much greater than compared to that of the baseline of 11.4, which highlights the large spread of task frequencies during training. Overall, all of these factors intimate that when $g = -1$ we massively overfit to the training set in regards to the tasks that were most dissimilar to the current representation and didn't explore the entire policy space sufficiently. One interesting aspect of the results is that when $g = -1$ it mainly samples from the drawer-close-v2 task, however performed the worst on the drawer-open-v2 task. This indicates that the method over-trained on the drawer-close-v2 task and learnt only to memorise how to perform on the task rather than to extrapolate a task skill, and this was then negatively impacted by the other tasks—this is an issue both with MAML, but also the TRPO method.

The results suggest that gradient change does find a separation of tasks and therefore can generate a curriculum as defined in Section 5.2.1.4. We see that in terms of sampling differences, the reach-v2 task was included in the most sampled task when $g = -1$ however it was the least sampled task when $g = 1$. As such, as previously mentioned, the gradient change hyperparameter g acts a useful switch for dictating the sampling schedule of the agent.

Although we confirmed that we can form curricula, we can see that, by using the gradient change metric, the performance of the agent when adapting on the test tasks indicates that the curricula defined solely by the gradient change metric does not aid in the forming an agent that can adapt across unseen, but related tasks whilst maintaining performance. We have seen a large amount of overfitting, in particular when $g = -1$, which is when we encourage task to be sampled that are dissimilar as defined by the gradient change term, to the current representation with regards to the gradients. Overall, this provides an indication that when selecting the hyperparameter we can affect the balance between overfitting or underfitting on the train tasks with regard to the test task performance. As such, gradient change could be a useful term for improving the generalisability of meta-RL as long as it is used in conjunction with a term that can negate the impact of overfitting.

5.4.2 Co-Task Similarity

We now look at using only the co-task similarity term from Equation 5.4 within the meta-RL framework and analyse its usage as part of a similarity metric and its effect on task sampling. The co-task similarity aims to leverage the similarity across tasks, as defined by the mean cosine similarity between tasks' gradients. In particular, we are looking at whether the co-task similarity term can be used as an effective metric for task similarity, encouraging separation of tasks regarding their inter-task similarities, and therefore be useful in forming task similarity-based curricula. The values for the co-task similarity hyperparameter, c, are provided as their switch values: 0, 1 and -1. When $c = 0$, the resulting task distribution is uniform and is provided for comparison—this is labelled as MAML in the results. -1 gives preference in the sampling of further tasks from other tasks to be sampled more, and 1 encourages closer tasks to others (i.e. those tasks with tighter grouping) to be sampled more—only constant hyperparameters are chosen. We set the hyperparameter for the gradient change term to $g = 0$ so this term is not used here.

By using the co-task similarity term we generate a set of task sampling weights—we provide the values for both the co-task similarity term and the resultant task weights in Figure 5.7. We see that compared with the gradient change task weights, we don't see as large of a consistent separation in task weights, however certain tasks do have periods of the training phase where the sampling rates are different from the uniform task distribution. Overall, the co-task similarity values follow, in general, the same curve across all 3 hyperparameter values for co-task similarity—there are cases where there is a small degree of consistent separation in tasks including pick-and-place-v2, however these are only minor changes. We do however see separation in the sampling weights. However, these represent more spikes in sampling weights and only see temporary, instead of sustained and consistent, separation—this has been caused by slight changes from the co-task similarity values, possibly representing noise within the representation, being exacerbated in the weights. Across nearly all the tasks we see spikes in sampling weights when $c = -1$ which are especially evident in the door-open-v2, push-v2 and basketball-v2 tasks. When $c = 1$ we also see these spikes in sampling weights, however they are much smaller and happen less. Both for $c = 1$ and $c = -1$ the sampling weights alternate between above and below the uniform sampling weights when $c = 0$ which provides an indication that we are not finding a consistent representation of task similarity with regards to the co-task similarity term, and instead is being highly affected by the noise in RL gradients across task interactions, as discussed in Section 5.2.1.5, due to current model parameters.

The resulting task frequencies across the agent's training phase, a consequence of the sampling weights discussed above, are detailed in Figure 5.8—we provide details on how the sampling rates change due to co-task similarity during the training phase in Appendix C.2.2. We see that both $c = 1$ and $c = -1$ hyperparameters result in task groupings that are the most sampled during training, but that are in the least sampled groupings of the other. When $c = -1$ the most sampled tasks are door-open-v2, reach-v2 and peg-insert-side-v2, whilst reach-v2 is one of the last sampled tasks when $c = 1$. Whereas, when $c = 1$ the most sampled tasks are push-v2, pick-and-place-v and drawer-close-v2 which are the least sampled tasks when $c = -1$. This provides an indication that although the

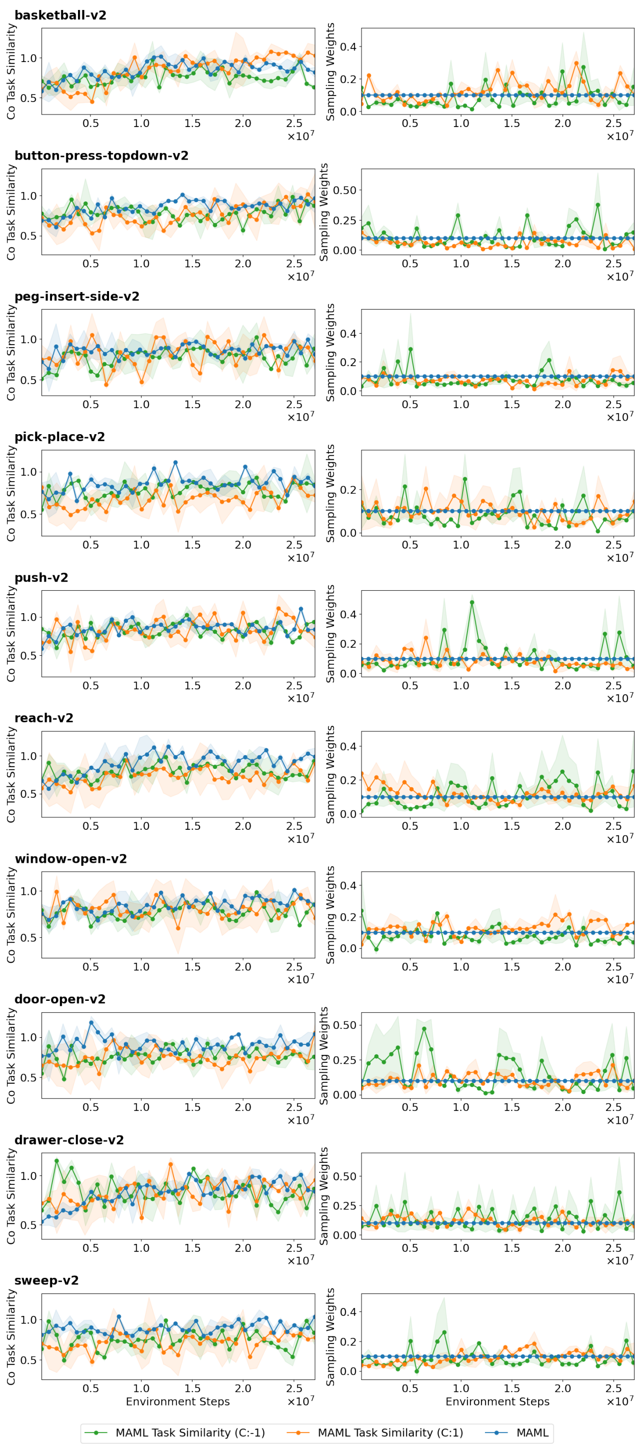


Figure 5.7: Co-task similarity and task sampling weights during training on the Meta-world ML10 environment [YQH+19] using co-task similarity to update the task sampling distribution, looking at different values of c, the hyperparameter that controls the influence of co-task similarity on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

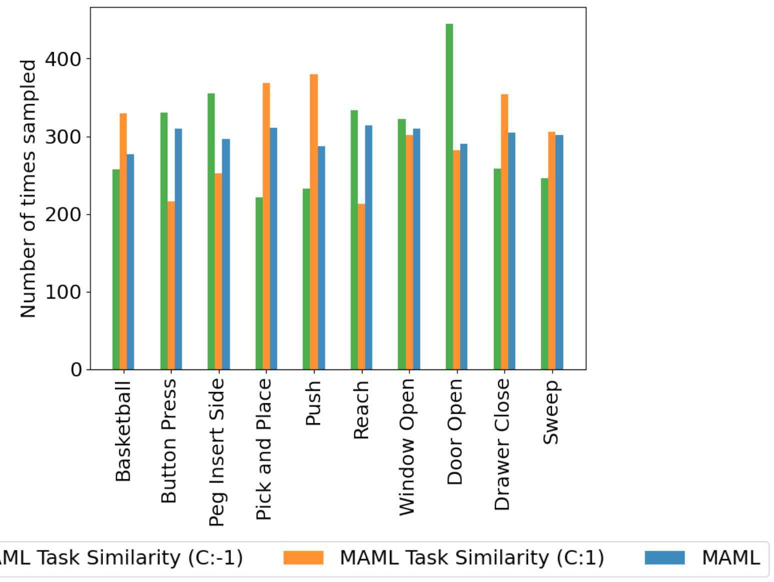
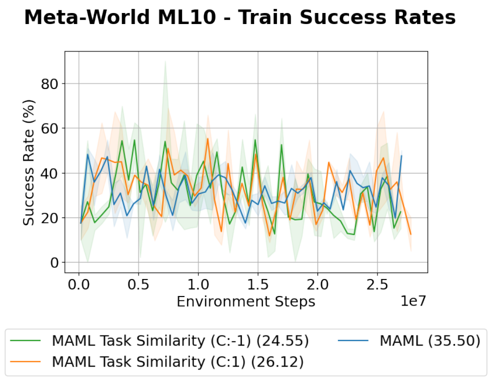


Figure 5.8: Overall task occurrences during training on the Meta-world ML10 environment [YQH+19], looking at different values of c, the hyperparameter that controls the influence of co-task similarity on the overall task sampling weights.

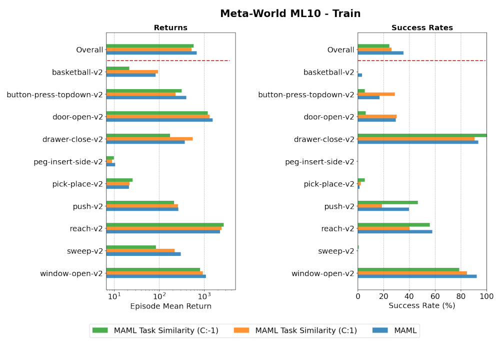
task weights only show temporary spikes in increase sampling weights, overall, the co-task similarity term is facilitating the creation of curricula that separate out particular task similarities. In particular, we see that the hyperparameters do facilitate the opposite curricula to be formed on the similarity defined by the co-task similarity term. We see this happen where the co-task similarity signifies that push-v2 and pick-and-place-v2 are least similar to other tasks, whilst peg-insert-side-v2, reach-v2 and door-open-v2 are the most similar.

From the task sampling weights calculated using the co-task similarity between task gradient we analyse the impact of the non-uniform task sampling distribution on performance—results of this are provided in Figure 5.9 which provides both the per task episode mean returns and success rates. Success rates are approximately 10% lower for both the agent's using co-task similarity to form the task sampling distribution, however mean episodic returns on each of the tasks are similar to the baseline across all tasks. Across all of the tasks, pick-place-v2 is the only task that both $c = 1$ and $c = -1$ have a higher success rate—this is a task which has a high deviation from the uniform sampling distribution, and is included in the most and least sampled, respectively, task groupings.

After the agent is trained, the agents' performance is analysed when acting on the test tasks provided by the Meta-World ML10 task distribution to test for

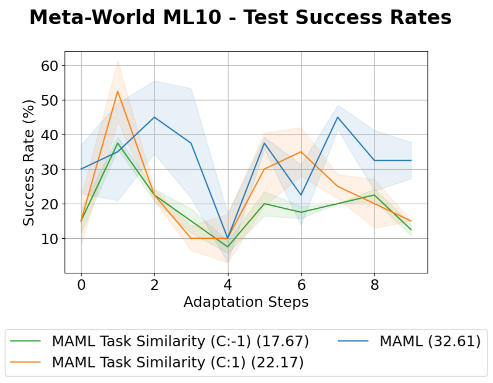


(a) Overall train success rate.

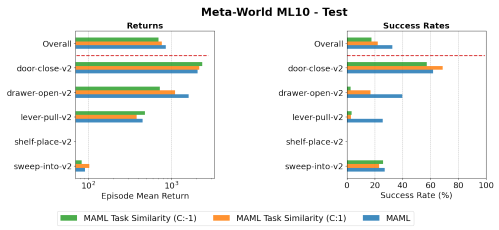


(b) Per task returns and success rates.

Figure 5.9: Returns and success rates during training on the Meta-world ML10 environment [YQH+19] using co-task similarity to update the task sampling distribution, looking at different values of c, the hyperparameter that controls the influence of co-task similarity on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.



(a) Overall train success rate.



(b) Per task returns and success rates.

Figure 5.10: Returns and success rates during testing on the Meta-world ML10 environment [YQH+19] using co-task similarity to update the task sampling distribution, looking at different values of c, the hyperparameter that controls the influence of co-task similarity on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

model adaptation. Figure 5.10 provides the test results for each of the hyperparameter choices against the baseline, providing a breakdown of per task returns and success rates, along with an success rate throughout test adaptation. We see that the curricula formed using co-task similarity have a dramatically worse affect on test adaptation performance than using a uniform task sampling distribution—this reduction in performance is almost consistent across all 10 adaptation steps for both values of c. Both the co-task similarity variants produce comparable performance on the door-close-v2 and sweep-into-v2 tasks, however experience almost no positive adaptation to the other three tasks with regard to success rates although they have comparable returns. These results indicate two aspects: the tasks within the Meta-World environment allow for large amounts of reward to be experienced without having the agent be successful in the task (this could be an indication that the reward structures could be improved in these tasks), along with that the co-task similarity variant agents have adapted to gain reward, but not complete the task. These results indicate that the curricula formed in training by both the co-task similarity based methods are not learning transferable knowledge of how to interact with the environment to enable fast adaptation to a new set of related tasks, especially those classified as harder tasks.

c	**Train Success**	**Num. Samples** Top	Bottom	**Sampling StdDev**	**Test Success**
0	35.50	—	—	11.4	32.61
1	26.12	pick-place-v2 push-v2	button-press-topdown-v2 reach-v2	56.7	22.17
-1	24.55	peg-insert-side-v2 door-open-v2	pick-place-v2 push-v2	66.2	17.67

Table 5.3: Comparison of co-task similarity hyperparameter values, c, on task sampling rates in relation to tasks sampled, top and bottom 20th percentile of samples, and success rates on Meta-world ML10 environment tasks. Standard deviation from 3 repetitions of each configuration.

We provide a summary of the affect of the co-task similarity hyperparameters in Table 5.3 discussed above, which provides comparison of the different tasks sampled, both the most and least, along with the train and test performance. We see the same degradation in performance, ~3-7%, on all methods with different c values. However, due to the low train success when using the co-task similarity term to affect task sampling, the test success is much lower when using this term. The curricula that we see have led to the learners underfitting to the task distribution and not being able to transfer useful information to enables the agent to adapt quickly, in particular to be successful at the ML10 tasks—this indicates that the agent is learning to memorise how to behave in the training tasks, rather than extrapolate understanding, without learning task-specific transferable skills.

The results when using the co-task similarity term indicate that it struggles to find a consistent separation of sampling weights, although we do have a separation in the overall task frequencies over the entire training phase. There is a grouping of tasks, however as shown in Table 5.3 there is a fairly low standard deviation in the task sampling frequencies which indicates that the separation is not fully decided upon—this is enforced by there not being consistent sampling separation throughout training, and only having spikes of increased sampling for certain tasks. This means that the curricula described in Section 5.2.1.4 are partially formed, however not sufficiently to indicate that co-task similarity is forming a useful metric of similarity amongst the tasks.

Overall, although we can form curricula that target a specific aspect of similarity in the tasks the agent is learning on by using the co-task similarity metric, the overall performance across train and test distribution adaption indicates that this term alone does not provide a useful enough signal for the training of an agent that can quickly adapt to new unseen, but related, tasks. However, the fact that the term does form a separation, does indicate that it may be useful along with other terms.

5.4.3 Combination: Gradient Change and Co-Task Similarity

Following the experiments of using the terms individually, we now investigate at the affect of using a combination of both of the terms in Equation 5.2, gradient change and co-task similarity, to form a task similarity metric to form a non-uniform task sampling distribution within the MAML method. This will analyse all the hyperparameter combinations for g and c with each taking values of -1 and 1, along with a baseline comparison of MAML with a uniform task sampling distribution (i.e. both hyperparameter values set to 0). As previously mentioned, the intuition is that using both of these terms should allow us to balance the overfitting and underfitting of performance to certain tasks, through forming the curricula discussed in Section 5.2.1.4, which we have seen in the previous result sections. In this section, we analyse whether by accounting for the task similarity to the current representation, using gradient change, and to the other tasks within the task distribution, using co-task similarity, they can both facilitate a learner that can quickly adapt to new tasks.

By using both the gradient change and co-task similarity terms we calculate the associated task sampling weights, as shown in Figure 5.11. Looking at the

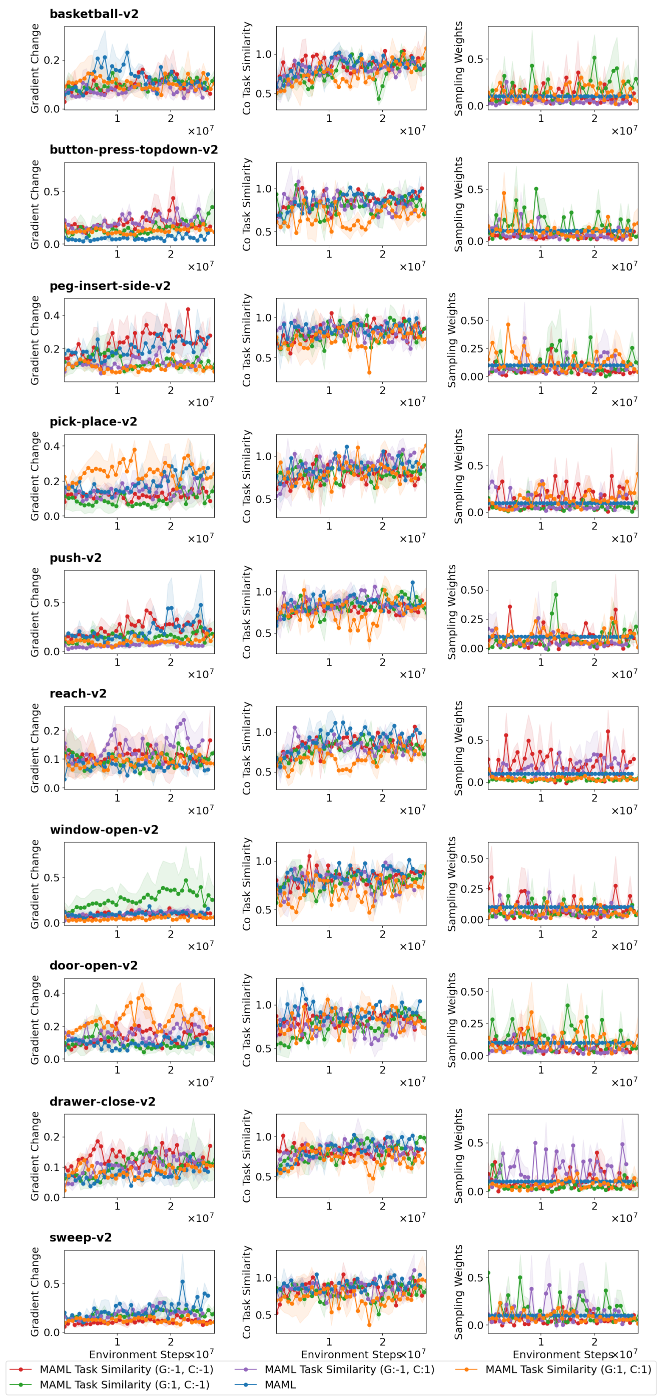


Figure 5.11: Gradient change, co-task similarity and task sampling weights during training on the Meta-World ML10 environment [YQH$^+$19] using a combination of gradient change and co-task similarity to update the task sampling distribution, looking at different values of g and c, the hyperparameter that controls the influence of gradient change and co-task similarity, respectively, on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

gradient change values, we see that there is a large separation across all variants, indicating that the overall task sampling weights are forcing the agent into different positions in the model parameter space. In particular, we are seeing consistent separation in the window-open-v2 task when $g = 1, c = -1$, and in the pick-place-v2 task when $g = 1, c = 1$.

With regards to co-task similarity, we see a large amount of noise around similar trajectories throughout the training phase. We do see specific divergences from the common pattern, of particular note is the agent using $g = 1, c = 1$ which sees separation compared to the other configurations in reach-v2, push-v2, button-press-topdown-v2 and window-open-v2 with lower co-task similarity on these tasks throughout large portions of the training phase—this is an indication that this method is finding similarities in the task distribution that other hyperparameter configurations are not finding especially as it is showing greater separation.

The effect that both of these terms have is evident in the task similarity weights. We can see that there is a large amount of noise in the weights, mainly due to the effect of the co-task similarity. However, there are instances where certain configurations are seeing consistent increases in sampling, in particular both $g = -1, c = -1$ and $g = -1, c = 1$ configurations on the reach-v2 task—both of these attempt to encourage dissimilar tasks from the current representation—and in the drawer-close-v2 task when $g = -1, c = 1$. The effect that the lower co-task similarity values from the $g = 1, c = 1$ configuration, is us seeing less sampling than the uniform weights on both reach-v2 and window-open-v2, both of which had lower values on co-task similarity.

Using the task weights, mentioned above, the agent is then trained with a task distribution which results in the overall task frequencies as detailed in Figure 5.12—we provide details on how the sampling rates change due to the combination of gradient change and co-task similarity during the training phase in Appendix C.2.3. Overall, we see that all methods form good separation in the overall task frequencies with groupings, but there is no consistent groupings across all hyperparameter configurations which indicates that each of them are forming different curricula, as intended. However it is evident that there is a difference in the amount of spread across task that occurs, which when we look at the performance of each of the configurations will be important to note with regard to

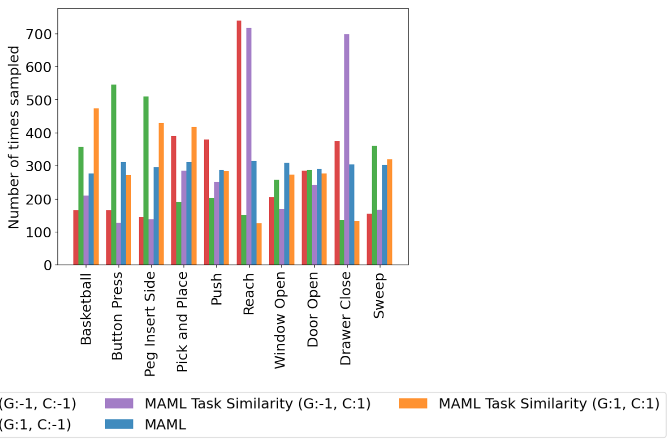
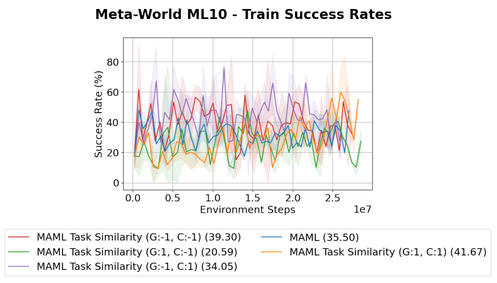
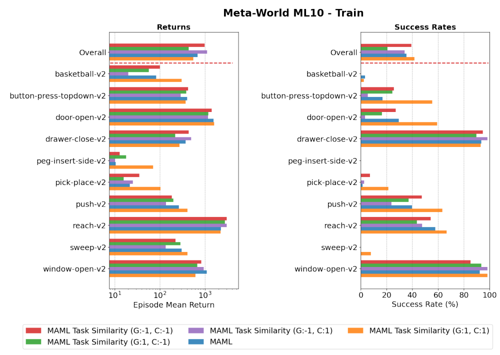


Figure 5.12: Overall task occurrences during training on the Meta-World ML10 environment [YQH+19], looking at different values of g and c, the hyperparameters that control the influence of both the gradient change and co-task similarity terms on the overall task sampling weights.

overfitting and underfitting. For example we see that the $g = 1, c = 1$ configuration has a much closer sampling frequency to the uniform distribution than when $g = -1, c = 1$. In particular, we see that when $g = -1, c = 1$ we see large sampling in both the reach-v2 and drawer-close-v2 tasks, and minimal on all other tasks which correlates with the task sampling weights detailed above—as such it will be important to analyse if this has overfitted to these tasks.



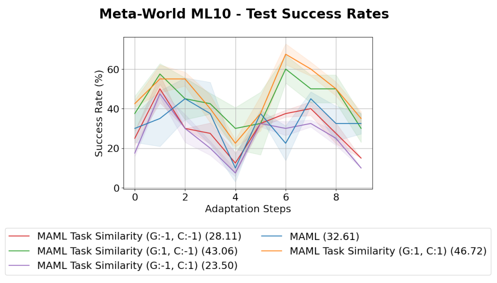
(a) Overall train success rate.



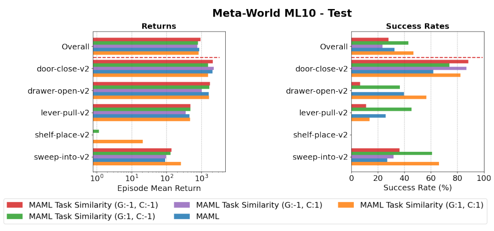
(b) Per task returns and success rates.

Figure 5.13: Returns and success rates during training on the Meta-World ML10 environment [YQH+19] using a combination of the gradient change and co-task similarity terms to update the task sampling distribution, looking at different values of g and c, the hyperparameters that control the influence of gradient change and co-task similarity, respectively, on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

The result of sampling the tasks with the frequencies defined above has had a large impact on training performance, as detailed in Figure 5.13, with regard to per task mean episodic returns and success rate. We see ∼5% higher success rate when $g=-1, c=-1$ and $g=1, c=1$ compared with using a uniform task sampling distribution. In particular, when using $g=1, c=1$ we see much higher success rate on the button-press-topdown-v2, door-open-v2 and pick-place-v2 tasks, whilst it is the only configuration to find success on the sweep-v2 task. Across all the tasks, we see very similar mean episodic returns, apart from increases when $g=1, c=1$ on basketball-v2, peg-insert-side-v2 and pick-place-v2—it will be vital to understand where this high performance translates to overfitting to the training tasks. In comparison, we see both low returns and overall success rates when $g=1, c=-1$, where success is consistently lower on all tasks, and is the only task with zero success rate on the pick-place-v2 task—this task is however seen as a difficult task to learn with other tasks in the training set as all other tasks do have a low, but non-zero, success rate.



(a) Overall train success rate.



(b) Per task returns and success rates.

Figure 5.14: Returns and success rates during training on the Meta-World ML10 environment [YQH+19] using a combination of the gradient change and co-task similarity terms (Equation 5.2) to update the task sampling distribution, looking at different values of g and c, the hyperparameters that control the influence of gradient change and co-task similarity, respectively, on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

Following the training of the agent's using each of the combinations of hyperparameters for gradient change and co-task similarity, we then test the agents ability to adapt to new tasks using the Meta-World ML10 test tasks. The results of testing each of these models indicates that the curricula formed during the training phase has had a significant affect on the agent's ability to adapt. We see that all tasks perform similarly with regard to mean episodic reward, except in the case of the shelf-place-v2 task. We see that both the $g = 1, c = -1$ and $g = 1, c = 1$ hyperparameter term configurations are the only agents that have learnt enough in the shelf-place-v2 task for the agent to have non-zero returns during training and produce high performance with regard to success rate (∼10-15% higher success rate than with a uniform task sampling distribution)—both of these configurations have $g = 1$, which is encouraging similar tasks to the current model parameters to be sampled more. This is in comparison to when $g = -1$ and any value of c, we are seeing low success rates compared with using a uniform task sampling distribution on all tasks apart from door-close-v2.

g	c	**Train Success**	**Num. Samples** Top	Bottom	**Sampling StdDev**	**Test Success**
0	0	35.50	—	—	11.4	32.61
1	1	41.67	basketball-v2 peg-insert-side-v2	reach-v2 drawer-close-v2	110.2	46.72
1	-1	20.59	button-press -topdown-v2 peg-insert-side-v2	reach-v2 drawer-close-v2	135.5	43.06
-1	1	34.05	reach-v2 drawer-close-v2	button-press -topdown-v2 peg-insert-side-v2	209.1	23.50
-1	-1	39.30	pick-place-v2 reach-v2	peg-insert-side-v2 sweep-v2	142.93	28.11

Table 5.4: Comparison of combination of gradient change and co-task similarity, g and c respectively, on task sampling rates in relation to tasks sampled, top and bottom 20th percentile of samples, and the resulting success rates on the Meta-World ML10 environment tasks, with standard deviation from 3 repetitions.

Table 5.4 provides a summation of the results presented in the previous figures and allows us to analyse the task selection with regard to overall performance, and the cross-over between tasks in this regard. The results indicate that the combination of the two terms represent a good overall metric for task similarity with good separation of sampling weights for particular tasks, and therefore overall task frequency over the training phase, along with high task frequency standard deviation and sampling weights.

The results suggest that when the gradient change term is used to encourage closer tasks to the current models parameters, $g = 1$, and we balance this with either training on tasks closer or further from each other, $c = 1 or c = -1$, we see test success rates are far greater than when using a uniform task distribution,

$g = 0, c = 0$. We also see that it doesn't seem to matter whether we are increasing the sampling of closer or further tasks to each other (i.e. the value of c doesn't impact the effectiveness of the agent) but that both are having the affect of reducing the standard deviation of tasks sampling and therefore reducing the overfitting to the closest tasks—both the configurations when $g = 1$ have lower standard deviation than when $g = -1$, which means that the spread of tasks being sampled is flatter and therefore we are ensuring all tasks are sampled sufficiently. In relation to the sampled tasks, both of these find that the least sampled tasks should be reach-v2 and drawer-v2.

In comparison to when $g = 1$, when $g = -1$ we are seeing a large amount of overfitting to the tasks that were found to be least important to forming an adaptive learner, in particular the reach-v2 task, as both agents have dramatically lower test success than train success. In particular, when $g = -1, c = 1$ we see a task sampling deviation of over 200, which is approximately double of both agents using $g = -1$—this indicates that the agent overfits on the task groupings, in particular the reach-v2 and drawer-close-v2 tasks. In this case, the co-task similarity term is unable to control for the overfitting to dissimilar tasks, according to the similarity to the current model representation, when both $c = 1$ and $c = -1$.

When using both the gradient change and co-task similarity terms within Equation 5.2, results have indicated that we can form curricula that can encourage the behaviour outlined in Section 5.2.1.4. When we compare the task samplings for both $g = 1, c = -1$ and $g = -1, c = 1$, which are inverse hyperparameter selections and therefore should produce inverse sampling, we see an inverse task sampling distribution where the top tasks for one configurations are the least sampled tasks for the other, and visa versa. However, we do note that the gradient change term seems to be the main controlling factor with regard to deciding the curriculum due to evident consistency of task selection across g values which is not the case for c values. This provides an indication that the curricula are formed, as we see consistent and inverse behaviour, and as such have gathered learner relevant information about the tasks in the environment.

Overall, by combining the gradient change and co-task similarity terms we are able to form a measure of task similarity across the task distribution that aids in the forming of curricula that can balance overfitting and underfitting, as indicated by an ability to control the standard deviation of task sampling whilst ensuring particular useful tasks are sampled at the appropriate times, in order to expose agents to a curriculum that encourages the learners' representation to be able to adapt to changes in the environment and across different tasks.

5.4.4 Discussion

The results presented during this section, indicate that gradients from the inner-loop of a meta-learning approach, in particular MAML, in producing a task similarity metric, defined by both gradient change and co-task similarity in the form defined in Equation 5.2, that provides useful information with regard to forming a curriculum targeting task similarities. We see that both the gradient change and co-task similarity metrics produce groupings of tasks of similarity during training, with gradient change in particular finding greater separation in relation to its similarity measure.

One of the intentions of the ablation study was to analyse whether the curricula stated in Section 5.2.1.4 would be produced in training by the respective terms. From each of the results across the models trained with the specific hyperparameter switch, we see that the gradient change term has the largest affect on defining a consistent curriculum, with task sampling weights being predominantly consistent when this term is used, in comparison to the fairly noisy representation of co-task similarity. We've seen that the hyperparameter choices have formed consistency in creating inverse curricula—with the top sampled tasks being in the least sampled tasks of the hyperparameter with the opposite switch value. However, it is important to note that with regards to the reusability of these metrics, due to the metrics being based on the gradients of the learner, they are innately associated to the learners performance. Under these conditions, the results have indicated that the metrics have revealed key information about the gradient-based task relationships across all the tasks in the environment. As such, we have seen that under these conditions certain tasks are defined as similar and/or dissimilar.

As we can see from the results, a non-uniform task sampling distribution can dramatically affect the performance of MAML. We see that the target of the curriculum has a large affect on the overall performance of MAML with regards to evidence of both overfitting and underfitting occurring when certain tasks are sampled too much or not enough. In particular, we see that when $g = 1$, when encouraging the sampling of close tasks, we see much higher performance than when $g = -1$, which is inline with the core ideas behind curriculum learning [BLCW09]. However, without the co-task similarity term we saw overfitting of the agent to the subset of tasks it classed as similar and dissimilar. We found that the introduction of the co-task similarity term countered the overfitting of the gradient change term and allowed for far greater performance in relation to train and test adaptation than when using a uniform task sampling distribution. In relation to the types of sampling distributions formed, we can see that there is still a large amount of noise in the resulting task distributions with regards to the frequency of sampled tasks over training phase.

Overall, we see that the task similarity metric, as defined by Equation 5.2, produces information that can be useful in forming a curriculum to encourage agent's that are able to adapt quickly to new tasks, as per the target of a meta-learning approach—the results in this section have provided evidence towards Hypothesis 1. However, in the current form, the defined metric has limitations that result in it not producing curricula that are fully optimal, as illustrated by the results in this section, for encouraging performance on new tasks and therefore generalisation. The main issues that we found are centred around the issue that the curricula formed are constant across the entire training phase. This led to either overfitting, or underfitting depending on the hyperparameters chosen, as discussed above. As such, it is evident that we need to find a method of allowing for the learning of an optimal curriculum that can balance these two during training. It is to be noted that the combination partially remedied these issues, with performance greater than the baseline, however we still had to make an appropriate hyperparameter choice to balance the effect—an appropriate curriculum may be different on another environment. One option is to adjust the hyperparameters controlling the effect of each of the terms. However, to have a curriculum that is able to change based on what the learner requires would involve learning a meta-parameter for each of the hyperparameters. This would incorporate additional complexity into task selection which is not preferable due to computational complexity.

5.5 Automatic Curriculum Learning Using Gradient-based Task Similarity

Following on from the findings of the ablation study, we now look into how we can optimally produce a non-uniform task sampling distribution to form a curriculum that encourages greater generalisation in MAML, using the terms defined in Equation 5.2. The aim is to fix some of the limitations of using the additive approach for combining the terms, as highlighted and discussed in the ablation study. In particular, the main issue with the approach discussed above is the need to use hyperparameters to dictate a curriculum for the agent. Although we found the optimal hyperparameters for the Meta-World environment, certain environments may require different approaches to facilitate an adaptive learner to be taught, and as such different curricula may exist—although we found the best performance with a curriculum that encouraged the interaction of closer tasks to the current representation, it may have been more optimal to have experienced different tasks during parts of training.

In order to do this we need to allow the learner to learn the optimal curriculum during the training phase; this is an area of machine learning called automatic curriculum learning. Automatic curriculum learning (ACL) is a technique for automatically adapting the training task distribution by learning the optimal learning situations for a particular performance target. This is an area which has seen an increase in focus within RL due to the impact that curricula can have on remedying issues in the paradigm [PCW$^+$20]. There has been recent further work looking at adaptive task distributions in RL in particular work by [PRHO21] which aim to meta-learn (in the sense of it being an auxiliary learning target) a task distribution to encourage generalisation in RL methods. This has the benefits that the curriculum is able to adapt itself to suit the requirements of the learner during learning, which means it can react to the current state of the learners understanding of the task. In this section, we use an adaptation of the gradient-based task similarity metric detailed in Equation 5.2 in the MAML method to perform ACL.

5.5.1 Method

We provide an update to Equation 5.2 which removes the hyperparameters g and c, both of which constrained the curriculum during training towards a particular target, and leverages the benefits of each term, as investigated in the ablation study, whilst facilitating the learning of an automatic curriculum that attempts to balance the sampling of tasks with regard to overfitting and underfitting in order to encourage generalisation in environments containing diverse task distributions. The updated equation is given as:

$$\text{Softmax}(-(\|\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)\|_2 * (\frac{1}{N}\sum_{n=1}^{N} \cos(\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta), \nabla_\theta \mathcal{L}_{\mathcal{T}_n}(f_\theta)))/\tau) \tag{5.8}$$

Equation 5.8 details a modification of Equation 5.2 where we expand on the work given in the ablation study and see that by taking the product of these two terms we can enable for a balancing between overfitting and underfitting whilst removing the hyperparameters locking in the curriculum for the entire training cycle—in this formulation we can see the use of each term as a hyperparameter for the other instead. The results from the ablation study indicated that when we set the hyperparameter for the gradient change and co-task similarity terms, g and c, to be constant across training we produced curricula that were consistent across the entire training cycle. However, this is sub-optimal as we would encourage either overfitting or underfitting with each term. The combination of the two terms remedied this problem to an extent we could balance the two terms, however we still relied on hyperparameters to defined a specific target intent for each term. We suggest that a possible solution for this is to use the product of the two terms instead, which allows for each term to directly affect the others impact.

It should be noted here that the definition for the co-task similarity equation is using the strict definition of cosine similarity between gradients of each tasks, which bounds the values between -1 and 1—this provides a means of replicating the switch values between different curricula provided by the hyperparameter terms in Equation 5.2—this results in us favouring tasks that share the same direction in relation to sampling. In summary, we can therefore interpret this new equation as co-task similarity being an adaptive hyperparameter for gradient change.

This change in formulation of the task similarity therefore impacts on how we must use the values of each term. It should be noted that we don't standardise the values of each term, gradient change and co-task similarity, as we don't have the situation of one out-scaling the other and dominating the effect on the task sampling distribution weights due to us taking the product of the terms instead. We also use a replay buffer here to reduce the effect of overfitting during training—due to this curriculum being adaptive compared to that of the single term cases investigated in the ablation study, the need to correct for overfitting should be reduced, however to ensure that we are sampling from all tasks during training, this mechanism is used. In Equation 5.8 we also include a negation to the product of the two terms which acts as a switch on an overall sampling curriculum—when the co-task similarity of a task is positive, meaning the mean direction of gradient is similar to other tasks, this will cause us to sample it less, compared to without the change in sign, we would sample it more, resulting us favouring tasks more unlike others than similar—the adaptation of this though causes a more balanced approach.

An advantage of using gradients as a base for task similarity metrics for adapting a curriculum, is that the performance of the agent to these tasks is embedded within the gradients themselves—due to this the curriculum is innately targeting performance as well. This means that when we understand the curriculum formed, we also need to account for the fact that it should be self-accounting for improved performance.

5.5.2 Results

In order understand how the modifications to the task similarity metric, as outlined in Equation 5.8, which aims to facilitate ACL within the MAML method through the forming of an adaptive task sampling distribution, affect the ability for the agent to learn across diverse multi-task distribution, the following experiments focus on analysing the type of curricula learnt through training, in particular the resulting task frequencies, along with the affect they have on train vs. test performance and generalisability.

These experiments are divided into three different sections. The first two sections focus on using both the Meta-World ML10 and ML45 environment's task distributions, both of which are detailed in Appendix A.1, in order to analyse the affect of increased diversity in the task distribution in particular investigating the impact of a greater number of tasks during training—this is intended to provide a means of further illustrating the impact of a non-uniform task sampling distribution. The final section details experiments explicitly focused on the analysis of generalisation which make use of the generalisation definition, as detailed in Chapter 4.

With regard to the mechanisms available to mitigate for the inherent noise within RL gradients, as detailed in Section 5.2.1.5, we use both an epsilon-greedy task sampling distribution selection policy and experience replay buffer in these experiments. All other experiment details are the same as detailed in Section 5.3, including the use of the MAML method with a uniform task sampling distribution as a baseline for comparison—this is labelled as MAML in the results. In these experiments we include two variations of the method, both defined by an overall switch on encouraging either to close or far tasks to investigate its impact and further investigate the impact of overall favouring closer tasks—as previously mentioned, this is controlled through the use of the negation term in the softmax. The hyperparameters used in these experiments are provided in Appendix C.1.

5.5.2.1 Meta-World ML10

This section details the results of using the updated gradient-based task similarity metric, as detailed in Equation 5.8, on the Meta-World ML10 environment [YQH+19]. Here, we provide an analysis of the results which focus on the way that both gradient change and co-task similarity interact to form task sampling weights, their affect on the curriculum throughout the training phase and the performance that they produce.

By using the combination of the gradient change and co-task similarity terms to form task sampling weights, as shown in Figure 5.15, we can see that the sampling of different tasks varies across the course of the training phase. In regard to gradient change, we see that although many of the tasks have the same values across the different methods, certain tasks have considerable separation and indicate that different policy spaces are being explored. On both the push-v2 and door-open-v2 tasks we see that the task similarity method encouraging close tasks has highly different gradient change values than the baseline across the entire training phase. We also see that the MAML baseline sees much greater gradient change than both the task similarity methods in the pick-place-v2 method which indicates that using a non-uniform task distribution is seeing this method as much closer to the representation. When looking at co-task similarity, we see all methods maintain the same trajectories over the entire training phase with considerable noise—this shows that the gradient change term will control most of the task sampling weight calculation. Using both these terms, the resultant task similarity weights show that the task similarity method encouraging close tasks tends towards the uniform distribution, as provided by the baseline MAML method—there are periods of higher and lower sampling a certain tasks but these are fairly infrequent and short. In comparison, the task similarity method that encourages far tasks shows highly varying task sampling weights across nearly all

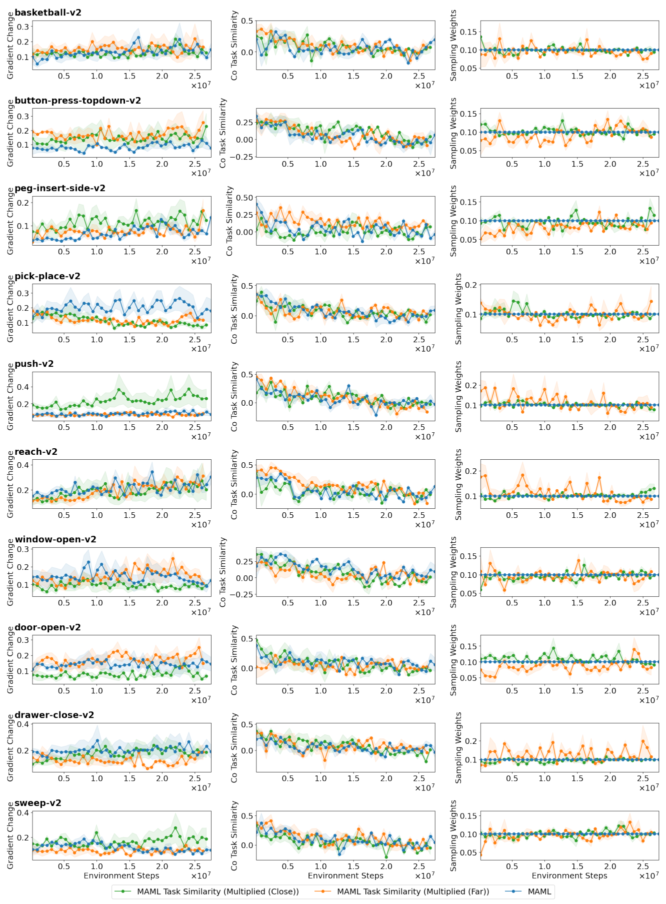


Figure 5.15: Gradient change and task sampling weights, with error bars showing standard deviation for 3 repetitions, formed using the automatic curriculum approach during training on the Meta-world ML10 environment [YQH+19].

tasks, where the associated sampling weights evolve over the course of the training phase. For example, in the button-press-topdown-v2 task we see sampling weights below the uniform weights for two-thirds of the training phase, where the sampling weights then switch to being higher for the last part. Overall, this indicates that the far method is favouring an adaptive curriculum, whereas the close method is encouraging a much more uniform task sampling distribution—this illustrates that the new method has the ability for different types of distributions to be formed depending on the agent's learning.

Using the task sampling weights detailed above, the resultant task frequencies during the training phase across each of the tasks are presented in Figure 5.16—we provide details on how the sampling rates change during the training phase in Appendix C.3.1. We see that, although the sampling of each of the tasks does

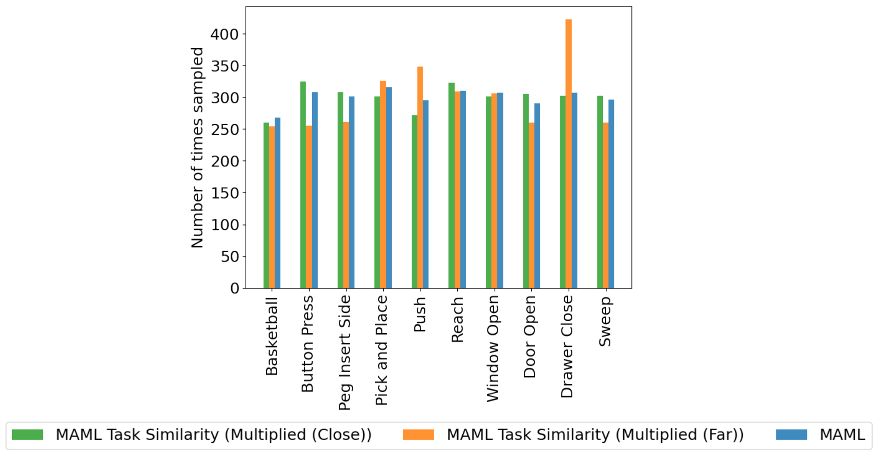
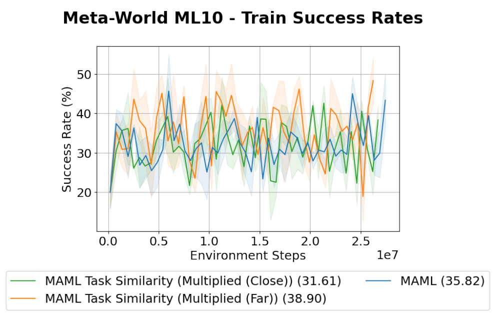
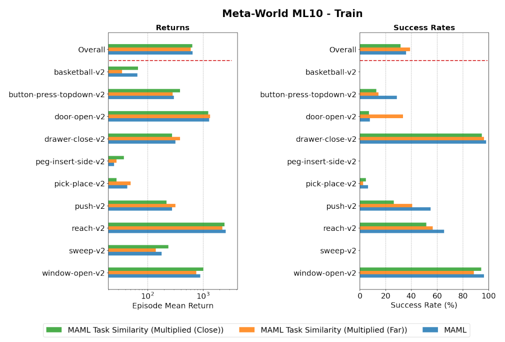


Figure 5.16: Overall task occurrences of the automatic curriculum approach during training on the Meta-World ML10 environment [YQH+19].

change over the course of the training phase, as previously shown, the overall task frequencies end up being virtually identical to that of the uniform task sampling distribution. This provides further indication of how the curriculum evolves over the entire training phase when using the new equation formulation, especially as during the ablation study we showed that both terms are capable of finding groupings of similar tasks. The only case of a task being sampled higher than others is the drawer-close-v2 task when the automatic curriculum is overall encouraging tasks that are far away. This highlights that the drawer-close-v2 task has an optimal representation most dissimilar to the other tasks.



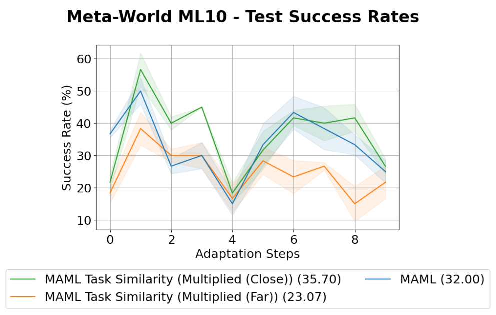
(a) Overall train success rate.



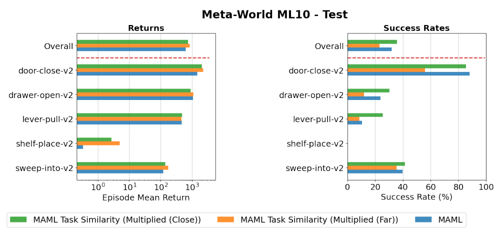
(b) Per task returns and success rates.

Figure 5.17: Returns and success rates of the automatic curriculum approach during training on the Meta-World ML10 environment [YQH+19] using Equation 5.8 to update the task sampling distribution, with error bars showing standard deviation for 3 repetitions of each method. Here we provide models using varying softmax temperatures and encouraging near or far.

When the agent is trained using the task frequencies detailed above, the performance of the agent, as detailed in Figure 5.17, is impacted by a small amount. Across all train tasks, the mean episodic returns are virtually the same to the baseline method, whilst success rates are very close in most tasks. Also, throughout the entire training phase, although it is noisy, the automated curriculum approach that overall encourages far tasks has the highest success rate by ∼3%. There is some separation in performance in the door-open-v2 and push-v2 with the far approach and baseline MAML approaches seeing highest performance, respectively. As the task frequencies are close to uniform for all methods here, these performances represent unbiased performance with regard to task sampling.



(a) Overall test success rate.



(b) Per task returns and success rates.

Figure 5.18: Returns and success rates, of the automatic curriculum approach during testing on the Meta-World ML10 environment [YQH+19], with error bars showing standard deviation for 3 repetitions of each method.

Following the training phase, we then test the agent on the Meta-World ML10 test task distribution, with the results shown in Figure 5.18. We see that although the automatic curriculum task similarity (far) method had the highest success rate during training, it shows dramatically worse performance than both the baseline MAML and the task similarity (close) method. We see a reduction in success rate across nearly all tasks in the test set. This shows that the task similarity (far) method overfit to the training set. This provides further evidence to suggest that the gradient change and co-task similarity terms are in line with generating behaviour the agrees with idea that we should encourage the learning of close tasks first. In regard to the highest performance, the task similarity (close) method has ∼3% higher success rate than the baseline, in particular showing higher performance than both other methods in the lever-pull-v2 task. As such, it is evident that this method has limited the overfitting found in the task similarity (far) method.

Method	**Train Success**	**Num. Samples** Top	Bottom	**Sampling StdDev**	**Test Success**
MAML	35.82	—	—	12.88	32.00
Multiplied (Close)	31.61	button-press -topdown-v2 reach-v2	basketball-v2 push-v2	19.02	35.70
Multiplied (Far)	38.90	push-v2 drawer-close-v2	basketball-v2 button-press -topdown-v2	51.95	23.07

Table 5.5: Summation of the effect of gradient-based automatic curriculum learning, using gradient change and co-task similarity, on task sampling rates in relation to tasks sampled, top and bottom 20th percentile of samples, and the resulting success rates on the Meta-World ML10 environment tasks, with standard deviation from 3 repetitions of each method.

A summation of the above results is detailed in Table 5.5 which allows us to compare how the task sampling rates affects both train and test performance. As mentioned, we can see the overfitting of the task similarity (far) method which has a much greater standard deviation in its task sampling and in particular has sampled the least from button-press-topdown-v2 which is one of the most sampled tasks in the task similarity (close) method. This ability to generate curricula that select opposite tasks to mainly sample from indicates that the switch does effect overall sampling and is not overcome by the terms. This result also follows along with the results in current research [BLCW09] along with the ablation study results, which suggests that sampling closer tasks encourages greater adaptation in the meta-learners model. However, we do see that they share a similar least sampled task, the basketball-v2 task, which indicates the adaptability of the curriculum, as it enables for both approaches to share similar information—this effect was not seen in the ablation study where the inverse hyperparameters generally led to inverse sampling distributions without cross-over.

Overall, on the Meta-World ML10 task distribution we see that by using the automatic curriculum learning approach we can adapt the task sampling distribution to increase the adaptability of a meta-learning agent, whilst reducing the overfitting evident in many of the hyperparameter configurations in the ablation study, without requiring for a set of hyperparameters to control the curriculum.

5.5.2.2 Meta-World ML45

We now show the results of using the updated gradient-based task similarity metric, as detailed in Equation 5.8, on the Meta-World ML45 environment [YQH+19], and provide an analysis of its affect on the generated curriculum and the resulting performance. In these experiments we focus on investigating how the approach works in a much larger and diverse task distribution than previous experiments, and whether the use of a non-uniform task distribution will decrease the impact that variation in the task distribution has on the agent's ability to adapt.

Across the training phase we calculate the values for both the gradient change and co-task similarity terms in Equation 5.8 to form the task sampling weights–these values are shown in Figure 5.19. We see that the gradient change values have a large separation across many of the tasks, with both the task similarity methods varying in the tasks that they are close and far from. This in particular is the case in the handle-press-side-v2, drawer-close-v2, dial-turn-v2, button-press-v2 and button-press-topdown-v2 tasks where we see consistent separation across the entire training phase. This means that all three method variations are finding themselves in very different locations within the parameter-space, providing an indication of the differences in curriculum with regard to non-uniform sampling. We also see that the co-task similarity measures reveal separation in the trajectories and tasks that they differ similarity in. In particular, we see more variation in the early stages of training with the co-task similarities which then converge to similar trajectories nearer the end—this happens predominantly in the plate-slide-side-v2, lever-pull-v2, stick-pull-v2 and sweep-v2 tasks. The resulting task sampling weights show that the task similarity (close) method is very similar to the uniform task sampling weights, with small periods of weight changes occurring throughout training. This is in comparison to the task sampling method that encourages far tasks, which shows periods of diverse task sampling during the early stages. This is in particular found in the push-v2, pick-place-v2 and level-pull-v2 tasks, which then converge towards the uniform task weights in the later stages of training. Overall, this does indicate that, in particular for the task similarity (far) method, we are seeing adaptive curriculums that are evolving over the entire training phase.

Using the task sampling weights detailed above, the resultant task frequencies during the training phase across each of the tasks are presented in Figure 5.20—we provide details on how the sampling rates change during the training phase

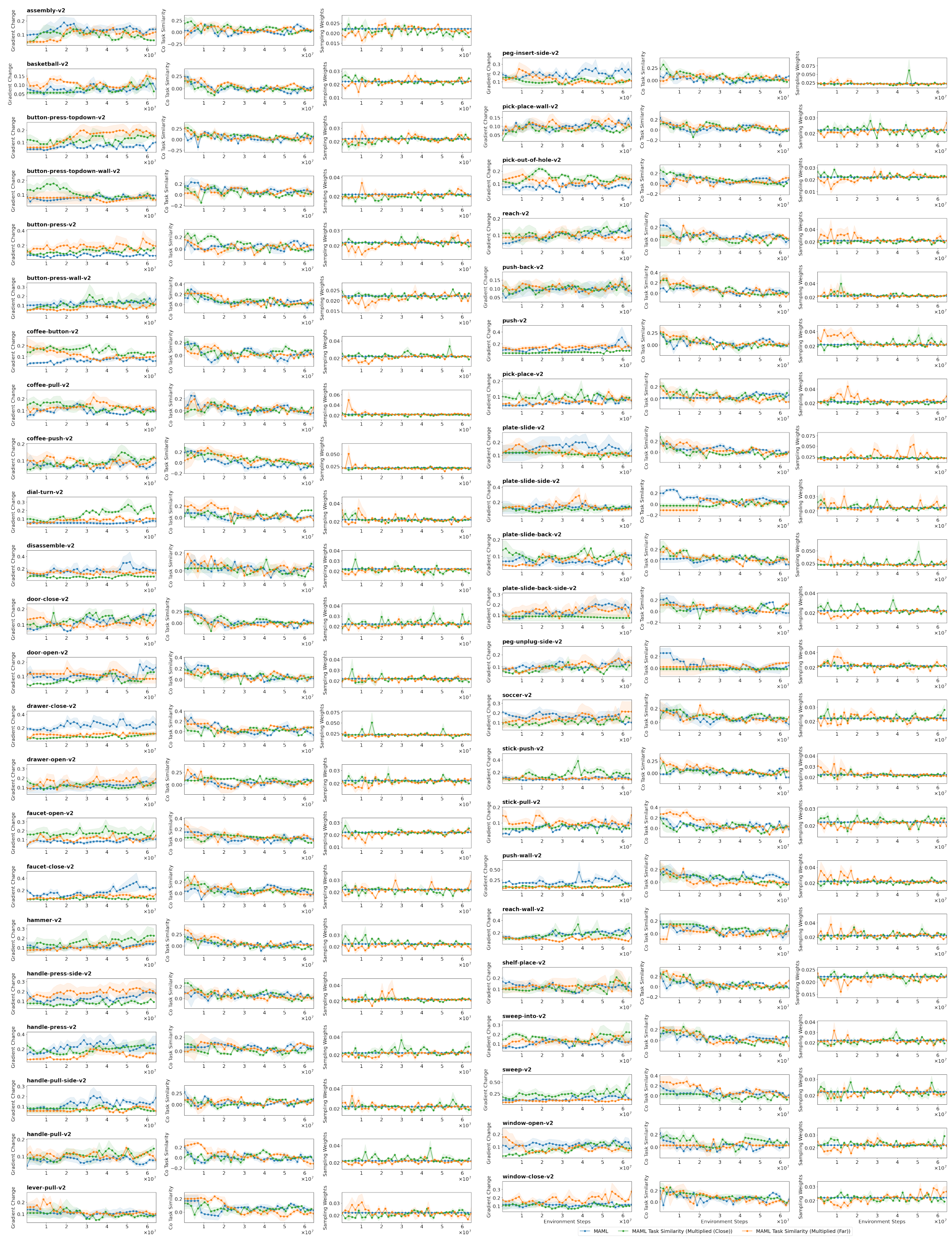


Figure 5.19: Gradient change and task sampling weights, with error bars showing standard deviation for 3 repetitions, of each method formed using the automatic curriculum approach during training on the Meta-World ML45 environment [YQH+19].

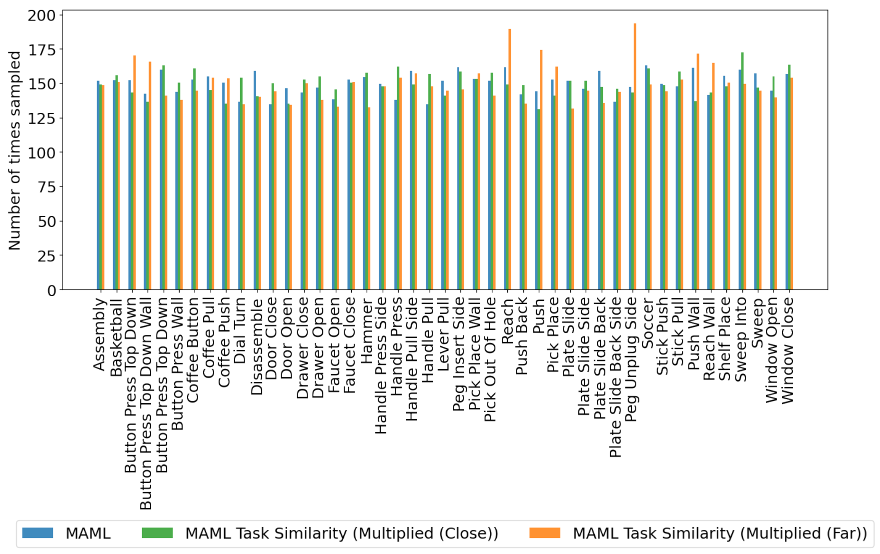
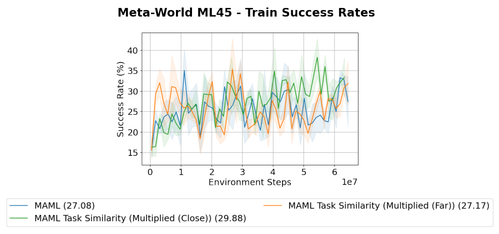


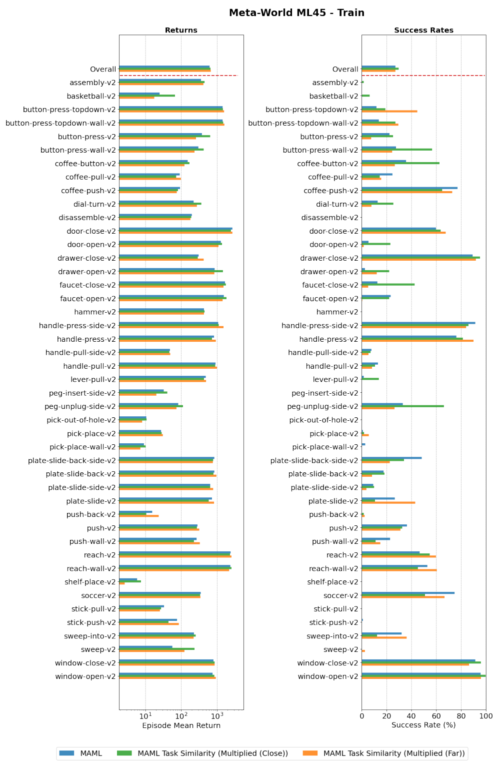
Figure 5.20: Overall task occurrences of the automatic curriculum approach during training on the Meta-world ML45 environment [YQH+19].

in Appendix C.3.2. We can see that both methods produce task frequencies that are very similar to the uniform task sampling distribution. The differences in task sampling occurs mainly in the task sampling approach which encourages far tasks where we see higher sampling rates in the reach-v2, peg-unplug-side-v2, push-v2 and push-wall-v2 tasks. Overall, these frequencies indicate that although we have task weight changes throughout training, the curricula are adapting over the course of the training phase which is resulting in the changes being balanced out when looking at the overall training frequency.

When the agent is trained with the task frequencies detailed above, the performance of the agent is highly affected, as shown in Figure 5.21. We can see that, although all of the methods have very similar returns across all of the tasks and overall success rates, there are several tasks where each of the method variations perform differently. In particular, the task similarity method that encourages close tasks sees much higher success rates on tasks including the button-press-wall-v2, coffee-button-v2, faucet-close-v2 and lever-pull-v2 tasks and is the only method that has non-zero success rates on both the assembly-v2 and basketball-v2 task. This however as meant that the performance on the other tasks are marginally worse than both the baseline and task similarity (far) methods, which each share very similar per-task success rates. Overall, this means that this method has been able to learn across a higher diversity of training tasks more



(a) Overall train success rate.

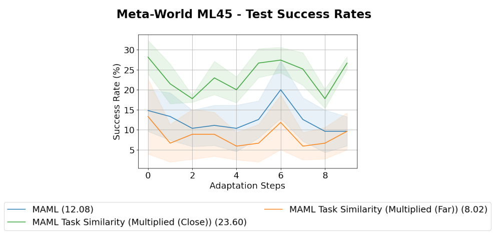


(b) Per task returns and success rates.

Figure 5.21: Returns and success rates of the automatic curriculum approach during training on the Meta-world ML45 environment [YQH+19] for tasks sampling methods, using Equation 5.8 to update the task sampling distribution and encouraging close or far tasks, along with the baseline uniform task sampling distribution method, with error bars showing standard deviation for 3 repetitions of each method.

evenly with regard to their success rate.

Having performed training, we then test the adaptation of the agents on the ML45 test task distribution. The results of which are presented in Figure 5.22. We can see that the returns of the methods are very similar, however the task similarity (close) method has the highest overall mean episodic returns due to significantly higher returns on both the bin-picking-v2 and hand-insert-v2 tasks. The result of the task similarity method that focuses on close tasks performing better across a broad range of tasks has resulted in an agent that performs adaptation to a new set of tasks far better than both the baseline approach and the task similarity approach that focuses on far tasks. Overall, we see a success rate increase of ~10-15% compared to the other methods. The results also show that on the box-close-v2 task, both the task similarity-based methods have non-zero


(a) Overall test success rate.

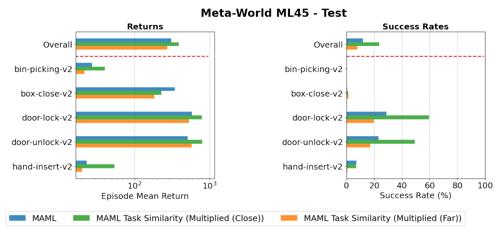

(b) Per task returns and success rates.

Figure 5.22: Returns and success rates of the automatic curriculum approach during testing on the Meta-World ML45 environment [YQH+19] for tasks sampling methods encouraging close or far tasks, along with the baseline MAML method, with error bars showing standard deviation for 3 repetitions of each method.

success compared to zero success rate on the baseline method. This shows that by using an adaptive curriculum we can form a learner that is able to adapt well to new tasks. However, the low overall success rate, and the zero success rate on the hand-insert-v2 from the task similarity (far) method suggests that this method overfit the training tasks. This provides further evidence towards the efficacy of a global target that encourages closer tasks in facilitate the greater adaptability of an agent.

Method	**Train Success**	**Num. Samples** Top	Bottom	**Sampling StdDev**	**Test Success**
MAML	27.08	—	—	7.82	12.08
Multiplied (Close)	29.88	button-press-v2 handle-press-v2 soccer-v2 sweep-into-v2 window-close-v2	button-press -topdown-wall-v2 coffee-push-v2 door-open-v2 push-v2 push-wall-v2	8.54	23.60
Multiplied (Far)	27.17	button-press -topdown-v2 reach-v2 push-v2 peg-unplug-side-v2 push-wall-v2	dial-turn-v2 door-open-v2 faucet-open-v2 hammer-v2 plate-slide-v2	13.68	8.02

Table 5.6: Summation of the effect of gradient-based automatic curriculum learning, using gradient change and co-task similarity, on task sampling rates in relation to tasks sampled, top and bottom 10th percentile of samples, and the resulting success rates on the Meta-World ML45 environment tasks, with standard deviation from 3 repetitions of each method.

We provide a summary of the results discussed above in Table 5.6 which allows us to compare how the tasks sampled, the top and bottom 10$^{\text{the}}$ percentile of tasks sampled, affected train and test success rates. The top and bottom sampled tasks for both of the task similarity-based sampling methods provide a mix of tasks in each categorisation. We see both methods sample the door-open-v2 task the least, whereas the task similarity (far) approach samples the push-wall-v2 task the most, but the task similarity (close) approach samples it the least. This cross-over of sampling indicates that neither of the methods represent the opposite sampling curricula, as would be insinuated by the switch, and instead both form adaptive curricula that are learnt based on the agent's current learning situation. As was indicated by the task frequencies, both of these curricula task sampling are very close to the standard deviation of the uniform task sampling distribution approach, only ∼5% higher, which has meant that the much of the overfitting found in the ablation study has been mitigated.

Overall, we see that by using a non-uniform task sampling distribution we can encourage the sampling of tasks that enable for an agent to learn an representation that can adapt to new, but related tasks, in a highly diverse task distribution. Of particular importance is the effectiveness of this approach in prioritising the sampling of close tasks, as defined by the task similarity metric.

5.5.2.3 Generalisation

In the previous section we have investigated the manner that Equation 5.8 learns a non-uniform task sampling distribution based on the task similarity metric terms of gradient change and co-task similarity. The have been shown to allows the task curriculum to adapt with regard to the agent's current behaviour in terms of its performance across the task distribution. To further investigate this method, we can analyse the generalisability of the method by using the definition of generalisation, as detailed in Chapter 4.

For these experiments we will be using the Meta-World ML10 task distribution task groupings, for both train and test, as provided in the meta-RL experiment setup in Section 4.4.1.2—as was detailed in this section, the performance measure is defined as the success rate across tasks, and the task distance as the pre-computed cosine distance between sampled observations. Using this environment allows us to further understand the method with regard to the Meta-World environment, following the results from the experiments above.

As shown by the previous results on both the ML10 and ML45 task distributions, the automatic curriculum approach shows greater performance when we negate the terms in Equation 5.8 and encourage closer tasks. As such, the task similarity method used in the generalisation experiments here uses the negated version of the equation. For use as a baseline, we provide the results of MAML with a uniform task sampling distribution—the results of the baseline are taken from Section 4.5.2.2 and are labelled in the results as MAML.

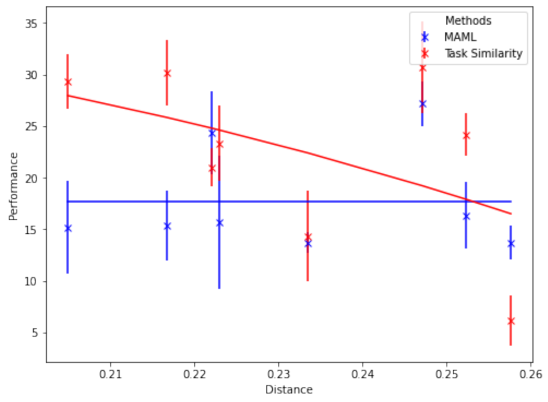


Figure 5.23: Performance (success rate of last 5 epochs, with error bars showing standard deviation for 3 repetitions) against task distance of the automatic curriculum approach on the Meta-World ML10 task distribution.

As can be seen in the comparison between performance and task distance, as shown in Figure 4.14, we see a reduction in the performance of the gradient-based task similarity approach, a negative correlation between performance and distance, compared to a constant performance as distance increases found in the baseline. We do however see a higher performance across most of the task distance domain, and therefore a higher function integral, with the gradient-based task similarity method, between train and test task distributions. This indicates that the task similarity metric is able to generalise across small adaptations in the observation space, whilst over larger adaptations performance degrades linearly to the point that its performance is lower than that of the baseline. However, the constant relationship shown in the baseline indicates that the task difficulty may be a factor in the gradients of these results.

As such, the next step is to look at the how train performance changes with regard to train to test distances. This is detailed in Figure 5.24. As we can see, there is no correlation between performance and task distance, however across both methods there is a large amount of noise in the results which indicates

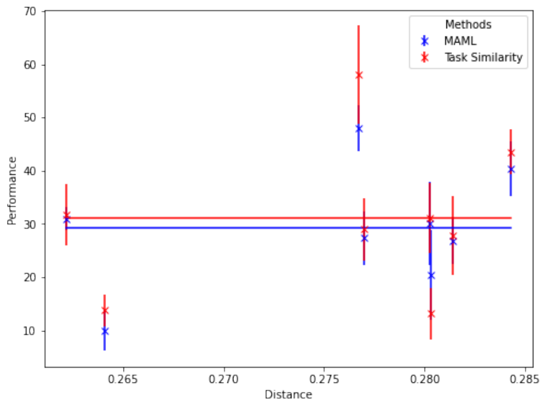


Figure 5.24: Train performance (mean success rate of last 5 epochs, with error bars showing standard deviation for 3 repetitions) against task distance of the automatic curriculum approach on the Meta-World ML10 task distribution.

that the difficulty of the train tasks will have an impact on the test performance to task distance relationship. As per the process described in Section 4.3.2.1, we must therefore perform normalisation of the test performance with the train performance in order to produce a fair representation of the generalisability of the agent.

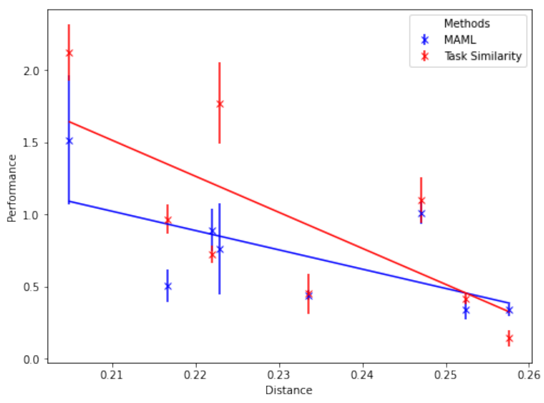


Figure 5.25: Normalised performance (test / train success rate of last 5 epochs, with error bars showing standard deviation for 3 repetitions) against task distance of the automatic curriculum approach on the Meta-World ML10 task distribution.

Once we perform performance normalisation, with the results of this shown in Figure 5.25, we see that the performance of the gradient-based task similarity method still, with comparison against the non-normalised performance, maintains higher performance across nearly all of the task distance domain, within the observation space, used in these experiments—this results in the gradient-based task similarity approach having a higher integral of its performance against task distance function and therefore we can state that it is more generalisable. It is important to note however, that the rate of performance degradation of the baseline is less than that of the task similarity method, but over results in a lower integral due to the much lower start performance within the distance domain within these experiments. In regard to test performance relative to train performance, we can see that the task similarity method is able to teach a meta-learner to adapt to tasks in a manner that facilitates performance increases between 1.5 to 2 times as found in the training task distribution, compared to equal to the training performance by the baseline. This suggests that the curriculum that the agent was exposed to using the task similarity (close) method facilitated the extrapolation of ideas from the training distribution that are highly useful in tasks within the test distribution.

As such, we can say that in regards to generalisation, as detailed in Section 4.3.1, the task similarity method is able to maintain a higher performance, in terms of success rate, across most of the task distance domain used in this experiment than the baseline, however its performance degrades at a faster rate. Overall, these two different facets of generalisation provide a useful insight into the conditions, in terms of task distribution properties, that both methods are applicable to. It also highlights that both methods are effective at generalisation in different regards, and as such the applicability of the method depends on the requirements that you have for the problem.

5.6 Discussion

Throughout these experiments the intention was to investigate: (a) whether, and how, gradients from a gradient-based meta-learners inner-loop produce an effective measure of task similarity, (b) how does a non-uniform task sampling distribution affect a gradient-based meta-learning method, MAML, with regards to performance, in particular its ability to form a generalisable representation that can adapt in diverse environments, and (c) whether we could leverage the gradient-based task similarity metric to form an automatic curriculum that optimises the current task distribution to the current state of the learner in order to balance overfitting and underfitting, whilst encouraging generalisation.

With the experiments, and associated results, we can see that gradients can be used to form a useful task similarity metric that produces effective task curricula within a gradient-based meta-learning framework. This work provides further empirical evidence to the work by [DCJ$^+$18] who has shown that gradients are a source of information for stating task similarity. The extended method used both the gradient change and task similarity terms, which encompass two specific aspects of a gradients similarity, with regard to the tasks sampled in a meta-learning framework. We've shown that using gradients computed during training allows for us to learn automatic curricula, however the task similarities found are then innately bound to the performance of the particular method using it—-this is indicated by the evolving values throughout training. As such, these task similarities are not necessarily able to be used by other methods. Comparing the usefulness of gradient change and co-task similarity, we found throughout the experiments that the gradient change term had a larger affect on defining overall separation in task sampling weights. Overall, we found that the co-task similarity measure didn't form a large amount of separation from the other methods and in general all methods followed the same trajectory throughout training. One of the reasons for this is that the co-task similarity measure was taken as the mean across task similarities which reduced any, although seemingly weak, signal—we did investigate using other approaches including taking the max of the similarities, however this introduced a large amount of noise into the term which reduced the curriculum consistency, and had a detrimental affect on performance in certain cases—however, other approaches such as clustering analysis [CPS03] may be applicable here to remedy this issue. Although both of the task similarity metric terms used in Equation 5.2 are defined as low-level behaviour-based task similarity metrics, as described in Chapter 3, we have seen that we are able to increase generalisability in an environment that has a diverse task distribution containing various adaptations, where all experiments included sampling adaptations in the goal space for each task, and the overall skills required differed between tasks.

We see that a non-uniform task distribution can heavily affect how generalisable an agent learnt through meta-learning can be, as shown by the large variability in performance that the MAML method had when exposed to different forms of distribution for task sampling. A curriculum learning approach formed using a non-uniform task distribution can facilitate improved generalisation in MAML, and should also be applicable in other gradient-based meta-learning approaches. In particular, we have seen through both the ablation study and the automatic curriculum learning extension the benefit of learning closer tasks to the current representation first, as defined in particular by the gradient change term, with the co-task similarity term providing a balance to overfitting on these tasks. In both results, the approaches that encouraged similar tasks created the agents who could adapt quicker to task distribution changes. We found that the gradient change and co-task similarity terms were used on their own heavily suffered from overfitting and underfitting. The combination helped to remedy this—in particular when we encouraged close tasks with gradient change and then balanced the overfitting with the co-task similarity term (either encouraging close or far task groupings), however the automatic curriculum learning approach did this further by adapting the curriculum throughout training, and allowed for diverse sampling whilst maintaining a standard deviation close to a uniform task sampling distribution that ensured overfitting didn't occur. We also propose that the work conducted here allows allows for the mitigation of negative transfer through the selection of tasks which will aid in the agent's learning, effectively producing the same behaviour as work by [YKG+20] who perform gradient muting to remove contrasting gradients from the parameters updates in order for the learner to only learn positively. Further, the results from the curriculum learning approach showed that a task similarity metric that is allowed to evolve over the course of training can allow for a more effective representation of the requirements of the agent throughout its learning phase, which is counter to the advice of [CS05] who recommend a task similarity metric be pre-computed as discussed in Section 3.3.2.

5.6.1 Limitations

The extension discussed in this chapter facilitates a learnt task curriculum in gradient-based meta-learning. However, the method does have limitations which either constrain its application to certain environments, whilst also introducing a few hyperparameters that must be tuned in order for the curriculum to be useful.

The main limitation in the current method is this it requires task identifiers from the environment to indicate which task is currently sampled from (i.e. the current work focuses on perfect information over what task the gradients are taken from to match to task sampling distribution). This information is not always available when sampling from an environment, especially in real-world applications. We have seen that gradients don't provide enough discriminating information to be used for task identifiers due to the amount of noise in the gradients themselves—approaches have been suggested for dealing with the variance in RL gradients [HCSD20], but this remains an open problem. Another issue is that the metrics used here evolve over the entire training phase as they represent the similarity with regard to the agent's representation of the tasks, and therefore lack consistency. We would therefore need to incorporate other sources of information, such as observations, to produce effective task identifiers that can be used to discriminate between tasks.

Another limitation are the use of hyperparameters in Equation 5.2. This introduces switch values for controlling the effect of the gradient change and co-task similarity terms, along with a softmax temperature and an epsilon value used in the epsilon-greedy task sampling distribution selection policy. These hyperparameters highly control the type of curricula that are created, and therefore must be appropriately tuned for the given use case. We were able to account for the gradient change and co-task similarity hyperparameters using the automatic curriculum learning approach described in Equation 5.8, however both other hyperparameter values still require tuning. For these we proposed the following heuristics: for the softmax temperature, as discussed in Section 5.3, we set the value to $\frac{1}{\text{Number of Tasks}}$, which allowed for separation in task sampling, and the epsilon value was set to 0.2 for selection of the uniform task distribution—this value however affects the amount of overfitting that occurs when forming the curriculum and should be increased if excessive overfitting occurs in order to balance exploitation of performance against the exploration of new experiences. Both of these hyperparameters need to be balanced as they will partially control the amount of overfitting and underfitting that occurs.

5.6.2 Research Avenues

From the results shown here, we can see that this approach should also be applicable to other methods, as long as the method has access to the separate gradient updates from each task interaction, rather than directly updating by combining all task trajectories together. Although the standard MAML method was used here, this approach should also be applicable to it's extensions, such as e-MAML [SYH$^+$19], as it leverages gradient similarity to form a more effective task curriculum. Furthermore, although the results in this chapter focused on applying this mechanism on a meta-RL approach due its overall relevance to remedying the issue of generalisation in RL, we suggest that this approach is not confined to RL and instead could also be used in the supervised learning approaches of this method due to the task representation and overall similarity metric is not RL specific. Along with meta-learning approaches we also suggest that this should be applied to a MTL setting. An example of this would be within the policy distillation method [RCG$^+$15] to form a curriculum for the scheduling of task sampling, rather than sampling tasks iteratively.

Although the approach that we have presented here produces increased generalisability in the agent through evolving the curriculum throughout the training phase, both the gradient change, co-task similarity metrics and their combination are classified as low-level behaviour-based task similarity metrics by the taxonomy as defined in Chapter 3. As such, a possible avenue for further work would be to look at using gradients to form a higher-order task representation space, in order to analyse the possibility of producing higher-order similarities across tasks in order to increase the degree of adaption and diversity of task distribution that this method can be effective across.

5.7 Summary

In this chapter, the overall aim was to investigate whether a task similarity metric using the gradients of a gradient-based meta-learning method would provide useful information with regard to forming task curricula that could encourage greater agent generalisation in RL environments consisting of diverse multi-task distributions.

Through the results of the ablation study we have shown that the the gradient change and co-task similarity terms introduced in Equation 5.2 are able to extrapolate information from the gradients of a meta-learner to formulate a task similarity metric that can indicate task relevance with regard to the specific requirements of the agent's representation of optimal behaviour in the environment. This provides further evidence of the effectiveness of low-level behaviour-based task similarity metrics on environments containing several adaptations in the task distribution.

Following these findings, this work contributes a further understanding of the effect of a non-uniform task sampling distribution in a meta-RL setting. We showed that by using a task similarity-based non-uniform task sampling distribution we can form curricula that controls for the optimal amount of interaction of an agent with a particular task. This contributes to the overall aim of providing approaches that have been shown to be capable of remedying the issue of generalisation in RL.

We then expanded upon this work by introducing an ACL approach that remedied some of the limitations of using hyperparameters to control the curriculum formed. This was done by using a task similarity metric that allowed for curricula to be evolved throughout the training of a meta-learner, thus continually exposing the agent to relevant experiences according to the agent's representation of optimal behaviour. This was found to be highly effective at mitigating both overfitting and underfitting, and thus facilitating greater generalisation. This highlighted the importance of using the agents current state as a guide for further experience.

Overall, this method showed the efficacy of combining gradient-based meta-RL with a metric learning approach to form a similarity metric that can influence the sampling of relevant tasks during training to form a curriculum that encourages generalisation.

Chapter 6

Conclusions

The overall goal of this thesis was to further the understanding of generalisation in current RL literature by answering the following question: *How does task similarity affect the generalisability of reinforcement learning agents?*. In this chapter, we look at the contributions we made in this thesis, and detail how these helped to answer the research questions defined in order to complete this goal. With this, we detail how the contributions fit in with, and aid in the expanding of the current research within the area. Finally, we close the thesis by discussing the future research directions motivated by the work in this thesis.

6.1 Summary of Research

In this section, we provide a summary of the contributions of each chapter in this thesis, and outline how these fit in with current research.

6.1.1 Understanding the Similarity of Reinforcement Learning Tasks

In Chapter 3, we discussed and highlighted the work in current literature that make use of different forms of task similarity in dealing with the problem of sharing knowledge across RL tasks. The was performed by providing a taxonomy that categorised the approaches by the level of abstraction of their task similarity metrics, and whether they use a task representation that was formed from environment- or behaviour-based properties. By looking at these methods we gave a means of discussing the current drawbacks and avenues for future work that need to be investigated to improve the current state of RL methods and environments with regard to generalisation in order to increase the applicability of the paradigm to real-world problems.

We found that definitions of task similarity metrics in current literature mainly focus on MDP level characteristics, whereas the taxonomy we present also incorporates and compares behaviour-based approaches such as a gradient-based task representation. They also don't focus on the overall target of the task similarity metrics, and therefore we are lacking an overall understanding of the ways that similarity has been used, and the possibilities that we have for task representations that they can be used in, to encourage particular performance targets. In particular, the work discussed in Chapter 3 focuses on the level of abstraction of the task representation, with regard to what forms of learning can occur at these levels. We see that the environment-based methods were limited in the forms of adaptation they were able to share across, whilst behaviour-based methods allowed for shared knowledge across all forms of environment adaptation. In relation to the level of abstraction, we found that no matter the level of abstraction they facilitated greater generalisation, however a higher-level abstraction facilitated greater transfer over higher diversity in adaptation. As such, by focusing on the possible effect of the task similarity metric provides further relevance of task similarity metrics to solving current issues in a spectrum of machine learning paradigms, but in particular the work here will provide a reference for applying these techniques in RL.

6.1.2 A Multi-Task Definition of Generalisation in Reinforcement Learning

In Chapter 4, we provided a definition of generalisation, summarised as the relationship between task similarity and performance, which has provided an insight into how adaptations within a task distribution affect current state-of-the-art methods within multi-task and meta-RL. As part of the definition we provided guidance over how to use it in analysing generalisation in RL models trained in both a multi-task and meta-learning setting (i.e. in instances where we have only a set of tasks to learn on, or a train/test set split in the task distribution, respectively).

The challenge of improving the generalisability of RL agents has garnered a large amount of attention recently due to the impact it would have on the applicability of the paradigm in real-world applications. The work detailed in this chapter allows for the expanding of current research, which currently doesn't have a task and model agnostic approach to analysing generalisation, with regard to a particular type of adaptation as defined by the task similarity metric, to allow for the analysis of generalisation across all types of RL methods. This leverages recent work on environments that have adaptive task distributions where we can form a consistent task representation in order to compare the model's measure of performance against the task distributions similarity in a form of adaptation.

6.1.3 Gradient-based Task Similarity Generated Curriculum in Multi-Task Meta-Reinforcement Learning

In Chapter 5, we investigated the feasibility of using a gradient-based task similarity metric, using the low-level abstraction performance-based task representation formed from the gradients of the inner-loop of a gradient-based method, MAML, based on both the gradient change and co-task similarity terms. Following the results of the ablation study, which confirmed the metrics application in regard to separation, we provided an extension of the MAML method that formed automatically learnt curricula to encourage generalisation, as defined by the definition in Chapter 4. The results presented indicated that this gradient-based metric enabled for effective generalisation over several forms of adaptation within the task distribution due to increased performance over most of the task distribution against MAML using a uniform task sampling distribution, in particular in the observation space as shown in the generalisation experiments. This indicates the efficacy of behaviour-based approaches for targeting specific learning within the agent as discussed in Chapter 3. Furthermore, although this metric had a low-level of abstraction, it was able to leverage the agent's own requirements during training to produce a highly generalisable agent, providing evidence to suggest automated curricula are vital in training generalisable agents.

This work represents a step forward in tackling the major issue of a lack of research into remedying generalisation in RL using task similarity-based curricula—the work presented in this chapter explores this in a meta-learning setting, however the mechanics discussed are applicable to other approaches that use task distributions in the training phase as this work primarily focuses on approaches that deal with multiple tasks with diverse properties with regard to their task representation. Many of the current approaches use uniform task sampling distributions, whereas we have shown in this work that a non-uniform task sampling distribution can highly effective in exposing the agents to tasks that will encourage generalisation.

6.2 Outlook and Future Work

In this section we discuss possible directions of future research motivated by the work in this thesis.

6.2.1 Understanding the Similarity of Reinforcement Learning Tasks

In Chapter 3, we detailed a taxonomy of the task similarities, with regards to both the task representation and the measure of similarity within this representation space. We identified that one of the main aspects we found lacking in current research is the investigation into the impact of abstraction with regard to task representations, and their measure of similarity within tasks, with many of the approaches using low-level abstractions. This represents what we see as a vital area of future work, in particular due to the results in the rest of the thesis, to understand the particular task settings that current methods are able to generalise across. In order to understand why an agent is unable to generalise, we must understand what we need to transfer and whether a method is able to capture this information from the task distribution itself. Furthermore, many of the performance targets we found in the methods that use task similarity metrics did not focus on the generalisability of an agent's resulting representation, and instead only looking at performance on static environments. We hope that the work conducted in this thesis highlights the importance of analysing the generalisability of an agent through the use of a quantifiable measure of generalisation that uses the similarity of the task distribution on an environment consisting of a diverse and adaptive task distribution, as discussed in Chapter 4.

6.2.2 A Multi-task Generalisation Definition in Reinforcement Learning

The definition of generalisation in RL that we introduce in Chapter 4 allows us to understand whether a method facilitates the learning of an agent that is capable of, and to what degree, generalisation over a particular task similarity. Chapter 4 provided details of the current limitations and discusses possible research avenues in Section 5.6, in which we provide a summary of here.

The results focused on providing examples of the relationship of task similarity and performance, which provided evidence of *whether* a method is generalising, whilst this approach relies on the task similarity metric for explaining *why* the particular task distribution groupings cause performance to degrade with increasing task distance. This reliance on the task similarity metric however means that a metric that represents the similarity of a particular adaptation needs to be selected. In order to improve the process of investigating the limitation in approaches we suggest that further work should be conducted into forming a task similarity metric that is able to automatically select relevant features that are degrading performance, and therefore reduce the need to curate specific adaptation-focused metrics.

Combining these two methods of stating *whether* and *why* a method is generalising, we propose that the method should be a guiding principle used when analysing any new RL method, as we do in Chapter 5, due to the importance of generalisation in facilitating the application of RL methods to real-world problems. Further from this, there is currently not an accepted view of the state of RL with regards to generalisation. Although we provide an analysis for both a MTL and meta-RL method, further analysis across a broad range of methods within RL needs to be conducted to fully understand the current generalisation limitations of RL methods.

Finally, one of the grounding aspects of this work is effect of task distance on performance and how we define generalisability in terms of these. However, as shown in this Chapter 4, the relationship is heavily dependent on the consistency of task similarity metrics across methods. The findings we presented, which provided evidence of the efficacy of this relationship, are purely empirical, whereas a theoretical grounding of this relationship would provide further understanding of the uses of task similarity metrics, in particular when comparing approaches.

6.2.3 Gradient-based Task Similarity Generated Curricula in Multi-Task Meta-Reinforcement Learning

In Chapter 5, we introduced an extension to the standard MAML method which allows for an adaptive curriculum to be formed using a gradient-based task similarity metric, using both gradient change and co-task similarity. However, in the results we showed that the metric doesn't always form a curriculum that encourages generalisation, in part due to the inherent noise within gradients and the level of abstraction of representation not always extracting relevant features. Therefore, one approach that would be of interest is that of automatically generating a metric that accounts for this. The field of metric learning [CWZ18] has been applied in many different circumstances and could be used to form an optimal metric for any particular environment that optimises for generalisability—optimally this would incorporate the definition of generalisation, as outlined in Chapter 4, to optimise for a reduced task similarity to performance gradient.

In Chapter 5, we focus on using a non-uniform task sampling distribution within a gradient-based meta-learning approach using gradient similarity, however the mechanisms introduced are not limited to this method and setting. The findings from this work indicate that the use of non-uniform task sampling distribution in general could aid in the encouraging of generalisation with regard in particular to the balancing of overfitting and underfitting. As such, we propose that many of the approaches in both meta-learning and multi-task learning that currently use a uniform sampling distribution, for example policy distillation approaches, would benefit from the use of a non-uniform task sampling distribution, controlled by the gradients of RL updates on these approaches—one aspect of research that would need to be conducted is investigating the task separation found in the task similarity metrics space for the particular setting.

Finally, this approach currently is limited in that it requires perfect information from the environment with regards to the particular task that we are sampling from, which is not always the case. As such, it would useful to analyse how this method would manage with a task embedding (i.e. a learnt task identifier), as this would likely incorporate noise into the system and therefore affect the resultant curriculum—the gradients were considered for this case, however as shown by the results they don't form constant separation, and therefore would not be applicable. As part of the proposed method that uses gradients to form the task similarity metric used in curriculum learning, we incorporated mechanisms to counter the inherent noise in RL gradients, and therefore it will be important to analyse the effect of the addition of further noise within the system when applied in environments without perfect task information.

6.3 Closing Remarks

In this thesis, we provided an investigation into the topic of generalisation in RL. We provide a comprehensive review of task similarity and its usages in current literature, detailing a taxonomy that indicated gaps in current research. Following this, we then use our understanding of the usages of task similarity in a task and model-agnostic definition of generalisation to provide a framework for discussing the generalisability of an agent in an adaptive multi-task environment. Furthermore, we make use of a gradient-based task similarity metric within a meta-learning method which results indicated that it encourages generalisation through the learning of an automatically adapted curricula, in particular in highly diverse environments. All of these contribute evidence to the idea that an understanding and use of task similarity is pivotal in the training of agents that are able to generalise effectively in diverse and adaptive environments—as such, providing further understanding to the central question of this thesis.

Overall, this thesis represents an expanding on the previous work within the RL paradigm to applications that require generalisable agents in an area that has the potential to facilitate the uptake of the machine learning paradigm within several different real-world domains. Although recent work has been taken to expand the applicability of the paradigm to these domains, there still remains research needed to gain an understanding of the limitations in current methods, along with discovering approaches that will increase the generalisability of RL in situations where they need to deal with environments which include multi-task task distributions containing tasks with diverse differences. There has been an increase in the work focusing on the problem of generalisation in RL, however there still remains a large amount of work to do to fully understand why RL agent's aren't currently able to generalise in multi-task task distributions, and to uncover what we can do to fix it. This thesis has provided evidence that the use of task similarity metrics are a key component in enable us to do this.

Bibliography

[AET$^+$14] Haitham Ammar, Eric Eaton, Matthew Taylor, Constantin Decebal, Decebal Mocanu, Kurt Driessens, Gerhard Weiss, and Karl Tuyls. An automated measure of mdp similarity for transfer in reinforcement learning. 07 2014.

[AKKL18] Himani Arora, Rajath Kumar, Jason Krone, and Chong Li. Multi-task learning for continuous control. *arXiv preprint arXiv:1802.01034*, 2018.

[AMCB21] Rishabh Agarwal, Marlos C. Machado, Pablo Samuel Castro, and Marc G. Bellemare. Contrastive behavioral similarity embeddings for generalization in reinforcement learning, 2021.

[ARW21] Johannes Ackermann, Oliver Paul Richter, and Roger Wattenhofer. Unsupervised task clustering for multi-task reinforcement learning, 2021.

[BCP$^+$16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[BDM$^+$17] André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. In *Advances in neural information processing systems*, pages 4055–4065, 2017.

[BGP21] Glen Berseth, Florian Golemo, and Christopher Pal. Towards learning to imitate from a single video demonstration, 2021.

[BLCW09] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual* *international conference on machine learning*, pages 41–48. ACM, 2009.

[BNVB13] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, Jun 2013.

[Car98] Rich Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

[CHHS19] Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. *arXiv preprint arXiv:1912.01588*, 2019.

[CKH$^+$18] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. *arXiv preprint arXiv:1812.02341*, 2018.

[CPS03] James Carroll, Todd Peterson, and Kevin Seppi. Reinforcement learning task clustering. pages 66–72, 01 2003.

[Cra20] Michael Crawshaw. Multi-task learning with deep neural networks: A survey, 2020.

[CS05] James L Carroll and Kevin Seppi. Task similarity measures for transfer in reinforcement learning task libraries. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 2, pages 803–808. IEEE, 2005.

[CWZ18] Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Darkrank: Accelerating deep metric learning via cross sample similarities transfer. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[DCJ$^+$18] Yunshu Du, Wojciech M Czarnecki, Siddhant M Jayakumar, Razvan Pascanu, and Balaji Lakshminarayanan. Adapting auxiliary losses using gradient similarity. *arXiv preprint arXiv:1812.02224*, 2018.

[Del18] Tristan Deleu. Model-Agnostic Meta-Learning for Reinforcement Learning in PyTorch, 2018. Available at: https://github.com/tristandeleu/pytorch-maml-rl.

[DR19] Kshitij Dwivedi and Gemma Roig. Representation similarity analysis for efficient task taxonomy & transfer learning, 2019.

[EGIL18] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function, 2018.

[FAL17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017.

[FCPP12] Norman Ferns, Pablo Samuel Castro, Doina Precup, and Prakash Panangaden. Methods for computing state similarity in markov decision processes, 2012.

[FD02] David Foster and Peter Dayan. Structure in the space of value functions. *Machine Learning*, 49(2-3):325–346, 2002.

[FLBPP18a] Vincent François-Lavet, Yoshua Bengio, Doina Precup, and Joelle Pineau. Combined reinforcement learning via abstract representations. *arXiv preprint arXiv:1809.04506*, 2018.

[FLBPP18b] Vincent François-Lavet, Yoshua Bengio, Doina Precup, and Joelle Pineau. Combined reinforcement learning via abstract representations, 2018.

[FPP04] Norm Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite markov decision processes. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 162–169. AUAI Press, 2004.

[FTF$^+$20] Meire Fortunato, Melissa Tan, Ryan Faulkner, Steven Hansen, Adrià Puigdomènech Badia, Gavin Buttimore, Charlie Deck, Joel Z Leibo, and Charles Blundell. Generalization of reinforcement learners with working and episodic memory, 2020.

[FV06] Fernando Fernández and Manuela Veloso. Probabilistic policy reuse in a reinforcement learning agent. AAMAS '06, page 720–727, New York, NY, USA, 2006. Association for Computing Machinery.

[FYF$^+$16] Chelsea Finn, Tianhe Yu, Justin Fu, Pieter Abbeel, and Sergey Levine. Generalizing skills with semi-supervised reinforcement learning. *arXiv preprint arXiv:1612.00429*, 2016.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[GDL$^+$17] Abhishek Gupta, Coline Devin, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. *arXiv preprint arXiv:1703.02949*, 2017.

[GEFL20] Abhishek Gupta, Benjamin Eysenbach, Chelsea Finn, and Sergey Levine. Unsupervised meta-learning for reinforcement learning, 2020.

[GL21] Ricardo Luna Gutierrez and Matteo Leonetti. Information-theoretic task selection for meta-reinforcement learning, 2021.

[GPA07] Sertan Girgin, Faruk Polat, and Reda Alhajj. State similarity based approach for improving performance in rl. pages 817–822, 01 2007.

[GRK$^+$21] Dibya Ghosh, Jad Rahme, Aviral Kumar, Amy Zhang, Ryan P. Adams, and Sergey Levine. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability, 2021.

[GVK19] Sam Green, Craig M Vineyard, and Cetin Kaya Koç. Distillation strategies for proximal policy optimization. *arXiv preprint arXiv:1901.08128*, 2019.

[HAMS20] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey, 2020.

[HCSD20] Mohammed Sharafath Abdul Hameed, Gavneet Singh Chadha, Andreas Schwung, and Steven X. Ding. Gradient monitored reinforcement learning, 2020.

[HHS17] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *Advances in Neural Information Processing Systems*, pages 1731–1741, 2017.

[HLK18] Yen-Chang Hsu, Zhaoyang Lv, and Zsolt Kira. Learning to cluster in order to transfer across domains and tasks, 2018.

[HPR$^+$17] Irina Higgins, Arka Pal, Andrei A Rusu, Loic Matthey, Christopher P Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. Darla: Improving zero-shot transfer in reinforcement learning. *arXiv preprint arXiv:1707.08475*, 2017.

[HSB$^+$19] Felix Hill, Adam Santoro, David G. T. Barrett, Ari S. Morcos, and Timothy Lillicrap. Learning to make analogies by contrasting abstract relational structure, 2019.

[IRE16] David Isele, Mohammad Rostami, and Eric Eaton. Using task features for zero-shot knowledge transfer in lifelong learning. In *IJCAI*, pages 1620–1626, 2016.

[JGMF19] Yiding Jiang, Shixiang Gu, Kevin Murphy, and Chelsea Finn. Language as an abstraction for hierarchical deep reinforcement learning, 2019.

[JHE$^+$19] Allan Jabri, Kyle Hsu, Ben Eysenbach, Abhishek Gupta, Sergey Levine, and Chelsea Finn. Unsupervised curricula for visual meta-reinforcement learning, 2019.

[KB06] George Konidaris and Andrew Barto. Autonomous shaping: Knowledge transfer in reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 489–496. ACM, 2006.

[KB07] George Konidaris and Andrew G Barto. Building portable options: Skill transfer in reinforcement learning. In *IJCAI*, volume 7, pages 895–900, 2007.

[KB18] Thommen George Karimpanal and Roland Bouffanais. Self-organizing maps as a storage and transfer mechanism in reinforcement learning. *arXiv preprint arXiv:1807.07530*, 2018.

[KB19] Mahmut Kaya and H. Bilge. Deep metric learning: A survey. *Symmetry*, 11:1066, 08 2019.

[KGS11] Zhuoliang Kang, Kristen Grauman, and Fei Sha. Learning with whom to share in multi-task feature learning. In *ICML*, 2011.

[KOK+21] Taehyeon Kim, Jaehoon Oh, NakYil Kim, Sangwook Cho, and Se-Young Yun. Comparing kullback-leibler divergence and mean squared error loss in knowledge distillation, 2021.

[KSB17] Murat Kocaoglu, Karthikeyan Shanmugam, and Elias Bareinboim. Experimental design for learning causal graphs with latent variables. In *Advances in Neural Information Processing Systems*, pages 7018–7028, 2017.

[KSCP18] Khimya Khetarpal, Shagun Sodhani, Sarath Chandar, and Doina Precup. Environments for lifelong reinforcement learning, 2018.

[KSGG16] Tejas D Kulkarni, Ardavan Saeedi, Simanta Gautam, and Samuel J Gershman. Deep successor reinforcement learning. *arXiv preprint arXiv:1606.02396*, 2016.

[KT00] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 2000.

[LL18] Lucas Lehnert and Michael L Littman. Transfer with model features in reinforcement learning. *arXiv preprint arXiv:1807.01736*, 2018.

[LLSL20] Kimin Lee, Kibok Lee, Jinwoo Shin, and Honglak Lee. Network randomization: A simple technique for generalization in deep reinforcement learning, 2020.

[MDR+20] Bhairav Mehta, Tristan Deleu, Sharath Chandra Raparthy, Chris J. Pal, and Liam Paull. Curriculum in gradient-based meta-reinforcement learning, 2020.

[MKNT21] Bogdan Mazoure, Ilya Kostrikov, Ofir Nachum, and Jonathan Tompson. Improving zero-shot generalization in offline reinforcement learning using generalized similarity functions, 2021.

[MKS$^+$13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[MPRP16] Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning, 2016.

[MWB18] Chen Ma, Junfeng Wen, and Yoshua Bengio. Universal successor representations for transfer reinforcement learning. *arXiv preprint arXiv:1804.03758*, 2018.

[NAS18] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms, 2018.

[NIA$^+$18] Evgenii Nikishin, Pavel Izmailov, Ben Athiwaratkun, Dmitrii Podoprikhin, T. Garipov, Pavel Shvechikov, Dmitry P. Vetrov, and Andrew Gordon Wilson. Improving stability in deep reinforcement learning with weight averaging. 2018.

[NL19] Akshay Narayan and Tze-Yun Leong. Effects of task similarity on policy transfer with selective exploration in reinforcement learning. In *AAMAS*, 2019.

[NLL17] Akshay Narayan, Zhuoru Li, and Tze-Yun Leong. Seapot-rl: Selective exploration algorithm for policy transfer in rl. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017.

[NPL$^+$20] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey, 2020.

[ORW16] Ian Osband, Benjamin Van Roy, and Zheng Wen. Generalization and exploration via randomized value functions, 2016.

[OSLK17] Junhyuk Oh, Satinder Singh, Honglak Lee, and Pushmeet Kohli. Zero-shot task generalization with multi-task deep reinforcement learning. *arXiv preprint arXiv:1706.05064*, 2017.

[PBS15] Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv preprint arXiv:1511.06342*, 2015.

[PCW+20] Rémy Portelas, Cédric Colas, Lilian Weng, Katja Hofmann, and Pierre-Yves Oudeyer. Automatic curriculum learning for deep rl: A short survey, 2020.

[PGK+18] Charles Packer, Katelyn Gao, Jernej Kos, Philipp Krähenbühl, Vladlen Koltun, and Dawn Song. Assessing generalization in deep reinforcement learning. *arXiv preprint arXiv:1810.12282*, 2018.

[PP+99] Theodore J Perkins, Doina Precup, et al. Using options for knowledge transfer in reinforcement learning. *University of Massachusetts, Amherst, MA, USA, Tech. Rep*, 1999.

[PRHO21] Rémy Portelas, Clément Romac, Katja Hofmann, and Pierre-Yves Oudeyer. Meta automatic curriculum learning, 2021.

[PZG17] Ben Poole, Friedemann Zenke, and Surya Ganguli. Intelligent synapses for multi-task and transfer learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017.

[RCG+15] Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

[Rud17] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[SGWA16] Jinhua Song, Yang Gao, Hao Wang, and Bo An. Measuring the distance between finite markov decision processes. In *AAMAS*, 2016.

[SJT$^+$20] Xingyou Song, Yiding Jiang, Stephen Tu, Yilun Du, and Behnam Neyshabur. Observational overfitting in reinforcement learning. In *International Conference on Learning Representations*, 2020.

[SLA$^+$15] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[SLK$^+$17] Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Rob Fergus. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407*, 2017.

[SOL19] Sungryull Sohn, Junhyuk Oh, and Honglak Lee. Hierarchical reinforcement learning for zero-shot generalization with subtask dependencies, 2019.

[SPS99] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

[SWD$^+$17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[SWT18] Doron Sobol, Lior Wolf, and Yaniv Taigman. Visual analogies between atari games for studying transfer learning in rl, 2018.

[SXS17] Tianmin Shu, Caiming Xiong, and Richard Socher. Hierarchical and interpretable skill acquisition in multi-task reinforcement learning. *arXiv preprint arXiv:1712.07294*, 2017.

[SYH$^+$19] Bradly C. Stadie, Ge Yang, Rein Houthooft, Xi Chen, Yan Duan, Yuhuai Wu, Pieter Abbeel, and Ilya Sutskever. Some considerations on learning to explore via meta-reinforcement learning, 2019.

[SYZ$^+$18] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy 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.

[SZC$^+$20] Trevor Standley, Amir R. Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning?, 2020.

[TBC$^+$17] Yee Teh, Victor Bapst, Wojciech M Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 4496–4506, 2017.

[TET12] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, Oct 2012.

[TFR$^+$17] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 23–30. IEEE, 2017.

[TP98] Sebastian Thrun and Lorien Y. Pratt. Learning to learn: Introduction and overview. In *Learning to Learn*, 1998.

[TS09] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(Jul):1633–1685, 2009.

[TS11] Matthew Taylor and Peter Stone. An introduction to inter-task transfer for reinforcement learning. *AI Magazine*, 32, 03 2011.

[TSL07] Matthew Taylor, Peter Stone, and Yaxin Liu. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8:2125–2167, 09 2007.

[Van18] Joaquin Vanschoren. Meta-learning: A survey, 2018.

[VGF21] Álvaro Visuś, Javier Garciá, and Fernando Fernańdez. A taxonomy of similarity metrics for markov decision processes, 2021.

[VPL20] Guillermo Valle-Pérez and Ard A. Louis. Generalization bounds for deep learning, 2020.

[VVM20] Nelson Vithayathil Varghese and Qusay H. Mahmoud. A survey of multi-task deep reinforcement learning. *Electronics*, 9(9), 2020.

[WKNT$^+$16] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

[WKSF20] Kaixin Wang, Bingyi Kang, Jie Shao, and Jiashi Feng. Improving generalization in reinforcement learning with mixture regularization, 2020.

[WLCS19] Rui Wang, Joel Lehman, Jeff Clune, and Kenneth O. Stanley. Paired open-ended trailblazer (poet): Endlessly generating increasingly complex and diverse learning environments and their solutions, 2019.

[WLT$^+$18] Sam Witty, Jun Ki Lee, Emma Tosch, Akanksha Atrey, Michael Littman, and David Jensen. Measuring and characterizing generalization in deep reinforcement learning. *arXiv preprint arXiv:1812.02868*, 2018.

[WLW18] Lisheng Wu, Minne Li, and Jun Wang. Learning shared dynamics with meta-world models, 2018.

[WWGT18] Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018.

[YKG$^+$20] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning, 2020.

[YQH$^+$19] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019.

[ZS18] Shangtong Zhang and Richard S. Sutton. A deeper look at experience replay, 2018.

[ZSI$^+$19] Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning, 2019.

[ZSP18] Amy Zhang, Harsh Satija, and Joelle Pineau. Decoupling dynamics and reward for transfer learning. *arXiv preprint arXiv:1804.10689*, 2018.

[ZVMB18] Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning, 2018.

[ZY17] Yu Zhang and Qiang Yang. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*, 2017.

[ZY20] Yi Zhou and Fenglei Yang. Latent structure matching for knowledge transfer in reinforcement learning. *Future Internet*, 12(2), 2020.

Appendix A

Environment Details

In this chapter we provide details of the environments used in this thesis in both Chapter 4 and 5. This primarily focuses on both the Meta-World [YQH+19] benchmarking environment and the classical Pendulum task (adapted from the OpenAI implementation [BCP+16]).

A.1 Meta-World

The Meta-World [YQH+19] environment is a simulated benchmark for RL in both MTL and meta-learning consisting of 50 diverse robotic manipulation tasks. This was developed in order to provide a set of benchmark environments that would offer a diverse task distribution that would allow for the testing of the generalisability of RL agents.

This environment is a wrapper around many of the MuJoCo [TET12] robotic manipulation tasks. It provides several different task splits in the form of both the meta-learning setting (ML1, ML10, ML45), providing a task split of train/test, and MTL task pools (MT1, MT10, MT50) providing task sampling for all tasks within the task pool rather than a train/test split. Across all the tasks within the benchmark, the action (a continuous action space) and observation spaces are the same but diverse, however they each contain different rewards functions within the same space[1], whilst each benchmark environment provides adapted goal spaces (the location of the goal is moved, and therefore the reward for each trajectory is adapted) for each of the tasks it contains.

[1]The rewards are scaled as discussed in code repository issue: https://github.com/rlworkgroup/metaworld/pull/312

In Table A.1, we provide the details of the task distributions for both the MTL and meta-learning setting.

Name	**Environment type**	**Number of Tasks**	
		Train	Test
ML10	Meta-learning	10	5
ML45	Meta-learning	45	5
MT10	Multi-task learning	10	n/a
MT50	Multi-task learning	50	n/a

Table A.1: Details of the provided task distributions provided by the Meta-world environment [YQH+19].

For each of the benchmark environments listed in Table A.1, we provide a list of all tasks included in each of the Meta-World task distributions in Table A.2. We also define whether they are a within the train or test task distribution—test tasks are only stated for the tasks belonging to the meta-learning settings, ML10 and ML45, whilst all tasks in the MTL setting are defined as train tasks.

Name	**ML10**	**ML45**	**MT10**	**MT50**
pick-place-v2	■	■	■	■
drawer-open-v2	♦	■	■	■
handle-press-side-v2		■		■
shelf-place-v2	♦	■		■
disassemble-v2		■		■
faucet-close-v2		■		■
door-lock-v2		♦		■
push-back-v2		■		■
coffee-button-v2		■		■
window-open-v2	■	■	■	■
bin-picking-v2		♦		■
door-close-v2	♦	■		■
faucet-open-v2		■		■
hammer-v2		■		■
push-v2	■	■	■	■
sweep-into-v2	♦	■		■
lever-pull-v2	♦	■		■

assembly-v2		■		■
basketball-v2	■	■		■
pick-out-of-hole-v2		■		■
door-open-v2	■	■	■	■
button-press-topdown-wall-v2		■		■
handle-pull-v2		■		■
soccer-v2		■		■
sweep-v2	■	■		■
coffee-pull-v2		■		■
drawer-close-v2	■	■	■	■
door-unlock-v2		♦		■
dial-turn-v2		■		■
plate-slide-side-v2		■		■
button-press-topdown-v2	■	■	■	■
push-wall-v2		■		■
reach-wall-v2		■		■
plate-slide-back-side-v2		■		■
stick-pull-v2		■		■
handle-press-v2		■		■
plate-slide-back-v2		■		■
box-close-v2		♦		■
button-press-wall-v2		■		■
hand-insert-v2		♦		■
reach-v2	■	■	■	■
peg-insert-side-v2	■	■	■	■
coffee-push-v2		■		■
plate-slide-v2		■		■
window-close-v2		■	■	■
peg-unplug-side-v2		■		■
pick-place-wall-v2		■		■
stick-push-v2		■		■
handle-pull-side-v2		■		■
button-press-v2		■		■

Table A.2: Details of the v2 tasks in the Meta-World environment [YQH+19]. Each task details the task distribution it belongs to (ML10, ML45, MT10 or MT50) and whether it is a task used within the train (■) or test (⧫) task distribution.

A.2 Pendulum

The pendulum task is an adaptation of the classical pendulum task provided by the OpenAI gym environment [BCP+16]. This is task consisting of a continuous action space task, which controls the torque of the agent, where the agent must swing the pendulum, which has started in a random position, so it stays in an upright position. This is modified to allow for adaptations to be made to the underlying task MDP by providing the parameterisation of both the length, l, and mass, m, of the pendulum. We detail the parameters and their associated recommended domain in Table A.3—this domain was found through experimentation as a bounds on the complexity of the solution with regard to successful performance.

Parameters	**Bounds**
mass (m)	[0, 10]
length (l)	[0, 10]

Table A.3: Parameters of the adaptive Pendulum environment.

Appendix B

Supplementary Material for Chapter 4

This chapter provides supplementary material for Chapter 4. This includes details of the per environment task subsets used in the experiments, along with the hyperparameters used for both the policy distillation and MAML methods on both the Pendulum and Meta-World environments.

B.1 Environment Task Subsets

Here we provide the details of the task subsets provided in the experiments conducted within Section 4.4 for both the policy distillation and MAML methods on the Pendulum and Meta-World environments. For the Pendulum environment we provide both the task subsets and the associated measure of task similarity. For the Meta-World environment we provide a breakdown of the task similarities for the overall task distribution that the task groupings were sampled from.

B.1.1 Pendulum

We define the pendulum task, an adaptation of the classical pendulum task from the OpenAI gym environment [BCP$^+$16], as a task where the agent must swing the pendulum, which has started in a random position, so it stays in an upright position—further details are provided in Appendix A.2. Its MDP consists of a continuous action space which defines the amount of torque the applies to the pendulum. In the experiments detailed in Section 4.4, we define the task's representation by two parameters: the length, l of the pendulum and it's mass m. The task subsets for both of the MTL and meta-learning settings are provided in the following sections.

B.1.1.1 Multi-task Learning

In the MTL setting experiments for the pendulum task, as detailed in Section 4.5.1.1, using the task representation as defined by the length and mass parameter-space, we sample 6 locations uniformly, for each task grouping, from bounds centred on $l = 1$ and $m = 1$ evenly increasing to cover the entire domain of $l \to [0, 2]$ and $m \to [0, 2]$. These domain bounds were chosen as a region which facilitates task completion, however still provides an effective analysis on how task similarity effects performance. The resultant task groupings, where each point in the figure represents a task and each subfigure a different task grouping, are provided in Figure B.1. Each of these task groupings represent a different train set for use in a MTL setting, with the associated inter-task similarity—we define the inter-task distance as the mean pairwise euclidean distance between each task within the task grouping. The task group sizes allowed for representative performance and similarity results to be calculated.

B.1.1.2 Meta-Learning

In the meta-learning setting experiments on the pendulum task, as detailed in Section 4.5.1.2, using the task representation, as defined by the length and mass parameter-space, we sample 6 locations uniformly, for each train task grouping, from bounds dividing the entire domain of $l \to [0, 10]$ and $m \to [0, 10]$ up into segments in order to form train groupings with similar inter-task train distances, but varying train to test distances. These domain bounds were chosen as a region which facilitates task completion, however still analyses how task similarity will affect performance when train to test distance is altered. We then provide a constant test task grouping, consisting of 4 tasks, located near $l = 1$ and $m = 1$—this location facilitated simple generation of varying task similarities with regard to train task groupings. The resultant task groupings, where each point in the figure represents a task, either train or test, and each subfigure a different task group pairing, are provided in Figure B.2. Each of these task groupings represent a different train and test set, along with the associated inter-task similarity—we define here the inter-task distance as the mean pairwise euclidean distance

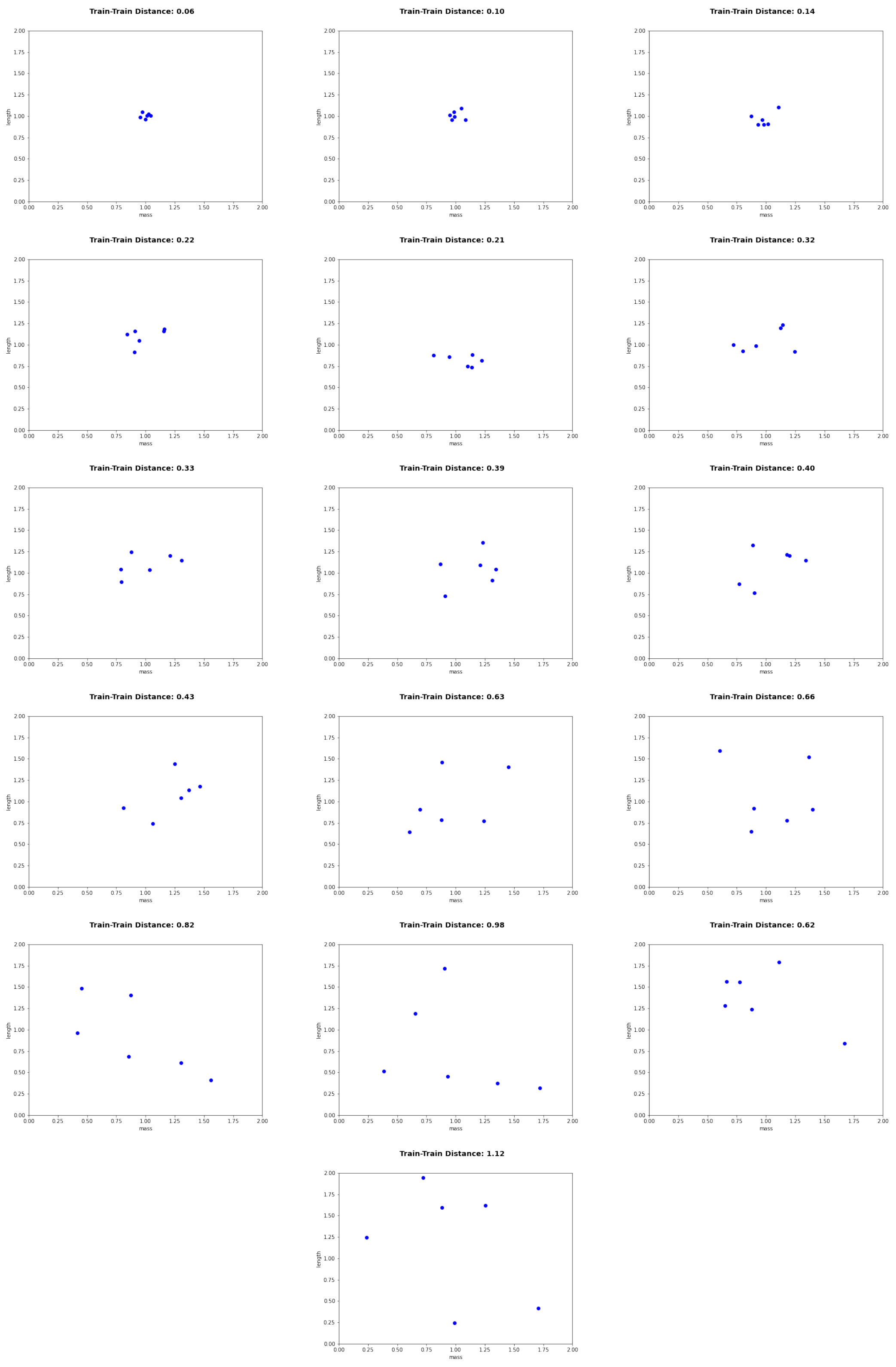


Figure B.1: Pendulum Multi-task Learning Task subsets

between each task within task grouping, with train to train distance being the distance between the tasks within the train task group, and train to test distance being the euclidean distance between the centroids of the train task grouping and test task grouping. The task group sizes allowed for representative performance and similarity results to be calculated.

B.1.2 Meta-World

The Meta-World environment, as further detailed in Appendix A.1, contains a diverse set of robotic manipulation tasks with a continuous action state that represents the agents torque in order to move the robotic arm around the state space with the aim of completing tasks. In the experiments in Section 4.4 we use subsets of the MT10, consisting of 10 train tasks, and the ML10 environment task distribution, which includes 10 train tasks and 5 test tasks. In this section we detail a breakdown of the task distances, defined by the cosine distance between task observations, from pre-computed randomly sampled actions, from 15 episodes of each of the tasks, taken in each of the tasks in the task distribution. Across all of the tasks we only use the mean values of the inter-task distances, however we provide here further details on the distances as a guide to understanding the noise inherent within the representation.

B.1.2.1 Multi-Task Learning

For the MTL setting experiments, used to analyse the generalisability of the policy distillation method across the observation space, as detailed in Section 4.5.2.1, we use the Meta-World task subsets taken from the overall MT10 task distribution. For this, the task distances are the mean task distance across the corresponding task subset. The mean inter-task distance across all tasks in the MT10 task distribution is 0.238, and the per-task inter-task distances are detailed in Figure B.3.

We see that in these observations there is a large amount of noise in the observation space of the MT10 train tasks when randomly sampled actions are taken, however certain tasks do have reduced noise—in particular the window-close-v2, window-open-v2 and drawer-close-v 2 task. This could indicate that for the observation space holds more validity in the representation of these tasks when compared to other tasks with more noise. Overall, we do see separation in

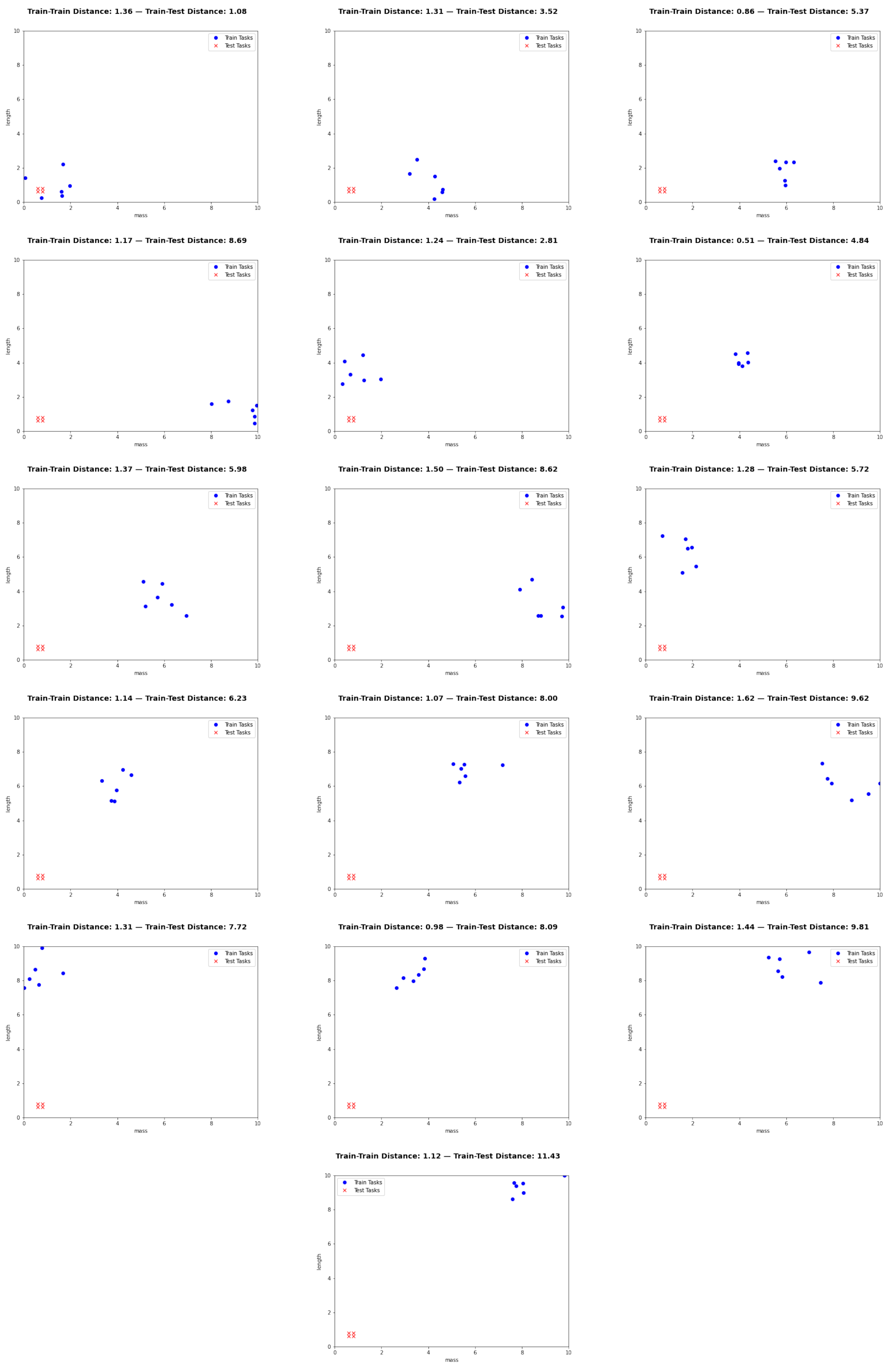


Figure B.2: Pendulum Meta-Learning Task subsets

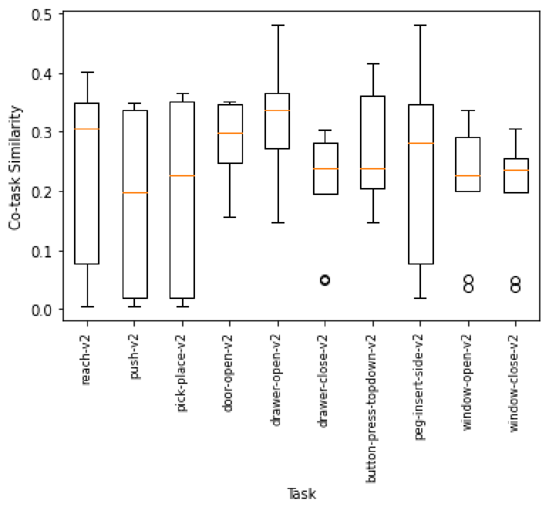


Figure B.3: Train to train task distances, defined by the cosine distance between pre-computed observations from randomly selected actions within the task, for each of the train tasks within the Meta-World MT10 [YQH$^+$19] task distribution. Error bars show standard deviation for 5 repetitions of sampling.

the task similarity metric, therefore this can be used as a signal for similarity.

B.1.2.2 Meta-Learning

For the meta-learning setting experiments, used to analyse the generalisability of the MAML method across the observation space, as detailed in Section 4.5.2.2, we use train task subsets and the whole of the test set, taken from the overall ML10 task distribution. From these, the mean train and test task distances across these task subsets are used for the task distances. The mean train to train inter-task distance for across all of the train tasks in ML10 is 0.274, whilst the mean train to test task inter-task distance over all of the train and test tasks in ML10 is 0.243. In Figure B.4, we provide a breakdown of the train to test task distances, which provide an indication of the similarity of each task within the train task set to the tasks within the test set.

We see from the per-task task distances calculated from the observations space, there is a large amount of noise across most of the ML10 train and test tasks, with regard to the train to test task distances in their observations. We do however, see much smaller noise in both the window-open-v2 and drawer-close-v2 tasks which could indicate that for these tasks this is a more valid representation than for other tasks with more noise. Overall, we do see separation in the task

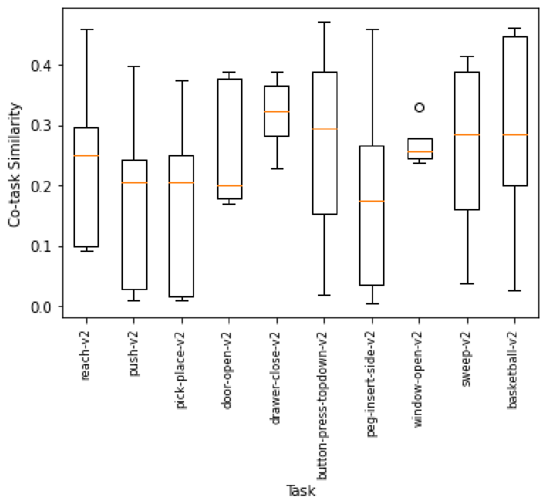


Figure B.4: Train to test task distances, defined by the cosine distance between pre-computed observations from randomly selected actions within the task, for each of the train tasks within the Meta-World ML10 [YQH+19] task distribution. Error bars show standard deviation for 5 repetitions of sampling.

similarity metric, therefore this can be used as a signal for similarity.

B.2 Experiment Hyperparameters

This section details the hyperparameters used in the experiments in Section 4.5 for both the policy distillation and MAML method. For both of these methods we detail the hyperparameters across both the Pendulum and Meta-World environments.

B.2.1 Policy Distillation

The following are the hyperparameters for policy distillation used in the MTL tasks on both the Pendulum and Meta-World MT10 environment task distributions.

B.2.2 MAML

The following are the hyperparameters for MAML [FAL17] used in the meta-learning experiments on both the Pendulum and Meta-World ML10 environment task distributions.

Hyperparameters	**Pendulum**	**Meta-World**
Hidden size	64	64
Number of layers	2	2
γ (discount factor)	0.995	0.995
λ (for Generalised Advantage Estimation)	0.97	0.97
Agent count	10	10
Max KL	1e-2	1e-2
Conjugate gradient damping	1e-2	1e-2
Number of iterations of conjugate gradient	10	10
Teacher batch size (agent)	1000	1000
Teacher sample batch size	10000	10000
Number of teacher episodes	100	100
Adam learning rate	1e-3	1e-3
Student batch size	1000	1000
Sample interval	10	10
Testing batch size	10000	
Number of student episodes	1000	500
Loss metric	KL	KL
Update algorithm	SGD	SGD

Table B.1: Policy Distillation hyperparameters.

Hyperparameter	**Pendulum**	**Meta-World**
γ (discount factor)	0.99	0.99
λ (for Generalised Advantage Estimation)	1.0	1.0
MAML (approximation)	First order	First order
Hidden size	64	64
Number of layers	2	2
Activation function	*tanh*	*tanh*
Inner-loop batch size (for each task)	20	20
Number of inner-loop steps	10	10
Inner-loop step size	0.05	
Number of outer-loop updates	100	50
Number of tasks in each batch	3	20
Max KL	1.0e-2	1.0e-2
Conjugate gradient damping	1e-5	1e-5
Number of iterations of conjugate gradient	10	10
Number of steps in line search	15	15
Line search backtracking ratio	0.8	0.8

Table B.2: MAML hyperparameters.

Appendix C

Supplementary Material for Chapter 5

In this chapter we provide supplementary material for Chapter 5. This includes the hyperparameter values used in the experiments in Section 5.4 and Section 5.5.2, along with further results and associated analysis from these experiments.

C.1 Experiment Hyperparameters

The following section details the hyperparameters for the MAML implementation provided by [Del18] and the gradient-based task similarity metric extensions used in the experiments discussed in Chapter 5. In Table C.1 we provide a breakdown of the hyperparameters for the associated experiments, either the ablation study or the automatic curriculum learning approach, in Section 5.4 and 5.5.2 on both the ML10 and ML45 Meta-World task distributions. The hyperparameters used in the experiments in Section 5.5.2.3 looking at the generalisability of the automatic curriculum learning using gradient-based task similarity approach are the same as detailed in Appendix B for the Meta-World ML10 MAML experiments, using the task similarity specific hyperparameters as listed at the end of Table C.1.

Hyperparameters	**Ablation**	**Auto Curriculum**	
	ML10	ML10	ML45
γ (discount factor)	0.99	0.99	0.99
λ (Generalised Advantage Estimation)	1.0	1.0	1.0
MAML (First order approximation)	True	True	True
Hidden size	64	64	64
Number of layers	2	2	2
Activation function	*tanh*	*tanh*	*tanh*
Inner-loop batch size (for each task)	10	10	10
Number of inner-loop steps	10	10	10
Inner-loop step size	0.05	0.05	0.05
Number of outer-loop updates	150	150	150
Number of tasks in each batch	20	20	45
Max KL	1.0e-2	1.0e-2	1.0e-2
Conjugate gradient damping	1e-5	1e-5	1e-5
Number of conjugate gradient iters	10	10	10
Number of steps in line search	15	15	15
Line search backtracking ratio	0.8	0.8	0.8
Task similarity replay buffer horizon	1	3	3
Task similarity replay discount	1.	0.99	0.99
τ (task sampling temperature)	0.1	0.1	0.03
Task sampling epsilon	N/A	0.2	0.2
Task similarity adapted	False	True	True
Co-task similarity is positive	True	False	False

Table C.1: Hyperparameters for the MAML implementation [Del18] with gradient-based task similarity extension for both the ablation study in Section 5.4, and the automatic curriculum approach in Section 5.5—task similarity specific parameters are listed at the end.

C.2 Additional Results: Ablation Study

This section provides additional results from the ablation study in Section 5.4 which look at using a gradient-based task representation space to form a task similarity metric and the effect it has on forming a non-uniform task sampling distribution in MAML. The results in this section will focus on how the frequency of each task's sampling changes during the training phase when generated from either the gradient change term, with hyperparameter g, the co-task similarity, with hyperparameter c, or a combination of the terms. We provide a comparison of the approach against a uniform task sampling distribution, when $g = 0$ and $c = 0$, which is used as a baseline (this is labelled as MAML).

C.2.1 Gradient Change

In Figure C.1 we provide the details over how the task occurrences in sampling change over the course of the training phase when influenced by the gradient change term, and its associated hyperparameter, g.

We can see that there are specific tasks that are experiencing consistent separation in terms of a different sampling frequency throughout the entire training phase. In particular, when $g = 1$ both the button-press-topdown-v2 and peg-insert-side-v2 tasks are consistently sampled more than both the baseline and when $g = -1$. In comparison, when $g = 1$ both the reach-v2 and drawer-close-v2 tasks experience consistently higher sampling rates. For the rest of the tasks, all hyperparameter configurations of gradient change result in fairly similar sampling frequencies across the entire training phase. We also see that the tasks most sampled for when $g = 1$ are being sampled less than the baseline of $g = 0$ when $g = -1$, and visa versa (i.e. inverse sampling rates). This consistency of separation across particular tasks, and the inverse relationship shown, indicate that we are finding good task similarity separation using the gradient change term, and forming curricula that encourages particular tasks consistency during training.

C.2.2 Co-Task Similarity

In Figure C.2 we provide the details over how the task occurrences in task sampling change over the course of the training phase when influenced by the co-task similarity term, and its associated hyperparameter, c.

The task frequencies across the entire training phase, when only the co-task similarity term is affecting the result, indicate a consistently adapting and noisy task similarity metric when $c = 1$ and $c = -1$. In particular, when $c = -1$, we have stages in the training phase where we are encountering a large standard deviation in the task sampling of certain tasks, for example in the door-open-v2, reach-v2 and drawer-close-v2 tasks, although it also happens, but to a smaller degree, in other tasks as well. We however only see this behaviour in the pick-place-v2 task when $c = 1$. Using the co-task similarity term it is evident that we don't have consistent separation of tasks samplings on any of the tasks with either co-task similarity configuration, thus a consistent curricula is not present. This effect will be partially due to the effect of the curriculum formed by the co-task similarity term, and it could be indication that the co-task similarity term is being negatively affected by the inherent noise in RL gradients.

C.2.3 Combination: Gradient Change and Co-Task Similarity

In Figure C.3 we provide the details over how the task occurrences in sampling change over the course of the training phase when influenced by the combination of the gradient change and co-task similarity terms, and their associated hyperparameters, g and c.

When we combine the gradient change and co-task similarity terms as in Equation 5.2, we can see that across all the hyperparameter configurations of g and c, there is a dramatic difference in the sampling of tasks in the ML10 task distribution. In particular, on the reach-v2, peg-insert-side-v2, drawer-close-v2 and door-open-v2 tasks we see large differences in the sampling frequencies across the training phase. This is in comparison to the rest of the tasks which show similar trajectories of task sampling across all hyperparameter configurations—there is however an small amount of variability in a couple of configurations. This indicates that these particular tasks are key to the curricula formed, and are therefore, with regards to the gradient change and co-task similarity terms, central to understanding the tasks that control the similarity of the entire task distribution. We see on these tasks that similar behaviour is seen when the gradient change term hyperparameter is set as the same value, in particular when $g = -1$. Overall, these results provide a strong indication that the curricula formed with the combination of the two similarity terms introduced in Section 5.2.1 provide a useful metric for generating curricula that are based on a task distributions task similarity.

C.3 Additional Results: Automatic Curriculum Learning via Gradient-based Task Similarity

This section provides additional results from the experiments from Section 5.5.2.3 looking at the automatic curriculum learning approach with a gradient-based task similarity metric, consisting of both the gradient change and co-task similarity terms on the inner-loop gradients of the MAML method. The results in this section focuses on how the frequency of each tasks' sampling changes during the training phase when we use Equation 5.8 to form an adaptive curriculum. We provide a comparison of the approach against a uniform task sampling distribution which is used as a baseline (this is labelled as MAML).

C.3.1 Meta-World ML10

In Figure C.4, we provide the details over how the task occurrences in sampling change over the course of the training phase when the task sampling distribution uses weights generated from Equation 5.8 on the ML10 Meta-World task distribution.

The task sampling rates that we see when we apply the gradient-based automatic curriculum learning approach are very similar for all configurations and the baseline across the entire training phase. What we do however see is short periods where certain tasks are sampled differently from the uniform task sampling distribution, in particular when we are encouraging far tasks—we see on the drawer-close-v2 and push-v2 tasks short periods of increased sampling, whereas on the peg-insert-side-v2 task we see a reduction in sampling during the first half of training. This is in comparison with the task similarity (close) method which doesn't show any separation from the uniform task sampling distribution task frequencies, as shown by the baseline, apart from for a short period on the button-press-topdown-v2 and pick-place-v2 tasks. These results show that on this task distribution we are not seeing much consistent separation in sampling trajectories, however the small perturbations evident across a large amount of environment steps could provide an indication that the method is forming curricula that can adapt over the course of the training phase, depending on the relevance of the task to the current agent's learning requirements.

C.3.2 Meta-World ML45

In Figure C.5, we provide the details over how the task occurrences in sampling change over the course of the training phase when the task sampling distribution uses weights generated from Equation 5.8 on the ML45 Meta-World task distribution.

On the ML45 task distribution, the overall task sampling rates across all of the tasks shows that both close and far configurations of Equation 5.8 closely follow the sampling rates of the uniform task sampling distribution. However, we see short periods of divergence across many of the tasks at different stages of the training phase. These differences in task occurrences are likely to be dampened in comparison to results on smaller task distribution sizes, such as ML10, due to the agent requiring to gather relevant information across a larger number of tasks. These results provide evidence that we are seeing an adaptive curriculum rather than the consistent separation of task sampling that was evident within the method investigated in the ablation study. The tasks that we see particular periods of diverse task sampling across the configurations include the drawer-open-v2, door-open-v2, peg-unplug-side-v2 and push-wall-v2 tasks. In these tasks we see at least two periods where we see separation in the task sampling occurrences. The tasks that we see the most separation in signify that these tasks are those that are seen by the similarity terms, gradient change and co-task similarity, to be most influential in the changing of curricula. We therefore state that these are tasks that, due to the gradient-based task similarity being innately linked to the agents' learning process, are pivotal to effecting the overall performance of the agent on this environment's task distribution.

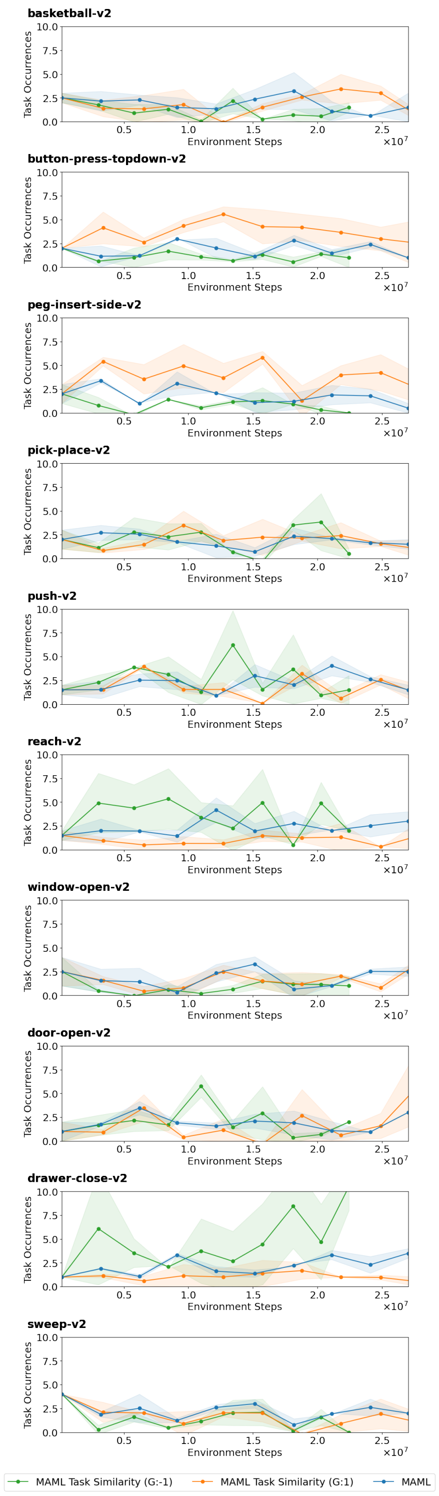


Figure C.1: Task occurrences during training on the Meta-world ML10 environment [YQH+19], looking at different values of g, the hyperparameter that controls the influence of gradient change on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

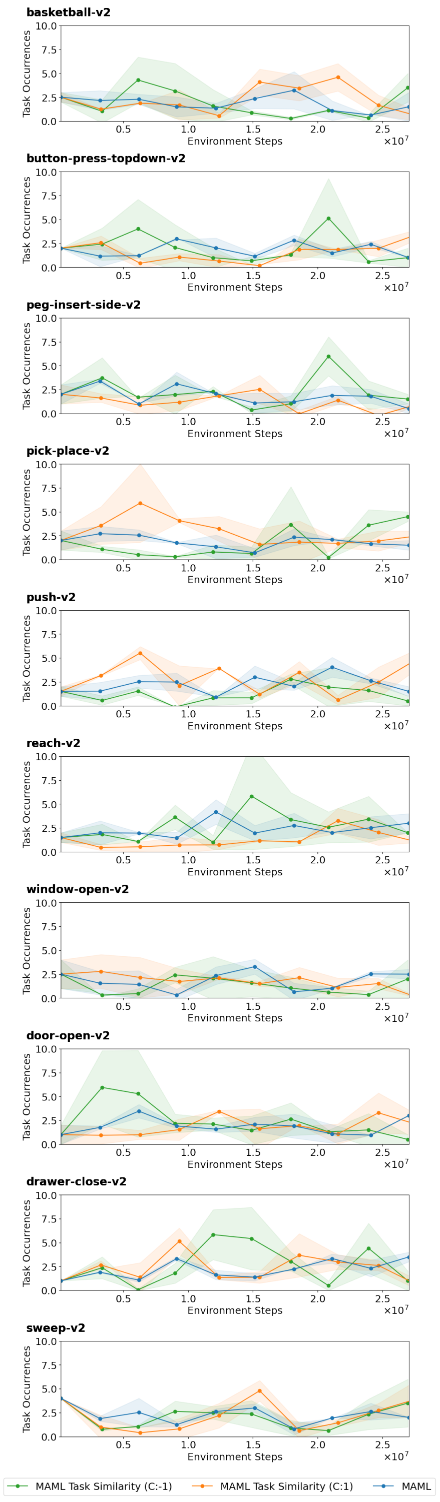


Figure C.2: Task occurrences during training on the Meta-world ML10 environment [YQH+19], looking at different values of c, the hyperparameter that controls the influence of co-task similarity on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

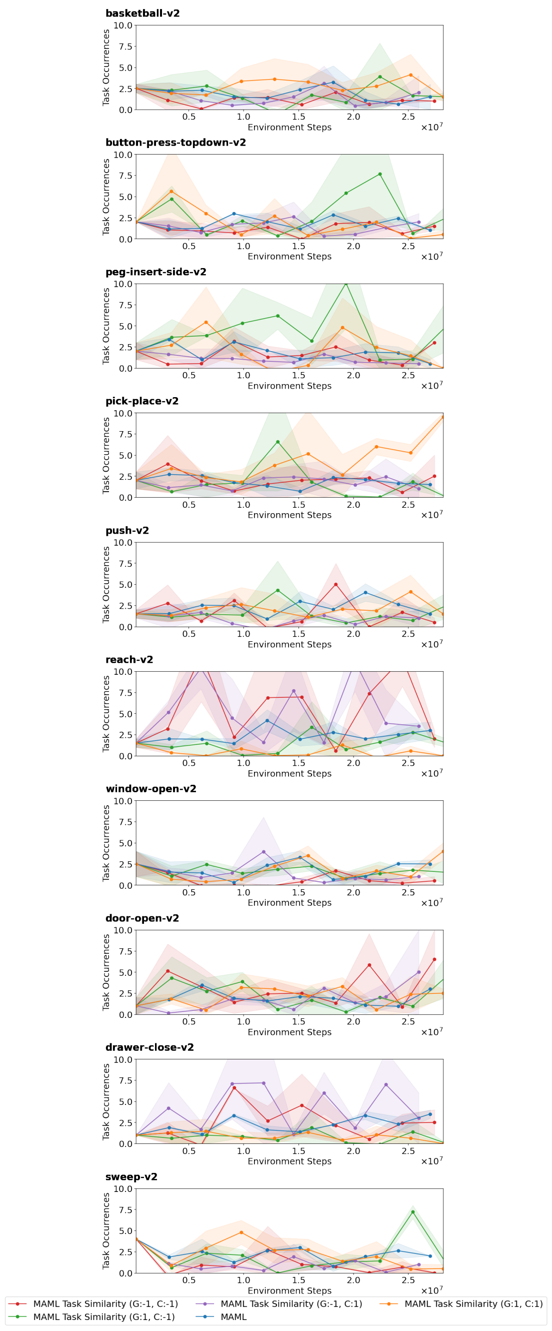


Figure C.3: Task occurrences during training on the Meta-world ML10 environment [YQH+19], looking at different values of g and c, the hyperparameters that control the influence of gradient change and co-task similarity, respectively, on the overall task sampling weights. Error bars show standard deviation for 3 repetitions of each configuration.

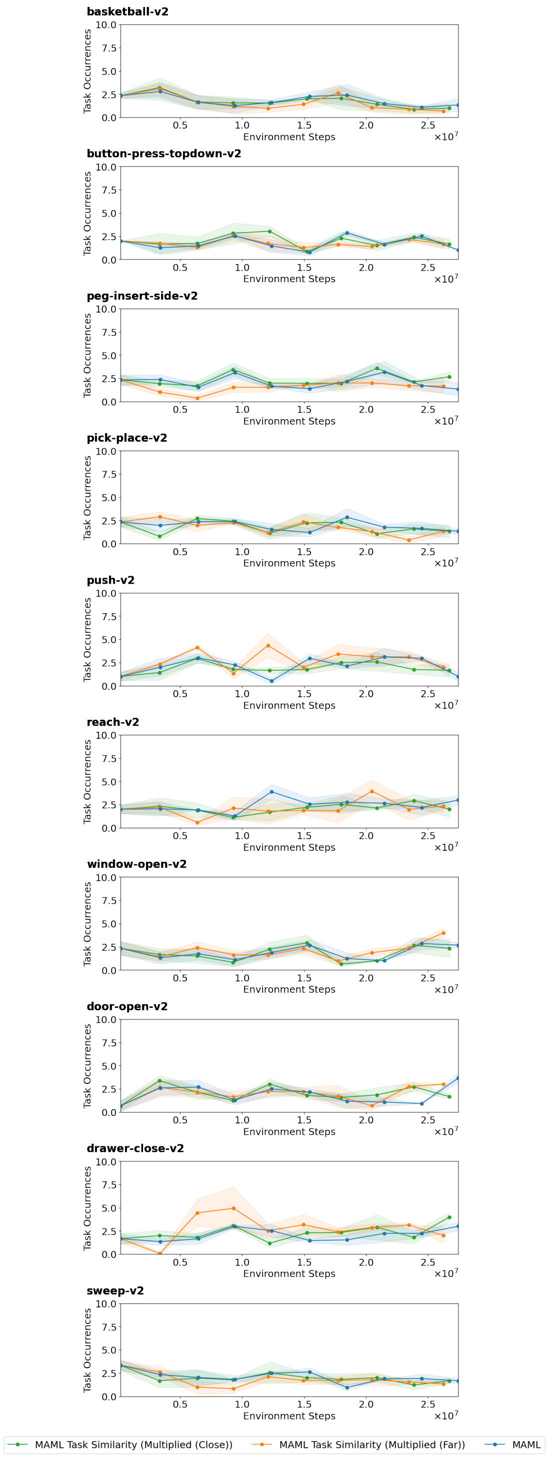


Figure C.4: Task occurrences during training on the Meta-world ML10 environment [YQH$^+$19] when using automatic curriculum learning based on gradient-based task similarity metrics in MAML. Error bars show standard deviation for 3 repetitions of each method.

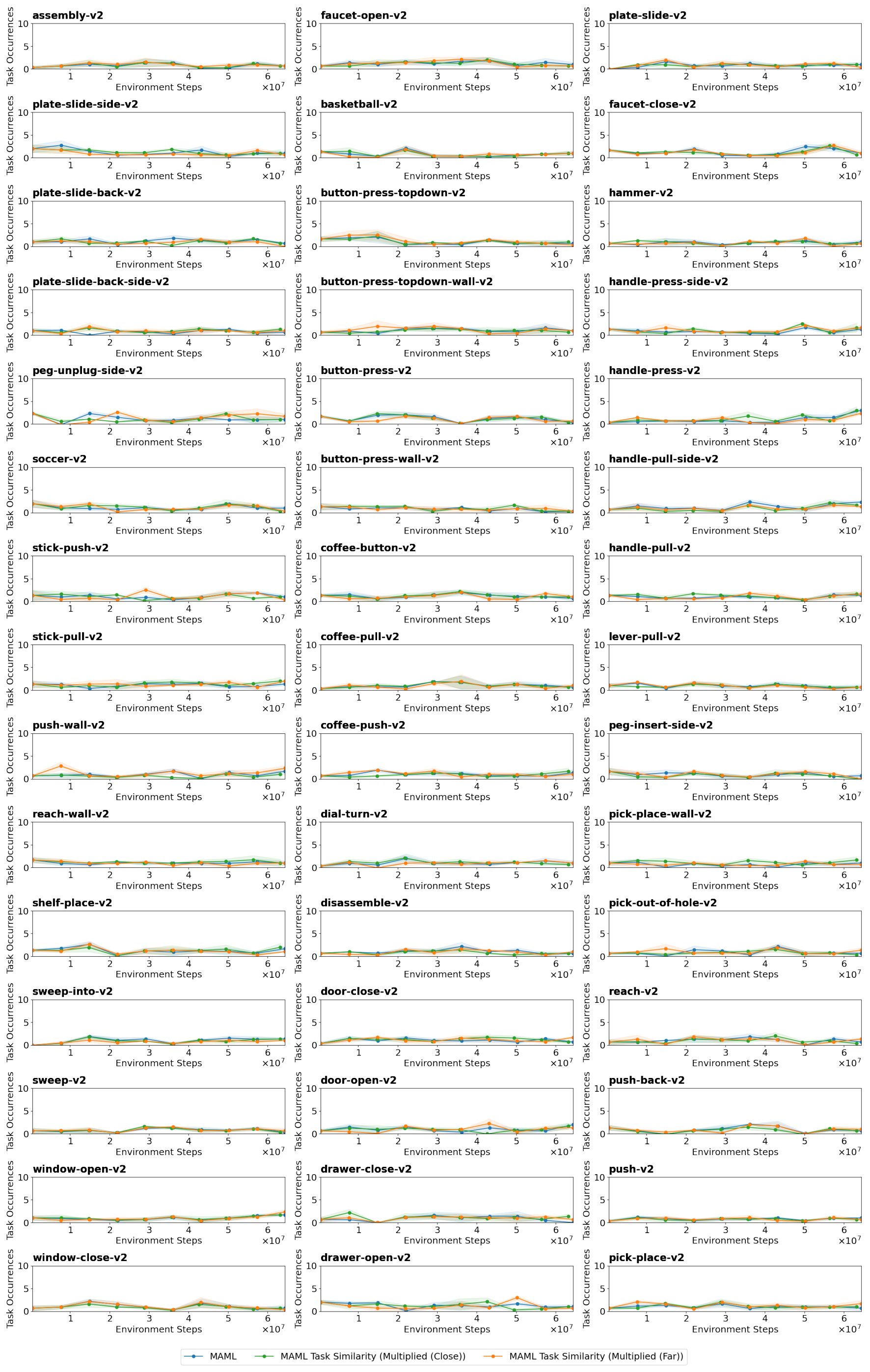


Figure C.5: Task occurrences during training on the Meta-world ML45 environment [YQH+19] when using automatic curriculum learning based on gradient-based task similarity metrics in MAML. Error bars show standard deviation for 3 repetitions of each method.