The Timeline of Meta-Reinforcement Learning - From the beginnings to the Adaptive Agent

Anonymous authors
Paper under double-blind review

Abstract

Humans are highly effective at utilizing prior knowledge to adapt to novel tasks, a capability standard machine learning models struggle to replicate due to their reliance on task-specific training. Meta-learning or 'learning to learn' overcomes this limitation by allowing models to acquire transferable knowledge from various tasks, enabling rapid adaptation to new challenges with minimal data. This survey presents a clear mathematical paradigm of meta-learning together with a formalization of common performance measures, distinguishes it from transfer-learning and multi-task learning, and utilizes it to derive the meta-reinforcement learning paradigm. A timeline of landmark meta-reinforcement learning developments from the earliest successes MAML and RL^2 to the Adaptive Agent is provided along with the corresponding paradigms and training schemes. This way, this work offers a comprehensive foundation for understanding meta-learning and meta-reinforcement learning, before giving an outlook on the latest developments and the connection of meta-learning to the path towards general intelligence.

1 Introduction

Humans possess a remarkable ability to tackle new tasks by leveraging their previous knowledge and experience. They can transfer skills and insights from one domain to another and refine them with very little experience, allowing them to adapt rapidly to novel situations. In contrast, standard machine learning models are typically designed to perform well on a single task by training on extensive amounts of task-specific data. Because they are optimized for peak performance within the domain of the trained task, they often excel in their predefined domains but struggle to generalize to new, unseen tasks. Meta-learning aims to overcome this limitation by "learning how to learn". Instead of focusing on one particular task, meta-learning models seek to extract higher-level knowledge and strategies from a variety of distinct yet related tasks, enabling them to adapt quickly to new challenges with minimal additional data by fine-tuning their parameters accordingly. In this way, the meta-learning paradigm aims to mimic human learning, where prior experiences inform the approach to new situations.

1.1 Evolution of Meta-Learning

Historically, the conceptual groundwork for self-adapting artificial intelligence was laid by Schmidhuber (1987), who introduced a paradigm for models that can adapt and generalize across tasks more effectively. However, meta-learning gained traction in the field of artificial intelligence just within the recent years, as the introduction of deep neural networks and the increasing availability of large data sets led to more powerful (standard) learners in several domains that, nevertheless, lacked the capability of human-like adaption to unseen situations. This substantial increase of task-specific learning capabilities on the one hand, and the lack of generalization throughout different tasks on the other hand, motivated the first meta-learning algorithms. Introducing "Model-agnostic Meta-Learning" (MAML), Finn et al. (2017a) founded the class of gradient-based meta-learners by abstracting the mere concept of gradient-based learning to a meta-level, where model parameters are aimed to be placed a priori in such a way that task-specific gradient-based learning has the most impact (see section 4.1). Almost at the same time, Duan et al. (2016) introduced

RL^2, the first memory-based meta-learner (see section 4.2) that implicitly incorporates previously gained knowledge over tasks by the memory structure of its underlying Recurrent neural network (RNN). These paradigm shifts to leveraging prior knowledge and experience for generalizing to unseen - but similar - tasks improved sample-efficiency and robustness (besides generalization) and were, thus, fundamental for developing artificial intelligence for real-life problems where requirements change over time.

Since then, various meta-learning algorithms have been developed in few-shot image classification He et al. (2023), Gharoun et al. (2024), Li et al. (2021), neural architecture search Elsken et al. (2019), Hospedales et al. (2022), Ren et al. (2021), Pereira (2024), neural language processing Yin (2020), Lee et al. (2022), Lee et al. (2021), and Reinforcement Learning (see section 2.2), so that the potential field of applications is almost as vast as the number of real world scenarios. From robotics , where different approaches aim to learn novel strategies from only a few demonstrations Finn et al. (2017b), or experiences Johannsmeier et al. (2019), over health care Rafiei et al. (2024), where patient or disease specific data is often sparse Tan et al. (2022) or diseases must be classified from only a few radio-logical images Maicas et al. (2018), up to adaptive control McClement et al. (2022), Duanyai et al. (2024), where system parameterizations change over time, many fields have been studied. The latter includes applications in energy forecasting Cui et al. (2016), Boutahir et al. (2024) and energy grid control Zhao et al. (2023), as well as aircraft control under uncertain wind conditions O'Connell et al. (2022) and fault detection Ahmed et al. (2020). Even for the preparation of space missions Gaudet et al. (2020), meta learning is used to pre-train models that can rapidly adapt in real time to changing environmental conditions which were unknown prior to the mission.

Since the topics related to or associated with meta-learning are widely spread and often difficult to distinguish from meta-learning itself or each other, several works surveying meta-learning and related topics have been published that categorize different sub-classes of meta-learning Beck et al. (2023b), Vettoruzzo et al. (2024) or differentiate between the numerous related topics Vettoruzzo et al. (2024), Barcina-Blanco et al. (2024), Upadhyay (2023). They address open problems Beck et al. (2023b), list a huge variety of existing meta-learning methods Vettoruzzo et al. (2024) or review overlaps between meta-learning and the most similar related topics Multi Task Learning (MTL) and Transfer Learning (TL) Upadhyay (2023).

1.2 Problem statement

The existing literature lacks a clear and detailed mathematical introduction of the concepts underlying meta-learning. A particularly surprising example of this lack of formalism is the absence of well-defined performance measures within the literature. When reviewing different studies, it remains uncertain whether the authors utilize identical metrics, as these are only implicitly defined, what complicates the comparison of results across different works.

The lack of formalism is especially relevant in Meta-RL, where agents interact with different but somehow similar environments in order to develop effective strategies. Since the agents' actions have a direct effect on the data gathered from their environments, it is crucial to adapt the meta-learning paradigm to include Meta-RL-specific concepts such as exploration, exploitation, and cumulative reward. Although some works attempt to address this problem Beck et al. (2023b), they fall short of providing rigorous mathematical formulations or situating the Meta-RL paradigm within the broader context of meta-learning, hence lacking a clean mathematical derivation. This general lack of formalization presents significant challenges in achieving a clear understanding of the underlying principles and methodologies when delving deeper into the field. It, moreover, explains the significant confusion between meta-learning and closely related paradigms like Multi-Task Learning (MTL) or Transfer-Learning (TL) throughout the literature. But, distinguishing between these paradigms, along with clearly defining the associated notions and concepts, is essential for understanding the developments towards generalist agents throughout the most recent years. A timeline of landmark advancements is needed, that draws a clear, ordered picture of the development from the earliest breakthroughs and their modifications until the current state of the art foundation models, while providing clear, formal definitions of the corresponding paradigms and training schemes.

1.3 Contribution and Paper outline

In response to the points mentioned above, the contributions of this work are manifold: It

- presents a clean and detailed mathematical formalization of meta-learning and meta-RL respectively.
- well defines and discusses the most common performance measures in meta-learning.
- clearly defines the closeliest related paradigms MTL and TL and distinguishes them from meta-learning.
- presents a timeline of meta-RL landmarks together with the corresponding paradigms and training schemes, that follows the history of meta-learning developments from the earliest landmarks MAML Finn et al. (2017a) and RL^2 Duan et al. (2016) to the Adaptive Agent Team et al. (2023), the most recent state of the art meta-RL algorithm published by DeepMind.
- discusses open problems and possible future research and connects it with the development path towards general intelligence.

Providing clear terminology and explicitly distinguishing meta-RL landmarks from their modifications, this work not only gives the reader a broad overview over the different classes of meta-learning algorithms, but also lays the foundation for delving much deeper into the advanced literature in the field of meta-RL. However, this work focuses particularly on meta-RL and does hence not cover all meta-learning developments of the previous years in all different sub-fields like classification, computer vision or natural language processing. This would go by far beyond the scope of this work and, more importantly, misdraw the reader's attention to details that are not helpful for the conceptual introduction of meta-RL this work is aiming for. The same holds for the comparison between meta-learning and its various related topics. Although meta-learning is distinguished from its closest (and potentially overlapping) neighbors in section 3, fields like continual learning, self-supervised learning and active learning remain largely untouched. The curious reader is therefore referred to Vettoruzzo et al. (2024) or specific surveys instead.

Accordingly, this work distributes as follows: In section 2 meta-learning is first distinguished from standard machine learning (section 2.1) by providing and discussing detailed mathematical formulations of meta-learning paradigm, training scheme and common performance measures, before this comparison is transferred to standard RL and meta-RL respectively in section 2.2. Section 3 then draws a line between meta-learning and its most related topics TL and MTL thereby also shortly introducing foundation models as "meta-variants" of TL models. In section 4, the timeline of meta-RL developements is presented along with formalization of paradigms and training schemes of the most relevant landmark algorithms. It starts with the simpler family of gradient-based meta-learners (section 4.1), before memory-based (black-box) algorithms are presented in subsections 4.2, 4.3 and 4.4, however, only for meta-RL. Starting with RNNs as the simplest memory architectures in section 4.2 followed by transformer-based meta-RL and the Adaptive Agent in sections 4.3 and 4.4, this outline of memory-based learners follows itself a timeline from the simplest architecture up to the Adaptive Agent, a generalist agent combining several concepts from the subsections before with self-supervised learning techniques like distillation and Automated Curriculum Learning, both of which are also described along with ADA's training scheme. Finally, section 5 discusses open problems of meta-learning and shades some light on most recent and possible future developments, before section 6 closes this work.

2 Paradigm

This section formally introduces Meta-Learning and Meta-RL by comparing them to standard machine and Reinforcement Learning respectively. It, thereby, creates a consistent structure of terms and notions used throughout the rest of the entire work.

2.1 Meta-Learning Paradigm

In standard machine learning a learner f_θ with parameters θ is trained to solve a particular task T of the form [1]

$$T := (\mathcal{L}, \mathcal{X}, \mu, \mathbb{T}, h), \tag{1}$$

[1]This formalization of a task is inspired by Finn et al. (2017a), Upadhyay (2023), Rakelly et al. (2019) etc.

by minimizing the loss function $\mathcal{L} : \mathcal{X} \to \mathbb{R}$ on some training data X_{train} out of the observation space $\mathcal{X} \subseteq \mathbb{R}^d$. This can be formalized as the optimization problem

$$\text{Minimize} \quad \mathcal{L}_\theta \quad \text{wrt. } \theta, \tag{2}$$

where $\mathcal{L}_\theta$ denotes that observations $x \in \mathcal{X}$ are processed or collected via the function f_θ. The goal is to best approximate the (theoretical) optimal function f_T^* [2] best solving the task T on all subsets $X \in \mathcal{X}$.

The distribution $\mathbb{T} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ in equation 1 represents the transition from one observation to another, while $\mu : \mathcal{X} \to [0, \infty]$ defines the distribution over initial observations x and $h \in$
$mathbbN$ is the horizon or episode length of the task defining the length of a single shot of learning.

The parameters θ denote the degrees of freedom of the learner (or model) f directly defined by its architecture, and are generally assumed to be a multi-dimensional vector of real values. If θ corresponds to the weights of a deep neural network (NN), f_θ is called a deep learner, and one speaks of deep learning rather than standard machine learning.

In practice, the observation space $\mathcal{X}$ is separated into three disjoint spaces, a training, validation and testing space $\mathcal{X}_{\text{train}}$, $\mathcal{X}_{\text{val}}$ and $\mathcal{X}_{\text{test}}$, where $\mathcal{X}_{\text{train}}$ normally is the much larger space reserved for training. During training, a training set X_{train} gets sampled from the training space in each training episode. The validation space is used to evaluate the learner's performance throughout the training process. Since generalization is the goal of training, $\mathcal{X}_{\text{val}}$ must be different from the training space in order to evaluate how well the learner can perform on unseen, but similar data. The test space $\mathcal{X}_{\text{test}}$ gets used to measure the final performance of the model f_θ after the training ends.

The model weights θ remain unchanged during validation and testing. Both aim to evaluate the performance of the current model. But while the validation is still part of the overall training process and just functions as an evaluation of the current training progress, the testing is carried out only once after training to evaluate the final model. However, the loss function $\mathcal{L}$ as well as all other task-specific components like observation space $\mathcal{X}$ or transition $\mathbb{T}$ remain unchanged throughout the whole training and testing process. As a consequence, the model f_θ is tailored to particularly solve the task T, while it generally fails to generalize to out-of-domain tasks, i.e. to tasks with different loss, transition or observation space Upadhyay (2023).

[2]The optimal function f_T^* is not neccessarily unique, but can, without loss of generality, be assumed as one particular function as each optimal function is per definition as good as another one.

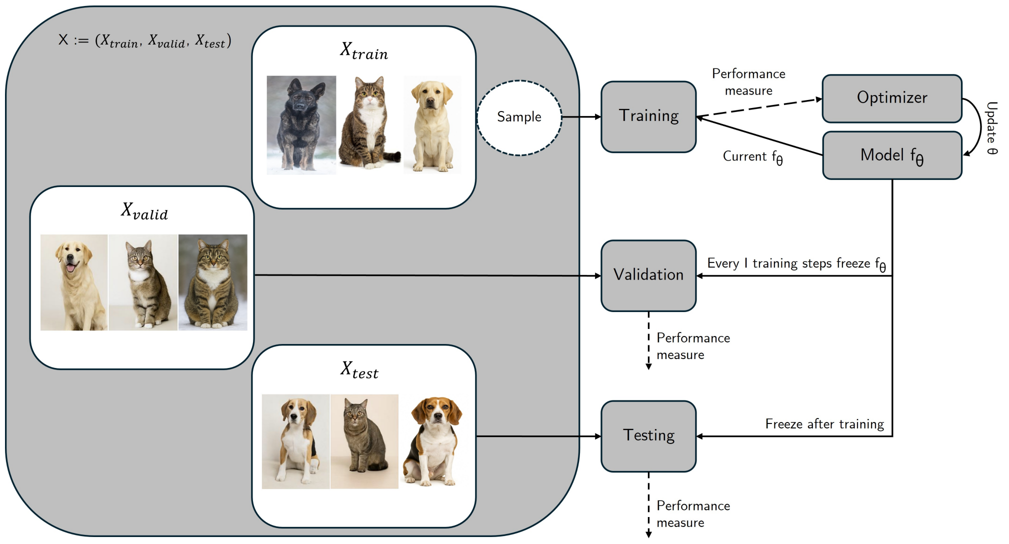


Figure 1: A general image classification task. Images are sampled w.r.t. the sampling strategy of the corresponding algorithm and classified by the current model f_θ. Performance measures, i.e. the corresponding loss is calculated until the optimizer updates the model weights θ according to the update rule of the corresponding algorithm. Every l training episodes, the whole validation set is classified and the respective losses are calculated in order to evaluate the performance of the current model. After training, the whole test set gets treated analogously.

Example 1 - Image Classification:

Let X be a labeled image set out of the observation space $\mathcal{X}$ of images showing cats or dogs and let $X_{\text{train}}, X_{\text{val}}$ and X_{test} be disjoint subsets of X with a split of 80% training data and 10% validation and test data. Then, the task T to classify images between 0 (dog) or 1 (cat), is formally defined by defining the components in equation 1, i.e.:

- The loss function $\mathcal{L}$ is, for example, the cross entropy loss. However, any suitable loss function can be selected.
- The initial distribution μ is the unit distribution over all images in X_{train} as all images are equally likely to be selected first.
- The transition $\mathcal{T}$ is a unit distribution conditioned on all images selected so far since all remaining images in X_{train} are always equally likely to be chosen next, while no image is selected twice in this example.
- Since each shot in image classification consists of only one image, the horizon h equals 1.

The optimal function f^* is the function perfectly classifying any image as cat or dog respectively, while the learner F_θ can be any suitable machine learning model, e.g. a convolutional neural network processing images to output whether they contain a cat or a dog. The test and validation sets X_{val} and X_{test} consist of images of cats and dogs not seen during training and the respective labels. This way, it can be validated whether the learner F_θ actually learned to distinguish between cats and dogs or if it only perfectly memorized the images in X_{train}. The latter is called overfitting. But, obviously, this way of testing the model f_θ does not provide any information if the same learner could also distinguish between cats and horses since the latter class does not exist in the task's observation space $\mathcal{X}$.

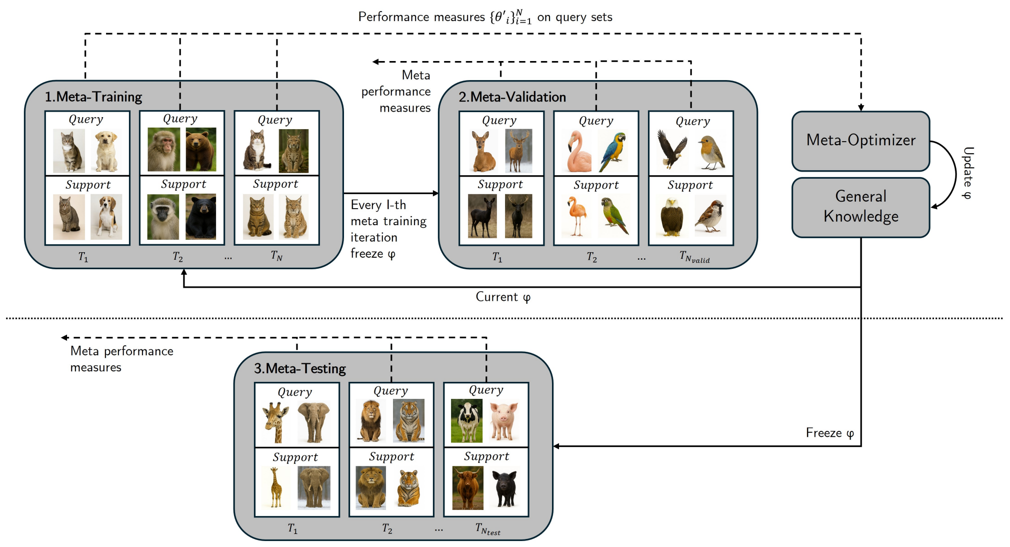


Figure 2: Meta-Learning of 2-way 1-shot animal classification tasks. During meta-training, the current meta-knowledge φ is fed into the task-specific model parameters θ_i in the way defined by the corresponding meta-learning algorithm. Task-specific fine-tuning on the support set results in updated parameters θ'_i whose performance is evaluated on the task-specific query set and fed into the meta-optimizer afterwards. The meta-optimizer uses all task-specific query set performances to update meta-knowledge φ according to the update rule of the corresponding meta-learning algorithm. Every l meta-episodes, N_{val} validation tasks are treated the same way for meta-validation, i.e. to evaluate meta-training progress. After meta-training, the same is done for the N_{test} tasks. Validation and test tasks were not seen during training, although getting sampled from the same task distribution $p(T)$.

Problem Setup

For many tasks the amount of training examples available is not sufficient for learning, and sometimes training a (standard) learner for a new but very similar task fully from scratch is too costly in terms of computation power, training time or money. For these reasons, Meta-Learning - in contrast to standard machine learning - aims to collect general knowledge over a family of similar tasks enabling best possible fine-tuning for each task within only a few examples i.e., adapting to a task within K shots rather than with a full training. Many frameworks even aim to optimize the learner's performance in the one-shot ($K = 1$) or zero-shot ($K = 0$) setting where the learner has only one shot (e.g. Finn et al. (2017a)) in the unseen task or even no shot at all.

Example 2 - Image classification Meta-Task:

Modifying the observation space $\mathcal{X}$ of example 1 to contain labeled images of several different animals, but only ten images per class, one obtains such a problem. With the same training-validation-test split as in example 1, this means eight images per class for training and only one for validation and testing respectively. This small amount of training data does not suffice for training a standard learner properly. However, on a meta-level distinguishing between all these different animals does not differ much from the task in example 1. Quite the opposite is the case since, on an abstract level, classifying dogs vs. cats requires similar high-level skills as classifying cats vs. horses: The model needs to identify the creature on the picture first, recognize shapes of noses, ears, or other body parts, examine the silhouette, etc. In other words, classifying cats vs. horses is just another task from the task distribution over different animal classification tasks of the form presented in example 1. In this way, the meta-task to classify any animal can be separated into several simplified tasks that share some common, high-level structure and belong to the same "meta-problem" of classifying animals.

Meta-Learning Paradigm

Formally, the meta-learning paradigm Thrun & Pratt (1998) consists of a distribution $p(T)$ over tasks of the form

$$T_i := (\mathcal{L}_i, \mathcal{X}, \mu_i, \mathbb{T}_i, h). \tag{3}$$

Each task of the form equation 3 (possibly) has its own loss function, transition dynamics or initial distribution, while the observation space $\mathcal{X}$ and the horizon h are generally assumed to equal between tasks from the same distribution p (see e.g. Finn et al. (2017a), Rakelly et al. (2019)), although this is no necessary condition and some works specifically focus on domain generalization Li et al. (2018), Triantafillou et al. (2020).

For the sake of this work both, the horizon h and the observation space $\mathcal{X}$ are, however, assumed to be shared within the family $\{T_I\}_{T_i \sim p(T)}$.

As highlighted in example 2, all tasks in $p(T)$ are assumed to share some common structure. If the model F_θ learned these certain skills (e.g. distinguishing shapes, identifying noses or ears, etc.), this would benefit the performance in any task T_i drawn from $p(T)$. At the same time, meta-learning aims to learn any task in $p(T)$ with only a few shots of task-specific training, i.e., the goal is to adapt fast to any task T_i. That means, there are two "components" of what must be learned - at least from a theoretical point of view:

- Common knowledge over all tasks T_i in $p(T)$ and
- Task-specific knowledge gained through few-shot fine-tuning.

This is why the meta-learning paradigm consists of two stages, the more abstract meta-task-level and the few-shot standard learning of a particular task $T_i \sim p(T)$. As a consequence, the training-validation-test split is as well two-staged. On meta-task-level, certain tasks get explicitly excluded from the meta-training task pool to function as validation throughout meta-training or as test tasks after meta-training respectively. The corresponding evaluation takes place on the stage of task-specific standard learning, which is why it is often referred to as inner learning. However, since every task T_i from the distribution $p(T)$ is assumed to be a few-shot learning task, a task-specific validation set X^i_{val} to evaluate the model during task-specific standard learning is not required for any task T_i in $p(T)$. This way, the training paradigm equation 2 of standard few-shot learning is mimicked within each task T_i while meta-training provides every such task-specific fine-tuning with a meaningful prior to enable fast adaption. The corresponding meta-training scheme is presented in the next subsection.

Although all tasks in $p(T)$ share the same observation space $\mathcal{X}$, each task T_i (possibly) has its own training and test sets X^i_{train} and X^i_{test}, as described in example 2. Moreover, all meta-learning frameworks presented in this work are designed to sample new training and test sets for each task in each iteration of meta-training (see e.g. Finn et al. (2017a), Rakelly et al. (2019) or figure 3).

Meta-Training

The concept of inner and outer learning is explicitly applied to all gradient-based meta-learning algorithms presented in section 4.1. However, even for the memory-based meta-learners presented in section 4.2 that do not explicitly divide learning into meta-learning and task-specific standard learning, the meta-training paradigm can be formalized as (implicitly) two-staged by introducing a meta-variable φ encoding common knowledge over all tasks seen so far. As highlighted in figure equation 3, in each meta-training iteration, this meta-variable is optimized wrt. the performance of the respective inner learning of each task T_i ($i = 1, \ldots, N$):

$$\text{Minimize} \quad \mathbb{E}_{T_i \sim p(T)} \mathcal{L}^{\text{meta}} \left(\theta^*_{T_i}(\varphi),\ \varphi,\ X^i_{\text{test}} \right) \quad \text{wrt. } \varphi \tag{4}$$

In other words, φ is optimized to be the best prior for fast task-specific fine-tuning, which is measured by the performance of the adapted task-specific parameters on the task-specific test set X^i_{test}.

The notation $\theta_i(\varphi)$ highlights the relationship between φ and the initial parameters of the inner learning, while $\mathcal{L}^{\text{meta}}$ denotes the meta loss function and $\theta^*_i(\cdot)$ the optimal task-specific parameters for Task T_i when starting inner learning with meta-parameter φ.

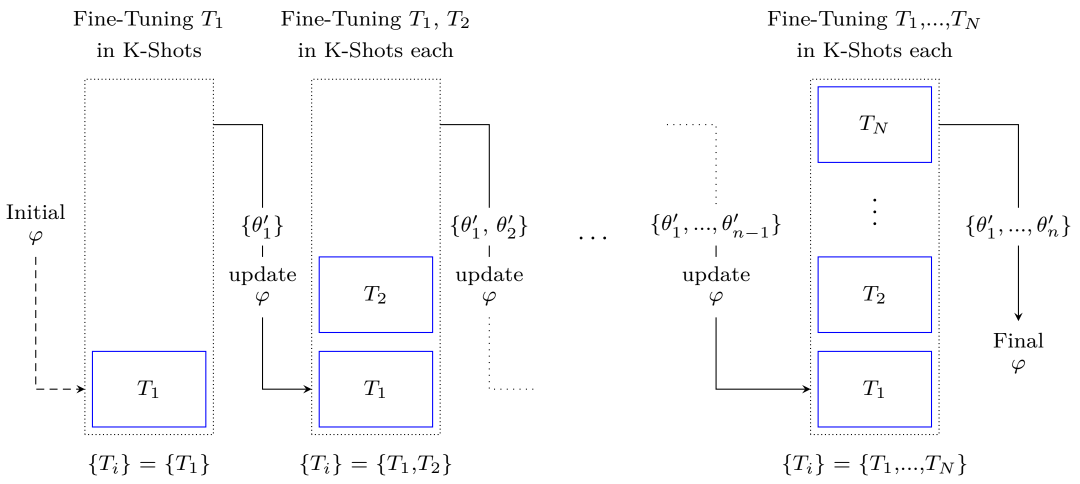

Fine-Tuning T_1 in K-Shots
Fine-Tuning T_1, T_2 in K-Shots each
Fine-Tuning T_1,...,T_N in K-Shots each
Initial φ
$\{\theta'_1\}$
update φ
$\{\theta'_1, \theta'_2\}$
update φ
...
$\{\theta'_1, ..., \theta'_{n-1}\}$
update φ
$\{\theta'_1, ..., \theta'_n\}$
Final φ
T_1
T_2
T_N
$\{T_i\} = \{T_1\}$
$\{T_i\} = \{T_1,T_2\}$
$\{T_i\} = \{T_1,...,T_N\}$

Figure 3: General Meta-Training - In each iteration a new task T_i is sampled from the family $p(T)$ and added to the set $\{T_i\}$ of training tasks. The meta-variable φ is used for fine-tuning each task individually in K shots, before the resulting parameters θ'_i of each task $T_i \in \{T_i\}$ are used to update φ. This process is repeated until N tasks are drawn from $p(T)$ in total.

However, the optimal task-specific parameters $\theta^*_i(\cdot)$ are rather a theoretical component of the formalization of meta-training taken from Upadhyay (2023) than a practical implementation. Inner learning aims to approximate the task-specific optimal parameters θ^*_i representing the optimal solution of the task T_i, and, thus, the resulting parameters must be denoted differently to distinguish them from the optimal ones. Throughout this work, the respective notation is $\theta'_i := \theta'_i(\varphi)$ for parameters trained in inner learning starting from $\theta_i(\varphi)$ i.e., initial parameters θ yielded from the prior φ.

Although the inner learning depends on the meta-learned prior φ, it fully corresponds to the training paradigm equation 2 of standard learning a particular task T_i. Consequentially, the inner optimization problem can be formalized as

$$\text{Minimize} \quad \mathcal{L}_i\left(\theta_i, \theta_i(\varphi), X^i_{\text{train}}\right) \quad \text{wrt. } \theta_i, \tag{5}$$

with $\theta_i(\varphi)$ denoting the starting point of training the parameters θ_i.

Besides some architectural choices, the choice of the meta-loss $\mathcal{L}_{\text{meta}}$ as well as of the inner learning determines the respective meta-learning framework. This holds for all algorithms presented in section 4, although the meta-loss is not explicitly given for the memory-based meta-learners in section 4.2. Moreover, the task sampling throughout meta-training is algorithm specific. Figure 3 shows a scheme, where tasks are iteratively drawn from $p(T)$ up to N tasks, but other algorithms might directly start with N tasks and re-sample them respectively. The numbers N, N_{val} and N_{test} of training, validation and testing tasks are itself hyperparameters of meta-training.

Meta-Testing

After training a meta-learner f_φ in the just described meta-training scheme, the resulting model must be evaluated on unseen test tasks. As already mentioned above, meta-testing rather aims to validate the learning process of adapting to a new task itself than the task-specific performance. The question answered here is how well the meta-trained φ boosts fine-tuning of θ_j to unseen tasks $\{T_j\}_{T_j \sim p(T)}$ with $t_j \neq T_i$ for all training tasks T_i and all j. Hence, the meta-variable φ must be fixed throughout the whole meta-testing process, while the task-specific parameters $\theta_j(\varphi)$ are adapted within K shots of inner learning for each task T_j. The corresponding θ'_j is then evaluated on the task-specific testing set X^j_{test} to yield the respective task-specific performance. This iterative procedure is shown in Figure 4, while the question of how to measure the overall meta-learning performance is answered in the next subsection.

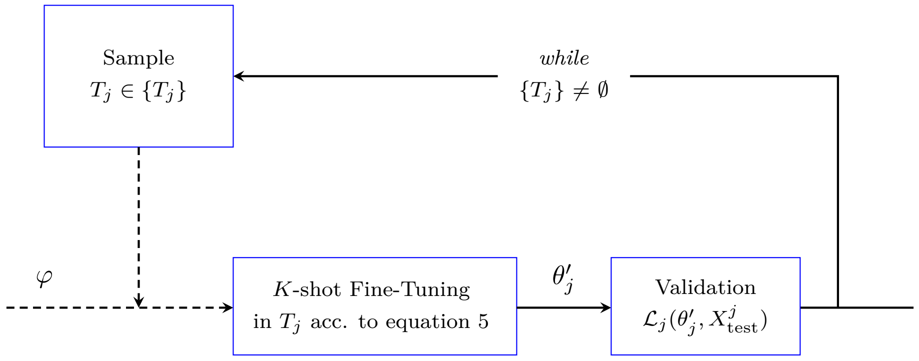


Figure 4: General Meta-Testing Paradigm - At the beginning of each iteration, a particular task T_j is drawn from the set of test tasks $\{T_j\}_{T_j \sim p(T)}$ without replacement. For each T_j the parameters $\theta_j(\varphi)$ are fine-tuned in K shots, before the resulting θ'_j get evaluated via the task-specific loss $\mathcal{L}_j$ to yield the performance.

In terms of example 2, meta-testing means to evaluate the meta-learnt knowledge, how to distinguish between different animals, on an animal combination not seen during meta-training. If, for example, only dogs were classified against cats, bears and monkeys during the meta-training process, giraffes can be classified against elephants during meta-testing, followed by lions against tigers and cows against pigs. All these tasks were not seen during meta-training but they are similar enough so that one would expect the prior knowledge to suffice for learning the new task quickly.

Performance Measures

In contrast to standard learning, where determining the performance of a learner f_θ simply means to evaluate its loss $\mathcal{L}$ on unseen test data, determining how well a meta-learned prior φ boosts fine-tuning any θ_j to an unseen test task $T_j \sim p(T)$ is more challenging. As meta-learning aims to enable fast adaption to unseen tasks, the mere results on the test tasks' test sets X^j_{test} after fine-tuning are not sufficient to properly evaluate the meta-learning performance. Instead, various measures must be used to draw a more detailed picture of the adaption capability the model f_φ developed through meta-training. For this reason, the following paragraphs define, motivate and discuss the most important performance measures used for evaluating meta-learning results. However, in most works, the notion performance refers to the total reward collected within one episode, if not stated otherwise, while the notion "asymptotic performance" refers to the accumulated rewards gained after a full, standard training as described at the beginning of this section. [3]

Generalization: In standard learning generalization refers to the ability of a learner to perform well on unseen data after training. It is generally quantified using the concept of the generalization error, which is the difference between the learner's performance on the training set and its performance on a test set, i.e.

$$\mathcal{L}_{\text{gen}}(\theta) := \mathcal{L}(\theta, X_{\text{test}}) - \mathcal{L}(\theta, X_{\text{train}}). \tag{6}$$

Analogously, the generalization ability of a meta-learner f_φ can be quantified by the accumulated generalization error

$$\mathcal{L}^{\text{acc}}_{\text{gen}}(\varphi) := \mathcal{L}^{\text{meta}}_{\text{test}}(\varphi) - \mathcal{L}^{\text{meta}}_{\text{train}}(\varphi) \tag{7}$$

over the unseen meta-test tasks, where the accumulated train and test losses are defined as the sum over the respective (standard learning) train and test losses

$$\mathcal{L}^{\text{meta}}_{\text{train}}(\cdot) := \sum_{T_j \sim p(T)} \mathcal{L}(\cdot, X^j_{\text{train}}), \qquad \mathcal{L}^{\text{meta}}_{\text{test}}(\cdot) := \sum_{T_j \sim p(T)} \mathcal{L}(\cdot, X^j_{\text{test}}),$$

and $\theta'_K(\varphi)$ denotes the adapted parameters after k training shots on the respective task T_j starting from prior φ.

[3]The notion "asymptotic" implies $K \to \infty$, which, from a theoretical point of view, is what happens in standard learning.

However, the accumulated generalization error equation 7 measures the generalization performance at task level, i.e. how well the learned prior φ boosts task-specific fine-tuning. In order to measure the meta-level generalization, equation 7 must be abstracted to the meta-generalization error

$$\mathcal{L}_{\text{gen}}^{\text{meta}}(\varphi) := \mathcal{L}_{\text{gen}}^{\text{test}}(\varphi) - \mathcal{L}_{\text{gen}}^{\text{train}}(\varphi) \tag{8}$$

where $\mathcal{L}_{\text{gen}}^{\text{train}}$ and $\mathcal{L}_{\text{gen}}^{\text{test}}$ denote the accumulated generalization error equation 7 summed over all meta-training tasks $T_i \sim p(T)$ or meta-test tasks $T_j \sim p(T)$ respectively. The meta-generalization error evaluates how well the model f_φ generalizes on meta-level by taking its task-specific generalization capability on the training tasks into account. A good meta-generalization means that φ boosts task-specific learning equally well for training and test tasks, while a bad meta-generalization hints φ to overfit to the training tasks in such a way, that fine-tuning has the maximal success, but this knowledge cannot be transferred to other tasks from the same distribution $p(T)$. In this work, the latter is called meta-overfitting, although it is a rather theoretical concept as the iterative sampling of training tasks throughout meta-training should avoid such an issue.

The difference between the task-level and the meta-level generalization can be highlighted by the meta-testing for example 2, as described in the previous subsections: If a model f_φ (e.g. a CNN) generalizes poorly on the test tasks (cats vs. horses, horses vs. fish and dogs vs. crocodiles) this can be due to two reasons:

1. The model f_φ is not able to generalize to classification tasks not seen during training, or
2. the model f_φ did not learn how to boost few-shot image classification at all.

The former is indicated by both, the task-specific and meta-level generalization errors, being high, while the latter follows from a bad task-specific generalization on all classification tasks, a property measured through the meta-generalization error. Consequentially, both, the task-level and meta-level errors, should be measured throughout meta-testing.

Adaption speed: Adaption speed refers to the rate at which a learner can effectively learn new tasks T_j from a limited number of examples i.e. in K shots. It is typically measured by tracking task-specific metrics over time, and examine the slope of the gained curve. A high slope indicates fast adaption, and vice versa. In the context of meta-learning adaption performance is therefore typically tracked during the individual training of each test task T_j, e.g. by the task-specific training loss $\mathcal{L}_{\text{train}}^j$. This way, one yields an adaption performance measure $\mu_{\varphi,T_j}^{\text{adapt}} : K \to \mathbb{R}$ with

$$\mu_{\varphi,T_j}^{\text{adapt}}(k) := \mathcal{L}_{\text{train}}^j(\theta_j^{(k)}, X_{\text{train}}^{j,(k)}) \tag{9}$$

for each task T_j, where (k) denotes the k-th iteration of learning, i.e. the parameters $\theta_j(\varphi)$ after k epochs of inner learning and the respective training set. Calculating equation 9 for each test task T_j and each fine-tuning iteration k yields the meta-adaption performance. Note, however, that the training loss $\mathcal{L}_{\text{train}}^j$ is only the most common example for the task-specific performance measure used for evaluating the adaption speed. In theory, any task-specific loss can be used. For example, using the task-specific generalization equation 6 one examines, how fast a model gains generalization capabilities within one particular task T_j.

Sample-Efficiency: Since gathering data is often difficult or expensive in the context of few-shot learning (especially for the reinforcement learning paradigm presented in the next section) it is also desirable to yield meta-learners that adapt to unseen situations without the need for large amounts of data. Such learners are referred to as sample-efficient. In contrast to adaption speed, which measures how fast a model can learn and improve within a certain amount of training iterations regardless of the samples needed, sample-efficiency is about learning effectively from fewer examples regardless of the training iterations required for extracting the data's information. Analogously to the adaption speed equation 9, it is measured by

$$\mu_{\varphi,T_j}^{\text{sample eff}}(S) := \mathcal{L}_{\text{train}}^j(\theta_j^{(S)}, X_{X \subseteq X_{\text{train}}^j}^{|X|=S}) \tag{10}$$

for each task individually, where $\theta_j^{(S)}$ denotes the parameters $\theta_j(\varphi)$ fine-tuned with S observations. In case of example 2, sample-efficiency is, however, the same as adaption speed as each training iteration contains of only one image to be classified, i.e. as $h = 1$ in classification. Note, moreover, that, similar to adaption speed, sample-efficiency can be measured with any kind of task-specific loss.

Out-of-distribution: While generalization refers to a learner's ability to perform well on unseen data that is drawn from the same distribution as the training data. Out-of-distribution (OOD) performance specifically evaluates how well a model can handle data that comes from a different distribution. In other words, while generalization measures how well the learner can apply learned patterns to new examples within the same context, OOD assesses a learner's ability to generalize to new, unseen situations not represented during training. Consequentially, high OOD performance indicates robustness and adaptability, while poor performance can lead to failures in real-world applications where conditions may vary significantly from the training scenarios.

On meta-level, this means to define the test tasks as the family $\{T_j\}_{T_j \not\sim p(T)}$ of tasks particularly not drawn from the task distribution $p(T)$. Then, measuring the performance means to calculate all the metrics mentioned above, i.e. task-specific generalization equation 7, meta-generalization equation 8, adaption speed 9 and sample efficiency equation 10.

In example 2, OOD on meta-level means classification tasks that do not specifically classify animals against each other. This can be the different dog breeds, their posture, the color of their coat or anything that is different from dogs vs. other animals while still using the same images. Even different images showing e.g. trees, cars or houses, are possible, but in this case one rather speaks of out-of-domain tasks.

2.2 Meta-RL Paradigm

Although the previous section gives a general introduction to the field of meta-learning, this work focuses on meta-reinforcement learning (meta-RL) in particular. For this reason, it is derived from the general meta-learning paradigm throughout the next subsections starting with the standard reinforcement learning (RL) paradigm and mimicking the structure of the previous section.

Standard Reinforcement Learning

In standard reinforcement learning (RL) an agent interacts with an environment $(\mathbb{A}, \mathbb{S}, R)$ whose underlying dynamics are assumed to form a Markov Decision Process (MDP)

$$M := (\mathbb{A}, \mathbb{S}, R, \gamma, \mathbb{T}, \mu, h). \tag{11}$$

In this manner, RL is a special case of the standard machine learning paradigm introduced in the previous section, where

- the standard task of the form equation 1 is extended by the action space $\mathbb{A}$ denoting the possibility of the agent to influence the environment - and hence the subsequent states - by an action $a \in \mathbb{A}$
- the state space $\mathbb{S}$ replaces the observation space $\mathcal{X}$.
- μ represents the distribution over initial states $s_0 \in \mathbb{S}$ rather than observations $x_0 \in \mathcal{X}$.
- the horizon h determines the end of an episode i.e., the terminal state s_h.

Figure 5 highlights this agent-environment interaction.

The strategy according to which action a_t is selected in state s_t for any given time t is called the policy. It is determined by the agent's weights θ and, hence, representative of the agent. If these parameters correspond to the weights of a neural network, one - analogous to the standard learning paradigm - speaks of Deep Reinforcement Learning (DRL). Throughout this work, only deep (Reinforcement) learning models are considered, so that terms DRL and RL as well as learning and deep learning will be used synonymously.

The policy $\pi_\theta : \mathbb{S} \to \mathbb{A}$ implicitly has an effect on the transition $\mathbb{T}$ from one state to another, as, in a MDP of the form equation 11, the subsequent state s_{t+1} always depends on the action a_t selected by π_θ. Hence,

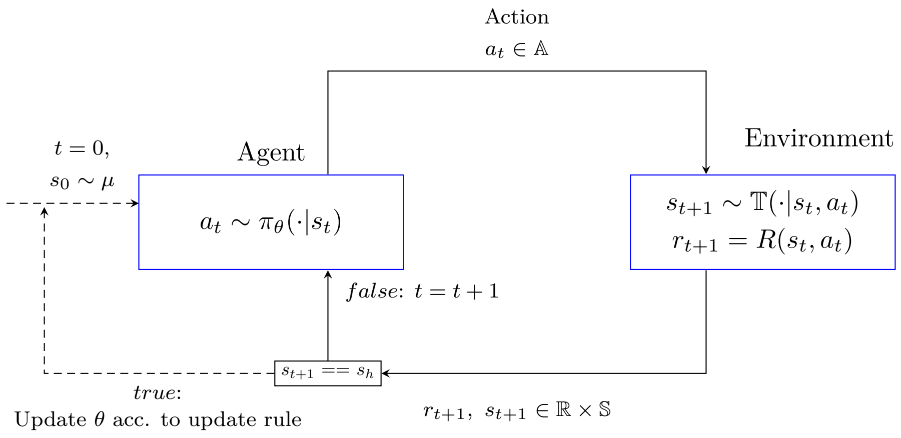

Action
$a_t \in \mathbb{A}$
Agent
Environment
$t = 0,$
$s_0 \sim \mu$
$a_t \sim \pi_\theta(\cdot|s_t)$
$s_{t+1} \sim \mathbb{T}(\cdot|s_t, a_t)$
$r_{t+1} = R(s_t, a_t)$
false: $t = t + 1$
$s_{t+1} == s_h$
true:
Update θ acc. to update rule
$r_{t+1},\ s_{t+1} \in \mathbb{R} \times \mathbb{S}$

Figure 5: Agent-Environment Interface: An agent receives the initial state $s_0 \sim \mu$ and selects an action $a_0 = \pi_\theta(s_0)$ within the environment w.r.t. its strategy parameterized by its weights. Then, the environment transits to the next state which, together with the reward, is processed back to the agent that selects the next action. This interaction is iteratively repeated until a terminal state s_h (determined by horizon h) is reached. Then, the agent updates its weights according to the respective RL algorithm's update rule in order to minimize the loss equation 12, and a new episode is started with another $s_0 \sim \mu$.

the transition of a general task equation 1 extends to the distribution of subsequent states given a state and a taken action

$$\mathbb{T} : \mathbb{S} \times \mathbb{A} \times \mathbb{S} \to \mathbb{R}.$$

This relationship between the current state s_t, action a_T and the subsequent state s_{t+1} has to fulfill the Markov property, i.e. earlier information than at time t must not be relevant. The same must hold for the reward function $R : \mathbb{A} \times \mathbb{S} \to \mathbb{R}$ that determines the loss $\mathcal{L}$ of a general task equation 1 together with the discount factor $\gamma \in [0, 1]$. A RL task T_i is hence defined as the goal to maximize rewards through agent-environment interaction in the corresponding MDP M_i.

If $h < \infty$, the number of agent-environment interactions before resetting the environment is bounded and the respective RL problem is called episodic. Since this is a rather practical assumption, h is considered to be bounded throughout this work. However, as some problems have differing episode lengths, h can be dynamic so that it changes at each environment reset. This is the case in example 3 below.

The loss $\mathcal{L}$ is generally referred to as the value function and represented by the negative expected sum over all rewards gained from the initial state s_0 up until the terminal state s_h following the strategy π_θ:

$$\mathcal{L} := -\mathbb{E}_{\pi_\theta(\cdot)}^{s_0 \sim \mu} \sum_{t=0}^{h} \gamma^t r_{t+1} \tag{12}$$

where $r_{t+1} = R(s_t, a_t)$ is the reward following from state s_t and action a_T and $\mathbb{E}_{\pi_\theta(\cdot)}^{s_0 \sim \mu}$ denotes the expectation under the assumptions that s_0 is drawn from the initial distribution μ and every action a_t is taken according to policy $\pi_\theta(s_t)$. Inserting this loss into the optimization problem equation 2 of standard learning yields the formal RL problem.

The value function equation 12 implies the agent to only update its parameters at the end of an episode. This is, however, not necessarily the case, although assumed throughout this paper. But generalizing equation 12 to the expected future reward starting from any state s_t at any time t is straightforward as it only requires for an index shift within the sum.

There are almost as many rules for updating θ as RL algorithms, and not all of them explicitly minimize equation 12 Sutton & Barto (2018), Francois-Lavet et al. (2018). For example, The family of function approximation RL algorithms aims to approximate the loss equation 12 (or modifications of it). Here, θ

is optimized to best approximate the loss, while the loss minimization is, afterwards, achieved by acting accordingly.

The training-validation-test split in RL is completely analogous to the one described for standard learning at the beginning of the previous section. However, sampling an observation set means something quite different in RL as the agent always has to interact with the environment in order to observe the next state. One rather speaks of "collecting experience" and this procedure always depends on the current policy π_θ. During validation and testing, this policy (i.e. its parameters θ) is fixed, while it normally changes throughout training.

Example 3 - Racing games:

Considering a racing game, where the agent's goal is to drive in a way that finishes a racing track as fast as possible, the different components of the underlying MDP equation 11 are defined as followed:

- The set of possible actions $\mathbb{A}$ contains all directions the racer can move to, its speed and whether or not to brake.
- The state space $\mathbb{S}$ contains all states, the racer can possibly find itself in. Additional to the current racing track (as pixels), it may include information on the racer's speed as well as the time since the start of the race or other useful information a player might have.
- The initial distribution over states μ refers to what the racer "sees" in its starting position. If that position is deterministic, μ is deterministic, too.
- The reward function can easily be modeled as one when reaching the goal and zero otherwise.
- The discount factor γ must be smaller than one to encourage the agent to finish the racing track as fast as possible.
- The strategy is the theoretical decision rule of the agent. It determines, how the racer actually drives at any time in the race.
- Each race is an episode. Hence, the time the agent needs to finish the racing track, determines the horizon h that is therefore dynamic.

In order to learn the optimal strategy for a particular racing track, the agent must - similar to a human player - explore different strategies, i.e., try out different ways to drive. At the same time, it must exploit its gained knowledge to finish the race faster. This trade-off is called the exploration-exploitation dilemma.

Problem Setup

Letting an agent interact with an environment for training purposes can be extremely financially expensive e.g., in robotics, trading or transportation. For such problems, an agent must be trained within a simulation and, thereafter, adapt as fast as possible to the real world. But a simulation is a model of the real world and can, hence, reflect the reality only up to a certain extent. Moreover, environmental dynamics often change over time e.g., when shocks permanently increase market prices, when a robot is assigned to a different task within the same physical environment or when a racing track (like in example 3) becomes more slippery due to heavy rain. In such scenarios, a standard RL agent must be trained fully from scratch, although it has already gained a lot of knowledge about the environment that is still valid. A vast amount of computational power and training time is getting wasted. Meta-RL agents are needed that can quickly, i.e. within K episodes of task-specific training, adapt to any kind of environmental changes and the real world.

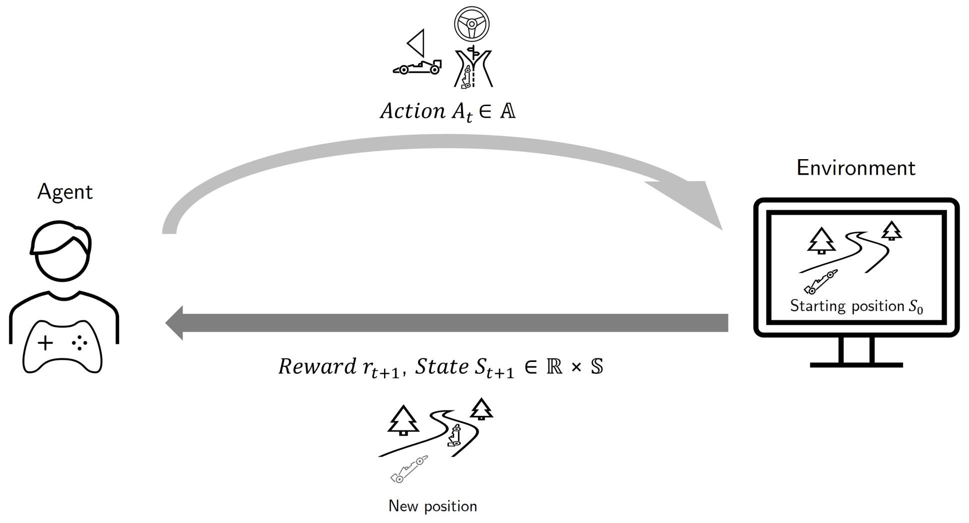

Action $A_t \in \mathbb{A}$
Agent
Environment
Starting position S_0
Reward r_{t+1}, State $S_{t+1} \in \mathbb{R} \times \mathbb{S}$
New position

Figure 6: Agent-environment interaction on a racing game. The agent observes its current position in the race and requests a left turn. The environment processes this turn and thereby transits to the next state that is given to the agent along with the reward signal in order to start the next loop. The initial state s_0 is defined by the starting position. Reaching the goal ends the episode.

Example 4 - Meta-Racing problem:

Example 3 can easily be modified to be a meta-RL problem by parameterizing how slippery the racing track is. Such families of MDPs are called parametric. For each different value of slipperiness the transition $\mathbb{T}$ of the underlying MDP slightly changes, while the overall structure of state and action spaces and even reward remain unchanged. Acting optimally in any of these environments requires almost the same high-level skills - the agent still needs to learn driving dynamics, when to brake and which direction to drive to, just with a different level of drifting in curves due to a change in slipperiness. In other words, solving the meta-task to learn how to drive and finish a randomly slippery racing track as fast as possible, remains the same.

Meta-Reinforcement Learning Paradigm

Analogous to meta-learning, the meta-RL paradigm formally consists of a distribution $p(M)$ over MDPs of the form

$$M_i := (\mathbb{S}, \mathbb{A}, R_i, \mathbb{T}_i, \mu_i, \gamma, h). \tag{13}$$

The MDPs from the same family $\{M_i\}_{M_i \sim p(M)}$ are considered to vary with respect to their underlying dynamics $\mathbb{T}_i$ or loss function - implicitly given by discount factor γ and the reward function R_i - while sharing some structure like state and action space - which, together, can be interpreted as the observation space $\mathcal{X}$ of a general task equation 3. These assumptions are rather practical (see e.g., Melo (2022) or Duan et al. (2016)) than an actual theoretical necessity. Since a RL task T_i is determined by the underlying MDP M_i, the notions task and MDP are used synonymously throughout this work.

Collecting knowledge within a family of MDPs (possibly) means two similar but different things:

1. Collecting general knowledge about structures like state or action space that are shared within the family, or

2. Identifying the current MDP as fast as possible to adjust the agent's policy π_θ accordingly (see e.g., the PEARL algorithm Rakelly et al. (2019) presented in section 4.1).

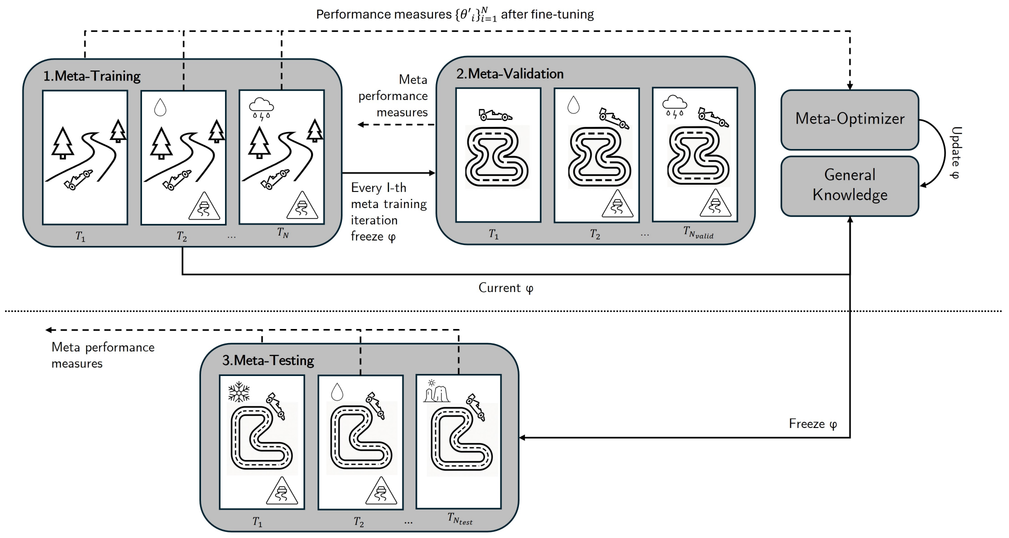

Performance measures $\{\theta'_i\}_{i=1}^N$ after fine-tuning
1.Meta-Training
T_1
T_2
...
T_N
Meta performance measures
Every l-th meta training iteration freeze φ
2.Meta-Validation
$T_{N_{valid}}$
Meta-Optimizer
General Knowledge
Update φ
Current φ
Meta performance measures
3.Meta-Testing
$T_{N_{test}}$
Freeze φ

Figure 7: Meta-learn how to generally finish different racing tracks with varying weather conditions. Starting from the meta-knowledge φ, K episodes of MDP-specific fine-tuning on the particular racing track yield performance measures as defined in section 2.1. The meta-optimizer uses these measures to update meta-knowledge. Every l episodes, different racing tracks are fine-tuned in order to evaluate the meta-training progress. After meta-training, test tracks are fine-tuned to finally evaluate how well the meta-knowledge enables adaption to unseen MDPs from the same MDP distribution $p(M)$.

In terms of example 4, the former means to learn how to process the pixels shown as the current state, remember general information like the route of the race, to understand how to drive a car on an abstract level, while the latter means to figure out as fast as possible how slippery the track actually is in order to adjust the style of driving accordingly. This way, meta-RL can also be considered to be two-staged. On meta-level common knowledge over all MDPs is collected, while on MDP-specific level the current MDP is identified to adjust the MDP-specific policy respectively. Both these interpretations have a direct impact on modeling the meta-training process. While memory-based meta-RL algorithms like RL^2 Duan et al. (2016) and TRMRL Melo (2022) presented in sections 4.2 and 4.3 respectively mainly follow the first idea, the PEARL framework Rakelly et al. (2019) discussed in section 4.1 models the algorithm in such a way that it quickly finds out in which MDP it is acting in to optimize its strategy for the remaining episodes. The training-validation-test split in the meta-RL paradigm thereby is completely analogous to the one in general meta-learning, i.e. due to K episode training, validation is not required for MDP-specific level, while certain MDPs are excluded from meta-training to function as validation or test MDPs respectively.

Meta-Training and Meta-Testing

Similar to general meta-learning and as highlighted in Fig. 3, the meta-training of the meta-RL paradigm is (theoretically) divided into an inner and an outer stage. Although the baseline meta-RL algorithms presented in 4 not necessarily use equation 12 as a MDP's loss $\mathcal{L}_i$, both, the inner optimization equation 5 as well as the outer optimization equation 4, are straightforward to adjust to meta- and standard RL respectively by introducing a meta-variable φ representing meta-knowledge. The only core structural difference is in the MDP-specific training and testing data X^i_{train} and X^i_{test}: In RL an agent must interact with an environment to obtain data rather than sampling it from some observation space $\mathcal{X}$. Hence, in the meta-RL paradigm, gathering data from a particular MDP M_i, in K shots means to act within the environment for K episodes following the strategy $\pi_{\theta_i(\varphi)}$ that might or might not change between episodes depending on the inner (standard) RL algorithm used. Formally, a training or testing set X is thus defined as

$$X := \{(s^k_t, a^k_t, r^k_{t+1}, s^k_{t+1}\}^{k=1,\ldots,K}_{t=0,\ldots h}, \tag{14}$$

where k represents the current episode and t counts the time steps taken within that episode.

The meta-testing in the meta-RL paradigm is also straightforward to adapt from the meta-testing explained in the previous section and highlighted in Fig. 4: The meta-variable φ is held fixed while for any unseen MDP M_j iteratively sampled from the family of test tasks $\{M_j\}_{M_j \sim p(M)}^{M_j \neq M_i}$ the MDP-specific policy $\pi_{\theta_j(\varphi)}$ is fine-tuned in K episodes of inner learning. The only major difference of meta-RL meta-testing is in the task-specific testing of $\pi_{\theta'_j}$ i.e. in the evaluation of the quality of fine-tuning $\pi_{\theta_j(\varphi)}$: Since, in RL, data can only be gathered by interacting with the environment, the fine-tuned policy π'_{θ_j} is held fixed for another number of K_{test} of episodes. The test loss $\mathcal{L}^j_{\text{test}}$ required for calculating the performance measures presented in the previous section is, thereby, represented by the (accumulated) reward gained throughout these episodes. For example 4, this can mean to let the meta-trained agent complete the racing track when it snows (as long as it has not seen snow during meta-training). The agent MDP-specifically adapts to the new situation within the next K races, in order to be evaluated on the same racing track with the same weather conditions without further adapting its strategy within another K_{test} number of races.

Bayesian Reinforcement Learning

An agent must explore the environment to uncover new actions and their potential rewards, which is essential for discovering optimal strategies. At the same time, it must exploit its existing knowledge to maximize immediate rewards based on previously learned information. Balancing these two strategies is crucial, particularly in a few-shot learning setting: If an agent focuses too much on exploration, it risks missing out on maximizing its returns from known actions, while excessive exploitation may result in suboptimal long-term performance due to undiscovered options, e.g., in example 4, it is important to figure out the slipperiness of the track before the first sharp bend. The goal is to select actions that maximize expected rewards while simultaneously refining the agent's understanding of the environment, enabling it to adapt its strategy as more information becomes available.

Bayesian Reinforcement Learning (BRL) provides a sophisticated approach to tackle this issue by allowing agents to quantify uncertainty and make informed decisions that enhance learning efficiency. It is the groundwork for task-representation algorithms like PEARL Rakelly et al. (2019) or VariBAD Zintgraf et al. (2020) presented in sections 4.1 and 4.2 respectively. The following paragraphs present the main idea of the paradigm of model-based BRL, where the agent employs a model of the dynamics of the environment $\mathbb{T}$ and a reward structure R to guide its decisions. As all BRL-based algorithms presented in section 4 use model-based approaches, model-free BRL is not discussed in this work.

The main idea of BRL is to leverage the gathered information of the agent together with Bayesian methods enabling the agent to adapt its strategy as more information becomes available. For this purpose, the notion of a MDP of the form equation 11 is extended to that of a Bayes-Adaptive MDP (BAMDP), [4] i.e. a tuple

$$M' := (\mathbb{A}, \mathbb{S}', R', \gamma, \mathbb{T}', \mu', h'), \tag{15}$$

with a hyperstate-space $\mathbb{S}' = \mathbb{S} \times B$ as the extension of the original state space $\mathbb{S}$ by the belief space B capturing the distribution over the model parameters, i.e. the parameters of the transition $\mathbb{T}$ and the reward R. The augmented transition and reward functions as well as the initial distribution and horizon are defined analogously to the hyper-state space (see Zintgraf et al. (2020), Ghavamzadeh et al. (2015)for further details). In terms of example 4, this means to capture a belief of the current slipperiness of the track while collecting more experience to make this belief more precise.

The current belief $b_t := p_t = p(R', \mathbb{T}'|c_t) \in B$ is defined as the posterior of the model parameters given the current experience c_t^T and a prior distribution $p_0 = p(R', \mathbb{T}')$ over the unknown reward and transition functions R and T of a MDP equation 11. The prior is often chosen as a normal distribution, while the posterior p_t is computed according to the Bayesian rule after collecting experience $c_t^T := \{s_{t+1}, r_t, a_t, s_t, \dots, s_1, r_0, a_0, s_0\}$, i.e.

$$p_t = p(R, T|c_t) = \frac{p(c_t|R, T)p(R, T)}{\int p(c_t|x)p(x)dx}. \tag{16}$$

[4]In Ghavamzadeh et al. (2015) it is done for discrete state and action spaces, but it is straight forward to adapt this for continuous spaces

The computation of the posterior equation 16 is generally intractable. Hence it must be approximated by the respective BRL algorithm, e.g. by representing it via a neural network.

The posterior captures the current knowledge of the model environment subject to the prior distribution. Hence it encodes the agent's uncertainty about model environment parameters. Since the agent aims to select the best possible action in each time step based on the current belief, while simultaneously refining the posterior of that belief whenever new information becomes available, exploration and exploitation are implicitly traded off this way.

Agents optimally trading-off exploration and exploitation are called Bayes-optimal agents. However, although a policy that maximizes the value function of a BAMDP of the form equation 15 is called a Bayes-optimal policy, the value (i.e. accumulated reward) of that policy might be lower than that of the optimal policy π^* as defined in the previous section since the agent needs to (optimally) explore the environment before finding the optimal model parameters. In example 4, the bayes-optimal behavior would initially require to drive and brake a few times to figure out how slippery the track is, although this increases the racing time in the first training episode.

3 Terminology

In the literature, meta-learning is often confused with the notions of transfer learning (TL) and multi-task learning (MTL). Although these concepts share quite some similarities - there are even algorithms exhibiting overlapping characteristics among them Upadhyay (2023) - the confusion mainly arises from a lack of clear definition. Therefore, the following paragraphs broadly introduce TL and MTL focusing on clear definitions. For a more detailed consideration, the reader is, nevertheless, referred to other works like Upadhyay (2023) particularly focusing on meta-learning, MTL, TL and their overlaps.

3.1 Transfer Learning

The significance of the concept of Transfer Learning was notably heightened following its integration with deep neural networks. Landmark advancements in computer vision, exemplified by architectures such as AlexNet Krizhevsky et al. (2012), GoogleNet Szegedy et al. (2015), and EfficientNet Tan & Le (2019), were pre-trained on the extensive ImageNet dataset before being employed in tasks such as image classification, segmentation, and object detection, respectively. These models demonstrated substantial improvements over traditional machine learning algorithms in terms of adaptability, sample efficiency, and few-shot performance. Motivated by these achievements, transfer learning has also been effectively applied in various other domains, including speech recognition (e.g. Wav2Vec Schneider et al. (2019)), healthcare applications Esteva et al. (2017), and reinforcement learning (RL). The latter itself distributes into various applications as knowledge transfer is used to close the gap between simulation and reality in robotics Yu et al. (2019), learn from demonstrations to incorperate expert knowledge Ravichandar et al. (2020), Sosa-Ceron et al. (2022), or adapt to new rules and gaming mechanics in game play using reward shaping OpenAI et al. (2019). All these various sub fields of TL and transfer RL present distinct challenges and are hence areas of ongoing research. Interested readers are thus encouraged to explore specific surveys on TL Niu et al. (2020), Zhuang et al. (2021), Tan et al. (2018), Panigrahi et al. (2021), and transfer RL Zhu et al. (2023), Zhao et al. (2020) for deeper insights.

Transfer Learning Paradigm

The main idea of TL is to extensively pre-train a model F_θ on a source task $T_1 := \{\mathcal{L}_1, \mathcal{X}_1, \mu, \mathbb{T}_1, h\}$ and transfer the acquired knowledge by fine-tuning f_θ on a related target task $T_2 := \{\mathcal{L}_2, \mathcal{X}_2, \mu, \mathbb{T}_2, h\}$. This approach leverages knowledge from pre-training to enhance few-shot performance and learning speed on the target task T_2, mirroring human learning. For example, one can extensively practice inline skating in the summer to prepare for skiing when it snows for only a few days in the winter. Therefore, the source task T_1 typically contains abundant, well-labeled data $\mathcal{X}_1$, while the target task's data $\mathcal{X}_2$ often is sparse, of low quality or expensive to obtain. Additionally, the goal encoded by $\mathcal{L}_1$ can differ from that represented by the loss function $\mathcal{L}_1$ or there may be a change in domain dynamics $\mathbb{T}$ that needs to be learned quickly. The latter is of particular importance in RL problems, where changes in environment dynamics can cause severe issues; for example when an agent is supposed to drive a car and the terrain becomes significantly more slippery. A change in the loss function can, however, become problematic if the loss $\mathcal{L}_2$ is considerably more computationally expensive to calculate as this slows down learning.

One can also define the TL paradigm with the possibility of several source tasks. However, this is rather a technical difference as one can define $T_1 = \bigcup_{i=1}^{N} T_1^i$, where T_1^i are the different source tasks. If, however, more than one target task exists, the notion "downstream tasks" is often used. This is the case in more recent developements, where TL, along with a vast amount of data and self-supervised learning approaches, led to much more powerful and broadly applicable models, namely foundation models.

Foundation Models

Foundation models serve as a common basis from which many task-specific models are built through adaptation Bommasani et al. (2021), i.e. they are pre-trained by a vast amount of source tasks as described above, before being fine-tuned to different downstream tasks. As a consequence, foundation models are inherently incomplete: they rather serve as a "foundation" for fine-tuning to various applications than as static general problem solvers. This way, the adaption of foundation models follows the meta-testing paradigm described in section 2.1, while their training is that of a TL algorithm not following the meta-training paradigm of section 2.1. As they are meant to be trained on a distribution $p(T)$ of training tasks rather than one single source task before being fine-tuned to a (potentially different) distribution of downstream tasks $p_{\text{test}}(T)$, they can be viewed as the "meta-level version" of TL.

TL is the key technique that makes foundation models possible, but scale is what makes them so powerful Bommasani et al. (2021), a fact that probably arises from the universal approximator property of neural networks: larger models can represent more complex functions of higher dimension, although they also tend to overfit more easily. The latter is the reason why besides model size, the data (e.g. the task pool in meta-learning) is also required to scale Team et al. (2023), Bommasani et al. (2021). However, as the amount of data increases, the data quality can not neccessarily be maintained. Instead, self-supervised learning techniques Gui et al. (2024), Ericsson et al. (2022) are used to learn patterns in available, but potentially unlabeled, data, so that foundation models are often defined as "large-scale transfer-learning models pre-trained via self-supervised learning" Bommasani et al. (2021).

There was a paradigm shift towards foundation models that get fine-tuned for downstream tasks within the recent years Bommasani et al. (2021). For example, almost all state-of-the-art NLP models are now adapted from one of a few foundation models, e.g. Bert Devlin et al. (2019), GPT3 Brown et al. (2020), GPT4 OpenAI et al. (2024), LLAMA Touvron et al. (2023). Similarly, foundation models were developed for various applications like agriculture Yin et al. (2025), medicine Zhang & Metaxas (2024), Liang et al. (2025), Moor et al. (2023), or robotics Firoozi et al. (2025). In RL, foundation models have been used for reward engineering Ye et al. (2024), Wang et al. (2024b), e.g. to generate rewards from only text description and the visual observations yielded from the environment Wang et al. (2024b). Additionally, the LLAMA Touvron et al. (2023) foundation model was successfully fine-tuned on RL downstream tasks without RL-specific training Rentschler & Roberts (2025) with remarkable results. As a consequence, NLP and RL paradigms were combined into RL from Human Feedback (RLHF), which is itself an ongoing and vast research field, so that interested readers are referred to particular RLHF surveys like Kaufmann et al. (2023), Casper et al. (2023), Chaudhari et al. (2025).

3.2 Multi-Task Learning

The concept of Multi-Task Learning (MTL) was first introduced by MTL as the formalization of the idea of training models on multiple tasks simultaneously. The main idea is to train one single model among several similar tasks simultaneously in order to find valuable, general feature representations that can be leveraged to improve task-specific performances.

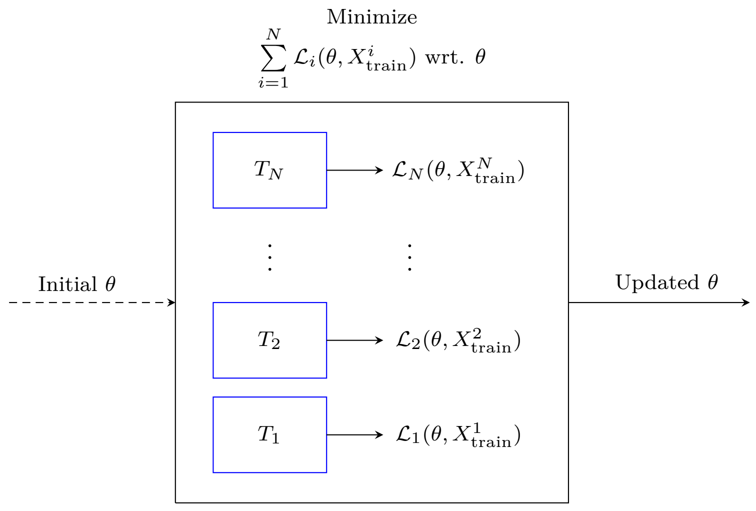

Minimize
$\sum_{i=1}^{N} \mathcal{L}_i(\theta, X^i_{\text{train}})$ wrt. θ
Initial θ
Updated θ
T_N
$\mathcal{L}_N(\theta, X^N_{\text{train}})$
T_2
$\mathcal{L}_2(\theta, X^2_{\text{train}})$
T_1
$\mathcal{L}_1(\theta, X^1_{\text{train}})$

Figure 8: General Multi-Task-Learning Paradigm:

Applying the paradigm of MTL to different neural network architectures like Transformers Torbarina et al. (2024) or Bayesian networks Lazaric & Ghavamzadeh (2010), ..., led to remarkable successes in neural language processing Chen et al. (2024), neural architecture search Gao et al. (2020), Pasunuru & Bansal (2019), segmentation Bischke et al. (2019), Zhou et al. (2021), Tang et al. (2023), medicine Bi et al. (2008), Zhang et al. (2023b), Zhou et al. (2021), and robotics Arcari et al. (2023), Gupta et al. (2021a), Deisenroth et al. (2014), to list only a few. Interested readers are, however, referred to MTL-focused surveys like Zhang & Yang (2022), Zhang & Yang (2018), Yu et al. (2024) or Multi-Task-RL (MTRL) specific works like Vithayathil Varghese & Mahmoud (2020) or D'Eramo et al. (2024) for a broader overview of MTL and MTRL algorithms. This section rather focuses on MTL and MTRL paradigms to help readers distinguish them from TL and meta-learning or transfer-RL and meta-RL respectively.

Multi-Task Learning Paradigm

Formally, in MTL a model is jointly trained to solve a given number N of tasks $(T_1, T_2, \dots, T_n)$ sharing some common structure. The corresponding training process mimics standard learning on task-specific level, but with parameters being shared throughout the tasks. A meta-level does not exist and hence there is no two-staged training process. Instead, the MTL paradigm aims to solve all N tasks equally well. The resulting optimization problem is visualized in figure 8 and can be formalized as

$$\text{Minimize} \quad \sum_{i=1}^{N} \mathcal{L}_i(\theta_i, X^i_{\text{train}}) \quad \text{wrt. } \theta \tag{17}$$

where $\theta_i \in \mathbb{R}^{d_i}$ denotes the task-specific parameters reserved for each individual task.

The task specific parts of the model are typically task-specific heads with one or more layers, so that the model parameters θ consist of task-specific parameters θ_i for each task T_i as well as common knowledge denoted by φ, but all in one single model. The performance of the MTL model is evaluated on task-level only, i.e. on the task-specific test sets X^i_{test}. A meta-test on unseen tasks does not take place.

Example 5 - Multi-Task Image classification: Example 4 can easily be modeled wrt. the multi-task-learning paradigm by trying to learn N different image classification tasks simultaneously. In each of these tasks, the images have to be decomposed into smaller pieces at first, before solving the task. This is the common knowledge φ, while the different heads are supposed to learn the particular classifications. Due to its broader definition of the notion "similar tasks", this MTL problem can easily be extended by tasks like detecting objects around the animals or augmenting the images by trees or houses - an extension that is not possible in the meta-learning paradigm.

In contrast to the meta-learning paradigm described in section 2.1, the MTL paradigm does not treat the parameters θ_i of each task as individual parameter sets. They are rather considered as the different parts of one bigger set of parameters $\theta = \{\theta_0, \theta_1, \dots, \theta_N\}$ with $\theta_0 \in \mathbb{R}^{d_0}$ denoting the common knowledge over all tasks. A deep neural network can, for example, learn some feature representations shared between all different tasks to extract the relevant information of its data before fine-tuning to the respective task. Such a NN is considered to consist of different blocks (of layers), a general part encoding all the information shared between tasks, and a smaller block (or head) for each individual task.

Multi Task Reinforcement Learning

In Multi-Task Reinforcement Learning (MTRL) one develops a policy for several similar but different tasks at once. Analogous to the general MTL paradigm, the common knowledge is about how to process the information the current state provides, which action has which effect on the environment and what are general dynamics shared among tasks. At the same time, rewards and transitions differ between the different MDPs, what requires for task-specific heads to decide for an action out of the (commonly) processed state information. Since the notion of a policy is rather a theoretical concept, this means the same neural network can function as the agent in the different MDPs with a task-specific head for each MDP. A second part of potentially common knowledge is task identification. However, compared to the more general meta-RL paradigm, this is rather simple as the same N tasks are always being considered. Most algorithms simply use one-hot encoding.

Example 5 - Multi-Task Racing:

Modifying example 3 so that the agent has to absolve N racing tracks at a time, one obtains a MTRL problem. As long as the agent observes the same information in each race - as mainly is the case among racing games when tracks change - processing the pixels or understanding car mechanics and general physics is common knowledge useful in all MDPs. Remembering the racing track and its respective characteristics is, however, task-specific.
In the literature, MTRL is often confused with Multi-Agent learning - where multiple agents act within the same environment while trying to increase individual or collective rewards - or Multi-Actor Learning - where multiple agents with the exact same goal collect experiences and share them for faster learning. These two scenarios correspond to the single-task setting. The former extends the notion of single-task to multiple agents, while the latter only uses multiple instances of the environment in order to parallelize training.

3.3 Comparison

For differentiating between the three different paradigms of TL, MTL and Meta-Learning, it is important to understand that they are answers of increasing levels of abstraction Upadhyay (2023) to the question of how to exploit knowledge from previously learnt tasks to solve new ones in only a few shots. TL, as the solution with the lowest level of abstraction, simply transfers the knowledge gained in one task (the source task) to a second one (the target task), while Meta-Learning, as the solution with the highest level of abstraction, generally tries to extract the core information to learn any task of a distribution p inw shots. In other words, Meta-Learning also extracts information from the source task(s) to use them as a prior for the target task(s), but this information can be much more general, and it results from the outer optimization rather than from solving the source task(s) in particular - a difference becoming even more unclear when foundation models get pre-trained on several source tasks in order to be meta-tested afterwards.

The most confusion in the literature is between MTL and meta-learning Upadhyay (2023) since both aim to solve several tasks as well as possible rather than exclusively focusing on one single target task (like in TL). While meta-learning iteratively and sequentially learns a number N of tasks not necessarily fixed and, afterwards, tests the thus acquired knowledge on test tasks unseen during training, MTL trains a fixed number N of tasks simultaneously and without testing the resulting model parameters θ on unseen tasks. The trained MTL model is, instead, evaluated on the validation set X^i_{test} of each of the learnt tasks, $T_1, \dots, T_N$ which corresponds to the standard machine learning test phase but for N tasks at the same time.

In this way, MTL rather focuses on best solving the learnt tasks than on adapting to new ones as fast and good as possible.

MTL can consist of tasks of different families, such as classification, regression, or segmentation, while meta-learning is normally reduced to one family of tasks represented by the distribution p over tasks of the form equation 3. And, most importantly, MTL does not consist of an outer learner extracting prior knowledge (or meta representations). In this specific aspect, MTL is more similar to transfer learning, where prior knowledge between tasks is only transported implicitly via the model parameters. However, the TL paradigm particularly requires a target task to transfer the gained knowledge to, while the MTL paradigm does not specifically include such an option. This further highlights the similarity between TL and Meta-Learning as well as the difference between Meta-Learning and MTL.

4 Timeline of meta-learning developements

The previous sections provided paradigms of meta-learning and meta-RL as well as the terminology necessary for differentiating between meta-learning, MTL and TL. Now, this section presents the evolution of meta-RL algorithms, starting with the simpler gradient-based methods, and closing with memory-based frameworks, which include more sophisticated neural network architectures like Transformers Vaswani et al. (2017). By explaining the concepts of the different landmark algorithms, this section implicitly follows the development history of meta-learners visualized in figure 9 Whenever a new concept is introduced, the initial landmark development is described first, before presenting more sophisticated modifications. Similarly, by starting of with gradient-based meta-learners, this section presents concepts in the order from simplest up to most sophisticated. While gradient-based meta-learners are very easy to understand as long as the reader is familiar with the mere concept of gradient descent, memory-based meta-learners like Transformers themselves need further explanation of their neural network architecture.

The consideration of memory-based meta-learning algorithms would go beyond the scope of this work. As this work's introduction already pointed out, there are numerous different topics where meta-learners - and especially memory-based ones - are applied to a huge variety of applications, and many works survey these meta-learning approaches. In contrast, no work draws a clear timeline from the beginnings of meta-RL to the current state of the art. Consequently, the general meta-learning case is only being examined for the simpler gradient-based algorithms. All other - memory-based - approaches are only discussed for meta-RL thereby tracing the timeline of developments that led to Deep Mind's Adaptive Agent (ADA) Team et al. (2023), the most powerful state-of-the-art meta-RL framework at the time presented in the next section 4.4. It is a transformer-based meta-learner incorporating many common techniques for boosting meta-training such as distillation and auto-curriculum learning briefly explained in sections 4.5.2 and 4.5.1 respectively.

4.1 Gradient-based Meta-Learning

Many standard machine learners - including deep neural networks - are gradient-based methods i.e. they are getting optimized via (stochastic) gradient descent methods (or similar variants of gradient based optimizer's like Adam Kingma & Ba (2017) or ADAMW Loshchilov & Hutter (2018)). But those algorithms often converge slowly or not even converge at all. The convergence highly depends - among other factors - on the choice of the starting point of optimization. An optimizer might find the global optimum (if it even exists) very quickly when starting the optimization relatively close to it, but is very likely to need many more iterations or even end up in a poor local minimum when starting with randomly initialized model parameters. In order to solve this issue, several initialization schemes were invented for neural networks - the Gloroth Glorot et al. (2011), He He et al. (2015) and Lecun LeCun et al. (1989) initializations to name the most important ones.

A second major issue of gradient-based updates is the choice of the optimizer's learning rate α. If it is set too high it can cause the optimizer to be unstable or alternate around some solution, while chosen too low it significantly slows down learning Nar & Sastry (2018).

Both these problems could be taggled, if it was possible to a priori place the parameters in a sufficiently narrow area around the desired optimum. Not only would learning be more stable and directed towards that

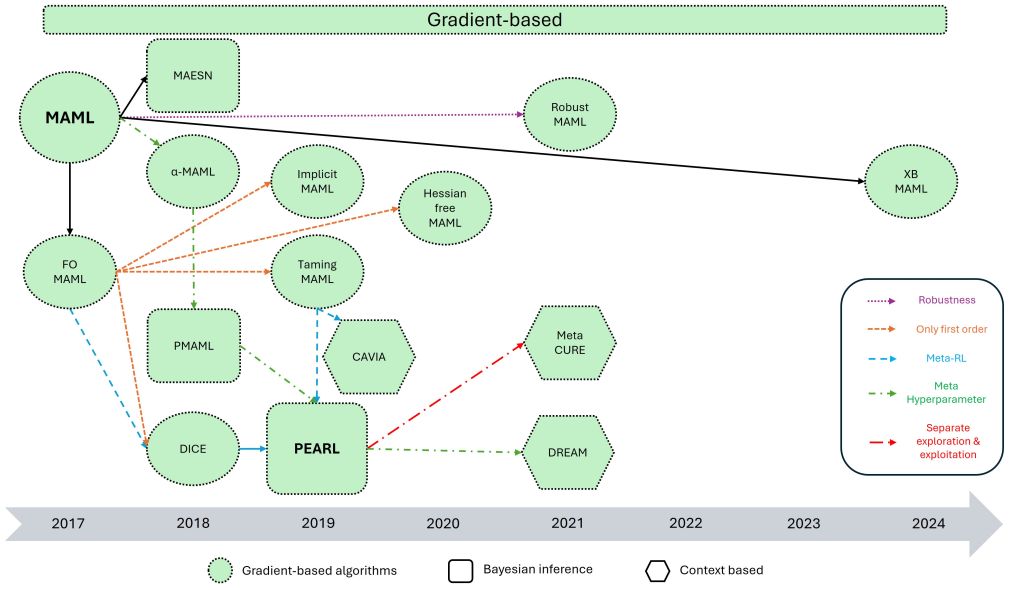


Figure 9: Timeline of gradient-based meta-learning algorithms from 2017 to 2024. The figure illustrates the chronological development of gradient-based methods and highlights the landmark algorithms MAML and PEARL, along with their subsequent developments. Arrows represent conceptual influences, showing which key features (i.e., meta-hyperparameter-learning, only first-order gradients, robustness, or separate exploration and exploitation) are incorporated into newer methods. The landmark algorithms are in bold type. Arrow color and style encode the nature of each contribution, while the shape of the nodes reflects the algorithmic categorization (bayesian inference shown in rectangles and context-based in pentagons).

optimum, but a small learning rate also would suffice to approach it within a few steps. This is the main idea of gradient-based meta-learning.

4.1.1 Model-Agnostic Meta-Learning

Model-Agnostic Meta-Learning (MAML), introduced by Finn et al. (2017a), combines the meta-learning paradigm with the idea of a priori placing the parameters θ in an area around the optimal solution $\theta*$. It is the foundational gradient-based meta-learning algorithm and one of the earliest and simplest landmarks in the evolution of meta-learning and meta-RL broadly applicable to any machine-learning problem where the loss function is sufficiently smooth to compute gradients. Hence, the MAML paradigm has successfully been adapted to various machine learning domains such as Neural Architecture Search Wang et al. (2022a), regression Sen & Chakraborty (2024), multi-object tracking Chen & Deng (2024), time-series classification Wang et al. (2024a), semi-supervised learning Boney & Ilin (2018), and reinforcement learning Rakelly et al. (2019), while getting applied to various fields like medicine Tian (2024), Alsaleh et al. (2024), Tian et al. (2024), Ranaweera & Pathirana (2024), Naren et al. (2021), biomass energy production Zhang et al. (2025), or fault diagnoses in bearings Lin et al. (2023).

Various works have examined the properties of the MAML paradigm. Assuming that MAML's model architecture consists of a sufficiently deep neural network with ReLU activations in the hidden layers and that the loss function is either cross-entropy or mean squared error (MSE), it has been proven that MAML serves as a universal function approximator for any training set X^i_{train} and test set X^i_{test} within an arbitrary task T_i Finn & Levine (2018). Furthermore, it has been shown Wang et al. (2022a) that MAML converges linearly to a global optimum under MSE loss when using an over-parameterized deep neural network along with a SGD optimizer like Adam Kingma & Ba (2017). This implies that a deeper model, with at least one hidden layer, is necessary, even if single tasks can be addressed using a shallower or linear model (as verified by e.g. Arnold et al. (2021)). Empirical findings support that MAML's meta-training typically achieves almost zero training loss and 100% training accuracy (indicating global convergence) when appropriate

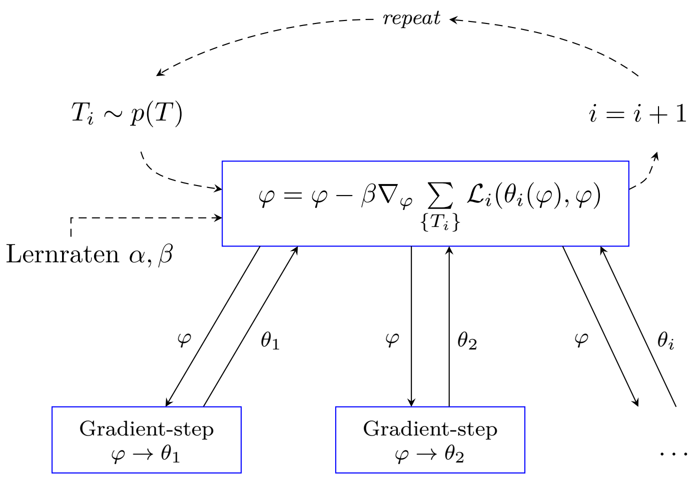

repeat
$T_i \sim p(T)$
$i = i + 1$
$\varphi = \varphi - \beta \nabla_\varphi \sum_{\{T_i\}} \mathcal{L}_i(\theta_i(\varphi), \varphi)$
Lernraten α, β
φ
θ_1
φ
θ_2
φ
θ_i
Gradient-step $\varphi \to \theta_1$
Gradient-step $\varphi \to \theta_2$
. . .

Figure 10: Model-Agnostic Meta-Learning

hyperparameters are utilized (refer to Wang et al. (2022a) for more details). However, it has been noted in Raghu et al. (2020) that MAML's meta-training leads to effective feature representations rather than a rapidly adaptable prior. During task-specific training, the initial layers of the underlying network exhibit minimal changes, suggesting that the fundamental feature representations remain stable.

Since MAML's universal approximation theorem Finn & Levine (2018) requires the size of the underlying neural network to scale with the number n of tasks, the number of different tasks to be learned simultaneously in the MAML paradigm is naturally bounded by the fact that the model size cannot be infinitely increased. Additionally, the time for learning the tasks scales with the complexity of tasks Parisotto et al. (2019) as more complex tasks require for better exploration and exploitation. Moreover, the optimal parameters $\theta_1^*, \theta_2^*, \ldots$ for the different tasks $T_1, T_2, \ldots$ are assumed to be sufficiently close to each other, so that a general prior φ (i.e. global optimum) can be found from which each optimal solution can be sufficiently well approximated within K gradient descent steps. But as parameter vectors are likely to be high-dimensional, this might be a rather strong assumption restricting the generalization capabilities of the method and limiting it to relatively simple parametric task distributions, i.e. to families of tasks whose task dynamics or loss functions can be parameterized with a low-dimensional set of parameters following a simple distribution. These drawbacks explain why the memory-based baseline algorithms presented in sections 4.2 and 4.3 outperform MAML, especially in terms of generalization and OOD performance. It is the main motivation for the development of MAML extensions like Robust MAML Nguyen et al. (2021) or XB-MAML Lee & Yoon (2024) particularly aiming to generalize to task slightly out of distribution.

As the original MAML paradigm consists of second-order gradients that are normally computationally expensive, Finn et al. (2017a) suggested First-Order (FO) MAML, a more efficient approach simply ignoring second-order derivatives. Although this modification does not guarantee global convergence Fallah et al. (2020) (what was the motivation for further modifications like Hessianfree MAML Fallah et al. (2020) or Implicit MAML Rajeswaran et al. (2019)) MAML in its first-order approximation is much easier to implement and computationally less expensive than memory-based meta-learning algorithms presented in the subsequent sections Finn et al. (2017a).

MAML Paradigm

The original MAML paradigm is directly derived from the general meta-learning paradigm presented in section 2.1. On the meta-level, the main idea is to place the prior φ in a region that is promising for all tasks from the distribution $p(T)$. so that, starting each task-specific fine-tuning from that prior, i.e. setting $\theta_i(\varphi) = \varphi$ for all θ_i, K gradient descent steps in fine-tuning have the maximum effect on task-specific performance.

Meta-Training

As the meta-training of MAML directly follows the general meta-training scheme presented in section 2.1, It consists of an outer, meta-level stage and inner, task-specific learning. One iteration of the corresponding meta-training scheme with $K = 1$ epochs of gradient descent is shown in figure 10. At the beginning of each iteration, a batch of tasks $\{T_i\}_{i=1,...,N} \sim p(T)$ are sampled from the task distribution $p(T)$. Starting with φ, each task-specific parameters $\theta_i = \varphi$ are updated in K epochs of task-specific gradient descent. The performance of the updated parameters θ_i' on the respective test set X^i_{test} is fed into the gradient descent update step of φ on the meta-level in order to optimize φ with respect to the fine-tuning performance resulting from it. The corresponding meta-testing is analogously derived by testing task-specific gradient descent learning with fixed φ. It completely follows the scheme presented in Figure 4 and is hence not shown here.

On the meta-level, the meta-loss $\mathcal{L}_{\text{meta}}$ is defined as the sum over all task-specific losses $\mathcal{L}_i$ on the respective test sets X^i_{test}:

$$\mathcal{L}_{\text{meta}} := \sum_{i=1}^{N} \mathcal{L}_i\big(\big(\theta_i',\ \varphi,\ X^i_{\text{test}}\big)$$

with θ_i' denoting the task-specific parameters resulting from the individual inner update steps equation 19. The respective optimization problem equation 4 is then solved with K stochastic gradient descent steps. For $K = 1$ this is

$$\varphi' = \varphi - \beta \nabla_\varphi \mathcal{L}_{\text{meta}} \tag{18}$$

with pre-defined meta-learning rate β. At task-level, the inner optimization problem equation 5 is also solved by K gradient descent steps of the form

$$\theta_i' = \varphi - \alpha \nabla_\varphi \mathcal{L}_i\left(\varphi, X^i_{\text{train}}\right), \tag{19}$$

with a pre-defined learning rate α, [5] and φ as the parameters of the corresponding model (e.g. weights of a neural network) solving the task T_i on the training set X^i_{train}. The loss $\mathcal{L}_i$ depends on the specific task and its training set X^i_{train}. It, generally, is assumed to be calculated with all K examples $x \in X^i_{\text{train}}$ at once, although this is no algorithmic necessity.

Meta-RL

Adjusting MAML to the meta-RL paradigm is straightforward. Besides the fact that task-specific training and test sets must be represented by K episodes of MDP-specific agent-environment interaction, the general MAML structure of embedding task-specific gradients in a meta-gradient descent scheme remains the same. However, the gradient descent steps equation 19 and equation 18 require gradients of the task-specific losses, which are the task-specific expected returns equation 12 in meta-RL. But these value functions are not differentiable due to unknown environment dynamics and the effect the agent has on them.

The policy currently used for the minimization of the value function equation 12 is updated throughout that minimization process which implicitly changes the value equation 12, i.e. the original MAML meta-RL paradigm is on-policy. Consequentially, only policy-gradient methods Sutton et al. (1999) can be used for MAML. How exactly the respective policy gradient is getting computed depends on the respective RL algorithm used on the task-level. However, many state-of-the-art standard RL algorithms like PPO Schulman et al. (2017) or TRPO Schulman (2015) utilize an off-policy learning scheme to learn from many policies at once, i.e. different copies of the agent gather experience in parallel with a fixed behavior policy in order to learn from these experiences a posteriori. This motivates the PEARL Rakelly et al. (2019) algorithm, the landmark off-policy modification of MAML presented in the next subsection.

For the meta-update of φ, the policy gradients $\nabla_\varphi \pi_{\theta_i'}$ must be calculated with the updated parameters θ_i' and on the respective task-specific test episodes X^i_{test}. In this way, the underlying policy-gradient algorithm is extended to meta-level in order to also update φ with respect to the performance of the fine-tuned parameters θ_i' on test sets X^i_{test}. From a conceptual point of view, this meta-gradient step corresponds to training a

[5]Finn et al. (2017a) state that the learning rate α can also be meta-learned, what is adressed in α-MAML Behl et al. (2019)

meta-policy π_φ to perform as good as possible after K epochs of task-specific gradient descent as the sum over task-specific losses $\sum_{i=1}^{N} \mathcal{L}_i$ defines the meta loss $\mathcal{L}_{\text{meta}}$. However, this meta-update scheme still consists of second-order terms that are difficult to derive. This is the motivation for meta-RL-specific MAML extensions like Taming-MAML Liu et al. (2019) or DICE Foerster et al. (2018).

4.1.2 Efficient Off-Policy Meta-RL

The identification of the current task T_i i.e., of the hidden dynamics of the corresponding MDP M_i, plays a crucial role in learning the task-specific optimal behavior $\pi_{\theta_i^*}$. During MDP-specific learning, this identification can only be based on the experiences collected so far in the particular MDP. Such experience is called context and many different meta-learning algorithms were developed to incorporate it into their paradigm Zintgraf et al. (2019), Ni et al. (2022), Ben-Iwhiwhu et al. (2022), Zhang et al. (2021), Fu et al. (2021), Deng et al. (2024). This way, the transitions and rewards of the current MDP are reconstructed faster so that the agents can tailor their MDP-specific policies accordingly. In example 4, the current context is defined by every part of the current racing track the racer has visited so far. The more of the track it visits, the more informative the context becomes and thus the easier the current track can be fully identified.

The current context is only a representation of the MDP dynamics $\mathbb{T}_i$ and R_i as the actions taken by the current agent determine which experiences it gathers. Hence, there is uncertainty about which MDP the agent is currently acting in. However, this uncertainty can be incorporated into the meta-RL framework by utilizing the Bayesian Reinforcement Learning paradigm introduced at the end of section 2.2. The context is used to a posteriori update the belief over the current MDP M_i, while the corresponding posterior distribution is used to inform the decisions of the agent. Such algorithms are called task-inference methods (see e.g. Beck et al. (2023b)), while algorithms like CAVIA Zintgraf et al. (2019) that utilize the current context without Bayesian inference are called context-based. Task-inference methods exist in the family of gradient-based meta-learners Ni et al. (2022), Ben-Iwhiwhu et al. (2022), Gupta et al. (2018), Zhang et al. (2021), as well as in the broader family of memory-based ones Zintgraf et al. (2020), Bing et al. (2024).

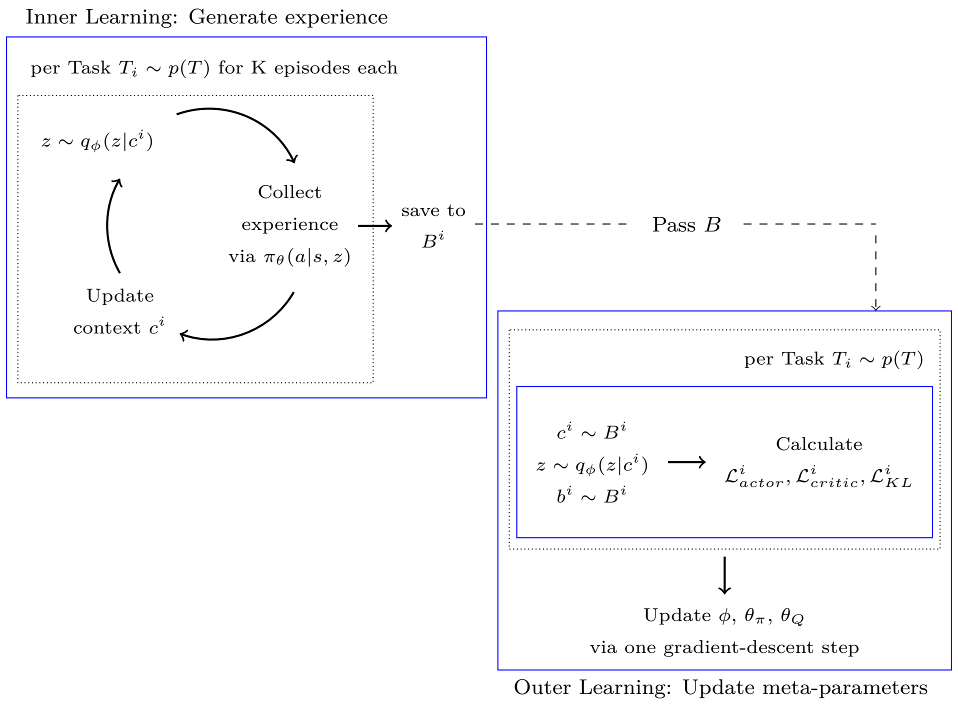
Efficient Off-Policy Meta-RL (PEARL) Rakelly et al. (2019) is such an off-policy learning framework based on Bayesian inference. It is the landmark gradient-based task-inference algorithm outperforming MAML as well as the simplest memory-based landmark RL^2 (presented in the next section) in terms of few-shot performance, adaption speed and sample-efficiency on the Mujoco benchmark Todorov et al. (2012), and is hence a frequently used benchmark for later meta-RL algorithms in the development timeline presented here. While other extensions of MAML like α-MAML Behl et al. (2019) or PMAML Finn et al. (2018) only slightly modify the original MAML paradigm, PEARL's meta-learning paradigm looks quite different. It not only includes task inference (as also done by e.g. Finn et al. (2018) or Grant et al. (2018)), but also reformulates the meta-training scheme to off-policy learning while staying on-policy within MDP-specific learning. This modification is likely to be the main reason for PEARL's superior adaption speed and sample efficiency. However, PEARL struggles in more complex MDP distributions as well as in MDPs where rewards are sparse. This motivates the more recent and sophisticated gradient-based algorithms DREAM Liu et al. (2021), MAESN Gupta et al. (2018) and MetaCure Zhang et al. (2021) that are more tailored to optimally explore at the beginning of MDP-specific adaption.

PEARL Paradigm

The main idea of PEARL is to infer a latent context variable z_i via Bayesian updates in each MDP-specific RL episode that encodes a MDP M_i based on the current context

$$c_t^i := \{(a_j, s_j, r_j, s_j')\}_{j=1,\dots t} \quad \text{with} \quad t = 1, \dots, K * h. \tag{20}$$

This context variable can either represent the value function equation 12 or other function approximations (model-free approach), or the transition $\mathbb{T}_i$ and reward R_i of the current MDP M_i (model-based approach). As at the end of section 2.2 only model-based BRL is introduced, z_i is assumed to represent MDP dynamics here.



Inner Learning: Generate experience
per Task $T_i \sim p(T)$ for K episodes each
$z \sim q_\phi(z|c^i)$
Collect experience via $\pi_\theta(a|s,z)$
Update context c^i
save to B^i
Pass B
per Task $T_i \sim p(T)$
$c^i \sim B^i$
$z \sim q_\phi(z|c^i)$
$b^i \sim B^i$
Calculate $\mathcal{L}^i_{actor}, \mathcal{L}^i_{critic}, \mathcal{L}^i_{KL}$
Update ϕ, θ_π, θ_Q via one gradient-descent step
Outer Learning: Update meta-parameters

Figure 11: PEARL's meta-learning - At the stage of inner learning, K episodes of experience are collected in each MDP and stored in a replay buffer for each task individually. The context variable z is updated at the beginning, and the task-specific context c^i at the end of each episode. After K episodes, minibatches are sampled from the replay buffer to update meta-parameters φ, θ_π and θ_Q.

The context variable z_i is sampled from a prior $p(z_i|c_i)$ with respect to the current context c^i_t at the beginning of each RL episode. The prior p is represented by a neural network p_{φ_z} with weights φ_z learned on a meta-level. Analogous to MAML, the corresponding meta-update of φ_z is one meta-gradient step of the form equation 18. However, the corresponding meta loss is modified to contain a KL-divergence penalty in order to stabilize training.

According to the BRL paradigm presented in section 2.2, the policy $\pi(\cdot|z_i)$ depends on the context variable z_i, i.e., on the belief in which task it is currently acting. It is also represented by a neural network with weights φ learned on the meta-level. Since the policy π_φ implicitly adapts to a particular M_i through the context, a MDP-specific gradient descent update of the policy weights φ is not required. This way, PEARL's paradigm is off-policy: The policy π_φ is updated on the meta-level while on the MDP-specific level only the belief is updated based on the gathered experience. The Bayesian update equation 16 is thereby approximated by the neural network architecture of p_{φ_z} that takes the context c^i_t as input.

The off-policy paradigm of PEARL enables off-policy algorithms for inner learning. In their original publication, Rakelly et al. (2019) used Soft-Actor-Critic Haarnoja et al. (2018). However, this is rather a design choice than a necessity of the PEARL paradigm, but it incorporates a critic Q_{φ_Q} into the framework whose parameters φ_Q must also be meta-learned via meta-gradient descent.

Meta-Training and Meta-Testing

Figure 11 shows the meta-training scheme of PEARL. At the beginning of each RL episode of MDP-specific training, z_i is sampled from the prior distribution $q_{\varphi_z}(\cdot|c_i)$ depending on the current context c_i. In each of the K episodes, the policy $\pi_\varphi(\cdot|z_i)$ is used to gather one RL episode of experience c^i_h (with horizon h denoting the end of the episode). The experience is stored in a MDP-specific replay buffer β_i so that it can be used for loss calculation for the meta-gradient descent steps of φ, φ_z and φ_Q after the K episodes end.

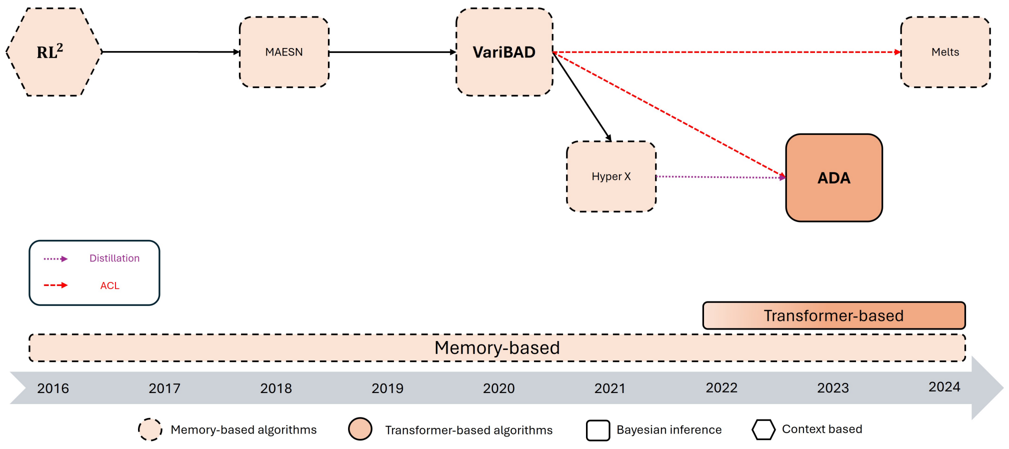


Figure 12: Timeline of memory-based meta-learning algorithms from 2016 to 2024. The figure illustrates the chronological development of memory-based methods and highlights landmark algorithms such as RL^2 and VARIBAD, along with their subsequent developments. Arrows represent conceptual influences, showing which key features (distillation and ACL) are incorporated into newer methods. Because of the connection to ADA, there is a colored transition shown to transformer-based algorithms. The landmark algorithms are in bold type. Arrow color and style encode the nature of each contribution, while the shape of the nodes reflects the algorithmic categorization (Bayesian inference is shown in rectangles and context-based in pentagons).

In order to calculate these losses, minibatches are sampled from the replay buffers β_i of all training MDPs $\{M_i\}_{i=1}^T$. Then, meta-parameters φ, φ_z and φ_Q get updated via one meta-gradient descent step of the form equation 18, with meta-losses for actor π_φ and critic Q_{φ_Q} as the sum over the respective MDP-specific losses defined by the Soft-Actor-Critic algorithm Haarnoja et al. (2018).

As mentioned above, the meta-loss for prior network parameters φ_z contains a KL-divergence regularizer

$$\mathcal{L}^i_{\mathrm{KL}} := \mathrm{D}_{\mathrm{KL}}(p(z_i|c^i_t), r(z_i))$$

with contexts c^i_t sampled from the replay buffers β_i, that keeps the prior network relatively close to a baseline distribution r e.g. the normally distributed prior chosen at the beginning of meta-training (see Sajid et al. (2021) for a more detailed explanation) that is a hyperparameter of the PEARL algorithm. Additionally, the MDP-specific critic losses are used. This is due to the fact, that the belief of which MDP the agent is in, directly affects value functions like equation 12, that are often used as the critic (see e.g. Schulman et al. (2017), Schulman (2015), Haarnoja et al. (2018)). The corresponding meta-loss for φ_z is

$$\mathcal{L}^{\varphi_z}_{\mathrm{meta}} := \sum_{T_i} \left(\mathcal{L}^i_{\mathrm{critic}} + \mathcal{L}^i_{\mathrm{KL}}\right).$$

A MDP-specific test set X^i_{test} does not exist. All meta-losses are calculated on the experience collected throughout learning and stored in the replay buffers β_i, because the context variables z_i are adapted after each RL episode. Nevertheless, it is rather a design choice than a theoretical necessity to not sample another K episodes with z_i or p_{φ_z} being fixed.

The meta-testing scheme of PEARL is analogous to the general meta-testing scheme presented in section 2.2 i.e., meta-parameters are fixed. However, since inner learning means to implicitly update the prior network $p_{\varphi_z}(z_i|c^i_t)$ wrt. the context c and resample z_i at the beginning of each episode respectively, the policy network $\pi_\varphi(\cdot|z_i)$ is only implicitly updated, too. This is analogous to meta-training, and similarly, no test episodes are sampled for performance evaluation.

4.2 Memory-based Meta-RL

In meta-RL, all MDPs are drawn from a meta-distribution $p(M)$ over MDPs that are different but similar. Memorizing these similarities is likely to facilitate the adaption to a new MDP which requires a similar strategy. Additionally, RL problems are sequential so that the ability to process sequences instead of single transitions (s, a, r, s') helps to cover time-dependencies and to solve RL problems where rewards are sparse, i.e. rewards are zero for most transitions. Altogether, this gives rise to the idea of utilizing memory architectures in RL and meta-RL respectively.

The simplest memory architecture is that of a Recurrent Neural Network (RNN). RNNs maintain a hidden state h that acts like a memory mechanism by storing information about past experiences. Although more recent advances like TRMRL Melo (2022) utilize transformer networks as the more sophisticated memory structure, following the timeline of memory-based meta-RL, this section discusses the landmark RNN-based meta-RL algorithms, starting with RL^2 as the earliest and simplest landmark in section 4.2.1.

RNNs are Turing-complete universal computers Tur. However, even in standard RL, RNN-based agents are very sensitive to architectural choices like their initialization, the choice of the underlying RL algorithm or the length of the context that is supposed to be captured by the hidden state h Ni et al. (2022). The latter is, moreover, highly MDP-specific Ni et al. (2022), which limits the flexibility of RNN-based meta-RL algorithms and makes them struggle to generalize between tasks Beck et al. (2023a), Ben-Iwhiwhu et al. (2022). This motivates additional modifications like using a hyper network Beck et al. (2023a) to learn initializations of the MDP-specific RNN-weights as well as incorporating Bayesian inference in the RNN-based meta-learners. The former was shown to significantly boost the corresponding RNN performance in Beck et al. (2023c) outperforming the latter ones. However, as the latter case yields a new family of meta-RL algorithm the most famous landmark RNN-based Bayesian meta-RL algorithms is presented in section 4.2.2.

4.2.1 Fast Reinforcement Learning via Slow Reinforcement Learning

The RL^2 algorithm in Duan et al. (2016) is the very first landmark presented in this work. As such, it is used as the main reference for black-box meta-RL in several surveys Beck et al. (2023b), Vettoruzzo et al. (2024), while it serves as a benchmark for all gradient-based meta-learners presented in the previous section as well as algorithms like VariBAD Zintgraf et al. (2020) or TRMRL Melo (2022) discussed later on. The main idea is to simply combine a standard RL algorithm with a RNN-based agent. This way, the general knowledge φ is - In contrast to gradient-based meta-RL methods - represented by the hidden state of the RNN. The corresponding RNN weights are only updated once every K episodes of MDP-specific learning, which aligns with the notion of "slow learning", while the activation of the RNNs hidden states serves as an implicit fast adaption scheme throughout these K episodes. In the original publication, TRPO Schulman (2015) was used as the standard RL algorithm, although RL^2 performs even better with more sophisticated RL algorithms Rakelly et al. (2019) like Proximal Policy Optimization (PPO) Schulman et al. (2017).

The RL^2 algorithm was experimentally shown to be Bayes-optimal on simple benchmark tasks like N-arm bandit problems Mikulik et al. (2020). The experiments indicated the Bayes-optimal solution to be a fix-point of meta-training. However, the task distributions used in Mikulik et al. (2020) were manually constructed so that calculating a bayes-optimal solution was tractable. Consequentially, the performance on broader, more complex task distributions is not guaranteed. The usage of only one single model is likely to limit the generalization capability of the algorithm Ben-Iwhiwhu et al. (2022). as the RNN has to cover all architectures theoretically needed for the optimal strategy π_i^* of each MDP M_i of the MDP distribution $p(M)$ Ben-Iwhiwhu et al. (2022). This results in poor asymptotic (i.e. long run) performance Gupta et al. (2018). In fact, RL^2 is outperformed by the gradient-based meta-RL algorithms presented in the previous section, which gave rise to more sophisticated methods like VariBAD Zintgraf et al. (2020) whose objective is explicitly tailored to optimize towards Bayes-optimality. The next paragraphs, nevertheless, describe RL^2 under the general meta-RL paradigm of section 2.2 since it is the foundational approach for memory-based meta-RL.

Paradigm

RL^2 can be differentiated into an inner and an outer learning stage Duan et al. (2016), although there, in fact, is only one particular policy π_θ represented by a RNN interacting with different MDPs through an environment. General knowledge is implicitly collected by updating the weights of the RNN once every K episodes, while the MDP-specific behavior is only updated implicitly via the model activations following from the currently gained experience. This way, the goal to bayes-optimally trade-off exploration and exploitation is just given implicitly, too. The objective to maximize rewards over K episodes results in the implicit need to explore the current MDP (i.e. its hidden dynamics and reward structure) as fast as possible in order to maximize rewards in the subsequent episodes.

Meta-Training and Meta-Testing

The meta-training scheme of RL^2 significantly differs from the general meta-RL meta-training. One meta-episode is defined as interacting with a particular MDP M_i for K episodes, i.e. as collecting experience X^i_{train} of the form equation 14. The experiences X^i_{train} are then used to update the policy parameters θ in the standard RL manner. For standard RL algorithms like TRPO Schulman (2015) or PPO Schulman et al. (2017) this means to sample single transitions (s, a, r, s') from the replay buffer β_i containing all experience in X^i_{train}, and use the corresponding batch for updating θ in several epochs of gradient descent.

Throughout MDP-specific learning, the parameters θ remain unchanged. Instead, the hidden state h_t of the RNN gets updated by the gathered experience. It is reset at the beginning of each MDP-specific training, but gets preserved throughout the K episodes - what is the major difference to simply modifying standard RL algorithms with a RNN agent. Conditioning the policy $\pi_\theta(\cdot|h_t)$ on that hidden state, one yields a MDP-specific adaption scheme like the inner learning of PEARL, since the hidden state h_t functions as the representation z of the context c. A MDP-specific training-test split does not exist as the inner learning performance is validated throughout MDP-specific training.

In contrast to its meta-training, the meta-testing scheme of RL^2 is analog to the meta-RL meta-testing presented in section 2.2. The policy parameters θ are held fixed, while the hidden state gets adapted during K episodes of MDP-specific learning. Again, this is very similar to PEARL, where only context variables z are adapted throughout meta-testing.

4.2.2 Variational Bayes-Adaptive Deep RL

As mentioned in the previous subsection, a single network - even a RNN - is naturally limited in the number of strategies it can represent. This aligns with the observation that actor-critic RNN-based standard RL already performs better when two distinct networks for actor and critic are used Ni et al. (2022). At the same time, the exploration-exploitation dilemma leads to poor reward during exploration that is nevertheless required for exploiting the gained knowledge in order to maximize future rewards. Altogether, this motivates the idea of separating the exploration process from the later exploitation, e.g., by distinct exploration and exploitation networks as presented in Stadie et al. (2018) for modifying MAML and RL^2, or as done in other works like Norman & Clune (2024), Liu et al. (2021) or Zhang et al. (2021).

Optimal exploration means collecting the most meaningful context possible, while exploitation requires optimally extracting the most essential information from that context in order to adjust the behavior respectively and maximize future rewards. How well the environment was explored is thus determined by the quality of the context Norman & Clune (2024), while a good representation of that context is substantial for good exploitation. This motivates leveraging context encoders as e.g., done in gradient-based algorithms like PEARL Rakelly et al. (2019), DREAM Liu et al. (2021), MAESN Gupta et al. (2018) or MetaCure Zhang et al. (2021), to inform the policy network with meaningful context embeddings z. However, it may be desirable to process the whole context sequence rather than single transitions, i.e. utilizing memory architectures for such context encoders.

This subsection discusses Variational Bayes-Adaptive Deep RL (VariBAD) Zintgraf et al. (2020), the landmark RNN-based task-inference method on the development path leading to the Adaptive Agent (ADA). The main idea of VariBAD is to incorporate a context-based Variational Auto-Encoder (VAE) into the memory-based meta-RL framework and abuse its context embeddings to represent task uncertainty and reconstruct MDP dynamics. The former is used to incorporate task uncertainty directly into the decision-making of the policy, while the latter enables the usage of model-based RL at task level. This way, VariBAD combines ideas from context-based task-inference (like presented for PEARL in section 4.1.2) with the RNN-based meta-RL framework introduced in the previous subsection.

VariBAD outperforms RL^2 in both a dynamic grid world and the Mujoco benchmark Todorov et al. (2012). exhibiting almost Bayes-optimal exploratory behavior that leads to higher returns during adaption Zintgraf et al. (2020). While both methods can adapt within a single episode, RL^2 demonstrates slower and less stable learning, and other meta-learning methods like PEARL perform poorly after just one episode of adaption Zintgraf et al. (2020). However, the Bayes-optimality of VariBAD has neither been shown nor proven across more sophisticated task distributions,and Melo (2022) even shows VariBAD to be unstable in out-of-distribution test tasks within the Mujoko environment which indicates worse generalization than implied by the original publication. This has led to modifications of VariBAD such as HyperX Zintgraf et al. (2021), which enhances meta-exploration by distilling policies from random networks to improve performance in distributions of sparsely rewarded MDPs, and MELTS Bing et al. (2024) specifically designed for effective zero-shot performance on non-parametric task distributions through task-clustering. Both these methods utilize modifications also used in the Adaptive Agent (i.e. distillation and task-selection), which are hence described in section 4.4.

VariBAD Paradigm

Besides a common policy π_θ, the main component of VariBAD is the VAE architecture that consists of a context-encoder RNN f_φ, a decoder g_ψ and a variational distribution $p_\varphi(z|c_t^i)$ that is the output of the RNN encoder. The VAE decoder is an ordinary neural network $g_{\varphi_{R,T}}$ with two heads, one representing a common transition function $\mathbb{T}$, the other a common reward R. The common policy $\pi_\theta(\cdot|s, p_\varphi(z|c))$ is an ordinary deep neural network parameterized by θ. Both, the common policy and the decoder receive the output distribution $p_\varphi(z|c)$ from the VAE encoder, so that, in practice, both these components are blocks of the same architecture, where the encoder output distribution p_φ is fed into the policy head π_θ and the model dynamics head g_ψ. This way, the VAE distribution p_φ represents "task uncertainty" Zintgraf et al. (2020), i.e. the belief over which MDP the policy is currently acting in.

The parameters φ, θ and ψ are altogether updated on the meta-level, while inner learning corresponds to the implicit change in the VAE distribution $p_\varphi(\cdot|c_t^i)$ resulting from updated context c_T^i. Using this task belief as an input, the common policy as well as the common dynamics R and $\mathbb{T}$ are implicitly adapted to the current, initially unknown MDP M_i. Consequentially, VariBAD is an on-policy algorithm, although the possibility of an off-policy modification of VariBAD was already mentioned in the original publication. Such a modification is presented in Dorfman et al. (2021). It is based on leveraging offline data collected by MDP-specific agents in order to extract meta-knowledge for meta-updates of φ, θ and ψ respectively.

Meta-Training and Meta-Testing

After carefully adjusting the meta-RL paradigm of sections 2.2 and 2.2 to the architecture of VariBAD, adapting the meta-training and meta-testing schemes is straightforward. For each of N MDPs $\{M_i\}_{i=1}^N$ sampled from $p(M)$, K episodes of MDP-specific training are absolved during which the meta-parameters φ, θ and ψ are held fixed. The resulting trajectories are stored in the replay buffer $\beta := \{c_t^i\}_{t=1,\dots,h_i}^{i=1,\dots,N}$ which is used for meta-level gradient descent updates. The corresponding meta-loss $\mathcal{L}_{\text{meta}}$ consists of two major components:

1. The RL loss simply is the accumulated loss over all trajectories, and
2. the ELBO loss, that itself consists of two components, a reconstruction loss for the decoder and a KL-divergence keeping the encoder output p_φ close to its update. [6]

The task inference happens at the output stage of the VAE encoder, since the encoder output distribution p_φ functions as the meta-learned prior. The corresponding posterior is updated in each timestep via variational inference. However, the corresponding scheme is rather technical and hence not shown here. Instead, the curious reader is referred to the original work in Zintgraf et al. (2020) or other works about task inference like Sajid et al. (2021).

[6]For a detailed derivation of the ELBO-loss and all corresponding components, the curious reader is referred to Sajid et al. (2021).

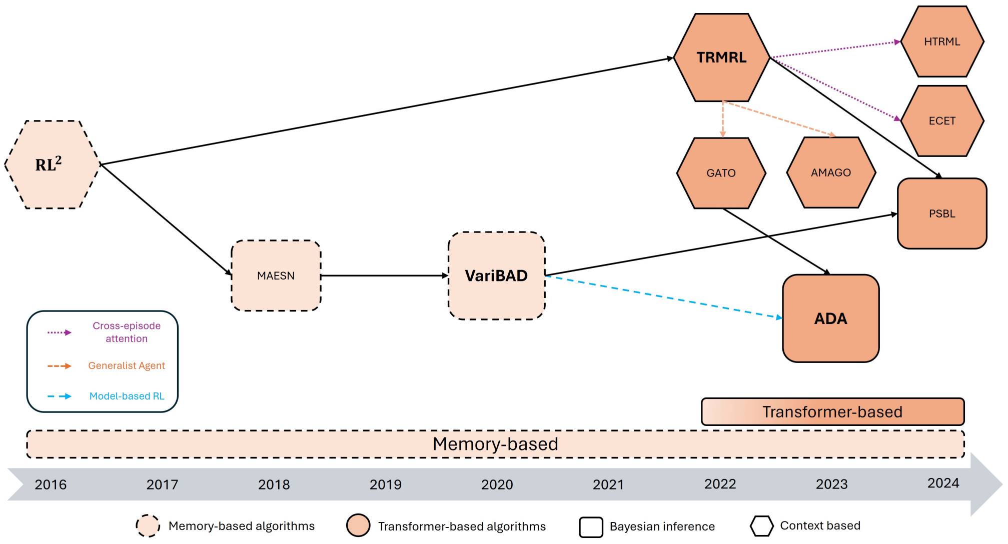


Figure 13: Timeline of transformer-based meta-learning algorithms from 2022 to 2024. The figure illustrates the chronological development of transformer-based methods and highlights landmark algorithms such as TRMRL and ADA, along with their subsequent developments. Arrows represent conceptual influences, showing which key features (cross-episode attention, generalist agent, and model-based RL) are incorporated into newer methods. Due to the development of RNN-based landmark algorithms RL^2 and Varibad, there is a colored transition to the transformer-based ones shown. The landmark algorithms are in bold type. Arrow color and style encode the nature of each contribution, while the shape of the nodes reflects the algorithmic categorization (Bayesian inference is shown in rectangles and context-based in pentagons).

The meta-testing of VariBAD is analogous to that of PEARL, i.e. all meta-parameters are fixed, while only the task uncertainty $p(z|c)$ is updated. This fully corresponds to the meta-testing described in section 2.2.

4.3 Transformer-based Meta-RL

The transformer network architecture was first proposed by Vaswani et al. (2017) as a language-to-language model i.e. as a model translating one language into another. As induced by its title, Vaswani et al. (2017) showed that "attention is all you need" by outperforming all other state-of-the-art language-to-language algorithms by a good margin. Motivated by its success, many modifications of the vanilla transformer"' have been developed to further boost performance for example the TransformerXL Dai et al. (2019) used in the Adaptive Agent. Simultaneously, modifications of the initial transformer were developed for tasks different from language-to-language translation, e.g., Vision Transformers for computer vision Han et al. (2023), visual Li et al. (2024a), Xiao et al. (2023) and semantic segmentation Strudel et al. (2021), forecasting Lim et al. (2021), and RL Hu et al. (2024). This way, transformer architectures got applied to various fields like medicine Xiao et al. (2023), e.g., for RNA prediction Chaturvedi et al. (2025) fault detection e.g., in manufactoring Wu et al. (2023), bio informatics Zhang et al. (2023a), automated driving Hu et al. (2022), and many other application fields. In RL, transformers were utilized to close the gap between simulation and reality in locomotion control Lai et al. (2023), act beneficial in various IOT (Internet of Things) environments like Smart-Homes or industrial buildings Rjoub et al. (2024) or better learn from a vast amount of (sub)-optimal demonstrations Liu & Abbeel (2023), Lee et al. (2023), Reed et al. (2022). Since each of the various extensions and modifications of the vanilla transformer builds up on the attention mechanism proposed in the original publication Vaswani et al. (2017), the original transformer architecture is broadly presented and discussed in Appendix 7.1, while this section focuses on the utilization of transformer architectures for meta-RL.

4.3.1 Transformers are Meta-Learners

Instead of translating a sentence from one language into another, a context-based RL agent (like PEARL or VariBAD) "translates" "' context trajectories of the form equation 20 or their context embeddings into the current action a_t. Such a "translation" does not require a decoder as the current action can directly be derived from a policy head. However, it is likely that the single transitions of a context trajectory have strong dependencies to each other or the time they occured, so that it is additionally desirable to contextualize them respectively. In example 4, it is relevant, when a certain turn was made, and, more importantly, particular actions a few time steps ahead might influence the current one. For example, how aggressively one speeds up on a straight road has a strong impact on how to navigate the next sharp turn. This motivates to abuse the transformer architecture whose self-attention mechanism provides the possibility to contextualize such trajectories, by modifying simple gradient-based algorithms like MAML (see section 4.1.1) respectively, e.g., in order to detect faults in bearings Li et al. (2024b), improve short-term load forecasting in scenarios where different clients require for federated learning Feng et al. (2025), or forecast stock prices Chen et al. (2025).

However, transformers additionally have a demonstrable long-term memory of up to 1500 steps into the past Ni et al. (2023), what particularly motivates to also utilize transformers in memory-based meta-learning Melo (2022), Grigsby et al. (2023), Xu et al. (2024), Shala et al. (2024b), Shala et al. (2024a). TRMRL Melo (2022) is such a transformer architecture tailored for meta-RL. It is the last landmark transformer-based meta-RL algorithm on the development path to the Adaptive Agent, and was shown to fulfill all necessary properties of a meta-learner Melo (2022). The main idea of TRMRL is to replace the RNN in the RL^2 paradigm with a transformer agent that is, additionally, initialized in a certain way to improve initial training stability.

Although PEARL demonstrates slightly better sample efficiency in locomotion tasks due to its off-policy learning approach, TRMRL outperforms the landmarks presented in sections 4.1 and 4.2 - namely MAML, PEARL, RL2 with PPO as an inner learner, and VariBAD - in the Mujoco environment Todorov et al. (2012) wrt. sample-efficiency, adaption speed and particularly OOD performance. This superior performance is even more significant in the more sophisticated Meta-World environment Yu et al. (2020). The only exception is Mujoco tasks with high task uncertainty. In these tasks, VariBAD slightly outperforms TRMRL, since it explicitely models task uncertainty. This motivates combining a transformer-based meta-RL agent with Bayesian inference like in PSBL Xu et al. (2024), or particularly with the VariBAD paradigm as done by the Adaptive Agent. While the latter achieves human-like performance, the former is shown to outperform TRMRL as well as other baselines like MAML, RL^2, PEARL or VariBAD in terms of asymptotic and OOD performance on sparse and dense reward tasks - an observation that becomes even clearer when the distribution shift increases Xu et al. (2024).

TRMRL Paradigm

TRMRL extends RL^2 by a transformer architecture i.e. by self-attention, while otherwise completely mimicking the RL^2 paradigm. This means, that the model weights are also only updated on the meta-level, i.e. after K MDP episodes, while inner learning corresponds to the model activations triggered by the collected experience. With respect to these two stages, Melo (2022) show that

1. the (inner learning) fast adaption of the multi-head attention serfs as a task representation mechanism, since each self-attention head contextualizes the embeddings of the context c_t^i collected in the current MDP M_i up until the current time step t.
2. the meta-learned model weights function as a long-term memory by design, through which the respective MDP can be identified and acted upon accordingly.

They hypothesis that any MDP can be represented by a distribution over working memories:

$$z_t = \sum_{t=1}^{T} \alpha_t f_\theta(o_t) \tag{21}$$

where o_t is a single transition of the form $(s_t, a_t, r_{t+1}, s_{t+1})$, N is the trajectory length, f_θ is a learnable, arbitrary linear function, and α_t are coefficients summing up to 1. However, the context scores of the last transformer encoder sub-block naturally have the form of such a task representation When given the context trajectory c_t^i of the form equation 20 as input at a particular time point t: Reformulating the self-attention softmax term (see equation 22), i.e. the softmax of the query-key multiplication, for a particular transition o_t and the multiplication with the corresponding value vector v_t, one yields representations for the coefficients α and the linear function f in equation 21 - the former through rewriting the softmax term, the latter by a sum over entries of the value vector. For the rather technical full derivation of that MDP representation property, the motivated reader is referred to the original publication Melo (2022). For the scope of this work, it is sufficient to conclude that this MDP representation property creates the implicit objective to "learn to make a distinction among the tasks in the embedding space" Melo (2022), i.e. the implicit goal to learn how to identify tasks through the experience collected within then. In fact, Melo (2022) additionally proof that any transformer self-attention layer minimizes the task uncertainty based on the current context c_t^i, i.e. that every transformer sub-block finds the best representation of the information gathered so far. In each transformer sub-block, the previous representation of c_t^i is recombined through the dense layers before the next self-attention layer persists in the form of a task representation. This way, the representation of c_t^i gets episodically refined, a process that "resembles a memory reinstatement operation" Melo (2022), i.e. an operation to reactivate long-term memory in order to identify the current task and act accordingly.

Meta-Training and Meta-Testing

In RL, transformers are often unstable during training Hu et al. (2024), especially in the begining Melo (2022). And while more sophisticated algorithms like AMAGO Grigsby et al. (2023) stabilize training by incorporating off-policy learning into their transformer architecture, TRMRL adresses this problem right at the beginning of meta-training by utilizing T-Fixup initialization Huang et al. (2020), an initialization scheme specially designed to avoid learning rate variation and layer normalization during early training (whose usefulness is particularly shown in the ablation studies of Melo (2022)). Besides that, the meta-training and meta-testing schemes of TRMRL are directly derived from RL^2.

4.4 The Adaptive Agent

Transformers still struggle with "credit assignment" Ni et al. (2023), i.e., with assigning current actions to rewards later on in the planning horizon - a problem that even increases when the data complexity is high Ni et al. (2023). This particularly motivates to utilize model-based RL, where the agent can base its decisions on the future effect the model predicts them to have on the environment. The only landmark algorithm on the development path presented in the previous subsections, that utilizes model-based RL for inner learning, is VariBad. It additionally models task uncertainty into its paradigm and, hence, Melo (2022) already identify the possibilities of combining the VariBad paradigm with a transformer architecture. The Adaptive Agent (ADA) Team et al. (2023) is this transformer-based modification of VariBad. It combines a large transformer architecture with self-supervised learning techniques in order to develop a generalist agent applicable for a vast number of "downstream tasks".

ADA builds up on other, similar approaches aiming for a generalist agent, i.e., RL foundation model, like GATO Reed et al. (2022) or the work of Team et al. (2021). It achieves human-like few- and zero-shot performance (in terms of generalization and adaption speed) for single and multi-agent tasks on an even more complex and dynamic extension of the XLAND environment Team et al. (2021), a "vast open-ended task space with sparse rewards" for single and multi-agent tasks. In multi-agent downstream tasks, the different Adaptive Agents clearly share labor and actively cooperate to improve their reward Team et al. (2023), while in single-agent tasks it is able to use one single episode of expert demonstrations in order to significantly improve performance. This makes ADA superior to the work of Team et al. (2021) and other generalist agents like GATO Reed et al. (2022) which exclusively learns from demonstrations. However, ADA is not generally superior to human-level performance as there are single- and multi-agent downstream tasks neither humans nor ADA are able to solve.

ADA's transformer memory length, i.e., the length of the context c used as model input, as well as its model and task pool sizes, scale demonstrably well, particularly when being scaled together with each other Team et al. (2023). For example, increasing the task pool, i.e., sampling a larger number of distinct tasks from a task distribution with more complex tasks, always has a positive effect on the agent's median performance, especially in the few-shot setting, But this effect is even bigger when model size or memory are also increased. The same holds for the model size and the transformer memory length: Larger models as well as models with longer context windows obtain better median performance in any K-shot setting, particularly when the task pool also increases. On a log-log plot, the effect of increasing the model or context window size additionally scales roughly linearly with the number K of shots, but only if the task pool is sufficiently large (otherwise the model overfits the small task pool). All these findings, additionally hold true in the 20th quantile i.e., in the 20,which indicates a more stable training beyond the human-like generalization and adaption performance, and highlights why Bommasani et al. (2021) identify scaling as the key factor of what makes foundation models so powerful.

Although ADA is capable of handling varying context lengths, it, like most RL transformers, assumes the dimension of observations to be constant throughout tasks. However, as ADA particularly utilizes an additional observation encoder prior to its transformer architecture, it may suffice to exchange only that encoder when observation spaces change. This is the motivation of AMAGO Grigsby et al. (2023), a more flexible approach that particularly utilizes an observation encoder mapping each context to a fixed-sized representation, so that the encoder is the only required architectural change across experiments. This way, AMAGO is particularly designed to incorporate off-policy actor-critic RL in combination with dynamic, long-term contexts, so that it can train agents from multiple rollouts in parallel from more distinct, long-term contexts, which makes it also applicable for multi-task RL.

Since approaches like AMAGO, GATO and ADA clearly are foundation models, they are computationally expensive during meta-training Shala et al. (2024b), what motivates hierarchical transformer architectures like HTRMRL Shala et al. (2024a) or ECET Shala et al. (2024b) that use additional self-attention sub-blocks to process cross-episode data in order to further increase sample efficiency and reduce model complexity. In contrast to the intra-episode self-attention sub-blocks utilized in TRMRL, AMAGO and ADA, these cross-episode blocks set whole episodes of experience in context to each other, so that the context mechanism of self-attention is also leveraged on episode level.

4.5 ADA Paradigm

Analogous to the VariBad paradigm presented in section 4.2.2, the ADA paradigm consists of a policy π_θ, a decoder g_ψ and a context encoder f_φ whose output is the variational distribution $p_\varphi(z|c_t^i)$ which is fed into policy and decoder in order to represent task uncertainty. However, the different components are modified:

- Prior to its transformer architecture, a ResNet architecture preprocesses the visual 3D-observations received from the XLand environment.

- The memory-based context-encoder is represented by a transformer architecture. [7] More precisely, the transformerXL-architecture Dai et al. (2019), a demonstrably more efficient and powerful modification of the vanilla transformer whose architecture allows it to model long-range dependencies.

- The architecture following the transformerXL mimics the Muesli algorithm Hessel et al. (2021), a computationally efficient, demonstrably powerful RL approach that combines model-based and model-free RL to leverage the advantages of both. This requires a critic network estimating the value equation 12 of the current state as well as an environment model for planning that predicts policy and critic network outputs along with future rewards.

In addition to these modifications, the number K of adaption episodes is varied ($k = \{1, 2, \ldots, 6\}$) and given as an additional input to the context encoder.

[7] Note that theoretically any transformer architecture can be used that is suitable for RL.

Meta-Training and Meta-Testing

In contrast to TRMRL, Team et al. (2023) do not use any particular initialization scheme. They rather use layer normalization and gating techniques to stabilize the transformer during training. Besides that, the VariBad meta-training scheme is extended by two self-supervised learning techniques:

1. An automatic curriculum (ACL) selecting tasks on the frontier of the current agent's capabilities for efficient learning, and
2. distillation for kickstarting the training process.

Both these techniques are described in more detail in the subsequent subsections.

4.5.1 Automated Curriculum Learning:

Autocurriculum-Learning (ACL) is a paradigm in meta-RL, which leverages the idea from human learning that training tasks should not be too hard or too easy for the current state of learning. The main idea originates from curriculum learning (CL), Dansereau (1978), where curricula are hand-crafted to guide the learning process of a standard learner (e.g. a neural network) on a single task. Such curricula were successfully applied in several supervised learning tasks like NLP, computer vision, medicine, NAS and RL - see e.g. Wang et al. (2022b), Soviany et al. (2022), Gupta et al. (2021b), for a detailed overview. The main idea of ACL is to automatically prioritize (sub) tasks at the frontier of the agent's capabilities during training, instead of manually selecting them. This ensures that the agent is consistently challenged and can progressively improve its skills Portelas et al. (2020), so that sample efficiency and generalization performance can be significantly improved Dennis et al. (2021), Jiang et al. (2021), Jabri et al. (2019), Wang et al. (2022b).

There exist several ACL methods (see e.g. Portelas et al. (2020), for example, Teacher-Student learning Matiisen et al. (2020), one of the earliest approaches, where a second model (the teacher) is simply used to select sub-tasks for the standard learner (the student) during training. For meta-RL, there also exist various techniques, e.g.

- Simple, but quite static methods like domain randomization (e.g., Volpi et al. (2021)), where environments are generated randomly, but independent of the current policy's capabilities.
- More flexible approaches like min-max-adversarial (e.g., Adv), where environments are created adversarial, i.e., to challenge the agent as much as possible.
- sophisticated paradigms like min-max-regret, e.g. Protagonist Antagonist Induced Regret Environment Design Dennis et al. (2021), where an adversary aims to maximize the regret between the protagonist (the agent) and its antagonist (an oponent assumed to be optimal).

For the latter, the regret is defined as the difference between the value of the protagonist's and the antagonist's action. This way, the adversary selects the simplest tasks the protagonist is not yet able to solve Dennis et al. (2021), what makes this method superior to min-max-adversarial, which, due to its objective, tends to generate tasks too difficult for the agent or even unsolvable Dennis et al. (2021). However, ADA leverages and compares two other ACL methods during meta-training, namely

1. NOOB-Filtering Team et al. (2021) maintains a control policy that takes no actions. For each newly sampled task, ADA and the control policy are rolled out for ten episodes. A task is selected for meta-training, if ADA's performance is neither too good nor too bad, the variance among trials is sufficiently high between episodes to indicate proper learning, and The control policy performs sufficiently bad to indicate the usefulness of proper decision making.
2. Robust Priorized Level Replay (PLR) Jiang et al. (2021) maintains a "level buffer" of tasks with a high learning potential, from which a task is sampled with a probability of p, while, otherwise, a new task is randomly sampled from the task distribution $p(T)$. The learning potential is estimated by

the "regret" of the task. For ADA, this regret is approximated by several fitness metrics like policy and critic losses. A task is added to the level buffer, if its regret is higher than those of the other level buffer tasks, so that the level buffer constantly gets refined.

In comparison to meta-training with uniformly sampled tasks, both, no-op filtering and PLR, improve ADA's sample efficiency and generalization as well as its few- and zero-shot performance. Additionally, PLR is shown to slightly outperform NOOB filtering, especially when the number K of adaption episodes is higher. However, any other approach from the ones described above might work just as well.

4.5.2 Distillation

The notion of distillation was first proposed by Hinton et al. (2015). It generally referees to model compression techniques transferring knowledge from a large, pre-trained Teacher model T [8] to a smaller, more efficient student model S Rusu et al. (2016). Hence, distillation can be referred to as a transfer-learning approach (see section 3.1 or Mansourian et al. (2025)). Classically, the student is trained to learn from the teacher's demonstrations, i.e. it learns to predict the teacher's softmax output distribution (in the discrete case) that is smoothened by choosing a sufficiently high softmax temperature parameter in order to yield soft targets Rusu et al. (2016), Mansourian et al. (2025), Hinton et al. (2015). Analogously, in RL, one of the best-known distillation approaches is policy distillation Rusu et al. (2016), an approach also known as immitation learning, where the student policy is trained to imitate the teacher's action distribution. Other approaches are value function distillation and dynamic reward-guided distillation (see Xu et al. (2025) for further details).

Generally, the two main questions in selecting a knowledge distillation scheme are, what knowledge to transfer and which architectures to respectively choose for teacher and student models Mansourian et al. (2025), Gou et al. (2021). In Team et al. (2023), dynamic reward-guided distillation is utilized, where the student policy is guided by the teacher policy rather than particularly mimicking it. They implement an additional distillation loss throughout the first four billion meta-updates that consists of the KL-divergence between student and teacher policy action distributions as well as a L^2-regularization term like in Schmitt et al. (2018). This way, the student policy can also explore states different from those visited by the corresponding teacher model, which is trained from scratch with a model size of 23 million parameters.

For the sake of comparison, Team et al. (2023) train four different models: A large (256 million parameters) and a small (23 million parameters) model both, with and without, distillation. They show the larger distilled model to strongly outperform the smaller one as well as the large ordinary model, while The larger ordinary model gets outperformed by the smaller one. This not only indicates a significant positive effect of distillation on model performance but also show the necessity of distillation for largely scaled (foundation) models.

5 Discussion

Meta-Learning describes the paradigm of learning how to learn. But, as reality poses a vast amount of potential tasks, the question necessarily arises, which general knowledge to acquire. The earlier algorithms presented and discussed in the previous sections primarily focused on meta-learning, the skill to take decisions in RL environments by pre-training model weights (MAML), basing the decisions on the currently explored context (PEARL) or the current context-based belief over which task they are acting in (Varibad) with manually designed architecture modifications, while memory-based agents, especially those based on transformer architectures, are meta-learners just by definition (as e.g., showed by Melo (2022)). Although they were not specifically designed to be meta-learners, these models learn how to learn just by emergence. The path towards general intelligence quite often looks like that Bommasani et al. (2021): While earlier approaches are hand-designed for particular purposes, later ones yield more general, unsupervised and unguided intelligence, learning different skills emergently. The consequence is a shift towards "homogeneity" i.e., the usage of one model architecture for several different tasks or task distributions Bommasani et al. (2021). In other words, as soon as a model architecture generalizes better, it is used for more general purposes, a

[8]It is also possible to use multiple teachers for different subtasks Schmitt et al. (2018). However, these teachers can be seen as one teacher model consisting of different task-specific components.

development that is highlighted by the timeline of meta-RL developments and visualized by the respective figures 9, 12 and 13. Even within the transformer-based development path, this shift towards homogeneity can be clearly observed: While early transformer-based meta-RL architectures like TRMRL require for hand-designed architectures for different observation spaces, generalist agents like ADA and AMAGO abstract this problem by observation space specific encoder blocks. Observing this scheme of developments (like e.g., outlined by Bommasani et al. (2021) and Clune (2020)), one gets an idea of how future developments might look like: As, from a theoretical point of view, any kind of skill can be meta-learned, meta-learning does not have to focus particularly on the task-specific adaption itself. The easiest example is meta-learning hyperparameters, - an approach that occurred quite early in the timeline of meta-learning developments with algorithms like α-MAML Behl et al. (2019) or PEARL Rakelly et al. (2019). But even far more complicated skills like neural-architecture search or designing curricula or populations of evolutionary algorithms can be meta-learned. However, before such approaches are discussed in the last of the following subsections, the relevance of meta-learning as well as the comparability and applicability of its algorithms are outlined in the next subsections.

Relevance of Meta-Learning

With the shift from manually designed, relatively simple to understand meta-learning algorithms like MAML or PEARL to largely scaled, blackbox foundation models like GATO Reed et al. (2022), ADA Team et al. (2023) or AMAGO Grigsby et al. (2023), the question arises whether the paradigm of meta-learning is actually still of relevance. Largely scaled foundation models from companies like DeepMind, OpenAI, or Meta can be applied to a vast amount of in- and out-of-distribution downstream tasks of different types - as e.g., done for the LAMA model in Rentschler & Roberts (2025) for RL tasks - without even thinking about data quality, distribution shifts or the meta-training paradigm. However, foundation models require a vast amount of computation power, even when only being rolled out for downstream tasks without model updates. During training, they were shown to necessitate scaling of their model size, the task pool and the task complexity Team et al. (2023), Bommasani et al. (2021). Although techniques like distillation and ACL can decrease the model size and boost the training progress, the amount required for (meta-)training and testing is not available for all researchers, companies and people. Moreover, in recent years there has been a significant shift from publicly available models with revealed source code trained on open source data towards secretly training models of unknown sizes and shapes on largely collected data sets that are hidden from the public - and, as such, hidden from public control and revision. But how models behave highly depends on the data they were trained on, so that this privatisation of model training comes with a high risk of unexpected, harmful behaviour Bommasani et al. (2021) along with the societal risk of an increasing gap between institutions with a high amount of computational power and data and others who can not utilize large resources.

Modern research can, regardless of the available computation power, focus on different niches of application Togelius & Yannakakis (2024) as well as on developing more efficient architectures like Hierarchy transformers Shala et al. (2024a), Shala et al. (2024b), or on gaining a better understanding of blackbox models, their behaviour, the reasons for their failures, and the societal and ethical impact their application has Togelius & Yannakakis (2024). The latter can support the development of even better general problem solvers Clune (2020), so that collaboration between university researchers and Big Tech companies is probably beneficial for both Togelius & Yannakakis (2024), Bommasani et al. (2021).

The Need for Specialized Meta-Learners

In application, agents must often be deployed on small edge devices with a very limited amount of computation power that does not allow for generalist agents but might benefit from meta-trained models that adapt fast. Such problems are likely to cover only a very narrow, low-dimensional task distribution that does not require generalist agents. In other words, often largely scaled transformer-based architectures go far beyond what is needed to solve a particular problem. This is probably why most meta-learning applications are still based on simple, sample-efficient gradient-based algorithms like MAML although they have worse OOD performance (see e.g., the applications for medicine Tian (2024), Alsaleh et al. (2024), Tian et al. (2024), Ranaweera & Pathirana (2024), Naren et al. (2021), biomass energy production Zhang et al.

(2025), or fault diagnoses Lin et al. (2023)). However, understanding the advantages and drawbacks of the various components of the landmark algorithms presented in the previous section can support applied researchers in manually designing the best fitting meta-learner for their particular problem. For example, autonomous driving requires extremely fast adaption, very good zero-shot performance and high robustness and reliability, while sample-efficiency does not really matter, especially during training. In such an application, gradient-based algorithms are most likely the wrong choice, while memory-based Bayesian inference learners like VariBAD or ADA are much more promising. Medical applications, in contrast, often have only a very limited amount of data per patient, so that they particularly require good sample-efficiency. For such applications, gradient-based meta-learners are a good first choice, as they are simple and sample-efficient.

Comparability

There is a lack of comparability among the landmark algorithms presented in the timeline of the previous section. The utilized performance measures differ between the different works and are often not clearly defined, an issue this work aims to address by defining general performance measures in section 2.1. In addition, several different benchmark environments were utilized. The development of the latter seems to follow the development of meta-learning algorithms itself: Early landmarks like MAML, RL^2 or PEARL were mainly tested on simple grid-world environments, the Atari benchmark Bellemare et al. (2013), or the Mujoco engine Todorov et al. (2012). But Fakoor et al. (2020) identified that those benchmarks are too simple to evaluate actual adaption, which motivates more generalistic, complex and sparsely rewarded environments like Meta-World Yu et al. (2020), Crafter Hafner (2021) or XLand Team et al. (2021). This lack of homogeneity makes it difficult for researchers and developers to compare different algorithms and modifications. However, this is a general problem of ongoing research and development and can, hence, only be tackled by several empirical studies.

The three Pillars of the Path towards General Intelligence

The development path towards human-like or even superior generally intelligent agents consists of three pillars Clune (2020):

- Meta-Learning Algorithms that can be viewed as the "evolution of intelligence" itself,
- meta-NAS Pereira (2024), Hospedales et al. (2022) creating the "physical body" of the gained "intelligence", and
- the generation of more meaningful, complex and challenging environments that function as the "world" the "physical body" of the learned "intelligence" is acting in.

While research in the meta-learning community mainly focussed on the learning algorithms presented in the timeline of the previous section, and a few meta-NAS approaches can be found in Pereira (2024), meta-environment research is sparse Clune (2020). However, as the growing complexities and capabilities of recent developments come with the necessity of a further increase in complexity for benchmark environments, most recent attention is shifted towards this research field by combining meta-RL with self-supervised learning techniques. This is - along various other foundation model approaches - reflected in the ADA paradigm, where initial learning is guided by a teacher model (Distillation) and an automatic curriculum (ACL) selecting tasks on the frontiers of the agent's capabilities. The latter is, however, a skill that can be meta-learned, so that a (teacher) model learns to create a personalized curriculum for any student model Portelas et al. (2021), Xu et al. (2023). The corresponding Meta-ACL paradigm is inspired by classroomscenarios, where a teacher teaches several students at once, ideally with respect to their personal capabilities and special needs.

Additionally, meta-design of environments can be inspired by examining the evolution on earth Clune (2020). Instead of developing algorithms generating open-ended environments like XLand Team et al. (2021) and meaningful rewards, one can aim for agents that explore on their own, i.e. out of curiosity Alet et al. (2020) or through intrinsic reward (see e.g., Zhang et al. (2021)) in environments providing a high variety of different niches to adapt to. The former corresponds to techniques like novelty search Lehman & Stanley (2011), while algorithms like quality diversity Pugh et al. (2016) can be utilized to generate high-variety environments.

Considering the current work of Big Tech companies, a very likely future path of meta-learning and foundation model development will be towards combining these pillars respectively. The most recent landmark of that kind is the Evolution Transformer Lange et al. (2024) of DeepMind. It combines the pillars of meta-learning and meta-NAS by learning how to design the population of RL agents (individual) for evolutionary RL algorithms. In other words, on top of the evolutionary approach (e.g., survival of the fittest) the Evolution Transformer learns how the population of agents (breed of creatures) must be designed to have the highest chance of survival as a whole, i.e. it focusses on the continued existence of the species rather than of maximization the performance of single individual.

Combining the different pillar results, along with a potential increase in model capability, in several potential ethical, societal and economic concerns that are very difficult to predict. When models are "taught to play god" - even in a very limited world like the currently existing environments - it is essential to track their behavior, the basis on which they make their decisions and the harm they can potentially cost from different views like law Chan et al. (2024), ethics, economy and various others. In this respect, the future development might be examined and analysed analogously to that of foundation models in Bommasani et al. (2021), where a huge group of various researchers of several domains and backgrounds collaborate in order to identify potential risks, challenges and benefits of the current shift towards largely scaled foundation models.

6 Conclusion

In this comprehensive survey, the timeline of meta-reinforcement learning developments from the foundational algorithms MAML and RL^2 to the Adaptive agent, the current state of the art, was provided. Through detailed mathematical formalizations of meta-learning and Meta-RL paradigms, alongside a clear distinction from related paradigms like transfer learning and multi-task learning this survey addressed the lack of detailed formalism within the literature and layed the groundwork for understanding the paradigms and training schemes of the landmark algorithms presented along the timeline of meta-RL developments. By the exploration of landmark developments in Meta-RL, the significance of memory-based approaches, particularly the paradigm shift towards transformer architectures and largely scaled foundation models through the usage of self-supervised learning techniques like distillation and automated curriculum learning, was underscored and set in context to the development path towards more generalist agents Looking forward, the insights into the path towards general intelligence offered in this survey focused on the trend of enhancing the automated generation of environments through meta-learning of self-supervision and evolutionary approaches. Besides a valuable insight, this highlighted the constantly growing need for collaborative, interdisciplinary teams of researchers that permanently analyze the behavior of generalist agents and their ethical implications for our society.

References

Multitask Learning. *Machine Learning*, 28(1). ISSN 1573-0565. doi: 10.1023/A:1007379606734. URL `https://doi.org/10.1023/A:1007379606734`.

New York, NY, USA.

Ibrahim Ahmed, Marcos Quinones-Grueiro, and Gautam Biswas. Complementary Meta-Reinforcement Learning for Fault-Adaptive Control. *Annual Conference of the PHM Society*, 12(1):8, November 2020. ISSN 2325-0178, 2325-0178. doi: 10.36001/phmconf.2020.v12i1.1289. URL `http://arxiv.org/abs/2009.12634`. arXiv:2009.12634 [cs].

Jay Alammar. The illustrated transformer. `http://jalammar.github.io/illustrated-transformer/`, 2018. Accessed: 2025-07-01.

Ferran Alet, Martin F. Schneider, Tomas Lozano-Perez, and Leslie Pack Kaelbling. Meta-learning curiosity algorithms, March 2020. URL `https://ui.adsabs.harvard.edu/abs/2020arXiv200305325A`. ADS Bibcode: 2020arXiv200305325A.

Aqilah M. Alsaleh, Eid Albalawi, Abdulelah Algosaibi, Salman S. Albakheet, and Surbhi Bhatia Khan. Few-Shot Learning for Medical Image Segmentation Using 3D U-Net and Model-Agnostic Meta-Learning (MAML). *Diagnostics*, 14(12):1213, January 2024. ISSN 2075-4418. doi: 10.3390/diagnostics14121213. URL `https://www.mdpi.com/2075-4418/14/12/1213`. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

Elena Arcari, Maria Vittoria Minniti, Anna Scampicchio, Andrea Carron, Farbod Farshidian, Marco Hutter, and Melanie N. Zeilinger. Bayesian Multi-Task Learning MPC for Robotic Mobile Manipulation. *IEEE Robotics and Automation Letters*, 8(6):3222–3229, June 2023. ISSN 2377-3766. doi: 10.1109/LRA.2023.3264758. URL `https://ieeexplore.ieee.org/abstract/document/10093028`.

Sébastien Arnold, Shariq Iqbal, and Fei Sha. When MAML Can Adapt Fast and How to Assist When It Cannot. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pp. 244–252. PMLR, March 2021. URL `https://proceedings.mlr.press/v130/arnold21a.html`. ISSN: 2640-3498.

Marcos Barcina-Blanco, Jesus L. Lobo, Pablo Garcia-Bringas, and Javier Del Ser. Managing the unknown in machine learning: Definitions, related areas, recent advances, and prospects. *Neurocomputing*, 599: 128073, September 2024. ISSN 0925-2312. doi: 10.1016/j.neucom.2024.128073. URL `https://www.sciencedirect.com/science/article/pii/S0925231224008440`.

Jacob Beck, Matthew Thomas Jackson, Risto Vuorio, and Shimon Whiteson. Hypernetworks in Meta-Reinforcement Learning. In *Proceedings of The 6th Conference on Robot Learning*, pp. 1478–1487. PMLR, March 2023a. URL `https://proceedings.mlr.press/v205/beck23a.html`. ISSN: 2640-3498.

Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A Survey of Meta-Reinforcement Learning, January 2023b. URL `http://arxiv.org/abs/2301.08028`. arXiv:2301.08028 [cs].

Jacob Beck, Risto Vuorio, Zheng Xiong, and Shimon Whiteson. Recurrent Hypernetworks are Surprisingly Strong in Meta-RL. *Advances in Neural Information Processing Systems*, 36:62121–62138, December 2023c. URL `https://proceedings.neurips.cc/paper_files/paper/2023/hash/c3fa3a7d50b34732c6d08f6f66380d75-Abstract-Conference.html`.

Harkirat Singh Behl, Atılım Guneş Baydin, and Philip H. S. Torr. Alpha MAML: Adaptive Model-Agnostic Meta-Learning, May 2019. URL `http://arxiv.org/abs/1905.07435`. arXiv:1905.07435 [cs].

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, June 2013. ISSN 1076-9757. doi: 10.1613/jair.3912. URL `https://www.jair.org/index.php/jair/article/view/10819`.

Eseoghene Ben-Iwhiwhu, Jeffery Dick, Nicholas A. Ketz, Praveen K. Pilly, and Andrea Soltoggio. Context meta-reinforcement learning via neuromodulation. *Neural Networks*, 152:70–79, August 2022. ISSN 0893-6080. doi: 10.1016/j.neunet.2022.04.003. URL `https://www.sciencedirect.com/science/article/pii/S0893608022001368`.

Jinbo Bi, Tao Xiong, Shipeng Yu, Murat Dundar, and R. Bharat Rao. An Improved Multi-task Learning Approach with Applications in Medical Diagnosis. In Walter Daelemans, Bart Goethals, and Katharina Morik (eds.), *Machine Learning and Knowledge Discovery in Databases*, pp. 117–132, Berlin, Heidelberg, 2008. Springer. ISBN 978-3-540-87479-9. doi: 10.1007/978-3-540-87479-9_26.

Zhenshan Bing, Yuqi Yun, Kai Huang, and Alois Knoll. Context-Based Meta-Reinforcement Learning With Bayesian Nonparametric Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(10):6948–6965, October 2024. ISSN 1939-3539. doi: 10.1109/TPAMI.2024.3386780. URL `https://ieeexplore.ieee.org/abstract/document/10495171`.

Benjamin Bischke, Patrick Helber, Joachim Folz, Damian Borth, and Andreas Dengel. Multi-Task Learning for Segmentation of Building Footprints with Deep Neural Networks. In *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 1480–1484, September 2019. doi: 10.1109/ICIP.2019.8803050. URL `https://ieeexplore.ieee.org/abstract/document/8803050`. ISSN: 2381-8549.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the Opportunities and Risks of Foundation Models, August 2021. URL `https://ui.adsabs.harvard.edu/abs/2021arXiv210807258B`. ADS Bibcode: 2021arXiv210807258B.

Rinu Boney and Alexander Ilin. Semi-Supervised Few-Shot Learning with MAML. January 2018. URL `https://openreview.net/forum?id=r1n5Osurf`.

Mohamed Khalifa Boutahir, Abdelaaziz Hessane, Yousef Farhaoui, Mourade Azrour, Mbadiwe S. Benyeogor, and Nisreen Innab. Meta-Learning Guided Weight Optimization for Enhanced Solar Radiation Forecasting and Sustainable Energy Management with VotingRegressor. *Sustainability*, 16(13):5505, January 2024. ISSN 2071-1050. doi: 10.3390/su16135505. URL `https://www.mdpi.com/2071-1050/16/13/5505`. Number: 13 Publisher: Multidisciplinary Digital Publishing Institute.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html`.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, Tony Wang, Samuel Marks, Charbel-Raphaël Segerie, Micah Carroll, Andi Peng, Phillip Christoffersen, Mehul Damani, Stewart Slocum, Usman Anwar, Anand Siththaranjan, Max Nadeau, Eric J. Michaud, Jacob Pfau, Dmitrii Krasheninnikov, Xin Chen, Lauro Langosco, Peter Hase, Erdem Bıyık, Anca Dragan, David Krueger, Dorsa Sadigh, and Dylan Hadfield-Menell. Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback, September 2023. URL `http://arxiv.org/abs/2307.15217`. arXiv:2307.15217 [cs].

Alan Chan, Carson Ezell, Max Kaufmann, Kevin Wei, Lewis Hammond, Herbie Bradley, Emma Bluemke, Nitarshan Rajkumar, David Krueger, Noam Kolt, Lennart Heim, and Markus Anderljung. Visibility into AI Agents. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '24, pp. 958–973, New York, NY, USA, June 2024. Association for Computing Machinery. ISBN 9798400704505. doi: 10.1145/3630106.3658948. URL `https://dl.acm.org/doi/10.1145/3630106.3658948`.

Mayank Chaturvedi, Mahmood A. Rashid, and Kuldip K. Paliwal. Transformers in RNA structure prediction: A review. *Computational and Structural Biotechnology Journal*, 27:1187–1203, January 2025. ISSN 2001-

0370. doi: 10.1016/j.csbj.2025.03.021. URL https://www.sciencedirect.com/science/article/pii/S200103702500090X.

Shreyas Chaudhari, Pranjal Aggarwal, Vishvak Murahari, Tanmay Rajpurohit, Ashwin Kalyan, Karthik Narasimhan, Ameet Deshpande, and Bruno Castro da Silva. RLHF Deciphered: A Critical Analysis of Reinforcement Learning from Human Feedback for LLMs. *ACM Comput. Surv.*, June 2025. ISSN 0360-0300. doi: 10.1145/3743127. URL https://dl.acm.org/doi/10.1145/3743127. Just Accepted.

Jiayi Chen and Chunhua Deng. MAML MOT: Multiple Object Tracking Based on Meta-Learning. In *2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 4542–4547, October 2024. doi: 10.1109/SMC54092.2024.10831964. URL https://ieeexplore.ieee.org/abstract/document/10831964.

Shijie Chen, Yu Zhang, and Qiang Yang. Multi-Task Learning in Natural Language Processing: An Overview. *ACM Comput. Surv.*, 56(12):295:1–295:32, July 2024. ISSN 0360-0300. doi: 10.1145/3663363. URL https://dl.acm.org/doi/10.1145/3663363.

Yunzhu Chen, Neng Ye, Wenyu Zhang, Jiaqi Fan, Shahid Mumtaz, and Xiangming Li. Meta-LSTR: Meta-Learning with Long Short-Term Transformer for futures volatility prediction. *Expert Systems with Applications*, 265:125926, March 2025. ISSN 0957-4174. doi: 10.1016/j.eswa.2024.125926. URL https://www.sciencedirect.com/science/article/pii/S0957417424027933.

Jeff Clune. AI-GAs: AI-generating algorithms, an alternate paradigm for producing general artificial intelligence, January 2020. URL http://arxiv.org/abs/1905.10985. arXiv:1905.10985 [cs].

Can Cui, Teresa Wu, Mengqi Hu, Jeffery D. Weir, and Xiwang Li. Short-term building energy model recommendation system: A meta-learning approach. *Applied Energy*, 172:251–263, June 2016. ISSN 0306-2619. doi: 10.1016/j.apenergy.2016.03.112. URL https://www.sciencedirect.com/science/article/pii/S030626191630438X.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285.

DONALD Dansereau. The Development of a Learning Strategies Curriculum1. In HAROLD F. O'neil (ed.), *Learning Strategies*, pp. 1–29. Academic Press, January 1978. ISBN 978-0-12-526650-5. doi: 10.1016/B978-0-12-526650-5.50006-X. URL https://www.sciencedirect.com/science/article/pii/B978012526650550006X.

Marc Peter Deisenroth, Peter Englert, Jan Peters, and Dieter Fox. Multi-task policy search for robotics. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3876–3881, May 2014. doi: 10.1109/ICRA.2014.6907421. URL https://ieeexplore.ieee.org/abstract/document/6907421. ISSN: 1050-4729.

Qi Deng, Ruyang Li, Qifu Hu, Yaqian Zhao, and Rengang Li. Context-Aware Meta-RL With Two-Stage Constrained Adaptation for Urban Driving. *IEEE Transactions on Vehicular Technology*, 73(2):1567–1581, February 2024. ISSN 1939-9359. doi: 10.1109/TVT.2023.3312495. URL https://ieeexplore.ieee.org/abstract/document/10242012.

Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design, 2021.

Carlo D'Eramo, Davide Tateo, Andrea Bonarini, Marcello Restelli, and Jan Peters. Sharing Knowledge in Multi-Task Deep Reinforcement Learning, January 2024. URL http://arxiv.org/abs/2401.09561. arXiv:2401.09561 [cs].

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

Ron Dorfman, Idan Shenfeld, and Aviv Tamar. Offline Meta Reinforcement Learning – Identifiability Challenges and Effective Data Collection Strategies. In *Advances in Neural Information Processing Systems*, volume 34, pp. 4607–4618. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/248024541dbda1d3fd75fe49d1a4df4d-Abstract.html.

Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. RL^2: Fast Reinforcement Learning via Slow Reinforcement Learning, November 2016. arXiv:1611.02779 [cs, stat].

Worrawat Duanyai, Weon Keun Song, Poom Konghuayrob, and Manukid Parnichkun. Event-triggered model reference adaptive control system design for SISO plants using meta-learning-based physics-informed neural networks without labeled data and transfer learning. *International Journal of Adaptive Control and Signal Processing*, 38(4):1442–1456, 2024. ISSN 1099-1115. doi: 10.1002/acs.3758. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/acs.3758. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/acs.3758.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural Architecture Search: A Survey. *Journal of Machine Learning Research*, 20(55):1–21, 2019. ISSN 1533-7928. URL http://jmlr.org/papers/v20/18-598.html.

Linus Ericsson, Henry Gouk, Chen Change Loy, and Timothy M. Hospedales. Self-Supervised Representation Learning: Introduction, advances, and challenges. *IEEE Signal Processing Magazine*, 39(3):42–62, May 2022. ISSN 1558-0792. doi: 10.1109/MSP.2021.3134634. URL https://ieeexplore.ieee.org/abstract/document/9770283.

Andre Esteva, Brett Kuprel, Roberto A. Novoa, Justin Ko, Susan M. Swetter, Helen M. Blau, and Sebastian Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639): 115–118, February 2017. ISSN 1476-4687. doi: 10.1038/nature21056. URL https://www.nature.com/articles/nature21056. Publisher: Nature Publishing Group.

Rasool Fakoor, Pratik Chaudhari, Stefano Soatto, and Alexander J. Smola. Meta-Q-Learning, April 2020. arXiv:1910.00125 [cs, stat].

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. On the Convergence Theory of Gradient-Based Model-Agnostic Meta-Learning Algorithms. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, pp. 1082–1092. PMLR, June 2020. URL https://proceedings.mlr.press/v108/fallah20a.html. ISSN: 2640-3498.

Changsen Feng, Liang Shao, Jiaying Wang, Youbing Zhang, and Fushuan Wen. Short-term Load Forecasting of Distribution Transformer Supply Zones Based on Federated Model-Agnostic Meta Learning. *IEEE Transactions on Power Systems*, 40(1):31–45, January 2025. ISSN 1558-0679. doi: 10.1109/TPWRS.2024.3393017. URL https://ieeexplore.ieee.org/abstract/document/10531071.

Chelsea Finn and Sergey Levine. Meta-Learning and Universality: Deep Representations and Gradient Descent can Approximate any Learning Algorithm, February 2018. arXiv:1710.11622 [cs].

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR, August 2017a.

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-Shot Visual Imitation Learning via Meta-Learning. In *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 357–368. PMLR, October 2017b. URL `https://proceedings.mlr.press/v78/finn17a.html`. ISSN: 2640-3498.

Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic Model-Agnostic Meta-Learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, Brian Ichter, Danny Driess, Jiajun Wu, Cewu Lu, and Mac Schwager. Foundation models in robotics: Applications, challenges, and the future. *The International Journal of Robotics Research*, 44(5):701–739, April 2025. ISSN 0278-3649. doi: 10.1177/02783649241281508. URL `https://doi.org/10.1177/02783649241281508`. Publisher: SAGE Publications Ltd STM.

Jakob Foerster, Gregory Farquhar, Maruan Al-Shedivat, Tim Rocktaeschel, Eric Xing, and Shimon Whiteson. DiCE: The Infinitely Differentiable Monte Carlo Estimator. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1529–1538. PMLR, July 2018. URL `https://proceedings.mlr.press/v80/foerster18a.html`. ISSN: 2640-3498.

Vincent Francois-Lavet, Peter Henderson, Riashat Islam, Marc G. Bellemare, and Joelle Pineau. An Introduction to Deep Reinforcement Learning. *Foundations and Trends in Machine Learning*, 11(3-4):219–354, 2018. ISSN 1935-8237, 1935-8245. doi: 10.1561/2200000071. arXiv:1811.12560 [cs, stat].

Haotian Fu, Hongyao Tang, Jianye Hao, Chen Chen, Xidong Feng, Dong Li, and Wulong Liu. Towards Effective Context for Meta-Reinforcement Learning: an Approach based on Contrastive Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):7457–7465, May 2021. ISSN 2374-3468. doi: 10.1609/aaai.v35i8.16914. URL `https://ojs.aaai.org/index.php/AAAI/article/view/16914`. Number: 8.

Yuan Gao, Haoping Bai, Zequn Jie, Jiayi Ma, Kui Jia, and Wei Liu. MTL-NAS: Task-Agnostic Neural Architecture Search Towards General-Purpose Multi-Task Learning. pp. 11543–11552, 2020. URL `https://openaccess.thecvf.com/content_CVPR_2020/html/Gao_MTL-NAS_Task-Agnostic_Neural_Architecture_Search_Towards_General-Purpose_Multi-Task_Learning_CVPR_2020_paper.html`.

Brian Gaudet, Richard Linares, and Roberto Furfaro. Adaptive guidance and integrated navigation with reinforcement meta-learning. *Acta Astronautica*, 169:180–190, April 2020. ISSN 0094-5765. doi: 10.1016/j.actaastro.2020.01.007. URL `https://www.sciencedirect.com/science/article/pii/S0094576520300072`.

Hassan Gharoun, Fereshteh Momenifar, Fang Chen, and Amir H. Gandomi. Meta-learning Approaches for Few-Shot Learning: A Survey of Recent Advances. *ACM Comput. Surv.*, 56(12):294:1–294:41, July 2024. ISSN 0360-0300. doi: 10.1145/3659943. URL `https://dl.acm.org/doi/10.1145/3659943`.

Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, and Aviv Tamar. Bayesian reinforcement learning: A survey. *ArXiv*, abs/1609.04436, 2015. URL `https://api.semanticscholar.org/CorpusID:207179119`.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In Geoffrey Gordon, David Dunson, and Miroslav Dudík (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 315–323, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL `https://proceedings.mlr.press/v15/glorot11a.html`.

Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge Distillation: A Survey. *International Journal of Computer Vision*, 129(6):1789–1819, June 2021. ISSN 1573-1405. doi: 10.1007/s11263-021-01453-z. URL `https://doi.org/10.1007/s11263-021-01453-z`.

Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting Gradient-Based Meta-Learning as Hierarchical Bayes, January 2018. URL `http://arxiv.org/abs/1801.08930`. arXiv:1801.08930 [cs].

Jake Grigsby, Linxi Fan, and Yuke Zhu. AMAGO: Scalable In-Context Reinforcement Learning for Adaptive Agents. October 2023. URL `https://openreview.net/forum?id=M6XWoEdmwf`.

Jie Gui, Tuo Chen, Jing Zhang, Qiong Cao, Zhenan Sun, Hao Luo, and Dacheng Tao. A Survey on Self-Supervised Learning: Algorithms, Applications, and Future Trends. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9052–9071, December 2024. ISSN 1939-3539. doi: 10.1109/TPAMI.2024.3415112. URL `https://ieeexplore.ieee.org/abstract/document/10559458`.

Abhishek Gupta, Russell Mendonca, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Meta-Reinforcement Learning of Structured Exploration Strategies. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper/2018/hash/4de754248c196c85ee4fbdcee89179bd-Abstract.html`.

Abhishek Gupta, Justin Yu, Tony Z. Zhao, Vikash Kumar, Aaron Rovinsky, Kelvin Xu, Thomas Devlin, and Sergey Levine. Reset-Free Reinforcement Learning via Multi-Task Learning: Learning Dexterous Manipulation Behaviors without Human Intervention. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6664–6671, May 2021a. doi: 10.1109/ICRA48506.2021.9561384. URL `https://ieeexplore.ieee.org/abstract/document/9561384`. ISSN: 2577-087X.

Kashish Gupta, Debasmita Mukherjee, and Homayoun Najjaran. Extending the Capabilities of Reinforcement Learning Through Curriculum: A Review of Methods and Applications. *SN Computer Science*, 3(1):28, October 2021b. ISSN 2661-8907. doi: 10.1007/s42979-021-00934-9. URL `https://doi.org/10.1007/s42979-021-00934-9`.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/haarnoja18b.html`.

Danijar Hafner. Benchmarking the Spectrum of Agent Capabilities. December 2021. URL `https://openreview.net/forum?id=76iypkcozLU`.

Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, Zhaohui Yang, Yiman Zhang, and Dacheng Tao. A Survey on Vision Transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):87–110, January 2023. ISSN 1939-3539. doi: 10.1109/TPAMI.2022.3152247. URL `https://ieeexplore.ieee.org/abstract/document/9716741`.

Kai He, Nan Pu, Mingrui Lao, and Michael S. Lew. Few-shot and meta-learning methods for image understanding: a survey. *International Journal of Multimedia Information Retrieval*, 12(2):14, June 2023. ISSN 2192-662X. doi: 10.1007/s13735-023-00279-4. URL `https://doi.org/10.1007/s13735-023-00279-4`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *CoRR*, abs/1502.01852, 2015. URL `http://arxiv.org/abs/1502.01852`.

Matteo Hessel, Ivo Danihelka, Fabio Viola, Arthur Guez, Simon Schmitt, Laurent Sifre, Theophane Weber, David Silver, and Hado Van Hasselt. Muesli: Combining Improvements in Policy Optimization. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 4214–4226. PMLR, July 2021. ISSN: 2640-3498.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-Learning in Neural Networks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5149–5169, September 2022. ISSN 1939-3539. doi: 10.1109/TPAMI.2021.3079209. URL `https://ieeexplore.ieee.org/abstract/document/9428530`. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

Shengchao Hu, Li Shen, Ya Zhang, Yixin Chen, and Dacheng Tao. On Transforming Reinforcement Learning With Transformers: The Development Trajectory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):8580–8599, December 2024. ISSN 1939-3539. doi: 10.1109/TPAMI.2024.3408271. URL `https://ieeexplore.ieee.org/abstract/document/10546317`.

Zhongxu Hu, Yiran Zhang, Yang Xing, Yifan Zhao, Dongpu Cao, and Chen Lv. Toward Human-Centered Automated Driving: A Novel Spatiotemporal Vision Transformer-Enabled Head Tracker. *IEEE Vehicular Technology Magazine*, 17(4):57–64, December 2022. ISSN 1556-6080. doi: 10.1109/MVT.2021.3140047. URL `https://ieeexplore.ieee.org/abstract/document/9694454`.

Xiao Shi Huang, Felipe Perez, Jimmy Ba, and Maksims Volkovs. Improving Transformer Optimization Through Better Initialization. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 4475–4483. PMLR, November 2020. URL `https://proceedings.mlr.press/v119/huang20f.html`. ISSN: 2640-3498.

Allan Jabri, Kyle Hsu, Abhishek Gupta, Ben Eysenbach, Sergey Levine, and Chelsea Finn. Unsupervised Curricula for Visual Meta-Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper_files/paper/2019/hash/d5a28f81834b6df2b6db6d3e5e2635c7-Abstract.html`.

Minqi Jiang, Edward Grefenstette, and Tim Rocktaeschel. Prioritized Level Replay. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 4940–4950. PMLR, July 2021. ISSN: 2640-3498.

Lars Johannsmeier, Malkin Gerchow, and Sami Haddadin. A Framework for Robot Manipulation: Skill Formalism, Meta Learning and Adaptive Control. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5844–5850, May 2019. doi: 10.1109/ICRA.2019.8793542. URL `https://ieeexplore.ieee.org/abstract/document/8793542`. ISSN: 2577-087X.

Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Huellermeier. A Survey of Reinforcement Learning from Human Feedback, December 2023. URL `https://ui.adsabs.harvard.edu/abs/2023arXiv231214925K`. ADS Bibcode: 2023arXiv231214925K.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL `https://arxiv.org/abs/1412.6980`.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL `https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf`.

Hang Lai, Weinan Zhang, Xialin He, Chen Yu, Zheng Tian, Yong Yu, and Jun Wang. Sim-to-Real Transfer for Quadrupedal Locomotion via Terrain Transformer. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5141–5147, May 2023. doi: 10.1109/ICRA48891.2023.10160497. URL `https://ieeexplore.ieee.org/abstract/document/10160497`.

Robert Lange, Yingtao Tian, and Yujin Tang. Evolution Transformer: In-Context Evolutionary Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '24 Companion, pp. 575–578, New York, NY, USA, August 2024. Association for Computing Machinery. ISBN 9798400704956. doi: 10.1145/3638530.3654393. URL `https://dl.acm.org/doi/10.1145/3638530.3654393`.

Alessandro Lazaric and Mohammad Ghavamzadeh. Bayesian Multi-Task Reinforcement Learning. pp. 599. Omnipress, June 2010. URL `https://inria.hal.science/inria-00475214`.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. doi: 10.1162/neco.1989.1.4.541.

Hung-yi Lee, Ngoc Thang Vu, and Shang-Wen Li. Meta Learning and Its Applications to Natural Language Processing. In David Chiang and Min Zhang (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Tutorial Abstracts*, pp. 15–20, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-tutorials.3. URL `https://aclanthology.org/2021.acl-tutorials.3/`.

Hung-yi Lee, Shang-Wen Li, and Ngoc Thang Vu. Meta Learning for Natural Language Processing: A Survey, July 2022. URL `http://arxiv.org/abs/2205.01500`. arXiv:2205.01500 [cs].

Jae-Jun Lee and Sung Whan Yoon. XB-MAML: Learning Expandable Basis Parameters for Effective Meta-Learning with Wide Task Coverage. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, pp. 3196–3204. PMLR, April 2024. URL `https://proceedings.mlr.press/v238/lee24b.html`. ISSN: 2640-3498.

Jonathan Lee, Annie Xie, Aldo Pacchiano, Yash Chandak, Chelsea Finn, Ofir Nachum, and Emma Brunskill. Supervised Pretraining Can Learn In-Context Reinforcement Learning. *Advances in Neural Information Processing Systems*, 36:43057–43083, December 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/hash/8644b61a9bc87bf7844750a015feb600-Abstract-Conference.html`.

Joel Lehman and Kenneth O. Stanley. Novelty Search and the Problem with Objectives. In Rick Riolo, Ekaterina Vladislavleva, and Jason H. Moore (eds.), *Genetic Programming Theory and Practice IX*, pp. 37–56. Springer, New York, NY, 2011. ISBN 978-1-4614-1770-5. doi: 10.1007/978-1-4614-1770-5_3. URL `https://doi.org/10.1007/978-1-4614-1770-5_3`.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to Generalize: Meta-Learning for Domain Generalization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018. ISSN 2374-3468. doi: 10.1609/aaai.v32i1.11596. URL `https://ojs.aaai.org/index.php/AAAI/article/view/11596`. Number: 1.

Xiangtai Li, Henghui Ding, Haobo Yuan, Wenwei Zhang, Jiangmiao Pang, Guangliang Cheng, Kai Chen, Ziwei Liu, and Chen Change Loy. Transformer-Based Visual Segmentation: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10138–10163, December 2024a. ISSN 1939-3539. doi: 10.1109/TPAMI.2024.3434373. URL `https://ieeexplore.ieee.org/abstract/document/10613466`.

Xianze Li, Hao Su, Ling Xiang, Qingtao Yao, and Aijun Hu. Transformer-based meta learning method for bearing fault identification under multiple small sample conditions. *Mechanical Systems and Signal Processing*, 208:110967, February 2024b. ISSN 0888-3270. doi: 10.1016/j.ymssp.2023.110967. URL `https://www.sciencedirect.com/science/article/pii/S0888327023008750`.

Xiaoxu Li, Zhuo Sun, Jing-Hao Xue, and Zhanyu Ma. A concise review of recent few-shot meta-learning methods. *Neurocomputing*, 456:463–468, October 2021. ISSN 0925-2312. doi: 10.1016/j.neucom.2020.05.114. URL `https://www.sciencedirect.com/science/article/pii/S0925231220316222`.

Pengchen Liang, Bin Pu, Haishan Huang, Yiwei Li, Hualiang Wang, Weibo Ma, and Qing Chang. Vision Foundation Models in Medical Image Analysis: Advances and Challenges, February 2025. URL `http://arxiv.org/abs/2502.14584`. arXiv:2502.14584 [eess].

Bryan Lim, Sercan O. Arık, Nicolas Loeff, and Tomas Pfister. Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, October 2021. ISSN 0169-2070. doi: 10.1016/j.ijforecast.2021.03.012. URL `https://www.sciencedirect.com/science/article/pii/S0169207021000637`.

Jian Lin, Haidong Shao, Xiangdong Zhou, Baoping Cai, and Bin Liu. Generalized MAML for few-shot cross-domain fault diagnosis of bearing driven by heterogeneous signals. *Expert Systems with Applications*, 230:120696, November 2023. ISSN 0957-4174. doi: 10.1016/j.eswa.2023.120696. URL https://www.sciencedirect.com/science/article/pii/S0957417423011983.

Evan Z. Liu, Aditi Raghunathan, Percy Liang, and Chelsea Finn. Decoupling Exploration and Exploitation for Meta-Reinforcement Learning without Sacrifices. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 6925–6935. PMLR, July 2021. URL https://proceedings.mlr.press/v139/liu21s.html. ISSN: 2640-3498.

Hao Liu and Pieter Abbeel. Emergent Agentic Transformer from Chain of Hindsight Experience. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 21362–21374. PMLR, July 2023. URL https://proceedings.mlr.press/v202/liu23a.html. ISSN: 2640-3498.

Hao Liu, Richard Socher, and Caiming Xiong. Taming MAML: Efficient unbiased meta-reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 4061–4071. PMLR, May 2019. URL https://proceedings.mlr.press/v97/liu19g.html. ISSN: 2640-3498.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. September 2018. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Gabriel Maicas, Andrew P. Bradley, Jacinto C. Nascimento, Ian Reid, and Gustavo Carneiro. Training Medical Image Analysis Systems like Radiologists. In Alejandro F. Frangi, Julia A. Schnabel, Christos Davatzikos, Carlos Alberola-López, and Gabor Fichtinger (eds.), *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*, pp. 546–554, Cham, 2018. Springer International Publishing. ISBN 978-3-030-00928-1. doi: 10.1007/978-3-030-00928-1_62.

Amir M. Mansourian, Rozhan Ahmadi, Masoud Ghafouri, Amir Mohammad Babaei, Elaheh Badali Golezani, Zeynab Yasamani Ghamchi, Vida Ramezanian, Alireza Taherian, Kimia Dinashi, Amirali Miri, and Shohreh Kasaei. A Comprehensive Survey on Knowledge Distillation, March 2025. URL http://arxiv.org/abs/2503.12067. arXiv:2503.12067 [cs].

Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher–Student Curriculum Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 31(9):3732–3740, September 2020. ISSN 2162-2388. doi: 10.1109/TNNLS.2019.2934906. URL https://ieeexplore.ieee.org/abstract/document/8827566.

Daniel G. McClement, Nathan P. Lawrence, Michael G. Forbes, Philip D. Loewen, Johan U. Backstroem, and R. Bhushan Gopaluni. Meta-Reinforcement Learning for Adaptive Control of Second Order Systems. In *2022 IEEE International Symposium on Advanced Control of Industrial Processes (AdCONIP)*, pp. 78–83, August 2022. doi: 10.1109/AdCONIP55568.2022.9894150. URL https://ieeexplore.ieee.org/abstract/document/9894150.

Luckeciano C Melo. Transformers are Meta-Reinforcement Learners. *arXiv preprint arXiv:2206.06614*, 2022.

Vladimir Mikulik, Grégoire Delétang, Tom McGrath, Tim Genewein, Miljan Martic, Shane Legg, and Pedro Ortega. Meta-trained agents implement Bayes-optimal agents. In *Advances in Neural Information Processing Systems*, volume 33, pp. 18691–18703. Curran Associates, Inc., 2020.

Michael Moor, Oishi Banerjee, Zahra Shakeri Hossein Abad, Harlan M. Krumholz, Jure Leskovec, Eric J. Topol, and Pranav Rajpurkar. Foundation models for generalist medical artificial intelligence. *Nature*, 616(7956):259–265, April 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-05881-4. URL https://www.nature.com/articles/s41586-023-05881-4. Publisher: Nature Publishing Group.

Kamil Nar and Shankar Sastry. Step Size Matters in Deep Learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/e8fd4a8a5bab2b3785d794ab51fef55c-Abstract.html.

Tarun Naren, Yuanda Zhu, and May Dongmei Wang. COVID-19 diagnosis using model agnostic meta-learning on limited chest X-ray images. In *Proceedings of the 12th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, BCB '21, pp. 1–9, New York, NY, USA, August 2021. Association for Computing Machinery. ISBN 978-1-4503-8450-6. doi: 10.1145/3459930.3469517. URL `https://dl.acm.org/doi/10.1145/3459930.3469517`.

Thanh Nguyen, Tung Luu, Trung Pham, Sanzhar Rakhimkul, and Chang D. Yoo. Robust Maml: Prioritization Task Buffer with Adaptive Learning Process for Model-Agnostic Meta-Learning. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3460–3464, June 2021. doi: 10.1109/ICASSP39728.2021.9413446. URL `https://ieeexplore.ieee.org/abstract/document/9413446`. ISSN: 2379-190X.

Tianwei Ni, Benjamin Eysenbach, and Ruslan Salakhutdinov. Recurrent Model-Free RL Can Be a Strong Baseline for Many POMDPs, June 2022. URL `http://arxiv.org/abs/2110.05038`. arXiv:2110.05038 [cs].

Tianwei Ni, Michel Ma, Benjamin Eysenbach, and Pierre-Luc Bacon. When Do Transformers Shine in RL? Decoupling Memory from Credit Assignment. *Advances in Neural Information Processing Systems*, 36:50429–50452, December 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/hash/9dc5accb1e4f4a9798eae145f2e4869b-Abstract-Conference.html`.

Shuteng Niu, Yongxin Liu, Jian Wang, and Houbing Song. A Decade Survey of Transfer Learning (2010–2020). *IEEE Transactions on Artificial Intelligence*, 1(2):151–166, October 2020. ISSN 2691-4581. doi: 10.1109/TAI.2021.3054609. URL `https://ieeexplore.ieee.org/abstract/document/9336290`. Conference Name: IEEE Transactions on Artificial Intelligence.

Ben Norman and Jeff Clune. First-Explore, then Exploit: Meta-Learning to Solve Hard Exploration-Exploitation Trade-Offs. *Advances in Neural Information Processing Systems*, 37:27490–27528, December 2024. URL `https://proceedings.neurips.cc/paper_files/paper/2024/hash/30754e5f4cd69d64b5527cdd87d3cf62-Abstract-Conference.html`.

Michael O'Connell, Guanya Shi, Xichen Shi, and Soon-Jo Chung. Meta-Learning-Based Robust Adaptive Flight Control Under Uncertain Wind Conditions, May 2022. URL `http://arxiv.org/abs/2103.01932`. arXiv:2103.01932 [cs].

OpenAI, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with Large Scale Deep Reinforcement Learning, December 2019. URL `http://arxiv.org/abs/1912.06680`. arXiv:1912.06680 [cs].

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, C. J. Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. GPT-4 Technical Report, March 2024. URL http://arxiv.org/abs/2303.08774. arXiv:2303.08774 [cs].

Santisudha Panigrahi, Anuja Nanda, and Tripti Swarnkar. A Survey on Transfer Learning. In Debahuti Mishra, Rajkumar Buyya, Prasant Mohapatra, and Srikanta Patnaik (eds.), *Intelligent and Cloud Computing*, pp. 781–789, Singapore, 2021. Springer. ISBN 9789811559716. doi: 10.1007/978-981-15-5971-6_83.

Emilio Parisotto, Soham Ghosh, Sai Bhargav Yalamanchi, Varsha Chinnaobireddy, Yuhuai Wu, and Ruslan Salakhutdinov. Concurrent Meta Reinforcement Learning, March 2019. URL http://arxiv.org/abs/1903.02710. arXiv:1903.02710 [cs].

Ramakanth Pasunuru and Mohit Bansal. Continual and Multi-Task Architecture Search, June 2019. URL http://arxiv.org/abs/1906.05226. arXiv:1906.05226 [cs].

Gean Trindade Pereira. Meta-Learning applied to Neural Architecture Search. Towards new interactive learning approaches for indexing and analyzing images from expert domains, 2024.

Rémy Portelas, Cédric Colas, Lilian Weng, Katja Hofmann, and Pierre-Yves Oudeyer. Automatic Curriculum Learning For Deep RL: A Short Survey, May 2020. URL http://arxiv.org/abs/2003.04664. arXiv:2003.04664 [cs].

Rémy Portelas, Clément Romac, Katja Hofmann, and Pierre-Yves Oudeyer. Meta Automatic Curriculum Learning, September 2021. URL http://arxiv.org/abs/2011.08463. arXiv:2011.08463 [cs].

Justin K. Pugh, Lisa B. Soros, and Kenneth O. Stanley. Quality Diversity: A New Frontier for Evolutionary Computation. *Frontiers in Robotics and AI*, 3, July 2016. ISSN 2296-9144. doi: 10.3389/frobt.2016.00040. URL https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2016.00040/full. Publisher: Frontiers.

Alireza Rafiei, Ronald Moore, Sina Jahromi, Farshid Hajati, and Rishikesan Kamaleswaran. Meta-learning in Healthcare: A Survey. *SN Computer Science*, 5(6):791, August 2024. ISSN 2661-8907. doi: 10.1007/s42979-024-03166-9. URL https://doi.org/10.1007/s42979-024-03166-9.

Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid Learning or Feature Reuse? Towards Understanding the Effectiveness of MAML, February 2020. URL `http://arxiv.org/abs/1909.09157`. arXiv:1909.09157 [cs, stat].

Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-Learning with Implicit Gradients. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper_files/paper/2019/hash/072b030ba126b2f4b2374f342be9ed44-Abstract.html`.

Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables, March 2019. arXiv:1903.08254 [cs, stat].

Kanishka Ranaweera and Pubudu N. Pathirana. Optimizing Diagnosis in Sparse Data Environments: A Model Agnostic Meta Learning Approach. In *2024 46th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 1–4, July 2024. doi: 10.1109/EMBC53108.2024.10782770. URL `https://ieeexplore.ieee.org/abstract/document/10782770`. ISSN: 2694-0604.

Harish Ravichandar, Athanasios S. Polydoros, Sonia Chernova, and Aude Billard. Recent Advances in Robot Learning from Demonstration. *Annual Review of Control, Robotics, and Autonomous Systems*, 3(Volume 3, 2020):297–330, May 2020. ISSN 2573-5144. doi: 10.1146/annurev-control-100819-063206. URL `https://www.annualreviews.org/content/journals/10.1146/annurev-control-100819-063206`. Publisher: Annual Reviews.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A Generalist Agent, November 2022. URL `http://arxiv.org/abs/2205.06175`. arXiv:2205.06175 [cs].

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A Comprehensive Survey of Neural Architecture Search: Challenges and Solutions. *ACM Comput. Surv.*, 54 (4):76:1–76:34, May 2021. ISSN 0360-0300. doi: 10.1145/3447582. URL `https://dl.acm.org/doi/10.1145/3447582`.

Micah Rentschler and Jesse Roberts. RL + Transformer = A General-Purpose Problem Solver, January 2025. URL `http://arxiv.org/abs/2501.14176`. arXiv:2501.14176 [cs].

Gaith Rjoub, Saidul Islam, Jamal Bentahar, Mohammed Amin Almaiah, and Rana Alrawashdeh. Enhancing IoT Intelligence: A Transformer-based Reinforcement Learning Methodology. In *2024 International Wireless Communications and Mobile Computing (IWCMC)*, pp. 1418–1423, May 2024. doi: 10.1109/IWCMC61514.2024.10592607. URL `https://ieeexplore.ieee.org/abstract/document/10592607`. ISSN: 2376-6506.

Andrei A. Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation, 2016.

Noor Sajid, Philip J. Ball, Thomas Parr, and Karl J. Friston. Active Inference: Demystified and Compared. *Neural Computation*, 33(3):674–712, March 2021. ISSN 0899-7667. doi: 10.1162/neco_a_01357. URL `https://doi.org/10.1162/neco_a_01357`.

Jürgen Schmidhuber. *Evolutionary Principles in Self-referential Learning: On Learning how to Learn: the Meta-meta-meta...-hook*. 1987.

Simon Schmitt, Jonathan J. Hudson, Augustin Zidek, Simon Osindero, Carl Doersch, Wojciech M. Czarnecki, Joel Z. Leibo, Heinrich Kuttler, Andrew Zisserman, Karen Simonyan, and S. M. Ali Eslami. Kickstarting deep reinforcement learning, 2018.

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised Pre-training for Speech Recognition, September 2019. URL http://arxiv.org/abs/1904.05862. arXiv:1904.05862 [cs].

John Schulman. Trust region policy optimization. *arXiv preprint arXiv:1502.05477*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, August 2017. arXiv:1707.06347 [cs].

Snigdha Sen and Pavan Chakraborty. Evaluating Efficacy of MAML Based Approach on Regression Using Astronomical Imbalanced Dataset. In *2024 International Conference on Computational Intelligence and Network Systems (CINS)*, pp. 1–8, November 2024. doi: 10.1109/CINS63881.2024.10864404. URL https://ieeexplore.ieee.org/abstract/document/10864404.

Gresa Shala, André Biedenkapp, and Josif Grabocka. Hierarchical Transformers are Efficient Meta-Reinforcement Learners, February 2024a. URL http://arxiv.org/abs/2402.06402. arXiv:2402.06402 [cs].

Gresa Shala, André Biedenkapp, Pierre Krack, Florian Walter, and Josif Grabocka. Efficient Cross-Episode Meta-RL. October 2024b. URL https://openreview.net/forum?id=UENQuayzr1.

Arturo Daniel Sosa-Ceron, Hugo Gustavo Gonzalez-Hernandez, and Jorge Antonio Reyes-Avendaño. Learning from Demonstrations in Human–Robot Collaborative Scenarios: A Survey. *Robotics*, 11(6):126, December 2022. ISSN 2218-6581. doi: 10.3390/robotics11060126. URL https://www.mdpi.com/2218-6581/11/6/126. Number: 6 Publisher: Multidisciplinary Digital Publishing Institute.

Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. Curriculum Learning: A Survey. *International Journal of Computer Vision*, 130(6):1526–1565, June 2022. ISSN 1573-1405. doi: 10.1007/s11263-022-01611-x. URL https://doi.org/10.1007/s11263-022-01611-x.

Bradly Stadie, Ge Yang, Rein Houthooft, Peter Chen, Yan Duan, Yuhuai Wu, Pieter Abbeel, and Ilya Sutskever. The Importance of Sampling inMeta-Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/d0f5722f11a0cc839fa2ca6ea49d8585-Abstract.html.

Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for Semantic Segmentation, May 2021. URL https://ui.adsabs.harvard.edu/abs/2021arXiv210505633S. ADS Bibcode: 2021arXiv210505633S.

Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: an introduction.* Adaptive computation and machine learning series. The MIT Press, Cambridge, Massachusetts, 2. edition, 2018. ISBN 978-0-262-03924-6.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/hash/464d828b85b0bed98e80ade0a5c43b0f-Abstract.html.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper With Convolutions. pp. 1–9, 2015. URL https://www.cv-foundation.org/openaccess/content_cvpr_2015/html/Szegedy_Going_Deeper_With_2015_CVPR_paper.html.

Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A Survey on Deep Transfer Learning. In Věra Kůrková, Yannis Manolopoulos, Barbara Hammer, Lazaros Iliadis, and Ilias Maglogiannis (eds.), *Artificial Neural Networks and Machine Learning – ICANN 2018*, pp. 270–279, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01424-7. doi: 10.1007/978-3-030-01424-7_27.

Mingxing Tan and Quoc Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 6105–6114. PMLR, May 2019. URL https://proceedings.mlr.press/v97/tan19a.html. ISSN: 2640-3498.

Yanchao Tan, Carl Yang, Xiangyu Wei, Chaochao Chen, Weiming Liu, Longfei Li, Jun Zhou, and Xiaolin Zheng. MetaCare++: Meta-Learning with Hierarchical Subtyping for Cold-Start Diagnosis Prediction in Healthcare Data. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, pp. 449–459, New York, NY, USA, July 2022. Association for Computing Machinery. ISBN 978-1-4503-8732-3. doi: 10.1145/3477495.3532020. URL https://dl.acm.org/doi/10.1145/3477495.3532020.

Suigu Tang, Xiaoyuan Yu, Chak Fong Cheang, Yanyan Liang, Penghui Zhao, Hon Ho Yu, and I Cheong Choi. Transformer-based multi-task learning for classification and segmentation of gastrointestinal tract endoscopic images. *Computers in Biology and Medicine*, 157:106723, May 2023. ISSN 0010-4825. doi: 10.1016/j.compbiomed.2023.106723. URL https://www.sciencedirect.com/science/article/pii/S0010482523001889.

Adaptive Agent Team, Jakob Bauer, Kate Baumli, Satinder Baveja, Feryal Behbahani, Avishkar Bhoopchand, Nathalie Bradley-Schmieg, Michael Chang, Natalie Clay, Adrian Collister, Vibhavari Dasagi, Lucy Gonzalez, Karol Gregor, Edward Hughes, Sheleem Kashem, Maria Loks-Thompson, Hannah Openshaw, Jack Parker-Holder, Shreya Pathak, Nicolas Perez-Nieves, Nemanja Rakicevic, Tim Rocktaeschel, Yannick Schroecker, Jakub Sygnowski, Karl Tuyls, Sarah York, Alexander Zacherl, and Lei Zhang. Human-Timescale Adaptation in an Open-Ended Task Space, January 2023. arXiv:2301.07608 [cs].

Open Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarina Barros, Charlie Deck, Jakob Bauer, Jakub Sygnowski, Maja Trebacz, Max Jaderberg, Michael Mathieu, Nat McAleese, Nathalie Bradley-Schmieg, Nathaniel Wong, Nicolas Porcel, Roberta Raileanu, Steph Hughes-Fitt, Valentin Dalibard, and Wojciech Marian Czarnecki. Open-Ended Learning Leads to Generally Capable Agents, July 2021. URL http://arxiv.org/abs/2107.12808. arXiv:2107.12808 [cs].

Sebastian Thrun and Lorien Pratt. Learning to Learn: Introduction and Overview. In Sebastian Thrun and Lorien Pratt (eds.), *Learning to Learn*, pp. 3–17. Springer US, Boston, MA, 1998. ISBN 978-1-4615-5529-2. doi: 10.1007/978-1-4615-5529-2_1. URL https://doi.org/10.1007/978-1-4615-5529-2_1.

Yuanyi Tian. A Meta-Learning Prediction Model for Identifying miRNA-Disease Associations Based on MAML. In *2024 5th International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT)*, pp. 1073–1077, March 2024. doi: 10.1109/AINIT61980.2024.10581492. URL https://ieeexplore.ieee.org/abstract/document/10581492.

Yuanyi Tian, Quan Zou, ChunYu Wang, and Cangzhi Jia. MAMLCDA: A Meta-Learning Model for Predicting circRNA-Disease Association Based on MAML Combined With CNN. *IEEE Journal of Biomedical and Health Informatics*, 28(7):4325–4335, July 2024. ISSN 2168-2208. doi: 10.1109/JBHI.2024.3385352. URL https://ieeexplore.ieee.org/abstract/document/10493116.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

Julian Togelius and Georgios N. Yannakakis. Choose Your Weapon: Survival Strategies for Depressed AI Academics [Point of View]. *Proceedings of the IEEE*, 112(1):4–11, January 2024. ISSN 1558-2256. doi: 10.1109/JPROC.2024.3364137. URL https://ieeexplore.ieee.org/abstract/document/10458714.

Lovre Torbarina, Tin Ferkovic, Lukasz Roguski, Velimir Mihelcic, Bruno Sarlija, and Zeljko Kraljevic. Challenges and Opportunities of Using Transformer-Based Multi-Task Learning in NLP Through ML Lifecycle: A Position Paper. *Natural Language Processing Journal*, 7:100076, June 2024. ISSN 2949-7191. doi: 10.1016/j.nlp.2024.100076. URL https://www.sciencedirect.com/science/article/pii/S2949719124000244.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models, February 2023. URL https://ui.adsabs.harvard.edu/abs/2023arXiv230213971T. ADS Bibcode: 2023arXiv230213971T.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-Dataset: A Dataset of Datasets for Learning to Learn from Few Examples, April 2020. URL http://arxiv.org/abs/1903.03096. arXiv:1903.03096 [cs].

Richa Upadhyay. Sharing to learn and learning to share : Fitting together metalearning and multi-task learning. 2023. Publisher: Luleå University of Technology.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Anna Vettoruzzo, Mohamed-Rafik Bouguelia, Joaquin Vanschoren, Thorsteinn Roegnvaldsson, and KC Santosh. Advances and Challenges in Meta-Learning: A Technical Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(7):4763–4779, July 2024. ISSN 1939-3539. doi: 10.1109/TPAMI.2024.3357847. URL https://ieeexplore.ieee.org/abstract/document/10413635. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

Nelson Vithayathil Varghese and Qusay H. Mahmoud. A Survey of Multi-Task Deep Reinforcement Learning. *Electronics*, 9(9):1363, September 2020. ISSN 2079-9292. doi: 10.3390/electronics9091363. URL https://www.mdpi.com/2079-9292/9/9/1363. Number: 9 Publisher: Multidisciplinary Digital Publishing Institute.

Riccardo Volpi, Diane Larlus, and Gregory Rogez. Continual Adaptation of Visual Representations via Domain Randomization and Meta-Learning. pp. 4443–4453, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Volpi_Continual_Adaptation_of_Visual_Representations_via_Domain_Randomization_and_Meta-Learning_CVPR_2021_paper.html.

Haoxiang Wang, Yite Wang, Ruoyu Sun, and Bo Li. *Global Convergence of MAML and Theory-Inspired Neural Architecture Search for Few-Shot Learning*, pp. 9797–9808. 2022a. URL https://openaccess.thecvf.com/content/CVPR2022/html/Wang_Global_Convergence_of_MAML_and_Theory-Inspired_Neural_Architecture_Search_for_CVPR_2022_paper.html.

Jiahao Wang, Zutong Sun, Hang Yan, and Hao Chen. MAML-TSC: Model-Agnostic Meta-Learning for Time Series Classification. In *2024 7th International Conference on Pattern Recognition and Artificial Intelligence (PRAI)*, pp. 1043–1049, August 2024a. doi: 10.1109/PRAI62207.2024.10827760. URL https://ieeexplore.ieee.org/abstract/document/10827760.

Xin Wang, Yudong Chen, and Wenwu Zhu. A Survey on Curriculum Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4555–4576, September 2022b. ISSN 1939-3539. doi: 10.1109/TPAMI.2021.3069908. URL https://ieeexplore.ieee.org/abstract/document/9392296.

Yufei Wang, Zhanyi Sun, Jesse Zhang, Zhou Xian, Erdem Biyik, David Held, and Zackory Erickson. RL-VLM-F: Reinforcement Learning from Vision Language Foundation Model Feedback, June 2024b. URL http://arxiv.org/abs/2402.03681. arXiv:2402.03681 [cs].

Haiyue Wu, Matthew J. Triebe, and John W. Sutherland. A transformer-based approach for novel fault detection and fault classification/diagnosis in manufacturing: A rotary system application. *Journal of Manufacturing Systems*, 67:439–452, April 2023. ISSN 0278-6125. doi: 10.1016/j.jmsy.2023.02.018. URL https://www.sciencedirect.com/science/article/pii/S0278612523000419.

Hanguang Xiao, Li Li, Qiyuan Liu, Xiuhong Zhu, and Qihang Zhang. Transformers in medical image segmentation: A review. *Biomedical Signal Processing and Control*, 84:104791, July 2023. ISSN 1746-8094. doi: 10.1016/j.bspc.2023.104791. URL `https://www.sciencedirect.com/science/article/pii/S1746809423002240`.

Tengye Xu, Zihao Li, and Qinyuan Ren. Meta-Reinforcement Learning Robust to Distributional Shift Via Performing Lifelong In-Context Learning. June 2024. URL `https://openreview.net/forum?id=laIOUtstMs`.

Zeqiu Xu, Jiani Wang, Xiaochuan Xu, Peiyang Yu, Tianyi Huang, and Jingyuan Yi. A Survey of Reinforcement Learning-Driven Knowledge Distillation: Techniques, Challenges, and Applications. In *2025 IEEE 6th International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT)*, pp. 1–6, April 2025. doi: 10.1109/AINIT65432.2025.11035605. URL `https://ieeexplore.ieee.org/abstract/document/11035605`.

Zifan Xu, Yulin Zhang, Shahaf S. Shperberg, Reuth Mirsky, Yuqian Jiang, Bo Liu, and Peter Stone. Model-Based Meta Automatic Curriculum Learning. In *Proceedings of The 2nd Conference on Lifelong Learning Agents*, pp. 846–860. PMLR, November 2023. URL `https://proceedings.mlr.press/v232/xu23a.html`. ISSN: 2640-3498.

Weirui Ye, Yunsheng Zhang, Haoyang Weng, Xianfan Gu, Shengjie Wang, Tong Zhang, Mengchen Wang, Pieter Abbeel, and Yang Gao. Reinforcement Learning with Foundation Priors: Let the Embodied Agent Efficiently Learn on Its Own, October 2024. URL `http://arxiv.org/abs/2310.02635`. arXiv:2310.02635 [cs].

Shuolei Yin, Yejing Xi, Xun Zhang, Chengnuo Sun, and Qirong Mao. Foundation Models in Agriculture: A Comprehensive Review. *Agriculture*, 15(8):847, January 2025. ISSN 2077-0472. doi: 10.3390/agriculture15080847. URL `https://www.mdpi.com/2077-0472/15/8/847`. Number: 8 Publisher: Multidisciplinary Digital Publishing Institute.

Wenpeng Yin. Meta-learning for Few-shot Natural Language Processing: A Survey, July 2020. URL `http://arxiv.org/abs/2007.09604`. arXiv:2007.09604 [cs].

Jun Yu, Yutong Dai, Xiaokang Liu, Jin Huang, Yishan Shen, Ke Zhang, Rong Zhou, Eashan Adhikarla, Wenxuan Ye, Yixin Liu, Zhaoming Kong, Kai Zhang, Yilong Yin, Vinod Namboodiri, Brian D. Davison, Jason H. Moore, and Yong Chen. Unleashing the Power of Multi-Task Learning: A Comprehensive Survey Spanning Traditional, Deep, and Pretrained Foundation Model Eras, April 2024. URL `http://arxiv.org/abs/2404.18961`.

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 *Proceedings of the Conference on Robot Learning*, pp. 1094–1100. PMLR, May 2020. URL `https://proceedings.mlr.press/v100/yu20a.html`. ISSN: 2640-3498.

Wenhao Yu, Visak CV Kumar, Greg Turk, and C. Karen Liu. Sim-to-Real Transfer for Biped Locomotion. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3503–3510, November 2019. doi: 10.1109/IROS40897.2019.8968053. URL `https://ieeexplore.ieee.org/abstract/document/8968053`. ISSN: 2153-0866.

Jin Zhang, Jianhao Wang, Hao Hu, Tong Chen, Yingfeng Chen, Changjie Fan, and Chongjie Zhang. MetaCURE: Meta Reinforcement Learning with Empowerment-Driven Exploration. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 12600–12610. PMLR, July 2021. URL `https://proceedings.mlr.press/v139/zhang21w.html`. ISSN: 2640-3498.

Shaoting Zhang and Dimitris Metaxas. On the challenges and perspectives of foundation models for medical image analysis. *Medical Image Analysis*, 91:102996, January 2024. ISSN 1361-8415. doi: 10.1016/j.media.2023.102996. URL `https://www.sciencedirect.com/science/article/pii/S1361841523002566`.

Shuang Zhang, Rui Fan, Yuti Liu, Shuang Chen, Qiao Liu, and Wanwen Zeng. Applications of transformer-based language models in bioinformatics: a survey. *Bioinformatics Advances*, 3(1):vbad001, January 2023a. ISSN 2635-0041. doi: 10.1093/bioadv/vbad001. URL `https://doi.org/10.1093/bioadv/vbad001`.

Yi Zhang, Yanji Hao, Yu Fu, Yijing Feng, Yeqing Li, Xiaonan Wang, Junting Pan, Yongming Han, and Chunming Xu. GAN-MAML strategy for biomass energy production: Overcoming small dataset limitations. *Applied Energy*, 387:125568, June 2025. ISSN 0306-2619. doi: 10.1016/j.apenergy.2025.125568. URL `https://www.sciencedirect.com/science/article/pii/S0306261925002983`.

Yingying Zhang, Xian Wu, Quan Fang, Shengsheng Qian, and Changsheng Xu. Knowledge-Enhanced Attributed Multi-Task Learning for Medicine Recommendation. *ACM Trans. Inf. Syst.*, 41(1):17:1–17:24, January 2023b. ISSN 1046-8188. doi: 10.1145/3527662. URL `https://dl.acm.org/doi/10.1145/3527662`.

Yu Zhang and Qiang Yang. An overview of multi-task learning. *National Science Review*, 5(1):30–43, January 2018. ISSN 2095-5138. doi: 10.1093/nsr/nwx105. URL `https://doi.org/10.1093/nsr/nwx105`.

Yu Zhang and Qiang Yang. A Survey on Multi-Task Learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609, December 2022. ISSN 1558-2191. doi: 10.1109/TKDE.2021.3070203. URL `https://ieeexplore.ieee.org/abstract/document/9392366`. Conference Name: IEEE Transactions on Knowledge and Data Engineering.

Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 737–744, December 2020. doi: 10.1109/SSCI47803.2020.9308468. URL `https://ieeexplore.ieee.org/abstract/document/9308468`.

Yincheng Zhao, Guozhou Zhang, Weihao Hu, Qi Huang, Zhe Chen, and Frede Blaabjerg. Meta-Learning Based Voltage Control for Renewable Energy Integrated Active Distribution Network Against Topology Change. *IEEE Transactions on Power Systems*, 38(6):5937–5940, November 2023. ISSN 1558-0679. doi: 10.1109/TPWRS.2023.3309536. URL `https://ieeexplore.ieee.org/abstract/document/10244062`. Conference Name: IEEE Transactions on Power Systems.

Yue Zhou, Houjin Chen, Yanfeng Li, Qin Liu, Xuanang Xu, Shu Wang, Pew-Thian Yap, and Dinggang Shen. Multi-task learning for segmentation and classification of tumors in 3D automated breast ultrasound images. *Medical Image Analysis*, 70:101918, May 2021. ISSN 1361-8415. doi: 10.1016/j.media.2020.101918. URL `https://www.sciencedirect.com/science/article/pii/S1361841520302826`.

Zhuangdi Zhu, Kaixiang Lin, Anil K. Jain, and Jiayu Zhou. Transfer Learning in Deep Reinforcement Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):13344–13362, November 2023. ISSN 1939-3539. doi: 10.1109/TPAMI.2023.3292075. URL `https://ieeexplore.ieee.org/abstract/document/10172347`. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A Comprehensive Survey on Transfer Learning. *Proceedings of the IEEE*, 109(1):43–76, January 2021. ISSN 1558-2256. doi: 10.1109/JPROC.2020.3004555. URL `https://ieeexplore.ieee.org/abstract/document/9134370`. Conference Name: Proceedings of the IEEE.

Luisa Zintgraf, Kyriacos Shiarli, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast Context Adaptation via Meta-Learning. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 7693–7702. PMLR, May 2019. URL `https://proceedings.mlr.press/v97/zintgraf19a.html`. ISSN: 2640-3498.

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, February 2020. arXiv:1910.08348 [cs, stat].

Luisa M. Zintgraf, Leo Feng, Cong Lu, Maximilian Igl, Kristian Hartikainen, Katja Hofmann, and Shimon Whiteson. Exploration in Approximate Hyper-State Space for Meta Reinforcement Learning. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 12991–13001. PMLR, July 2021. URL `https://proceedings.mlr.press/v139/zintgraf21a.html`. ISSN: 2640-3498.

7 Appendix

7.1 Attention is all you need

The following paragraphs broadly discuss the architecture of the vanilla transformer Vaswani et al. (2017). The goal is to give the reader an intuition how self-attention "creates context" and "focusses" on important information on the example of language-to-language translation. For a detailed description of the vanilla transformer architecture the interested reader is nevertheless referred to the original paper or Alammar (2018).

The vanilla transformer consists of an encoder and a decoder, which both divide into six 2- or 3-layer sub-blocks respectively [9]. Metaphorically speaking, the encoder translates an input sentence like

The child plays football because it likes it very much.

into "machine language", i.e. word embeddings, while the decoder translates these embeddings back into the desired output language. However, a word-by-word transformation from the input language into "‘machine language "’ and back into the output language preserves no information about what word the first "it" refers to and to which word the second "it" is related to. One needs a possibility to preserve the context between the words and, hence, the semantic structure of the sentence while encoding it. Similarly, while decoding, the syntactic structure of the sentence must be transformed so that it fits the grammar of the output language. For example, a German sentence has a totally different grammatical structure than its English counterpart with the same semantic meaning. In their original publication, Vaswani et al. (2017) solved these problems using two techniques: Self-attention layers and positional encodings. The former is presented in the following paragraph, while the latter is broadly discussed further below.

Self-Attention

In a self-attention layer, every word embedding x_i $(i = 1, 2, \dots, n)$ of a n-word input sentence x like the one above gets a query, key and value projection matrix Q_i, K_i and V_i assigned to it. These matrices are parameters of the self-attention and, as such, they get learned during training. They are used to project a word embedding x_i into query, key and value vectors q_i, k_i and v_i by multiplication. The "context" between words is then considered in the following way:

- Every word embedding x_i gets "compared" to every other word embedding x_j. This is done by multiplying every query vector q_i with every key vector k_j in a dot product. The result can be interpreted as the context between the corresponding words of the input sentence since, mathematically, the dot product puts the query and key vectors in geometrical relationship to each other.
- For each word embedding x_i, the result of its query-key multiplication with every other word embedding (including itself) is a vector representing the attention scores of the word embedding x_i to all other word embeddings x_j. This vector of attention scores gets fed into a softmax function to yield the respective attention weights summing up to one.
- At last, these "attention weights" are multiplied by the value vector v_i, resulting in a weighted sum over attention weights with weights v_i that directly depend on the learned projection V_i. Hence, the model can learn which attention scores $q_i \cdot k_j$ are more or less important by adjusting the values of the value projection matrix V_i respectively.

[9]This number of six sub-blocks for encoder and decoder is rather an architectural choice by Vaswani et al. (2017) than a necessity of the vanilla transformer.

In practice, these calculations are done simultaneously for all query-key pairs of the embedded input sentence x:

$$\text{Attention}(x) := \text{softmax}(\frac{QK^T}{\sqrt{d}})\ V, \tag{22}$$

with the hyperparameter d as the dimension of key, value and query vectors, $\sqrt{d}$ as the scaling factor and K, V and Q being the respective key, value and query matrices. The scaling factor $\sqrt{d}$ keeps the dot product logits small so that the soft-max does not slip into saturation. This maintains a proper gradient flow and thus stabilizes training.

Each encoder sub-block of the vanilla transformer consists of a self-attention layer so that the context between words (and hence the semantic structure of the sentence) is preserved throughout the whole embedding process. Encoding the above example sentence, the model can thus learn that the first word "The" refers to the next word "child" and no other word through the first step, the query-key multiplication $q_1 \cdot k_2$. At the same time, this connection is almost meaningless to the semantic of the whole sentence so that the model possibly learns to assign a very small value to the value vector v_1.

The self-attention layers of the decoder sub-blocks look slightly different to those described above. Since decoding the embeddings into output language is a sequential manner, no word can be set into context to words following later on in the sentence. Therefore, the corresponding query-key multiplications $q_i \cdot k_{i+a}$ with $a = 1, \ldots, n-i$ are set to minus infinity by construction so that the respective softmax results in zero attention weights.

In their original work, Vaswani et al. (2017) implemented multi-head-attention with the number of heads as a hyperparameter and showed this to further boost performance. Multi-Head Attention duplicates the input (i.e. the word embeddings x_i), projects it into the smaller subspace of each head (i.e. the queries, keys, values are smaller-dimensional vectors), executes the Self-Attention of the form equation 22, , concatenates the outputs of all heads to one single output matrix, and projects them back into the original output dimension d by another linear layer transformation.

Positional Encoding

In the self-attention equation 22 the order of input tokens does not matter and hence the order of the words of the input sentence is not automatically preserved throughout encoding. However, the order of words in a sentence is quite important, especially as syntactic structure of input and output languages might differ, e.g. when translating English into German. This is why, Vaswani et al. (2017) included a positional embedding for each word embedding x_i in the vanilla transformer architecture that has the same length as the embedding itself. The form of the positional encoding p_i is a hyperparameter of the vanilla transformer. In their original work, Vaswani et al. (2017) combine different sin and cosine functions, but state that they have tried different positional encodings without a major loss in performance. Other transformer variants even learn this positional embedding function along with all the above model parameters.