Remote Sensing for Water Quality: A Multi-Task, Metadata-Driven Hypernetwork Approach

Olivier Graffeuille[1] , **Yun Sing Koh**[1] , **Jörg Wicker**[1] and **Moritz Lehmann**[2]
[1]School of Computer Science, The University of Auckland
[2]Xerra Earth Observation Institute, The University of Waikato
ogra439@aucklanduni.ac.nz, y.koh@auckland.ac.nz, j.wicker@auckland.ac.nz,
moritz.lehmann@gmail.com

Abstract

Inland water quality monitoring is vital for clean water access and aquatic ecosystem management. Remote sensing machine learning models enable large-scale observations, but are difficult to train due to data scarcity and variability across many lakes. Multi-task learning approaches enable learning of lake differences by learning multiple lake functions simultaneously. However, they suffer from a trade-off between parameter efficiency and the ability to model task differences flexibly, and struggle to model many diverse lakes with few samples per task. We propose Multi-Task Hypernetworks, a novel multi-task learning architecture which circumvents this trade-off using a shared hypernetwork to generate different network weights for each task from small task-specific embeddings. Our approach stands out from existing works by providing the added capacity to leverage task-level metadata, such as lake depth and temperature, explicitly. We show empirically that Multi-Task Hypernetworks outperform existing multi-task learning architectures for water quality remote sensing and other tabular data problems, and leverages metadata more effectively than existing methods.

1 Introduction

Inland water quality has deteriorated globally, primarily due to excessive nutrient loading and global warming, leading to issues such as harmful algal blooms [Hou *et al.*, 2022]. Water quality is a critical factor influencing the health of aquatic ecosystems, the safety of drinking water supplies, and the sustainability of agriculture and industry. Traditional methods for monitoring water quality involve in-situ sampling and laboratory analysis, which are expensive, offer poor spatial and temporal coverage, and are limited by site accessibility. Remote sensing provides large-scale, continuous, and non-invasive observations, motivating research into water quality remote sensing models [Pahlevan *et al.*, 2020]. Effective monitoring aligns with the United Nations Sustainable Development Goals 6 and 14 [United Nations, 2015], aiming for universal access to clean water and promoting the Leave No One Behind principle [United Nations, 2016].

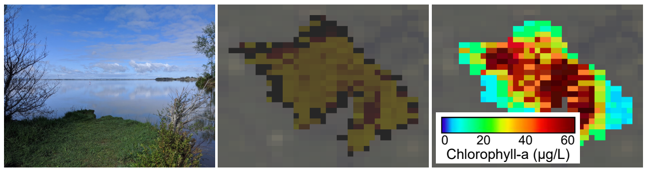


Figure 1: Left: Photo of Algal Bloom, Lake Waikare, New Zealand. Middle: Satellite image of lake. Right: Water quality estimation.

Remote sensing models aim to estimate chlorophyll-a concentration in lakes to use as an indicator of water quality, given multispectral satellite data that represent water colour (Figure 1). These models make pixel-level predictions, based on single-point ground truth samples corresponding to a single satellite pixel. This problem is, therefore, formulated as a regression task with tabular features. Training accurate models is challenging as datasets contain dozens or hundreds of lakes with varied relationships between water colour and chlorophyll-a concentration, due to differences in biological and geological conditions [Neil *et al.*, 2019]. Furthermore, training data is scarce due to expensive ground truth labels, with as few as five data points per lake, motivating data-efficient methods [Graffeuille *et al.*, 2022].

Multi-Task Learning (MTL) approaches can enable learning of lake differences, by modelling lake functions as distinct but related tasks and jointly learning these tasks with knowledge transfer to improve generalisation performance. MTL neural network architectures can be broadly categorised into hard parameter sharing and soft parameter sharing [Ruder, 2017]. Hard parameter sharing, which shares most parameters across tasks, offers parameter efficiency but limited flexibility due to a shared feature representation [Long *et al.*, 2017]. In contrast, soft parameter sharing allows for flexible representations by learning unique weights for each task, constrained by mechanisms like regularization, at the cost of reduced parameter efficiency [Misra *et al.*, 2016]. This highlights a trade-off between parameter-efficient task scaling and learning flexible task networks. As such, MTL approaches struggle to learn problems with many diverse tasks from few samples, such as water quality remote sensing. This necessitates the development of MTL architectures which are both parameter efficient and flexible.

An opportunity to enhance learning for water quality remote sensing is in auxiliary data sources which describe the lakes themselves, such as depth, temperature and weather. These task-level features, which we refer to as *metadata*, are informative of the environmental processes within lakes, indicating that lakes with similar features are likely to be highly related tasks. For example, a warm, shallow lake is likely to be similar to another lake of similar characteristics, but different from a cold, deep lake [Yang *et al.*, 2022]. When available, metadata may help the model learn between-task relationships, particularly as lakes have insufficient data to accurately learn task relationships implicitly during training [Zheng *et al.*, 2019; Zamir *et al.*, 2018]. However, existing MTL algorithms cannot leverage task metadata effectively because they are designed to learn from data only at the sample level, whereas metadata is task-level data and is therefore invariant across all samples within a task. This necessitates the development of MTL architectures that can effectively leverage metadata to improve learning.

To address these gaps, we propose *Multi-Task Hypernetworks*, a novel MTL architecture which learns flexible task networks with substantially fewer parameters per task than other soft parameter sharing approaches, and can effectively leverage task metadata. We achieve this by learning a unique low-dimensional embedding vector of each task, which captures task differences and relationships. Our architecture can generate flexible task networks and enable abstract knowledge transfer using task embeddings as the only task-specific parameters; in our experiments, we use only ten parameters per task. This task-level parameter framework also allows us to naturally inject task metadata into the model.

Our approach accomplishes this by utilising hypernetworks [Ha *et al.*, 2017] for soft parameter knowledge transfer across tasks. Notably, this represents the first attempt within the research area to explore this methodology. Hypernetworks are neural networks that generate the weights of another neural network. Instead of directly learning the weights of a "target" network, which models the task of interest, hypernetworks produce them dynamically based on some input. Our proposed Multi-Task Hypernetwork uses a hypernetwork to generate a different set of weights for multiple target networks which each model a different task. Network weights for all target networks are generated by the same hypernetwork, but use task-specific embeddings as input to the hypernetwork to generate entirely distinct weights (Figure 2). The use of a shared deep hypernetwork enables abstract knowledge transfer between tasks, while the task embeddings capture task differences and relationships. This is a conceptual departure from previous soft parameter sharing methods, which share knowledge at the parameter level. Additionally, by inputting metadata to the shared hypernetwork with the task embedding, the target network weights become a function of their metadata. This allows our architecture to learn task relationships and task functions explicitly from task-level metadata, as well as implicitly by joint task optimisation.

We show that our architecture can improve the accuracy of water quality estimation and other tabular MTL datasets compared to existing approaches, both with and without available metadata. Code and datasets are available at https://github.com/OGraffeuille/Multi-Task-Hypernetworks.

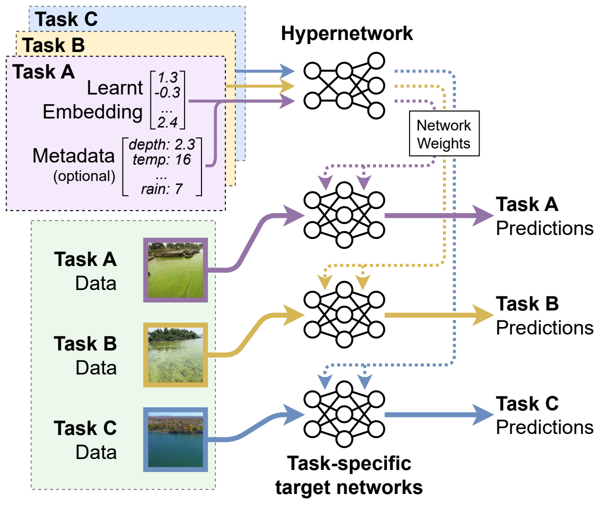


Figure 2: Multi-Task Hypernetwork conceptual diagram.

2 Related Work

2.1 Multi-Task Learning Architectures

Hard sharing MTL architectures [Caruana, 1997; Lopes *et al.*, 2023] have been adapted to share knowledge in task specific layers using a tensor normal distribution which learns task similarities [Long *et al.*, 2017], to learn task differences in hard sharing layers by learning task masks [Pascal *et al.*, 2021], to reduce generalisation error with an adversarial loss to encourage similar task latent distributions [Shui *et al.*, 2019], and to use architecture search [Guo *et al.*, 2020; Sun *et al.*, 2020]. However, hard sharing approaches have limited flexibility. Some soft parameter sharing architectures [Liu *et al.*, 2019; Sun *et al.*, 2021] instead transfer knowledge across tasks by linearly combining intermediate representations of each task [Misra *et al.*, 2016; Ruder *et al.*, 2019], or using multiple expert networks [Ma *et al.*, 2018]. However, these approaches are parameter-inefficient as they require learning of entire network weights for each task. Transferring knowledge by factorising model weights across tasks can improve parameter efficiency but has limited expressiveness [Yang and Hospedales, 2017]. By instead transferring knowledge through a deep hypernetwork weight generator, our approach can learn abstract task relationships and is substantially more parameter efficient than these architectures as it only learns a small embedding vector for each task.

Multi-task learning with metadata. Leveraging natural language task descriptions has been explored in natural language processing [Ye and Ren, 2021; You *et al.*, 2016] and reinforcement learning [Sodhani *et al.*, 2021]. One prior study leverages metadata for MTL, by clustering tasks on their metadata and then modelling task groups with hard parameter sharing [Zheng *et al.*, 2019]. However, it is incompatible with continuous metadata and therefore not applicable to the problem of water quality remote sensing. Further, this unsupervised metadata approach cannot learn metadata feature importances, interactions, or relationships with task functions, and is limited by its hard parameter sharing architecture.

Task embeddings. In a setting with many available historical models, task embeddings are used to select a model for a new MTL problem [Zhang *et al.*, 2018]. Task embeddings are also used to condition task-specific decoders [Sun *et al.*, 2021]; while this approach is limited to dense image prediction tasks, by using a hypernetwork our work is able to condition general network architectures on task embeddings.

2.2 Hypernetworks

Another field of neural architecture research is hypernetworks, originally introduced as a method to compress model weights [Ha *et al.*, 2017], but since applied to diverse domains [Littwin and Wolf, 2019; Shamsian *et al.*, 2021]. Transformer-based architectures use hypernetworks to condition task-specific prompts and adapters [Ye and Ren, 2021; Üstün *et al.*, 2022; Liu *et al.*, 2022].

Hypernetworks can inject model-level information into deep learning systems, such as our motivation of exploiting task-level metadata. In continual learning, frozen task-specific embeddings are a memory-efficient way to store previous task models [von Oswald *et al.*, 2020]. In MTL, hypernetworks can incorporate user-defined task preferences and compute requirements in a hard parameter sharing architecture search [Raychaudhuri *et al.*, 2022], or learn the pareto front of multi-objective task optimisation [Navon *et al.*, 2020]. These works generate weights for an entire hard parameter sharing MTL network conditioned on a model-level input of interest. In contrast, our work generates the weights for a single task network conditioned on task features, using a hypernetwork as a method for soft parameter sharing between tasks by generating multiple target networks.

3 Multi-Task Hypernetwork

Problem formulation. We define the multi-task learning problem as follows. Consider a set of T related tasks $\{\mathcal{T}_t\}_{t=1}^T = \{(\mathcal{D}_t, f_t)\}_{t=1}^T$. Each task $\mathcal{T}_t$ has a deep learning function $\{f_t : \mathcal{X} \mapsto \mathcal{Y}\}_{t=1}^T$, and a set of training data $\{\mathcal{D}_t\}_{t=1}^T$ which contains n_t training instances $\mathcal{D}_t = \{\boldsymbol{x}_i^t, y_i^t\}_{i=1}^{n_t}$ where $\boldsymbol{x}_i^t \in \mathcal{X}, y_i^t \in \mathcal{Y}$. The aim of MTL is to jointly learn each task function $\{f_t\}_{t=1}^T$ from the training data of all tasks $\{\mathcal{D}_t\}_{t=1}^T$.

Metadata. Metadata is data that describes and gives information about other data [Zheng *et al.*, 2019]. In the context of MTL, we define metadata as task-level data, that is, data describing the tasks themselves. For example, lake attributes in a problem where each lake is associated with a different task, or mechanical information about robot arms when modelling each arm's movements as its own task. Metadata may come from auxiliary datasets or from domain knowledge. We note that for some MTL applications, metadata is unavailable or undefined. Consider the learning of semantic segmentation and surface normal of an image as a multi-task problem [Misra *et al.*, 2016]; no relationship between these two tasks exists which can be expressed with task-level features.

Problem formulation with metadata. When metadata is available, we modify the multi-task learning problem formulation as follows. Each task $\mathcal{T}_t$ additionally has an associated metadata feature vector $\boldsymbol{m}_t$ of size d_m, such that $\mathcal{T}_t = (\mathcal{D}_t, \boldsymbol{m}_t, f_t)$. The aim of MTL with metadata is therefore to jointly learn the task functions $\{f_t\}_{t=1}^T$ from the metadata $\{\boldsymbol{m}_t\}_{t=1}^T$ as well as from the task training data $\{\mathcal{D}_t\}_{t=1}^T$. When metadata is unavailable, we use the original MTL problem formulation. If this metadata feature vector is informative with regards to task functions, leveraging this additional source of knowledge may improve the performance of MTL models. However, current approaches are not designed to effectively leverage metadata. A naive approach to learn from metadata with existing techniques would be to append it as supplementary features to training data $\{\hat{\boldsymbol{x}}_i^t\}_{i=1}^{n_t} = \{(\boldsymbol{x}_i^t, \boldsymbol{m}_t)\}_{i=1}^{n_t}$. This may not be an effective way to exploit this data as the appended features are constant for all data within a task.

3.1 Method Overview

The foundation of our MTL architecture is a hypernetwork h, which is shared by all tasks. The network weights θ_t of target function f_t are not trained directly. Instead, they are generated by h according to $\theta_t = h(\boldsymbol{e}_t)$, where $\boldsymbol{e}_t$ is a d_e dimensional embedding vector associated with task t.

The hypernetwork generates the weights for each task's target network. Task-specific embeddings provide the flexibility for the hypernetwork to generate different weights for each target network. The learnt embeddings are a low-dimensional representations of target functions, allowing our model to efficiently learn task similarities and relationships.

Multi-Task Hypernetworks are therefore parameterised by the weights θ_h that define h, and T task-specific embedding vectors $\{\boldsymbol{e}_t\}_{t=1}^T$. Most model parameters are shared between all tasks since $|\theta_h| \gg Td_e$. Despite this, our approach is best classified as soft parameter sharing, as the target networks are generated with flexible weights at all network layers. To perform inference on task t, we compute the weights of f_t with h then predict $\boldsymbol{x}$ with f_t. This is performed for all tasks in parallel. During training, loss gradients are backpropagated through the target networks, then pass through the hypernetwork weights and task embeddings, such that the entire architecture can be trained directly with any gradient optimizer. Task embeddings are trained identically to other parameters.

When metadata $\boldsymbol{m}_t$ is available, Multi-Task Hypernetworks can naturally leverage it by appending it to the task embeddings as input to the hypernetwork $\theta_t = h(\boldsymbol{e}_t, \boldsymbol{m}_t)$. Unlike task embeddings, metadata are treated as constants and are frozen during training. Considering that task-specific embeddings allow Multi-Task Hypernetworks to learn task differences and relationships, by appending static metadata to the embeddings, the metadata acts as pre-learnt embedding priors from an auxiliary source. Including informative metadata may therefore improve the model's ability to learn task relationships. Further, under the hypernetwork framework, the weights that define a task's target network are a function of that task's metadata. As our model is trained with metadata over multiple tasks, the hypernetwork will directly learn the relationship between the metadata and the task functions. Our

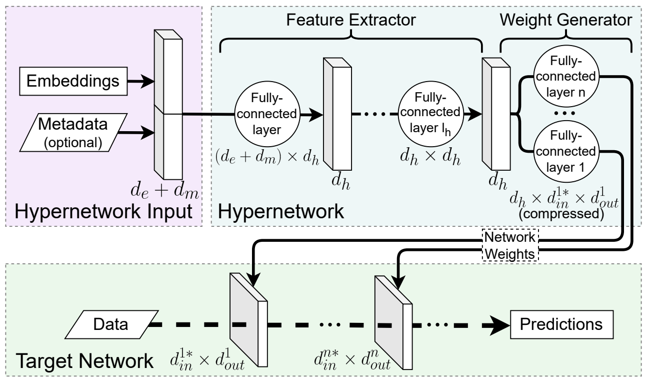


Figure 3: Multi-Task Hypernetwork architecture for a single task. The input to the hypernetwork (task embeddings and optionally task metadata) passes through the hypernetwork feature extractor layers. The resulting d_h dimensional latent vector passes through the hypernetwork weight generator layers to generate the target network weights. The weight generator layers use compressed weight tensors to reduce model parameters. The target network then makes predictions for that task.

architecture is therefore able to leverage metadata in learning both task relationships and task functions explicitly.

3.2 Hypernetwork Architecture

The hypernetwork maps from a low-dimensional input task embedding (and optionally task metadata) to the weight matrices that define a task network. Our hypernetwork architecture, displayed in Figure 3, achieves this with two components: a feature extractor and a weight generator.

To model the water quality remote sensing problem, the hypernetwork architecture described in this section generates linear target networks. Our approach can model different target network architectures by modifying the hypernetwork weight generator to instead generate different weight tensors, such as the convolutional layer generation in the original hypernetwork implementation [Ha *et al.*, 2017]. We note that hypernetworks are used for efficient weight generation of various target network architectures [Ha *et al.*, 2017; Lin *et al.*, 2020; Ye and Ren, 2021], indicating that our approach would be viable in diverse settings.

The hypernetwork takes as input a task embedding (and optionally task metadata) of dimension d_e (or $d_e + d_m$). The feature extractor is a feedforward neural network with l_h linear layers of d_h neurons, allowing our model to learn abstract representations of task embeddings and metadata. Similarly to features in a typical neural network, metadata is normalised to unit mean and variance.

The weight generator consists of linear layers mapping from the resulting d_h dimensional latent space to weight matrices which each define a layer in the target network. Let layer n of the target network be a linear layer with d_{in}^n inputs and d_{out}^n outputs. This layer has a weight matrix of size $d_{in}^n \times d_{out}^n$ and a bias of size d_{out}^n, for a total of $(d_{in}^n + 1) \times d_{out}^n$ parameters. For simplicity, let $d_{in}^{n*} = d_{in}^n + 1$, such that layer n has $d_{in}^{n*} \times d_{out}^n$ parameters. To generate these parameters, the hypernetwork requires a linear mapping from d_h to $d_{in}^{n*} \times d_{out}^n$. This can be achieved with a hypernetwork weight matrix $\boldsymbol{W}$ of size $d_h \times d_{in}^{n*} \times d_{out}^n$ and a bias $\boldsymbol{b}$ of size $d_{in}^{n*} \times d_{out}^n$. However, $\boldsymbol{W}$ contains d_h times as many parameters as the target network layer, which is generally prohibitively large.

Weight compression. To reduce the number of parameters in the hypernetwork weight generator, we implement a simple tensor compression technique for hypernetworks generating linear layers. This technique is analogous to the convolution chunking technique in the original hypernetwork implementation [Ha *et al.*, 2017]. Instead of learning $\boldsymbol{W}$ directly, we learn three smaller weight matrices, $\boldsymbol{W}^A, \boldsymbol{W}^B, \boldsymbol{W}^C$, of size $d_h \times d_{in}^{n*}, d_h \times d_{out}^n, d_{in}^{n*} \times d_{out}^n$ respectively. These matrices expand into $\boldsymbol{W}$ as follows: $\boldsymbol{W}_{ijk} = \boldsymbol{W}_{ij}^A \boldsymbol{W}_{ik}^B \boldsymbol{W}_{jk}^C$. This weight compression substantially reduces the total number of model parameters while remaining trainable via standard backpropagation. These three matrices each learn interaction terms between two of the three dimensions of $\boldsymbol{W}$: task features, target network layer input features and target network layer output features.

Complexity. The total number of parameters in our Multi-Task Hypernetwork is given by:

$$P = \underbrace{(d_e + d_m + d_h(l_h - 1))(d_h + 1)}_{\text{hypernetwork feature extractor}}$$
$$+ \underbrace{\textstyle\sum_{i=1}^{l_t} \left(2d_{in}^{i*} d_{out}^i + d_h d_{in}^{i*} + d_h d_{out}^i\right)}_{\text{hypernetwork weight generator}}$$
$$+ \underbrace{T d_e}_{\text{task embeddings}}$$

where l_t is the number of layers in the target network. Note that a single target network has $\sum_{i=1}^{l_t} d_{in}^{i*} d_{out}^i$ total parameters, or half of the first term representing the hypernetwork weight generator parameters. Despite having more parameters than a single target network, the hypernetwork weights are shared for all tasks, such that our approach requires only d_e additional parameters per task. This is substantially fewer than other soft parameter sharing MTL architectures, and in practice, our method has fewer parameters when modelling more than a few tasks. An experimental analysis of computational complexity validates that our method is suitable for training large target networks, but is omitted due to space constraints.

4 Experiments

We evaluate the effectiveness of our approach by comparing its performance to baseline methods, in the typical MTL setting without metadata, and with metadata. We include metadata in baseline methods naively as described in Section 3.

Experimental setup. Experiments were repeated across 50 seeds for each combination of hyperparameters. Data was partitioned into training, validation, and test sets in a 60/20/20 ratio. Performance on the validation set was used for hyperparameter selection and early stopping. Hyperparameter selection was performed independently with and without metadata. Regarding general hyperparameters, all methods used

Dataset	N	T	Task represents	Data features	Data label	Task metadata
Water Quality	796	88	a lake	multispectral lake colour ($D = 16$)	concentration of chlorophyll-a ($D = 1$)	features of lake *e.g. depth*, *weather* ($D = 11$)
Cubic	200	20	a cubic function	x ($D = 1$)	y ($D = 1$)	polynomial coefficients *i.e.* a, b, c, d ($D = 4$)
Algorithms	422	24	an ML algorithm	dataset statistics ($D = 11$)	performance on dataset ($D = 1$)	algorithm labels *e.g. is_tree*, *is_kernel* ($D = 8$)
Robot Arm	400	20	a robot arm	arm end position ($D = 2$)	arm joint angles ($D = 3$)	arm joint lengths ($D = 3$)

Table 1: Summary of datasets used in experiments. N represents total training instances, D represents data dimensionality. All datasets other than Water Quality have equal task sizes.

the learning rate $\eta \in \{10^p | p = -3, -3.5, -4, -4.5\}$ and random batches of size 64. All models used a network architecture of three fully connected layers of 32 neurons with ReLU activations. We empirically found that networks of this size had sufficient representation capacity to effectively model the test datasets. Regarding Multi-Task Hypernetwork hyperparameters, we tuned the number of layers in the hypernetwork feature extractor $l_h \in \{0, 1, 2\}$ where each layer had 32 neurons with ReLU activations. Task embeddings used $d_e = 10$ parameters and were initialised with a uniform distribution and variance 1, using identical initialisations across tasks. All experiments are implemented in PyTorch with a GeForce RTX 3080 GPU.

Water Quality dataset. This dataset was produced in collaboration with local council. We combined the GLORIA dataset for hyperspectral remote sensing reflectance measurements and co-located water quality attributes [Lehmann *et al.*, 2023] with the LakesATLAS dataset for global lake attribute data [Lehner *et al.*, 2022]. Lakes with at least five data points were included, resulting in 88 diverse lakes spanning glacial, riverine, volcanic origins, and artificial reservoirs, with mean chlorophyll-a concentrations ranging 0.7 to 290 mg/m^3.

Other datasets. To analyse the general effectiveness of our approach, we include three additional multi-task tabular regression datasets with metadata and limited data instances (Table 1). **Cubic** is a synthetic dataset which we design to have a simple and complete relationship between task metadata and the task functions. Each task is a third order 1D polynomial defined over $x \in [-1, 1]$ with random coefficients, where the goal is to estimate y given x. Polynomial coefficients are task metadata. The **Algorithms** dataset [Brazdil *et al.*, 1994] aims to estimate the performance of traditional ML algorithms on datasets, given dataset summary statistics, where each algorithm is a task. Metadata is not from auxiliary sources but manually constructed from domain knowledge by naively categorising algorithms across binary labels *e.g.* "is_tree", "is_kernel". The **Robot Arm** dataset [Duka, 2014] aims to estimate robot arm joint angles given the end arm position. Each task is highly nonlinear and represents an arm with different joint lengths, which are the task metadata.

Baseline methods. We include **Hard sharing** [Caruana, 1997], **MRN** (Multilinear Relational Networks) [Long *et al.*, 2017], **MaxRoam** (Maximum Roaming Networks) [Pascal *et al.*, 2021], **Cross-stitch** networks [Misra *et al.*, 2016], **Sluice** networks [Ruder *et al.*, 2019], **DMTRL** (Deep Multi-Task Representation Learning) [Yang and Hospedales, 2017]. Regarding traditional approaches, **STL-naive** is a single task learning network with no task information. **STL** is a single task learning network which is given task information to learn task differences, as metadata if available or as one-hot task embeddings otherwise. We exclude methods which use domain-specific components rendering them incompatible with tabular problems, from computer vision [Liu *et al.*, 2022; Sun *et al.*, 2021; Liu *et al.*, 2019; Bhattacharjee *et al.*, 2022; Lopes *et al.*, 2023] and natural language processing [Tay *et al.*, 2021; Üstün *et al.*, 2022; Ye and Ren, 2021].

4.1 Overall Performance

Results are displayed in Table 2 as mean $\pm$ standard error. To estimate the overall difference in performance between the methods for all datasets, we use a Friedman test with Nemenyi post-hoc test on the results, displayed in Figure 4.

Without metadata. On the Water Quality dataset, Multi-Task Hypernetworks achieve RMSE 20% lower than all other methods. STL achieved the second lowest RMSE, indicating that other MTL algorithms struggle to model problems with many diverse tasks, and highlighting the benefits of model flexibility and parameter efficiency for this application. In this setting, our approach also achieves the lowest RMSE for Algorithms and Robot Arm. The Friedman-Nemenyi test shows that overall, our method outperforms all baseline methods other than Sluice networks on these datasets, indicating that our architecture can outperform many existing MTL architectures in this setting.

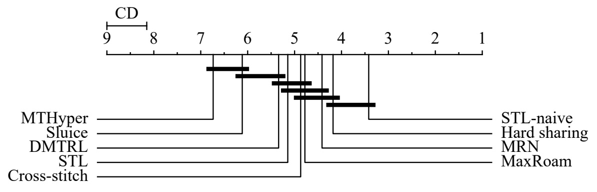
With metadata. Multi-Task Hypernetworks again outperform other methods by 20% on Water Quality. Overall, practitioners using our algorithm and leveraging lake metadata can substantially increase the accuracy of their chlorophyll-a estimations compared to existing modelling approaches. Our approach also achieves the lowest RMSE for all datasets. The Friedman-Nemenyi test shows that overall, our method substantially outperforms all baseline methods. Interestingly, despite not being designed for multi-task problems, STL overall outperforms all the multi-task learning methods except sluice networks. This indicates that current MTL architectures are not able to leverage metadata effectively.

Effects of Metadata

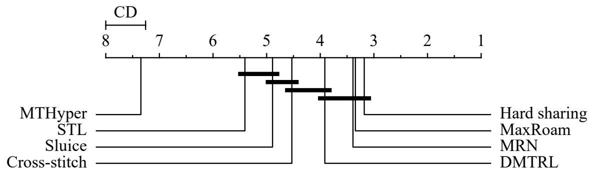
To analyse the impact of metadata on model performance, we consider the completeness, noisiness and complexity of the

	Method	Water Quality	Cubic	Algorithms	Robot Arm	P_{avg}
No Metadata	STL-naive	0.521 ± 0.007	0.624 ± 0.001	1.950 ± 0.062	0.559 ± 0.001	2.5k
	STL	0.512 ± 0.010	0.132 ± 0.007	1.465 ± 0.045	0.547 ± 0.002	3.7k
	Hard sharing	0.851 ± 0.041	**0.040** ± 0.002	1.553 ± 0.052	0.596 ± 0.004	4.0k
	MRN	0.911 ± 0.045	0.053 ± 0.003	1.488 ± 0.052	0.565 ± 0.003	7.3k
	MaxRoam	0.783 ± 0.042	**0.036** ± 0.002	1.505 ± 0.051	0.585 ± 0.004	9.2k
	Cross-stitch	0.554 ± 0.020	0.057 ± 0.005	**1.387** ± 0.049	0.624 ± 0.001	105.5k
	Sluice	0.534 ± 0.018	0.064 ± 0.007	1.447 ± 0.052	0.540 ± 0.003	126.0k
	DMTRL	0.535 ± 0.017	**0.038** ± 0.001	**1.342** ± 0.044	0.625 ± 0.001	20.0k
	MT Hypernet (ours)	**0.411** ± 0.004	0.122 ± 0.008	**1.397** ± 0.048	**0.485** ± 0.002	10.1k
Metadata	STL	0.489 ± 0.008	0.030 ± 0.001	1.607 ± 0.049	**0.469** ± 0.002	2.7k
	Hard sharing	0.973 ± 0.045	0.034 ± 0.001	1.520 ± 0.050	0.598 ± 0.004	4.2k
	MRN	0.938 ± 0.056	0.041 ± 0.003	1.479 ± 0.055	0.575 ± 0.005	7.6k
	MaxRoam	0.781 ± 0.031	0.036 ± 0.001	1.489 ± 0.048	0.589 ± 0.003	9.5k
	Cross-stitch	0.543 ± 0.015	0.032 ± 0.001	**1.325** ± 0.045	0.624 ± 0.001	115.9k
	Sluice	0.559 ± 0.020	0.038 ± 0.003	1.403 ± 0.049	0.546 ± 0.005	136.4k
	DMTRL	0.560 ± 0.016	0.034 ± 0.001	**1.344** ± 0.046	0.630 ± 0.001	16.0k
	MT Hypernet (ours)	**0.395** ± 0.004	**0.023** ± 0.000	**1.307** ± 0.046	**0.467** ± 0.001	14.4k
Improvement	STL	0.023 ± 0.012	0.102 ± 0.007	-0.142 ± 0.033	0.078 ± 0.002	-
	Hard sharing	-0.122 ± 0.051	0.006 ± 0.001	0.033 ± 0.036	-0.001 ± 0.004	-
	MRN	-0.026 ± 0.065	0.012 ± 0.002	0.009 ± 0.024	-0.010 ± 0.004	-
	MaxRoam	0.002 ± 0.049	0.001 ± 0.001	0.015 ± 0.021	-0.004 ± 0.003	-
	Cross-stitch	0.011 ± 0.020	0.024 ± 0.005	0.062 ± 0.021	0.000 ± 0.001	-
	Sluice	-0.026 ± 0.018	0.027 ± 0.006	0.043 ± 0.020	-0.006 ± 0.003	-
	DMTRL	-0.025 ± 0.017	0.004 ± 0.001	-0.002 ± 0.026	-0.005 ± 0.001	-
	MT Hypernet (ours)	0.015 ± 0.005	0.099 ± 0.008	0.090 ± 0.028	0.019 ± 0.001	-

Table 2: Average task performance (RMSE) of our approach and baseline methods: with metadata, without metadata, and improvement in performance (reduction in RMSE) from using metadata. P_{avg} represents the average number of network parameters across the four datasets with optimal hyperparameters, displayed in thousands.



(a) without metadata



(b) with metadata

Figure 4: Critical difference diagram for Nemenyi significance test on all datasets. A higher score indicates better overall performance.

metadata. The metadata of the Water Quality dataset are insufficient to entirely represent lake task functions, as lakes have many bio-geo-optical factors which cannot be entirely represented by 11 features, so we describe this metadata as incomplete. The metadata is sensor data, and hence contains noise. The metadata is also complicated, as these metadata features have abstract interactions. We observe that this metadata improves the performance of Multi-Task Hypernetworks and STL, but not other methods. This indicates that our approach can extract knowledge even from complex metadata, whereas other MTL approaches cannot.

Regarding the Cubic dataset, the metadata is complete as the metadata features define the task functions, is generated without noise, and is simple since features represent coefficients which are linearly related to the task function. We observe that this straightforward metadata improves the performance of all methods except MaxRoam, but that Multi-Task Hypernetworks outperform other methods by 25%. Robot Arm metadata is also complete and generated noiselessly, but is complicated, as the relationship between the robot arm lengths and the task functions is highly nonlinear. Similar to Water Quality, we observe that this complicated metadata only improves the performance of Multi-Task Hypernetworks and STL, further indicating that our approach can extract knowledge from complex metadata while other MTL approaches cannot. Algorithms metadata is manually labelled and therefore noiseless. It is simple, but incomplete as algorithm outcomes cannot be entirely represented by high-level categorical descriptions. Despite this, metadata improves the performance of Multi-Task Hypernetworks, Cross-stitch and Sluice networks. This demonstrates the utility of metadata in enhancing model performance, even when naively labelled.

Model parameters. In our experiments, our architecture has fewer network parameters than other soft sharing approaches of equal capacity, with an order of magnitude fewer parameters than Cross-stitch and Sluice networks, and fewer than DMTRL despite this method being designed to minimise network parameters.

4.2 Ablation Study

We carry out an ablation study to provide insight into the contributions of the different components in our architecture with and without metadata, displayed in Table 3.

	Method	Water Quality	Cubic	Algorithms	Robot Arm
None	MT Hypernet	**0.411**±0.004	**0.122**±0.008	**1.421**±0.040	**0.485**±0.002
	no compression	0.493±0.007	**0.103**±0.013	1.555±0.052	0.656±0.014
	no feature extractor	**0.413**±0.005	**0.122**±0.008	**1.421**±0.040	**0.485**±0.002
Metadata	MT Hypernet	**0.395**±0.004	**0.023**±0.000	**1.322**±0.040	**0.467**±0.001
	no compression	0.619±0.024	0.025±0.001	1.935±0.058	0.528±0.003
	no feature extractor	0.470±0.007	**0.023**±0.000	**1.342**±0.034	0.474±0.002
	no embeddings	0.434±0.006	0.024±0.000	1.460±0.046	**0.467**±0.001
	random metadata	0.406±0.006	0.066±0.004	1.506±0.050	0.517±0.005

Table 3: Ablation study of Multi-Task Hypernetworks (RMSE).

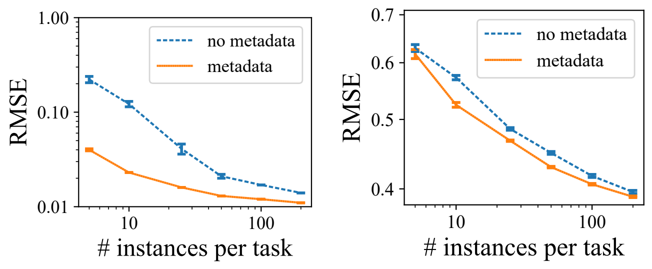
Weight matrix compression reduces overparameterisation. Using full-sized weight matrices in the hypernetwork weight generator without the matrix compression technique described in Section 3.2 substantially reduces performance for all datasets except Cubic without metadata. Without compression, our architecture has many more parameters and is more flexible, indicating that the weight matrix compression reduces overparameterisation and increases generalisability.

Feature extractor learns metadata representations. Removing the feature extractor layers from the hypernetwork decreases performance for Water Quality and Robot Arm with metadata. This may be because these datasets have complicated metadata, and so benefit from a non-linear feature extractor to learn abstract metadata representations. Datasets with simple metadata, or with learnt task embeddings but no metadata, may not benefit from this deep learning component. An optimal Multi-Task Hypernetwork architecture for each dataset can be found by tuning the depth of the feature extractor l_h, as is done in experiments in Section 4.1.

Task embeddings provide flexibility. Using only metadata as input to the hypernetwork without task-specific embeddings decreases performance in all datasets except Robot Arm. This may be because trainable embeddings give the hypernetwork degrees of freedom between tasks, to learn task relationships and differences more flexibly than with only static metadata. As Robot Arm has complete metadata which captures all task differences, this dataset may not require the additional flexibility from learnable task embeddings. We experimentally demonstrate that task embeddings learn "meaningful" task representations which are predictive of task-level knowledge, but omit this due to space constraints.

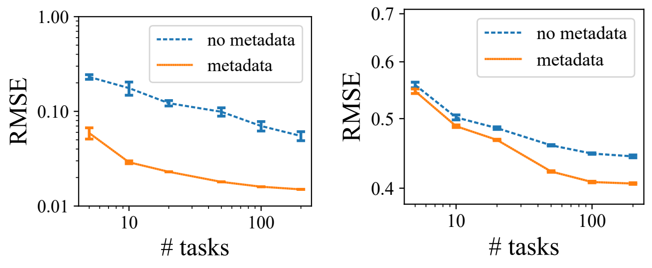
Metadata is informative. Randomising metadata by independently shuffling metadata features between tasks decreases performance in all datasets, indicating that our architecture can extract knowledge from informative metadata.

Dataset size affects metadata performance gain. The relationship between dataset size and the benefits of leveraging metadata is examined through sensitivity analysis on Cubic and Robot Arm datasets, which can be produced in various sizes. We vary the number of training instances per task (Figure 5) and the number of tasks in each dataset (Figure 6). Increasing the number of training instances per task decreases the gain in performance from leveraging metadata. Intuitively, as the task size increases, the model is better able to learn task functions and relationships implicitly from joint optimisation without metadata. This indicates that metadata



(a) Cubic (b) Robot Arm

Figure 5: Multi-Task Hypernetwork performance vs. training instances per task with and without metadata.



(a) Cubic (b) Robot Arm

Figure 6: Multi-Task Hypernetwork performance vs. number of tasks for datasets with and without metadata.

is most useful for small datasets. Increasing the number of tasks increases the gain in performance from leveraging metadata for Robot Arm and for Cubic with fewer than ten tasks. Intuitively, more tasks provide the hypernetwork with more metadata samples to learn the relationship between metadata and task functions. Cubic, with simpler metadata, may need fewer tasks to learn this relationship, while the complex Robot Arm metadata benefits from more tasks.

5 Conclusion

We propose Multi-Task Hypernetworks, a novel architecture for multi-task learning, which learns flexible task functions with fewer parameters than existing soft parameter sharing approaches. We formalise task-level data, called metadata, into the MTL problem. Uniquely, our architecture can leverage metadata to learn task functions and relationships explicitly. We reveal the potential for metadata to enhance performance on MTL problems, particularly with few data samples, but show that existing methods cannot effectively leverage it. We show that our architecture is effective for multi-task learning and effectively extracts knowledge from informative metadata. Our approach substantially improves the accuracy of remote sensing water quality estimation, a challenging problem which contains many diverse tasks with few samples, enabling more accurate global water monitoring without additional expensive data collection. A limitation of our work is the current lack of understanding on evaluating whether metadata will improve learning. Future research could explore the potential of transfer learning to lakes with no training data, generating task networks from metadata alone.

Acknowledgments

This research is partially funded by the New Zealand MBIE TAIAO data science programme. We thank Henry Gouk for his insight.

References

[Bhattacharjee *et al.*, 2022] Deblina Bhattacharjee, Tong Zhang, Sabine Süsstrunk, and Mathieu Salzmann. Mult: an end-to-end multitask learning transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12031–12041, 2022.

[Brazdil *et al.*, 1994] Pavel Brazdil, João Gama, and Bob Henery. Characterizing the applicability of classification algorithms using meta-level learning. In *European Conference on Machine Learning*, pages 83–102. Springer, 1994.

[Caruana, 1997] Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

[Duka, 2014] Adrian-Vasile Duka. Neural network based inverse kinematics solution for trajectory tracking of a robotic arm. *Procedia Technology*, 12:20–27, 2014.

[Graffeuille *et al.*, 2022] Olivier Graffeuille, Yun Sing Koh, Jörg Wicker, and Moritz K Lehmann. Semi-supervised conditional density estimation with wasserstein laplacian regularisation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6746–6754, 2022.

[Guo *et al.*, 2020] Pengsheng Guo, Chen-Yu Lee, and Daniel Ulbricht. Learning to branch for multi-task learning. In *International Conference on Machine Learning*, pages 3854–3863. PMLR, 2020.

[Ha *et al.*, 2017] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. In *Proceedings of the 5th International Conference on Learning Representations*, 2017.

[Hou *et al.*, 2022] Xuejiao Hou, Lian Feng, Yanhui Dai, Chuanmin Hu, Luke Gibson, Jing Tang, Zhongping Lee, Ying Wang, Xiaobin Cai, Junguo Liu, et al. Global mapping reveals increase in lacustrine algal blooms over the past decade. *Nature Geoscience*, 15(2):130–134, 2022.

[Lehmann *et al.*, 2023] Moritz K. Lehmann, Daniela Gurlin, Nima Pahlevan, Krista Alikas, Janet Anstee, Sundarabalan V. Balasubramanian, Cláudio C.F. Barbosa, Caren Binding, Astrid Bracher, Mariano Bresciani, Ashley Burtner, Zhigang Cao, Arnold G. Dekker, Nathan Drayson, Reagan M. Errera, Virginia Fernandez, Cédric G. Fichot, Peter Gege, Claudia Giardino, Anatoly A. Gitelson, Steven R. Greb, Hayden Henderson, Hiroto Higa, Abolfazl Irani Rahaghi, Cédric Jamet, Dalin Jiang, Kersti Kangro, Raphael Kudela, Lin Li, Martin Ligi, Hubert Loisel, Steven Lohrenz, Ronghua Ma, Daniel A. Maciel, Tim J. Malthus, Bunkei Matsushita, Camille Minaudo, Deepak R. Mishra, Sachidananda Mishra, Tim Moore, Wesley J. Moses, Hà Nguyễn, Evlyn M.L.M. Novo, Stéfani Novoa, Daniel Odermatt, David M. O'Donnell, Leif G. Olmanson, Michael Ondrusek, Natascha Oppelt, Waterloo Pereira Filho, Stefan Plattner, Antonio Ruiz Verdú, Salem I. Salem, John F. Schalles, Stefan G.H. Simis, Eko Siswanto, Brandon Smith, Ian Somlai-Schweiger, Mariana A. Soppa, Evangelos Spyrakos, Hendrik J. van der Woerd, Andrea Vander Woude, Vincent Vantrepotte, Marcel R. Wernand, Mortimer Werther, Linwei Yue, Thomas Jordan, Jeremy A. Kravitz, Arne S. Kristoffersen, Mark Matthews, Elinor Tessin, Ryan A. Vandermeulen, Dariusz Ficek, Courtney Di Vittorio, and Kyana Young. Gloria-a globally representative hyperspectral in situ dataset for optical sensing of water quality. *Scientific Data*, 10(1):100, 2023.

[Lehner *et al.*, 2022] Bernhard Lehner, Mathis L Messager, Maartje C Korver, and Simon Linke. Global hydro-environmental lake characteristics at high spatial resolution. *Scientific Data*, 9(1):1–19, 2022.

[Lin *et al.*, 2020] Xi Lin, Zhiyuan Yang, Qingfu Zhang, and Sam Kwong. Controllable pareto multi-task learning. *arXiv preprint arXiv:2010.06313*, 2020.

[Littwin and Wolf, 2019] Gidi Littwin and Lior Wolf. Deep meta functionals for shape representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1824–1833, 2019.

[Liu *et al.*, 2019] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1871–1880, 2019.

[Liu *et al.*, 2022] Yen-Cheng Liu, Chih-Yao Ma, Junjiao Tian, Zijian He, and Zsolt Kira. Polyhistor: Parameter-efficient multi-task adaptation for dense vision tasks. *Advances in Neural Information Processing Systems*, 35:36889–36901, 2022.

[Long *et al.*, 2017] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Philip S Yu. Learning multiple tasks with multilinear relationship networks. *Advances in Neural Information Processing Systems*, 30, 2017.

[Lopes *et al.*, 2023] Ivan Lopes, Tuan-Hung Vu, and Raoul de Charette. Cross-task attention mechanism for dense multi-task learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2329–2338, 2023.

[Ma *et al.*, 2018] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1930–1939, 2018.

[Misra *et al.*, 2016] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3994–4003, 2016.

[Navon *et al.*, 2020] Aviv Navon, Aviv Shamsian, Ethan Fetaya, and Gal Chechik. Learning the pareto front with hypernetworks. In *International Conference on Learning Representations*, 2020.

[Neil *et al.*, 2019] Claire Neil, Evangelos Spyrakos, Peter D Hunter, and Andrew N Tyler. A global approach for chlorophyll-a retrieval across optically complex inland waters based on optical water types. *Remote Sensing of Environment*, 229:159–178, 2019.

[Pahlevan *et al.*, 2020] Nima Pahlevan, Brandon Smith, John Schalles, Caren Binding, Zhigang Cao, Krista Ma, Ronghua anfd Alikas, Kersti Kangro, Daniela Gurlin, Nguyễn Hà, et al. Seamless retrievals of chlorophyll-a from sentinel-2 (msi) and sentinel-3 (olci) in inland and coastal waters: A machine-learning approach. *Remote Sensing of Environment*, 240:111604, 2020.

[Pascal *et al.*, 2021] Lucas Pascal, Pietro Michiardi, Xavier Bost, Benoit Huet, and Maria A Zuluaga. Maximum roaming multi-task learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9331–9341, 2021.

[Raychaudhuri *et al.*, 2022] Dripta S Raychaudhuri, Yumin Suh, Samuel Schulter, Xiang Yu, Masoud Faraki, Amit K Roy-Chowdhury, and Manmohan Chandraker. Controllable dynamic multi-task architectures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10955–10964, 2022.

[Ruder *et al.*, 2019] Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. Latent multi-task architecture learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4822–4829, 2019.

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

[Shamsian *et al.*, 2021] Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks. In *International Conference on Machine Learning*, pages 9489–9502. PMLR, 2021.

[Shui *et al.*, 2019] Changjian Shui, Mahdieh Abbasi, Louis-Émile Robitaille, Boyu Wang, and Christian Gagné. A principled approach for learning task similarity in multi-task learning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3446–3452, 2019.

[Sodhani *et al.*, 2021] Shagun Sodhani, Amy Zhang, and Joelle Pineau. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, pages 9767–9779. PMLR, 2021.

[Sun *et al.*, 2020] Ximeng Sun, Rameswar Panda, Rogerio Feris, and Kate Saenko. Adashare: Learning what to share for efficient deep multi-task learning. *Advances in Neural Information Processing Systems*, 33:8728–8740, 2020.

[Sun *et al.*, 2021] Guolei Sun, Thomas Probst, Danda Pani Paudel, Nikola Popović, Menelaos Kanakis, Jagruti Patel, Dengxin Dai, and Luc Van Gool. Task switching network for multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8291–8300, 2021.

[Tay *et al.*, 2021] Yi Tay, Zhe Zhao, Dara Bahri, Donald Metzler, and Da-Cheng Juan. Hypergrid transformers: Towards a single model for multiple tasks. In *International Conference on Learning Representations*, 2021.

[United Nations, 2015] United Nations. Transforming our world: the 2030 agenda for sustainable development, 2015. Accessed: 2014-01-15.

[United Nations, 2016] United Nations. Leave no one behind, 2016. Accessed: 2014-01-15.

[Üstün *et al.*, 2022] Ahmet Üstün, Arianna Bisazza, Gosse Bouma, Gertjan van Noord, and Sebastian Ruder. Hyper-x: A unified hypernetwork for multi-task multilingual transfer. In *Proceedings of EMNLP 2022*. Association for Computational Linguistics, 2022.

[von Oswald *et al.*, 2020] Johannes von Oswald, Christian Henning, Benjamin F Grewe, and João Sacramento. Continual learning with hypernetworks. In *Proceedings of the 8th International Conference on Learning Representations*, 2020.

[Yang and Hospedales, 2017] Yongxin Yang and Timothy Hospedales. Deep multi-task representation learning: A tensor factorisation approach. In *5th International Conference on Learning Representations*, 2017.

[Yang *et al.*, 2022] Xiao Yang, Catherine M. O'Reilly, John R. Gardner, Matthew R. V. Ross, Simon N. Topp, Jida Wang, and Tamlin M. Pavelsky. The color of earth's lakes. *Geophysical Research Letters*, 49, 2022.

[Ye and Ren, 2021] Qinyuan Ye and Xiang Ren. Learning to generate task-specific adapters from task description. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 646–653, 2021.

[You *et al.*, 2016] Qiang You, Ou Wu, Guan Luo, and Weiming Hu. Metadata-based clustered multi-task learning for thread mining in web communities. In *Machine Learning and Data Mining in Pattern Recognition: 12th International Conference, MLDM 2016, New York, NY, USA, July 16-21, 2016, Proceedings*, pages 421–434. Springer, 2016.

[Zamir *et al.*, 2018] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018.

[Zhang *et al.*, 2018] Yu Zhang, Ying Wei, and Qiang Yang. Learning to multitask. *Advances in Neural Information Processing Systems*, 31, 2018.

[Zheng *et al.*, 2019] Zimu Zheng, Yuqi Wang, Quanyu Dai, Huadi Zheng, and Dan Wang. Metadata-driven task relation discovery for multi-task learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 4426–4431, 2019.