Thèse n° 9878

Few-shot Learning for Efficient and Effective Machine Learning Model Adaptation

Présentée le 14 juin 2024

Faculté des sciences et techniques de l'ingénieur
Groupe de scientifiques IEL
Programme doctoral en génie électrique

pour l'obtention du grade de Docteur ès Sciences

par

Arnout Jan J DEVOS

Acceptée sur proposition du jury

Prof. A. Skrivervik, présidente du jury
Dr J.-M. Sallese, directeur de thèse
Dr D. Krotov, rapporteur
Dr A. Ilic, rapporteur
Prof. C. Hongler, rapporteur

École
polytechnique
fédérale
de Lausanne

2024

Acknowledgements

This manuscript exists due to the support, advice, guidance, and contributions from numerous collaborators, friends, and family members. I would like to express my profound gratitude to several individuals who have assisted me throughout this journey.

First, I would like to thank my PhD advisor Jean-Michel Sallese, whose guidance and support have been crucial to bringing my doctoral journey to a good end. Besides being an outstanding scientist and educator, I am thankful for his commitment as a thesis director and beyond to my academic success and that of many others.

I would like to thank my committee members: Alexander Ilic, Dmitry Krotov, Clément Hongler, and Anja Skrivervik for finding the time to review my thesis and for their thoughtful discussions during the defense.

I am also grateful to Isabelle Buzzi, Patricia Hjelt, Angela Devenoge, and Holly Cogliati-Bauereis for their assistance with administrative duties.

During my PhD, I feel fortunate to have collaborated with and be empowered by many talented people. My appreciation goes to them, including Lina Qiu, Yatin Dandi, Carlos Medina Temme, Sylvain Chatel, Hong Zheng, Heather Selby, Olivier Gevaert, Victor Kristof, Andreas Maggiori, El Mahdi Chayti, Ali Benlalah, Klavdiia Naumova, Daniil Dmitriev, Praneeth Karimireddy, Behzad Bozorgtabar, Anna Susmelj, Martin Jaggi, Marie-Anne Hartley, Olivier Gevaert, Sandra Siby, Matthias Bethge, and last but not least Jean-Michel Sallese. A number of other people have been crucial and inspiring in bringing my PhD to a good end; they know who they are. I am especially thankful to Lina Qiu, who in addition to being an exceptionally insightful researcher in machine learning also has great leadership and perseverance, which got our joint work eventually successfully published. I am also particularly grateful to Yatin Dandi, an outstanding mathematical mind with an insatiable and infectious passion for research and knowledge.

I am grateful for the many years of funding provided by the combination of the European Union's Marie Skłodowska-Curie Actions and the EPFL School of Computer and Communication Sciences, i.e., the European Union and Swiss taxpayers, respectively.

I would also like to acknowledge my dear EPFL PhD student friends Maksym, Dongyang, Nastia, Grzegorz, Xinrui, Khashayar, Willem, Ognjen, Viktor, Khashayar, Stefan, Thanasis, and Ekaterina. Without you and the many nice moments we shared, my PhD journey would have been much less exciting! Another special thanks goes to my friends and flatmates, who actually all went/are going through an EPFL PhD, and with whom I shared many years of my life, including the COVID-19 pandemic with some of them: Sylvain, Étienne, Quentin, Auguste, Kunal, and Philipp. It has been enjoyable and a privilege to share this time with you. A special shoutout to Sylvain, whose organizational skills, strategic mastermind, and cooking genius have been very helpful for life and work.

I want to thank my lab/floor colleagues including Gergely, Lars, Theresa, Mahsa, Victor, Bogdan, Sandra, Mathilde, Aswin, William, Farnood, Lucas, Jalal, Daniyar, Anthony, Maximilien, Surya, Guillaume, Saeed, Sadegh, Saber, Oscar, Daichi, Paula, and Sepehr. I owe a special mention to Gergely Ódor, my longtime officemate, whose mathematical rigor and diverse interests have been inspiring for me.

Within and beyond the scope of my PhD, I have been extremely fortunate to be embedded in multiple innovation ecosystems. I feel grateful to have collaborated with and be supported by some of the most ambitious and effective individuals, including Julia, Nhi, Lewis, Sephin, Liliane, Cynthia, Thomas, Laurynas, David, Jaques, Anirudhh, Maximilian, Christophe, Julian, Renato, Yevhen, and Armand. I owe a special mention to Julia Wagner, who is extremely resourceful and a master of getting things done - without you, the Sciencepreneurship Summer School and many other things we collaborated on would not be in the great state they were/are in. I also owe a special mention to Laurynas Lopata, a great tech & business collaborator, and an excellent thought partner.

During my PhD, I was fortunate to spend time in industry with very talented scientists. I want to thank Federico Tomasi, Zhenwen Dai, Andreas Damianou, Kamil Ciosek, and Mounia Lalmas at Spotify and Martin Lenz, Alexander Grabner, and Christian Leistner at Amazon for sharing with me their passion for research and real-world applications.

I want to thank my friends Nick, Stijn, and Jakob for their never-failing support and fun times together in Belgium and Switzerland. If I forgot to mention anyone above, *my bad.*

Finally, I would like to thank my parents, brother, and sister-in-law. Your ever-supportive and no-pressure family environment has enabled me to explore and pursue my ambitions in education, research, and innovation, however different or remote they were from yours. I am fully aware that these things cannot be taken for granted, and I am thankful for that. Your unconditional love and support have made all this possible.

Lausanne, April 15, 2024 A.J.J.P. D.

Abstract

Machine learning (ML) enables artificial intelligent (AI) agents to learn autonomously from data obtained from their environment to perform tasks. Modern ML systems have proven to be extremely effective, reaching or even exceeding human intelligence. Although these systems are highly effective at solving their specific tasks, they are not very efficient, requiring massive and costly amounts of task-specific high-quality data to reach that performance. In contrast, human intelligence can transfer knowledge from previous experiences to learn new related concepts effectively and very efficiently, i.e., from mostly one or only a few examples of a new concept. Few-shot learning is the ML area that deals with this. The research in this thesis focuses on investigating and developing few-shot learning solutions for more effective and more efficient machine learning model adaptation.

First, we investigate the existing linear regression meta-learning method R2D2 its reproducibility. We find that the findings are mostly reproducible, but different backbone models should be taken into consideration when comparing results of different methods. Continuing with the methodology of linear regression, we introduce REGRESSION NETWORKS, a metric-learning few-shot classification approach. The metric is the embedded regression error of a point to a subspace of shots. REGRESSION NETWORKS achieve excellent results, especially when subspaces can be formed with multiple shots per class. Subsequently, we address the inefficiency of needing many labeled training tasks by leveraging advances in self-supervised learning. Our approach PROTOTRANSFER has matching contrastive pre-training with prototypical task adaptation. We show that PROTOTRANSFER outperforms state-of-the-art unsupervised methods and even matches performance of supervised methods under domain-shift, at a fraction of the labeling cost. Additionally, we design a meta-learning approach for survival analysis of rare diseases based on high-dimensional RNA sequencing data. Addressing parameter count, a first-order meta-learning algorithm is adopted together with a Cox survival loss. We show that meta-learning is significantly more effective in this setting than regular fine-tuning. Lastly, we go beyond single task adaptation with a model-agnostic learning to meta-learn (MALTML) approach. Our algorithm enables a model to quickly exploit commonalities among related tasks from an unseen task distribution, before quickly adapting to a specific task. Our experiments show that MALTML generally improves adaptation.

Keywords: machine learning, deep learning, few-shot learning, meta-learning.

Résumé

L'apprentissage automatique permet aux agents intelligents artificiels (IA) d'apprendre de manière autonome à partir des données obtenues dans leur environnement afin d'effectuer des tâches. Les systèmes modernes d'apprentissage automatique se sont révélés extrêmement efficaces, atteignant, voire dépassant, l'intelligence humaine. Bien que ces systèmes soient très efficaces pour résoudre leurs tâches spécifiques, ils ne sont pas très efficients, car ils nécessitent des quantités massives et coûteuses de données de haute qualité spécifiques à la tâche pour atteindre cette performance. En revanche, l'intelligence humaine peut transférer des connaissances à partir d'expériences antérieures pour apprendre de nouveaux concepts connexes de manière efficace et très efficiente, c'est-à-dire à partir d'un seul ou de quelques exemples seulement d'un nouveau concept. L'apprentissage à partir d'un petit nombre d'exemples est le domaine de la ML qui traite de cette question. La recherche dans cette thèse se concentre sur l'étude et le développement de solutions d'apprentissage à court terme pour une adaptation plus efficace et plus efficiente des modèles d'apprentissage automatique.

Tout d'abord, nous étudions la méthode de méta-apprentissage de régression linéaire R2D2 et sa reproductibilité. Nous constatons que les résultats sont généralement reproductibles, mais que les différentes architectures de réseaux neuronaux doivent être prises en compte lors de la comparaison des résultats de différentes méthodes.

Poursuivant la méthodologie de la régression linéaire, nous présentons REGRESSION NETWORKS, une approche de classification de quelques plans par apprentissage métrique. La métrique est l'erreur de régression intégrée d'un point à un sous-espace de plans. REGRESSION NETWORKS obtiennent d'excellents résultats, en particulier lorsque des sous-espaces multidimensionnels peuvent être formés avec plusieurs plans par classe.

Ensuite, nous nous attaquons à l'inefficacité de la nécessité de nombreuses tâches d'apprentissage étiquetées en tirant parti des progrès de l'apprentissage auto-supervisé. Notre approche PROTOTRANSFER fait correspondre le pré-entraînement contrastif à l'adaptation de tâches prototypiques. Nous démontrons que PROTOTRANSFER est plus performant que les méthodes de méta-apprentissage non supervisé les plus récentes et qu'il atteint même les performances des méthodes supervisées en cas de changement de domaine, mais pour une fraction du coût d'étiquetage.

En outre, nous concevons une approche de méta-apprentissage pour l'analyse de survie des maladies rares basée sur des données de séquençage de l'ARN à haute dimension. En ce qui concerne le nombre de paramètres, un algorithme de méta-apprentissage du premier ordre est adopté avec une perte de survie de Cox. Nous démontrons que, comparé à un réglage fin régulier, le méta-apprentissage est nettement plus efficace dans ce contexte. Enfin, nous allons au-delà de l'adaptation à une seule tâche avec une approche d'apprentissage de méta-apprentissage agnostique de modèle (MALTML). Notre algorithme d'apprentissage permet à un modèle d'exploiter rapidement les points communs entre les tâches connexes à partir d'une distribution de tâches inédite, avant de s'adapter rapidement. Les expériences montrent que MALTML améliore généralement l'adaptation à quelques tâches par rapport aux lignes de base.

Mots clefs : apprentissage automatique, apprentissage profond, apprentissage à court terme, le méta-apprentissage.

Contents

Acknowledgements **i**

Abstract **iii**

1 Introduction **1**
- 1.1 Research . . . 3
  - 1.1.1 Organization . . . 4
  - 1.1.2 Papers related to the thesis . . . 4
- 1.2 Innovation . . . 5

2 Reproducing Meta-learning with differentiable closed-form solvers **7**
- 2.1 Preface . . . 7
- 2.2 Introduction . . . 8
- 2.3 Background in meta-learning . . . 8
- 2.4 Analysis of the R2D2 Classifier . . . 11
- 2.5 Reproducibility . . . 12
- 2.6 Results and contributions . . . 14
- 2.7 Conclusion . . . 16

3 Regression Networks for Meta-Learning Few-Shot Classification **17**
- 3.1 Preface . . . 17
- 3.2 Introduction . . . 18
- 3.3 Regression Networks . . . 19
  - 3.3.1 Notation . . . 19
  - 3.3.2 Model . . . 20
  - 3.3.3 Subspace Orthogonalization . . . 21
- 3.4 Experiments . . . 21
  - 3.4.1 Few-shot Image Classification: mini-ImageNet . . . 22
  - 3.4.2 Ablation study . . . 23
- 3.5 Related work . . . 23
- 3.6 Conclusion . . . 24

4 Self-Supervised Prototypical Transfer Learning for Few-Shot Classification 25
- 4.1 Preface 25
- 4.2 Introduction 26
- 4.3 A Self-Supervised Prototypical Transfer Learning Algorithm 27
  - 4.3.1 Preliminaries 27
  - 4.3.2 Self-Supervised Prototypical Pre-Training: ProtoCLR 28
  - 4.3.3 Supervised Prototypical Fine-Tuning: ProtoTune 29
- 4.4 Experiments 29
  - 4.4.1 In-Domain Few-shot Classification: Omniglot and mini-ImageNet . 30
  - 4.4.2 Cross-domain Few-Shot Classification: CDFSL benchmark 30
  - 4.4.3 Ablation Study: Batch Size, Number of Queries, and Fine-Tuning 32
  - 4.4.4 Number of Training Classes and Samples 33
  - 4.4.5 Task Generalization Gap 34
- 4.5 Related Work 35
- 4.6 Conclusion 37

5 A meta-learning approach for genomic survival analysis 39
- 5.1 Preface 39
- 5.2 Introduction 40
- 5.3 Methods 41
  - 5.3.1 Datasets 41
  - 5.3.2 Survival prediction model 42
  - 5.3.3 Meta-learning 42
  - 5.3.4 Experimental setup 45
  - 5.3.5 Evaluation 47
  - 5.3.6 Hyper-parameter selection 48
  - 5.3.7 Interpretation of the genes prioritized by the meta-learning model 50
- 5.4 Results 50
  - 5.4.1 Meta-learning outperforms regular pre-training and combined learning 50
  - 5.4.2 Meta-learning achieves competitive predictive performance compared to direct learning 51
  - 5.4.3 Risk score ranked genes are enriched in key cancer pathways . . . 52
  - 5.4.4 Validation on the small sample rare cancer cohort 53
  - 5.4.5 Validation on the independent lung cancer cohort 53
- 5.5 Discussion 56
- 5.6 Data Availability 60

6 Model-Agnostic Learning to Meta-Learn 61
- 6.1 Preface 61
- 6.2 Introduction 62
- 6.3 Related Work 63

- 6.4 Methodology . . . 63
  - 6.4.1 Learning to Meta-Learn Problem Setup . . . 64
  - 6.4.2 A Model-Agnostic Learning to Meta-Learn Algorithm . . . 64
- 6.5 Experimental Protocol . . . 66
  - 6.5.1 Illustrative preliminary experiment: regression . . . 67
  - 6.5.2 Specifics of main experiments . . . 68
- 6.6 Future Work . . . 69
- 6.7 Results . . . 69
  - 6.7.1 Experimental Setup and Hyperparameters . . . 69
  - 6.7.2 Classification . . . 70
  - 6.7.3 2D Navigation . . . 71
  - 6.7.4 Continual Regression . . . 72
  - 6.7.5 Continual Reinforcement Learning . . . 72
- 6.8 Findings . . . 74
  - 6.8.1 Rapid Meta-Learning . . . 74
  - 6.8.2 Rapid (Meta-)Learning is More Important than Feature Reuse . . . 74
  - 6.8.3 Overfitting . . . 75
  - 6.8.4 Reptile . . . 75
- 6.9 Documented Modifications . . . 75

7 Conclusion 77

A Appendix for chapter 3 79
- A.1 Algorithm . . . 79
- A.2 Experimental details . . . 80
  - A.2.1 Experimental Setup and Datasets . . . 80
  - A.2.2 Architectures and training . . . 80
- A.3 Extra experiments: mini-ImageNet and CUB . . . 81
- A.4 Domain Shift: mini-ImageNet to CUB and CUB to mini-ImageNet . . . 82

B Appendix for chapter 4 85
- B.1 Experimental Details . . . 85
  - B.1.1 Datasets . . . 85
  - B.1.2 Architecture and Optimization Parameters . . . 86
  - B.1.3 Augmentations . . . 87
  - B.1.4 Classes for t-SNE Plots . . . 89
  - B.1.5 Results With Full Confidence Intervals & References . . . 89

C Appendix for chapter 5 95

D Appendix for chapter 6 97
- D.1 Omniglot . . . 97
  - D.1.1 Hyperparameters . . . 97

D.1.2 training (30), validation (4), test (16) alphabets 97
D.2 Tiered-ImageNet 99
D.2.1 Hyperparameters 99
D.2.2 training (20), validation (6), test (8) categories 99
D.2.3 Family and task visualization 100
D.3 2D navigation 100
D.4 Continual Regression 100
D.4.1 Hyperparameters 100

Bibliography **115**

Curriculum Vitae **117**

1 Introduction

Intelligence can be defined as *"the ability to learn and perform suitable techniques to solve problems and achieve goals, appropriate to the context in an uncertain, ever-varying world"* (Manning, 2020). In 1956, John McCarthy coined the term artificial intelligence (AI) and defined it as *"the science and engineering of making intelligent machines, especially intelligent computer programs"*. This kind of intelligence is *artificial* (by design) in contrast to the *natural* (inherent) intelligence of living beings, mainly humans. Russell and Norvig (2010) further detail the definition of AI as *"the study of [artificial] agents that receive percepts from the environment and perform actions. Each such agent implements a function that maps percept sequences to actions"*.

Within AI, machine learning (ML) is a subfield that focuses on enabling AI agents with the capability to learn autonomously from data obtained from their environment to perform tasks. Practically, this entails that computer programs that process inputs to produce outputs can be *learnt* using ML algorithms on data instead of needing to be explicitly programmed. Similar to how a computer compiler enables programs written in a source language easy for humans to read to be translated into a binary computer-readable language, machine learning enables one to go from data and a task specification to a learned model (program) that processes inputs and produces outputs.

Modern machine learning systems have proven to be extremely effective, reaching or even exceeding human intelligence, in tasks such as object image classification (He et al., 2015) and the board game of Go (Silver et al., 2016). These breakthroughs have been made possible mostly by employing the general and powerful deep neural network model architecture combined with the stochastic gradient descent learning algorithm and efficient computational implementations (Krizhevsky et al., 2012).

Although these deep learning systems are highly *effective* at solving their specific tasks, they are not very *efficient*, requiring massive amounts of task-specific high-quality data to reach that performance. For example, the popular object image classification benchmark dataset ImageNet (Russakovsky et al., 2015a) consists of 1.28 million labeled high-resolution images across 1,000 categories. For many tasks and organizations, it is too expensive or even impossible to obtain this amount and quality of data. Yosinski et al. (2014) show that trained deep learning models exhibit excellent *transfer learning* capabilities; continuing training ("fine-tuning") on (part of) the parameters of a deep neural network "pretrained" on a related source task benefits learning on a target task. This enables to reduce the amount of data needed to collect for the target tasks. For example, Esteva et al. (2017) leverage transfer learning from a pretrained ImageNet model with an order of magnitude fewer examples (127,000 vs 1.28 million) across broadly the same number of categories (757 vs 1000), in a skin cancer image classification setting, to match the performance of human dermatologists.

Still, human intelligence can learn new related concepts *effectively* and very *efficiently*, i.e., from mostly one or only a few examples of a new concept (B. Lake et al., 2011). The area of *Few-shot Learning* covers adapting models with only one or a handful of examples per new concept, based on knowledge from related concepts (Fei-Fei et al., 2006; B. M. Lake et al., 2015). Few-shot learning is challenging since adapting sufficiently to a new data set has to be balanced with overfitting to that small data set.

The research in this thesis focuses on investigating and developing few-shot learning solutions for more *effective* and more *efficient* machine learning model adaptation. The typical transfer learning solution is to fine-tune an off-the-shelf pre-trained deep neural network model. Although this solution is efficient by avoiding the need for training a big model from scratch, it requires carefully choosing a number of hyperparameters to be effective as well. Determining these hyperparameters such as adaptation learning rate and layers to freeze is challenging because of the very small datasets encountered.

Addressing the few-shot learning problem more methodologically is *meta-learning*, which explicitly *learns to learn* few-shot tasks (Vinyals et al., 2016; Finn et al., 2017). This paradigm uses a collection of few-shot learning tasks during training that relate to the unknown few-shot testing goal tasks. Note that collecting many of these training few-shot learning tasks, especially with labels, is an expensive inefficient task in its own respect, which can be addressed by sampling from existing (single task) datasets (Vinyals et al., 2016). Black-box meta-learning and optimization-based meta-learning approaches are generally applicable across a variety of learning problems (e.g., classification, regression, & reinforcement learning), but they also have challenging training problems including second order gradients and high memory requirements (K. Li and Malik, 2017; Ravi and Larochelle, 2017; Finn et al., 2017; Nichol et al., 2018). For the popular setting of few-shot *classification*, these challenges are largely addressed by semi-parametric meta-learning (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018). Advances in few-shot classification approaches mostly focus on formulating more expressive and efficient class representations leading to more effective performance (Bertinetto et al., 2019; Devos and Grossglauser, 2020; Medina et al., 2020). In this thesis, we investigate, develop, and apply few-shot learning methods for effective and efficient machine learning model adaptation.

1.1 Research

Chapter 2 investigates the existing linear regression-based meta-learning method R2D2 (Bertinetto et al., 2019) its reproducibility (Devos et al., 2019). We find it to be generally reproducible, with a few missing parameters. After bringing our work to the authors' attention, they updated the original paper and released code. An official anonymous reviewer publicly calls our work a *"worthy contribution to science"* (Anonymous, 2019).

Chapter 3 continues with the methodology of linear regression and introduces Regression Networks (Devos and Grossglauser, 2020). Regression Networks is a few-shot classification metric learning approach we develop that follows the line of research of more expressive class representations. It extends the embedded single point class representations of Prototypical Networks (Snell et al., 2017) with more expressive class subspace representations. We meta-learn classification of embedded points by regressing the closest approximation in every class subspace while using the regression error as a distance metric. Regression Networks achieve excellent results, especially when more aggregate class representations can be formed with multiple shots per class.

Chapter 4 addresses the costly requirement of having many labeled training tasks for few-shot classification by introducing ProtoTransfer (Medina et al., 2020). ProtoTransfer uses advances in self-supervised learning to construct a metric embedding that clusters unlabeled prototypical samples and their augmentations together. This pre-trained embedding is a starting point for few-shot classification by summarizing class clusters and fine-tuning. We show that ProtoTranser outperforms state-of-the-art unsupervised meta-learning methods on in-domain tasks, and even matches the performance of supervised methods under domain-shift, but at a fraction of the labeling cost.

Chapter 5 focuses on an application of meta-learning for survival analysis of rare diseases based on high-dimensional RNA sequencing data (Qiu et al., 2020). Addressing the high parameter count, a first-order meta-learning algorithm is adapted together with a Cox survival loss. We demonstrate that, compared to regular fine-tuning, meta-learning is significantly more effective in this biomedical regression setting. We also show that the meta-learning framework with a few samples can achieve competitive performance with learning from scratch with a significantly larger number of samples.

Chapter 6 goes beyond single task adaptation to task distribution adaptation with a model-agnostic learning to meta-learn (MALTML) approach (Devos and Dandi, 2021). Concretely, our approach generalizes the seminal model-agnostic meta-learning (MAML) work of Finn et al. (2017) to another level of learning. The algorithm enables any model to quickly exploit commonalities among related tasks from an unseen task distribution, before quickly adapting to specific tasks from that same distribution. Experiments on image classification, (continual) regression, and reinforcement learning tasks demonstrate that learning to meta-learn mostly improves adaptation.

1.1.1 Organization

The aforementioned contributions are described in detail in the next five chapters. Each chapter maps to a paper written by the author (with collaborators). Each chapter starts with a preface, consisting of a summary of the work and a list of author contributions using the CRediT framework (Brand et al., 2015). The appendices for all chapters are collected at the end.

1.1.2 Papers related to the thesis

All the papers included as chapters in this thesis have been accepted at their respective venues after peer-review. The chapter-paper mapping is as follows:

- Thesis Chapter 2:
  Arnout Devos, Sylvain Chatel, and Matthias Grossglauser (May 2019). "[Re] Meta-learning with differentiable closed-form solvers". In: *ReScience C Journal* 5.2. #1. DOI: 10.5281/zenodo.3160540
- Thesis Chapter 3:
  Arnout Devos and Matthias Grossglauser (June 2020). "Regression Networks for Meta-Learning Few-Shot Classification". In: *ICML 2020 Workshop on Automated Machine Learning (AutoML)*. URL: https://sites.google.com/view/automl2020/accepted-papers_1
- Thesis Chapter 4:
  Carlos Medina, Arnout Devos, and Matthias Grossglauser (June 2020). "Self-supervised prototypical transfer learning for few-shot classification". In: *ICML 2020 Workshop on Automated Machine Learning (AutoML)*. URL: https://sites.google.com/view/automl2020/accepted-papers_1
- Thesis Chapter 5:
  Yeping Lina Qiu, Hong Zheng, Arnout Devos, Heather Selby, and Olivier Gevaert (Dec. 2020). "A meta-learning approach for genomic survival analysis". In: *Nature communications* 11.1, p. 6350. URL: https://www.nature.com/articles/s41467-020-20167-3
- Thesis Chapter 6:
  Arnout Devos and Yatin Dandi (Dec. 2021). "Model-Agnostic Learning to Meta-Learn". In: *Proceedings of Machine Learning Research*. Vol. 148. PMLR, pp. 155–175. URL: https://proceedings.mlr.press/v148/devos21a.html

1.2 Innovation

The chapters in this thesis are a selection of research work that the author contributed to during their PhD and that are related to machine learning model adaptation. Next to research, the author also completed EPFL's entrepreneurial PhD program as an *EPFLinnovators* fellow in computer science (Licina, 2021). The program is officially summarized as (Hunen, 2018):

> *EPFLinnovators is a four-year doctoral program based in the dynamic environment of the EPFL campus. To develop the innovation potential of PhD students, the EPFLinnovators programme integrates a six-months to two-year secondment in the non-academic sector. Furthermore, the programme provides the PhD students with the training, experience and advice they need to become tomorrow's entrepreneurs.*

Hence, the author completed the following two secondments in the non-academic sector:

- At Amazon, we developed an approach for synthetic sensor data generation for model training and evaluation of autonomous drones.
- At Spotify, we developed an approach for cold-start recommendation. The cold-start setting relates to zero-shot learning, where using only auxiliary information (e.g., meta-data), a model should make adapted predictions (Jingjing Li et al., 2019).

Within the scope of the program, next to completing its 4 mandatory courses on entrepreneurship, public speaking, and teaching, the author also co-founded the Sciencepreneurship Summer School (Devos and Wagner, 2023), the first of its kind in Switzerland. The first edition was recognized by the official EPFL-ETH Zurich summer school program and it took place in Zurich in April 2023. Qualitatively, the first edition consisted of a week-long course (1 ECTS credit) gathering graduate students, experienced sciencepreneurs, and ecosystem players. The course's goal is to share knowledge, experiences, and pitfalls about transforming scientific advances into societal impact through entrepreneurship. Quantitatively, the first edition attracted 113 applicants, 30 student participants (10 EPFL, 10 ETH Zurich, 10 others), of which over 40% identified as women, and $45,000 in funding. After completing the course, one participant from the student team of three that won the course's pitching competition co-founded an ETH Zurich spinoff and two others joined venture building programs in Germany and the UK.

2 Reproducing Meta-learning with differentiable closed-form solvers

2.1 Preface

Summary: In this paper, we present a reproduction of the paper of Bertinetto et al. (2019) *"Meta-learning with differentiable closed-form solvers"* as part of the ICLR 2019 Reproducibility Challenge. In successfully reproducing the most crucial part of the paper, we reach a performance that is comparable with or superior to the original paper on two benchmarks for several settings. We evaluate new baseline results, using a new dataset presented in the paper. Yet, we also provide multiple remarks and recommendations about reproducibility and comparability. After we brought our reproducibility work to the authors' attention, they have updated the original paper on which this work is based and released code as well. Our contributions mainly consist in reproducing the most important results of their original paper, in giving insight in the reproducibility and in providing a first open-source implementation.

This chapter is an edited version of Devos et al. (2019).

Code: https://github.com/ArnoutDevos/r2d2

Co-authors: Sylvian Chatel (SC) and Matthias Grossglauser (MG).

Contributions:
AD: Conceptualization, Methodology, Software, Visualization, Investigation, Writing - Original Draft.
SC: Conceptualization, Methodology, Software, Visualization, Investigation, Writing - Original Draft.
MG: Supervision, Writing - Review and Editing, Project Administration.

2.2 Introduction

The ability to adapt to new situations and learn quickly is a cornerstone of human intelligence. When given a previously unseen task, humans can use their previous experience and learning abilities to perform well on this new task in a matter of seconds and with a relatively small amount of new data. Artificial learning methods have been shown to be very effective for specific tasks, often times surpassing human performance (Silver et al., 2016; Esteva et al., 2017). However, by relying on standard supervised-learning or reinforcement learning training paradigms, these artificial methods still require much training data and training time to adapt to a new task.

An area of machine learning that learns and adapts from a small amount of data is called *few-shot learning*. A *shot* corresponds to a single example, e.g. an image and its label. In few-shot learning, the learning scope is expanded to a variety of tasks with a few shots each, compared to the classic setting of a single task with many shots. A promising approach for few-shot learning is the field of *meta-learning*. Meta-learning, also known as *learning-to-learn*, is a paradigm that exploits cross-task information and training experience to perform well on a new unseen task.

In this work we reproduce the paper of Bertinetto et al. (2019) (referenced as *"their paper"*); it falls into the class of gradient-based meta-learning algorithms that learn a model parameter initialization for rapid fine-tuning with a few shots (Finn et al., 2017; Nichol et al., 2018). The authors present a new meta-learning method that combines a deep neural network feature extractor with differentiable learning algorithms that have closed-form solutions. This reduces the overall complexity of the gradient-based meta-learning process, while advancing the state-of-the-art in terms of accuracy across multiple few-shot benchmarks.

We interacted with the authors through OpenReview[I], bringing our reproducibility work and TensorFlow code[II,III] to their attention. Because of this, they have updated their original paper with more details to facilitate reproduction and they have released an official PyTorch implementation[IV].

2.3 Background in meta-learning

The objective of few-shot meta-learning is to train a model that can quickly adapt to a new task by using only a few datapoints and training iterations. In our work, we will consider only classification tasks, but it should be noted that meta-learning is also generally applicable to regression or reinforcement learning tasks (Finn et al., 2017).

[I] https://openreview.net/forum?id=HyxnZh0ct7&noteId=BkxDPnDZMV

[II] our R2D2 and R2D2*: https://github.com/ArnoutDevos/r2d2

[III] our MAML on CIFAR-FS: https://github.com/ArnoutDevos/maml-CIFAR-FS

[IV] Bertinetto et al. (2019) code: https://github.com/bertinetto/r2d2

In order to provide a solid definition of meta-learning, we need to define its different components. We denote the set of tasks by $\mathbb{T}$. A task $\mathcal{T}_i \in \mathbb{T}$ corresponds to a classification problem, with a probability distribution of example inputs $\boldsymbol{x}$ and (class) labels y, $(\boldsymbol{x}, y) \sim \mathcal{T}_i$. For each task, we are given training samples $\mathcal{Z}_{\mathcal{T}} = \{(\boldsymbol{x}_i, y_i)\} \sim \mathcal{T}$ with K shots per class and evaluation samples $\mathcal{Z}'_{\mathcal{T}} = \{(\boldsymbol{x}'_i, y'_i)\} \sim \mathcal{T}$ with Q shots (*queries*) per class, all sampled independently from the same distribution $\mathcal{T}$. In meta-learning, we reuse the learning experience used for tasks $\mathcal{T}_i,\ i \in [0, L]$ to learn a new task $\mathcal{T}_j$, where $j > L$, from only K examples, for every single one of the N classes in the task. Commonly, this is denoted as an *N-way K-shot* problem. To this end, in meta-learning two different kinds of learners can be at play: (1) a *base-learner* that works at the task level and learns a single task (e.g. classifier with N classes) and (2) a *meta-learner* that produces those model parameters that enable the fastest average fine-tuning (using the *base-learner*) on unseen tasks.

The authors put a specific view of meta-learning forward. Their meta-learning system consists of a generic feature extractor $\mathit{\Phi}(\boldsymbol{x})$ that is parametrized by ω, and a task-specific predictor $f_{\mathcal{T}}(X)$ that is parametrized by $w_{\mathcal{T}}$ and adapts separately to every task $\mathcal{T} \in \mathbb{T}$ based on the few shots available. In the case of a deep neural network architecture, this task-specific predictor $f_{\mathcal{T}}$ can be seen as the last layer(s) of the network and is specific to a task $\mathcal{T}$. The preceding layers $\mathit{\Phi}$ can be trained across tasks to provide the best feature extraction on which the task-specific predictor can finetune with maximum performance.

The base-learning phase in their paper assumes that the parameters ω of the feature extractor $\mathit{\Phi}$ are fixed and computes the parameters $w_{\mathcal{T}}$ of $f_{\mathcal{T}}$ through closed-form learning process $\mathit{\Lambda}$. $\mathit{\Lambda}$, on its own, is parametrized by ρ. The meta-learning phase in the paper learns a parametrization of $\mathit{\Phi}$ and $\mathit{\Lambda}$ (respectively ω and ρ). In order to learn those meta-parameters, the algorithm minimizes the expected loss on test sets from unseen tasks in $\mathbb{T}$ with gradient descent. The base-learning and meta-learning phases are shown in Figures 2.1 and 2.2, respectively.

Most of the recent meta-learning works are tested against image datasets and their feature extractor consists of a convolutional neural network (CNN). The variability between works resides mainly in the base learner $f_{\mathcal{T}}$ and its parameter obtaining training procedure $\mathit{\Lambda}$. Examples are an (unparametrized) k-nearest-neighbour algorithm (Vinyals et al., 2016), a CNN with SGD (Mishra et al., 2017), and a nested SGD (Finn et al., 2017). Systems in Vinyals et al. (2016) and Snell et al. (2017) are based on comparing new examples in a learned metric space and rely on matching. In particular, MatchingNet from Vinyals et al. (2016) uses neural networks augmented with memory and recurrence with attention in a few-shot image recognition context. Mishra et al. (2017) build on this attention technique by adding temporal convolutions to reuse information from past tasks. Another example of a matching-based method is introduced in Garcia and Bruna (2017), where a graph neural network learns the correspondence between the training and testing sets. A different approach is to consider the SGD update as a learnable function

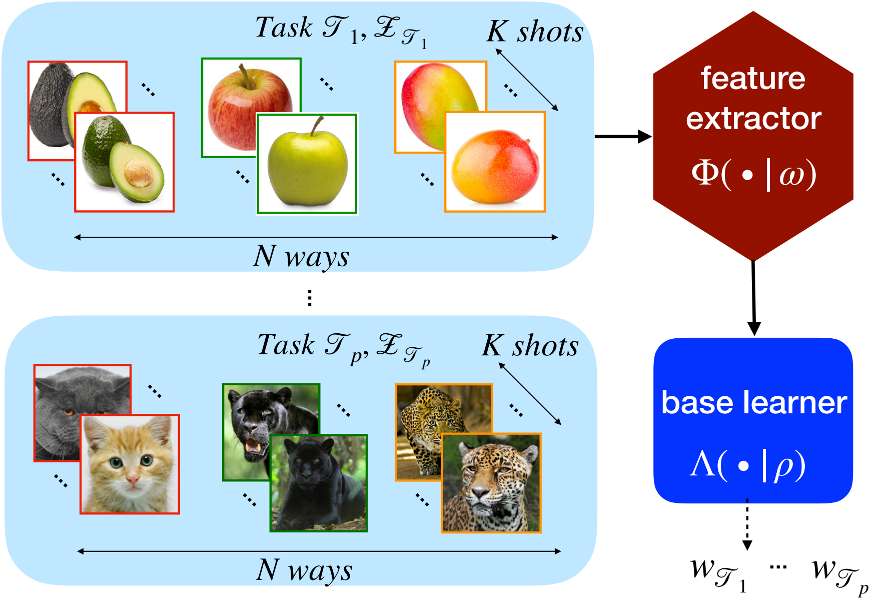

Task $\mathscr{T}_1, \mathscr{Z}_{\mathscr{T}_1}$
K shots
N ways
Task $\mathscr{T}_p, \mathscr{Z}_{\mathscr{T}_p}$
K shots
N ways
feature extractor
$\Phi(\bullet \mid \omega)$
base learner
$\Lambda(\bullet \mid \rho)$
$w_{\mathscr{T}_1}$ ··· $w_{\mathscr{T}_p}$

Figure 2.1: Base-learning of the task-specific parameters $w_{\mathcal{T}_i}$ over p tasks following steps 3 to 6 of Algorithm 1.

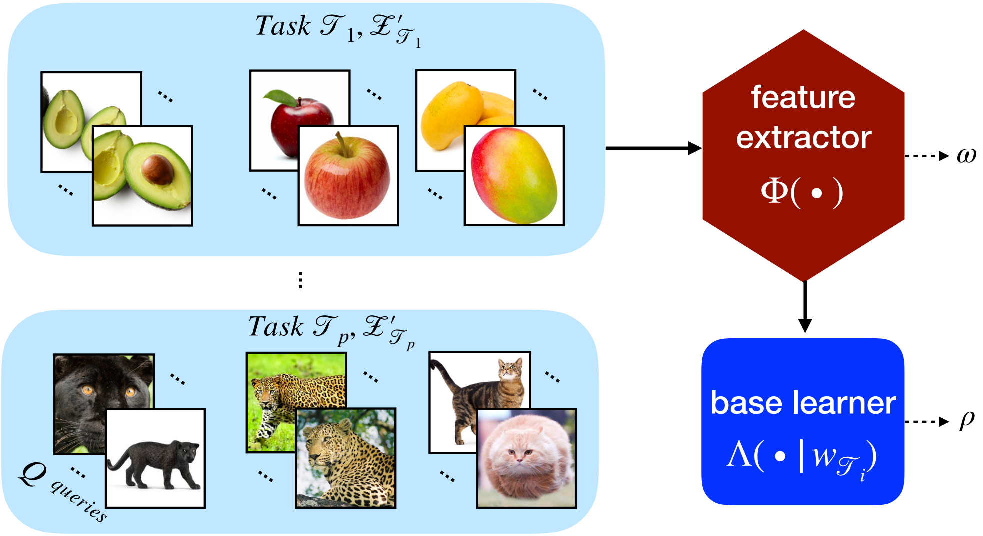

Task $\mathscr{T}_1, \mathscr{Z}'_{\mathscr{T}_1}$
Task $\mathscr{T}_p, \mathscr{Z}'_{\mathscr{T}_p}$
Q queries
feature extractor
$\Phi(\bullet)$
ω
base learner
$\Lambda(\bullet \mid w_{\mathscr{T}_i})$
ρ

Figure 2.2: Meta-learning of the meta-parameters ω and $\boldsymbol{\rho}$ over the evaluation sets of each task $\mathcal{Z}'_{\mathcal{T}_i}$ using the previously learned $w_{\mathcal{T}_i}$ following steps 7 to 9 of Algorithm 1.

for meta-learning. In particular, sequential learning algorithms, such as recurrent neural networks and LSTM-based methods, enable the use of long-term dependencies between the data and gradient updates as pointed out by Ravi and Larochelle (2017). Finally, Finn et al. (2017) introduce a technique called model-agnostic meta-learning (MAML). In MAML, meta-learning is done by backpropagating through the fine-tuning stochastic gradient descent update of the model parameters.

2.4 Analysis of the R2D2 Classifier

In their paper, Bertinetto et al. (2019) present a new approach that relies on using fast and simple base learners such as *ridge regression differentiable discriminator* (R2D2) or *(regularized) logistic regression differentiable discriminator* (LRD2). In our reproducibility work we will focus on the R2D2 algorithm, because it is the only proposed algorithm with a truly closed-form solver for the base-learner. For reproducibility purposes, we transformed the original textual description of R2D2 in their paper into an algorithmic description in Algorithm 1, elaborated upon in the following.

Algorithm 1 Ridge Regression Differentiable Discriminator (R2D2)

```
Require: Distribution of tasks $\mathbb{T}$.
Require: Feature extractor $\Phi$ parameterized by $\omega$.
Require: Finetuning predictor $f_{\mathcal{T}}$ with base-learning algorithm $\Lambda$ and task-specific
    parameters $w_{\mathcal{T}}$, and meta-parameters $\boldsymbol{\rho} = (\alpha, \beta, \lambda)$
Initialize $\Phi$, $\Lambda$, and $f_{\mathcal{T}}$ with pre-trained or random parameters $\omega_0$ and $\boldsymbol{\rho_0}$
while not done do
    Sample batch of tasks $\mathcal{T}_i \sim \mathbb{T}$
    for all $\mathcal{T}_i$ do
        Sample $K$ datapoints for every class from $\mathcal{T}_i$ and put in them in the training set
$\mathcal{Z}_{\mathcal{T}_i}$
        Base-learn $f_{\mathcal{T}_i}$ using $\Lambda$:
$W_i = w_{\mathcal{T}_i} = \Lambda(\mathcal{Z}_{\mathcal{T}_i}) = X_i^T(X_i X_i^T + \lambda.I)^{-1} Y_i$
with $X_i = \Phi(\mathcal{Z}_{\mathcal{T}_i})$ and $Y_i$ the one-hot labels from $\mathcal{Z}_{\mathcal{T}_i}$.

        Sample datapoints for every class from $\mathcal{T}_i$ and put in them in the evaluation set
$\mathcal{Z}'_{\mathcal{T}_i}$
    end for
    Update meta-parameters $\theta = (\omega, \boldsymbol{\rho})$ through gradient descent :

        $$\theta \leftarrow \theta - \varepsilon . \sum_i \nabla_\theta \mathcal{L}(f_{\mathcal{T}_i}(\Phi(\mathcal{Z}'_{\mathcal{T}_i})), Y'_i)$$

    with $\varepsilon$ the learning rate, $\mathcal{L}$ the cross-entropy loss, and $f_{\mathcal{T}_i}(X'_i) = \alpha X'_i W_i + \beta$.
end while
```

In R2D2, during base-learning with $\mathcal{Z}_{\mathcal{T}}$, the linear predictor $f_{\mathcal{T}}$ is adapted for each training task $\mathcal{T}$, by using the learning algorithm Λ; and the meta-parameters ω (of Φ) and $\boldsymbol{\rho}$ (of Λ) remain fixed. It is only in the meta-training phase that meta-parameters ω and $\boldsymbol{\rho}$ are updated, by using $\mathcal{Z}'_{\mathcal{T}}$. The linear predictor is seen as $f_{\mathcal{T}}(x) = xW$ with W a matrix of task-specific weights $w_{\mathcal{T}}$, and x the feature extracted version of $\boldsymbol{x}$, $x = \Phi(\boldsymbol{x})$. This approach leads to a ridge regression evaluation such that it learns the task weights $w_{\mathcal{T}}$:

$$\Lambda(X, Y) = \arg\min_{W} \|XW - Y\|^2 + \lambda \|W\|^2 \tag{2.1}$$

$$= (X^T X + \lambda I)^{-1} X^T Y \tag{2.2}$$

where X contains all NK feature extracted inputs from the training set of the considered task. A key insight in their paper is that the closed-form solution of Equation 2.2 can be simplified using the *Woodbury matrix identity* yielding $W = \Lambda(X, Y) = X^T(XX^T + \lambda I)^{-1}Y$. This considerably reduces the complexity of the matrix calculations in the special case of few-shot learning. Specifically, XX^T is of size $NK \times NK$, in the case of an N-way K-shot task; this matrix will, together with the regularization, be relatively easily inverted. Normally, regression is not adequate for classification, but the authors noticed that it still has considerable performance. Therefore, in order to transform the regression outputs (which are only effectively calculated when updating the meta-parameters using $\mathcal{Z}'_{\mathcal{T}}$) to work with the cross-entropy loss function, the meta-parameters $(\alpha, \beta) \in \mathbb{R}^2$ serve as a scale and bias, respectively:

$$\hat{Y'} = \alpha X' \left[X^T(XX^T + \lambda I)^{-1} Y \right] + \beta \tag{2.3}$$

2.5 Reproducibility

As a first step in the reproducibility, we reproduce the results of a baseline algorithm on different datasets used in their paper. In this perspective, we first consider the MAML algorithm from Finn et al. (2017). We use the official TensorFlow implementation of MAML (Finn et al., 2017) to reproduce the baseline's results. Then, we amend this MAML implementation to reproduce the results on the new CIFAR-FS dataset proposed by their paper (Bertinetto et al., 2019).

When reproducing the R2D2 algorithm, our first consideration is that the feature extractors in MAML and R2D2 are very different. MAML uses four convolutional blocks with an organization of [32, 32, 32, 32] filters. Whereas, R2D2's four blocks employ a [96, 192, 384, 512] scheme, as shown in Figure 2.3. In other words, the feature extractor in R2D2 is more complex hence is expected to yield better results (Mhaskar et al., 2016). In order to provide a meaningful comparison, we implement and evaluate both the simple and more complex feature extractors for the R2D2 algorithm, denoted by R2D2* and R2D2 respectively.

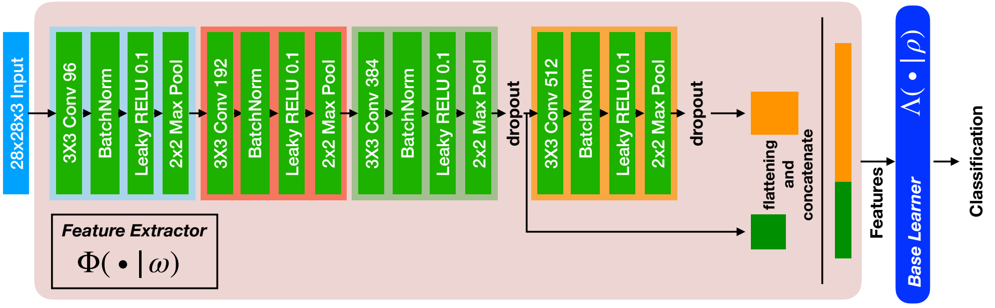


Figure 2.3: Overall architecture of the R2D2 system considering [96, 192, 384, 512] filters in the feature extractor with 4 convolutional blocks for the CIFAR-FS dataset.

In order to make a working reproduction of their paper we had to make the following assumptions. We first considered the aforementioned complex architecture and feature extractor. In particular, for the feature extractor, we made assumptions on the convolutional block options. We considered a 3x3 convolution block with a 'same' padding and a stride of 1. For the 2x2 maximum pooling, we use a stride of 2 and no padding. Second, concerning the ridge regression base-learner, we opted for a multinomial regression that returns the class with the maximum value through one-hot encoding. Following the guidelines for the feature extractor presented in Section 4.2 of their paper, we were not successful in reproducing the exact number of features at the output of the feature extractor. In their paper, the overall numbers of features at the output of the extractor are 3584, 72576 and 8064 for Omniglot, miniImageNet and CIFAR-FS, respectively. However, by implementing the feature extractor described in their paper, we obtain 3988, 51200 and 8192 respectively.

For comparison purposes, we use the same number of classes (e.g. 5) and shots during (e.g. 1) training and testing, despite their paper using a higher number of classes during training (16 for miniImageNet, 20 for CIFAR-FS) than during testing (5 for miniImageNet and CIFAR-FS). Regarding the amount of shots, their paper uses a random number of shots during training. This is different from the way most baselines are trained using the same number of shots per class during training and testing (Finn et al., 2017; Nichol et al., 2018; Vinyals et al., 2016). For comparability, it is paramount to keep the training and testing procedures similar, if not the same. In particular, as in their paper the 5-way results are exactly the same as those reported in MAML (Finn et al., 2017), using the same number of classes and shots during training and testing allows for a justifiable comparison.

Finally, a last assumption is made on the algorithm's stopping criterion. In their paper, the stopping criterion is vaguely defined as "*the error on the meta-validation set does not decrease meaningfully for 20,000 episodes*". Therefore, in line with the MAML training procedure, we meta-train using 60,000 iterations. To update the meta-parameters, in line with their paper, we use the Adam optimizer (Kingma and J. L. Ba, 2015) with an

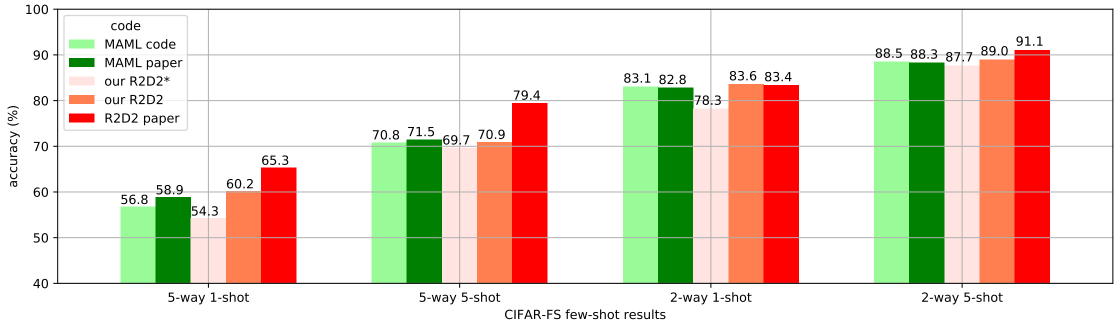


Figure 2.4: N-way K-shot classification accuracies on CIFAR-FS. Detailed results in Table 2.1.

Table 2.1: N-way K-shot classification accuracies on CIFAR-FS with 95% confidence intervals.

Method	MAML paper Bertinetto et al. (2019)	MAML ours	R2D2* ours	R2D2 ours	R2D2 paper Bertinetto et al. (2019)
5-way, 1-shot	58.9 ± 1.9%	56.8 ± 1.9%	54.3 ± 1.8%	60.2 ± 1.8%	65.3 ± 0.2%
5-way, 5-shot	71.5 ± 1.0%	70.8 ± 0.9%	69.7 ± 0.9%	70.9 ± 0.9%	79.4 ± 0.1%
2-way, 1-shot	82.8 ± 2.7%	83.1 ± 2.6%	78.3 ± 2.8%	83.6 ± 2.6%	83.4 ± 0.3%
2-way, 5-shot	88.3 ± 1.1%	88.5 ± 1.1%	87.7 ± 1.1%	89.0 ± 1.0%	91.1 ± 0.2%

initial learning rate of 0.005, dampened by 0.5 every 2,000 episodes. We use 15 examples per class for evaluating the post-update meta-gradient. We use a meta batch-size of 4 and 2 tasks for 1-shot and 5-shot training respectively. For MAML we use a task-level learning rate of 0.01, with 5 steps during training and 10 steps during testing.

2.6 Results and contributions

The results of the different implemented architectures and algorithms for several datasets are shown in Figures 2.4 and 2.5. More detailed results with 95% confidence intervals are shown in Tables 2.1 and 2.2. The first and last column correspond to the baselines in original papers.

Table 2.2: N-way K-shot classification accuracies on miniImageNet with 95% confidence intervals

Method	MAML paper Finn et al. (2017)	MAML code Finn et al. (2017)	R2D2* ours	R2D2 ours	R2D2 paper Bertinetto et al. (2019)
5-way, 1-shot	48.7 ± 1.8%	47.6 ± 1.9%	45.7 ± 1.8%	51.7 ± 1.8%	51.5 ± 0.2%
5-way, 5-shot	63.1 ± 0.9%	62.3 ± 0.9%	63.7 ± 1.3%	63.3 ± 0.9%	68.8 ± 0.2%
2-way, 1-shot	74.9 ± 3.0%	78.8 ± 2.8%	74.7 ± 2.9%	74.6 ± 2.9%	76.7 ± 0.3%
2-way, 5-shot	84.4 ± 1.2%	82.6 ± 1.2%	83.0 ± 1.2%	84.6 ± 1.2%	86.8 ± 0.2%

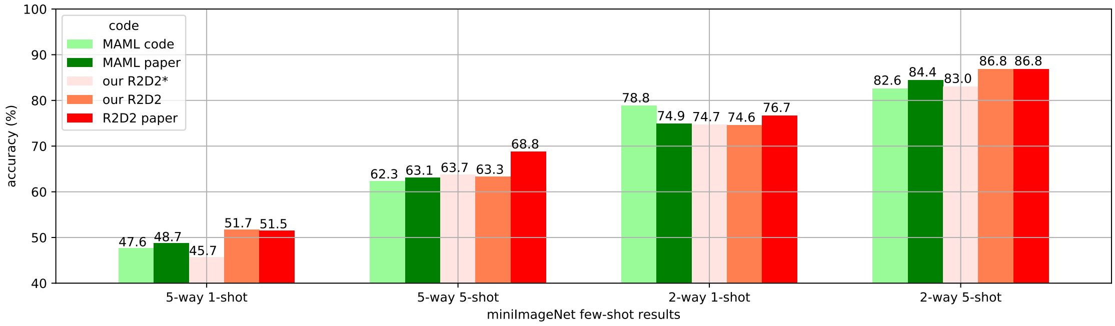


Figure 2.5: N-way K-shot classification accuracies on miniImageNet. Detailed results in Table 2.2.

Our implementations were made in Python 3.6.2 and TensorFlow 1.8.0 (Abadi et al., 2016). The source code of all implementations is available[V] online[VI]. The simulations were run on a machine with 24 Xeon e5 2680s at 2.5 GHz, 252GB RAM and a Titan X GPU with 12 GB RAM.

Although our results differ slightly from the original paper of Bertinetto et al. (2019), R2D2 (with its more complex network architecture) performs better than the MAML method for most simulations. It is not a surprise that, in most of the cases, with a more complex feature extractor better results are obtained for the same algorithm (R2D2 vs R2D2*). Overall, our study confirms that the R2D2 meta-learning method, with its corresponding complex architecture, yields better performance than basic MAML (with its simpler architecture). The differences between reproduced results and reported values might be due to our assumptions or the stopping criterion in the training. Also, as expected, the complexity (N-ways) and the amount of data (K-shots) play a major role in the classification accuracy. The accuracy drops when the number of ways increases and number of shots decreases. An outlier worth mentioning is our MAML simulation on miniImageNet: the 2-way 1-shot classification accuracy of $78.8 \pm 2.8\%$ is much better than the $74.9 \pm 3.0\%$ reported in Finn et al. (2017).

In summary, we successfully reproduced the most important results presented in Bertinetto et al. (2019). Although our reproduced results and their paper results differ slightly, the general observations of the authors remain valid. Their meta-learning with differentiable closed-form solvers yields state-of-the-art results and improves over another state-of-

---

[V] R2D2 and R2D2*: https://github.com/ArnoutDevos/r2d2

[VI] finn2017model with CIFAR-FS: https://github.com/ArnoutDevos/maml-CIFAR-FS the-art method. The assumptions made, however, could have been clarified in their original paper. Indeed, these assumptions could be the source of the discrepancy in the reproduction results. In this reproducibility work we did not focus on the logistic regression based algorithm (LRD2) from their paper because the logistic regression solver does not have a closed-form solution.

Overall, with this reproducibility project we make the following contributions:

- Algorithmic description of the R2D2 version of meta-learning with differentiable closed-form solvers (Algorithm 1).
- Evaluation of the MAML pipeline from Finn et al. (2017) on two datasets: the existing miniImageNet and new CIFAR-FS for different few-shot multi-class settings.
- Implementation of R2D2* in TensorFlow on the pipeline following Algorithm 1 with the original MAML feature extractor.
- Implementation of R2D2 in TensorFlow on the pipeline following Algorithm 1 with the Figure 2.3 architecture as mimicked from in the original paper (Bertinetto et al., 2019).
- Evaluation and insights in the reproducibility of Bertinetto et al. (2019).

2.7 Conclusion

In this work we have presented a reproducibility analysis of the ICLR 2019 paper *"Meta-learning with differentiable closed-form solvers"* by Bertinetto et al. (2019). Some parameters and training methodologies, which would be required for full reproducibility, such as *stride* and *padding* of the convolutional filters, and a clear stopping criterion, are not mentioned in the original paper or in its appendix (Bertinetto et al., 2019)). However, by making reasonable assumptions, we have been able to reproduce the most important parts of the paper and to achieve similar results. Most importantly we have succeeded in reproducing the increase in performance of the proposed method over some reproduced baseline results, which supports the conclusions of the original paper. However, the different neural network architectures should be taken into consideration when comparing results.

3 Regression Networks for Meta-Learning Few-Shot Classification

3.1 Preface

Summary: We propose *regression networks* for the problem of few-shot classification, where a classifier must generalize to new classes not seen in the training set, given only a small number of examples of each class. In high dimensional embedding spaces the direction of data generally contains richer information than magnitude. Next to this, state-of-the-art few-shot metric methods that compare distances with aggregated class representations, have shown superior performance. Combining these two insights, we propose to meta-learn classification of embedded points by regressing the closest approximation in every class subspace while using the regression error as a distance metric. Similarly to recent approaches for few-shot learning, regression networks reflect a simple inductive bias that is beneficial in this limited-data regime and they achieve excellent results, especially when more aggregate class representations can be formed with multiple shots.

This chapter is an edited version of Devos and Grossglauser (2020).

Code: https://github.com/ArnoutDevos/RegressionNet

Co-author: Matthias Grossglauser (MG).

Contributions:
AD: Conceptualization, Methodology, Software, Visualization, Investigation, Writing - Original Draft.
MG: Supervision, Writing - Review and Editing, Project Administration.

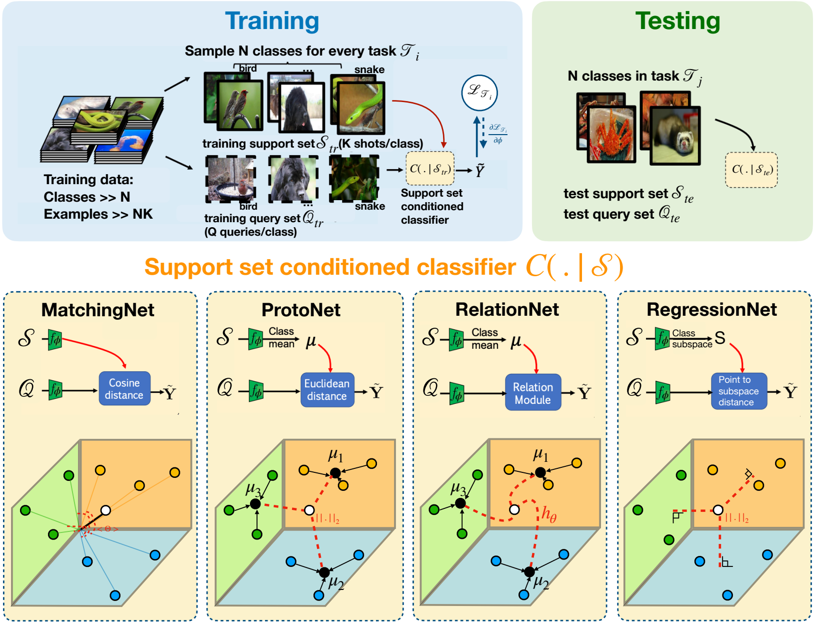

Training
Sample N classes for every task $\mathcal{T}_i$
bird
...
snake
training support set $\mathcal{S}_{tr}$ (K shots/class)
Training data:
Classes >> N
Examples >> NK
training query set $\mathcal{Q}_{tr}$
(Q queries/class)
$C(.|\mathcal{S}_{tr})$
$\tilde{Y}$
Support set conditioned classifier
$\mathcal{L}_{\mathcal{T}_i}$
$\frac{\partial \mathcal{L}_{\mathcal{T}_i}}{\partial \phi}$
Testing
N classes in task $\mathcal{T}_j$
test support set $\mathcal{S}_{te}$
test query set $\mathcal{Q}_{te}$
$C(.|\mathcal{S}_{te})$
Support set conditioned classifier $C(. | \mathcal{S})$
MatchingNet
ProtoNet
RelationNet
RegressionNet
$\mathcal{S}$
$\mathcal{Q}$
f_ϕ
Class mean
μ
Class subspace
S
Cosine distance
Euclidean distance
Relation Module
Point to subspace distance
$\tilde{Y}$
μ_1
μ_2
μ_3
$\|\cdot\|_2$
h_θ

Figure 3.1: Few-shot learning process (top) and metric-learning based methods (bottom), with $\mathcal{S}$ the support set (colored circles), $\mathcal{Q}$ the query set (white circle), $\widetilde{Y}$ the output distribution over classes for the query points, f_ϕ a neural embedding function, and h_θ a neural network based distance function. Figure inspired by W.-Y. Chen et al. (2019).

3.2 Introduction

The ability to adapt quickly to new situations is a cornerstone of human intelligence. Artificial learning methods have been shown to be very effective for specific tasks, often surpassing human performance (Silver et al., 2016; Esteva et al., 2017). However, by relying on standard training paradigms for supervised learning or reinforcement learning, these artificial methods still require much training data and training time to adapt to a new task.

An area of machine learning that learns and adapts from a small amount of data is called *few-shot learning* (Fei-Fei et al., 2006). A *shot* corresponds to a single example, e.g., an image and its label. In few-shot learning the learning scope is expanded from the classic setting of a single task with many shots to a variety of tasks with a few shots each. Several machine learning approaches have been used for this, including metric-learning (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018), meta-learning (Finn et al., 2017; Ravi and Larochelle, 2017), and generative models (Fei-Fei et al., 2006; B. M. Lake et al., 2015).

W.-Y. Chen et al. (2019) show that a metric-learning based method called *prototypical networks* (Snell et al., 2017), although simple in nature, achieves competitive performance with state-of-the-art meta-learning approaches such as MAML (Finn et al., 2017) and other metric-learning methods. Metric-learning methods approach the few-shot classification problem by "*learning to compare*". To learn high capacity nonlinear comparison models, most modern few-shot metric-learning methods use a neural embedding space to measure distance.

In image classification with neural embeddings, the embedding dimensions are usually high: e.g., 1600 for a *Conv-4* backbone (used later). In high dimensional image vector spaces "direction" of data generally contains richer information than "magnitude" (Zhe et al., 2019). MatchingNet (Vinyals et al., 2016) leverages purely directional information whereas ProtoNet (Snell et al., 2017) and RelationNet (Sung et al., 2018) mostly improve on this by comparing with aggregated class representations.

We propose *regression networks* which combine the good properties of directional information in high-dimensional vector spaces with rich aggregate class information. The proposed method is based on the idea that there exists an embedding in which every point from the same class can be approximated by a linear combination of other points in that same class. In order to do this, we learn a nonlinear mapping of the input space to an embedding space by using a neural network and regress the best embedded approximation, for each example. Classification of an embedded query point is then performed by simply finding the nearest class subspace by comparing regression errors.

Subspaces have been used to model images for decades in computer vision and machine learning (Fitzgibbon and Zisserman, 2003; Naseem et al., 2010). For example, the linear regression classification (LRC) method (Naseem et al., 2010) relies on the fact that the set of all reflectance functions produced by Lambertian objects, which parts of natural images are composed of, lie near a low-dimensional vector subspace (Basri and Jacobs, 2003). More recently, Simon et al. (2019) have explored few-shot learning with affine subspaces. Unlike our approach with vector subspaces, affine subspaces cannot be constructed with 1-shot learning, a key few-shot learning problem.

Figure 3.1 shows an overview of the few-shot learning process and a comparison of the proposed method with comparable state-of-the-art metric-learning based approaches.

3.3 Regression Networks

3.3.1 Notation

We formulate the N-way K-shot classification problem in an episodic way. Every episode has a small support set of N classes with K labeled examples $\mathcal{S} = \{(\mathbf{x}_{11}, y_{11}), \ldots, (\mathbf{x}_{NK}, y_{NK})\}$, and a query set of Q different examples $\mathcal{Q} = \{(\mathbf{x}_{1(K+1)}, y_{1(K+1)}), \ldots, (\mathbf{x}_{N(K+Q)}, y_{N(K+Q)})\}$. Note that the query set contains labels only during training, and the goal is to predict the labels of the query set during testing. In $\mathcal{S}$ and $\mathcal{Q}$ each $\mathbf{x}_{ij} \in \mathbb{R}^D$ is the D-dimensional feature vector of an example and $y_{ij} \in \{1, \dots, N\}$ is the corresponding label.

3.3.2 Model

Regression networks perform classification by regressing the best approximation to an embedding point in each aggregated class representation and subsequently using the regression error as a measure of distance. For this we start by constructing a K-dimensional embedded subspace $\mathbf{S}_n$ of each class n, given K shots per class, through an embedding function $f_\phi : \mathbb{R}^D \to \mathbb{R}^M$ with learnable parameters ϕ. With slight abuse of notation, every class is represented by its subspace matrix $\mathbf{S}_n \in \mathbb{R}^{M \times K}$, that contains the K vectors of the embedded class support points:

$$\mathbf{S}_n = \begin{bmatrix} f_\phi(\mathbf{x}_{n1}) & \dots & f_\phi(\mathbf{x}_{nK}) \end{bmatrix} \tag{3.1}$$

The *regression-error distance* $\widetilde{d}(\mathbf{e}_i, \mathbf{S}_n)$ of a point $\mathbf{e}_i \in \mathbb{R}^M$ in the embedding space to a class subspace $\mathbf{S}_n$ can be measured by regressing the closest point in the subspace in terms of a certain distance metric. The closest point can be constructed with a linear combination (represented by vector $\mathbf{a} \in \mathbb{R}^{K \times 1}$) of the embedded support examples spanning that space. By using a Euclidean distance metric, this can be formulated as a quadratic optimization problem in $\mathbf{a}$ of the following form:

$$\widetilde{d}(\mathbf{e}_i, \mathbf{S}_n) = \min_{\mathbf{a}} d(\mathbf{e}_i, \mathbf{S}_n \mathbf{a}) = \min_{\mathbf{a}} \|\mathbf{e}_i - \mathbf{S}_n \mathbf{a}\|_2 \tag{3.2}$$

The associated learning problem with this few-shot (K) high-dimensional (M) overdeterminded system is *least-squares linear regression.* Given that $M \geq K$, this is a well conditioned system, and it admits a *differentiable* closed-form solution (Friedman et al., 2001):

$$\begin{aligned} \widetilde{d}(\mathbf{e}_i, \mathbf{S}_n) &= \left\| \mathbf{e}_i - \mathbf{S}_n \left(\mathbf{S}_n^T \mathbf{S}_n \right)^{-1} \mathbf{S}_n^T \mathbf{e}_i \right\|_2 & (3.3) \\ &= \|\mathbf{e}_i - \mathbf{P}_n \mathbf{e}_i\|_2 & (3.4) \end{aligned}$$

where it is important to note that the matrix to be inverted is of size $K \times K$ and the number of shots K is usually small. The transformation matrix $\mathbf{P}_n$ projects the point $\mathbf{e}_i$ orthogonally onto the subspace spanned by the columns of $\mathbf{S}_n$ (Koç and Barkana, 2014). Note that this transformation matrix $\mathbf{P}_n$ has to be computed only for every class, not every example, which speeds up practical computation. Because the embedding function f_ϕ can output linearly dependent embeddings for different support examples, a small term $\lambda_1 > 0$ is added to avoid singularity:

$$\widetilde{d}(\mathbf{e}_i, \mathbf{S}_n) = \left\| \mathbf{e}_i - \mathbf{S}_n \left(\mathbf{S}_n^T \mathbf{S}_n + \lambda_1 \mathbf{I} \right)^{-1} \mathbf{S}_n^T \mathbf{e}_i \right\|_2 \tag{3.5}$$

With this point-to-subspace distance function $\widetilde{d} : \mathbb{R}^M \times \mathbb{R}^{M\times K} \rightarrow [0, +\infty)$, regression networks give a distribution over classes for a query point $\mathbf{x}$ based on a softmax over distances to each of the class subspaces in the embedding space:

$$p_\phi(y = n \mid \mathbf{x}) = \frac{\exp\left(-\widetilde{d}(f_\phi(\mathbf{x}), \mathbf{S}_n)\right)}{\sum_{n'} \exp\left(-\widetilde{d}(f_\phi(\mathbf{x}), \mathbf{S}_{n'})\right)} \tag{3.6}$$

Meta-learning continues by minimizing the negative log-probability function $\mathcal{L}_{base}(\phi) = -\log p_\phi(y = n \mid \mathbf{x})$ of the true class n via stochastic gradient descent (SGD). Training episodes are formed by randomly sampling a subset of N classes from the training set. Then, a subset of K examples within each class is chosen as the support set $\mathcal{S}$, and a subset of Q examples within each class is chosen as the query set $\mathcal{Q}$.

3.3.3 Subspace Orthogonalization

The K-dimensional class vector subspaces live in a much larger M-dimensional space. Therefore, we can exploit this freedom during training by making subspaces as directionally different as possible. Concretely, we propose to add a pairwise subspace orthogonolization term to the loss function:

$$\mathcal{L}_T = \mathcal{L}_{base} + \lambda_2 \sum_{i\neq j}^{N} \frac{\left\|S_i^T S_j\right\|_F^2}{\|S_i\|_F^2 \, \|S_j\|_F^2} \tag{3.7}$$

where $\|\cdot\|_F$ is the Frobenius norm and λ_2 is a scaling hyperparameter. Section 3.4.2 studies the effect of this term. We have also experimented with principal angles between vector subspaces (Bjorck and Golub, 1973), because they only depend on the subspaces, not on the (non-unique) set of points that define the subspaces as in Equation (3.7). Their results were comparable with our current approach, but come at a higher computational cost with a singular value decomposition.

Algorithm 5 in Appendix A.1 details the complete regression networks training procedure.

3.4 Experiments

In terms of few-shot learning evaluation, we focus on the natural image-based mini-ImageNet (Vinyals et al., 2016) dataset. To ensure a fair comparison with other methods, we perform experiments under the same conditions using the verified re-implementation (W.-Y. Chen et al., 2019) of MatchingNet, ProtoNet, RelationNet, MAML and extend it with R2D2 (Bertinetto et al., 2019). Compared to our direct approach, R2D2 is a meta-learning technique which leverages the closed-form solution of multinomial regression

	Conv-4		**ResNet-10**	
Method	**5-way 1-shot**	**5-way 5-shot**	**5-way 1-shot**	**5-way 5-shot**
MatchingNet	48.14±0.78	63.28±0.68	54.49±0.81	69.14±0.69
ProtoNet	44.42±0.84	65.15±0.67	51.98±0.84	73.77±0.64
RelationNet	49.31±0.85	65.33±0.70	52.19±0.83	69.97±0.68
MAML	46.47±0.82	62.71±0.71	54.69±0.89	66.62±0.83
R2D2	**50.07**±0.79	65.66±0.69	**55.71**±0.78	71.69±0.63
RegressionNet (ours)	47.02±0.77	**67.09**±0.69	55.44±0.86	**76.29**±0.59

Table 3.1: Average accuracies (%) of mini-ImageNet test tasks with 95% confidence intervals.

indirectly for classification (See Section 3.5). We decide to compare with these methods in particular, because they serve as the basis of many state-of-the-art few-shot classification algorithms (Oreshkin et al., 2018; Xing et al., 2019), and our method is easily interchanged with them. Experimental details can be found in Appendix A.2.

In this section, next to performance evaluation, we address the following research questions: (i) Can regression networks benefit from richer class representations and higher dimensions? (Section 3.4.1). (ii) How much effect does subspace orthogonalization have? (Section 3.4.2).

3.4.1 Few-shot Image Classification: mini-ImageNet

Table 3.1 shows the results for 5-way classification for mini-ImageNet for a different number of shots and backbones.

First, because regression networks are expected to benefit more from better subspace representations when more support examples are available per class, we investigate the effect of the number of shots. As expected, when increasing the number of shots K per class from 1 to 5, the classification accuracies increase for all methods. In the 5-shot case, RegressionNet significantly outperforms all other methods, showing the benefit of using rich class representations.

Secondly, as the backbone gets deeper, regression networks and prototypical networks begin to perform significantly better than matching networks and relation networks with R2D2 following close. Although the performance difference is small for mini-Imagenet, given a relatively deep feature extractor ResNet-10, regression networks outperform all other meta-learning and metric-learning methods when enough shots are available.

N-way K-shot	$\lambda_2 = 0$	$\lambda_2 > 0$
5-way 1-shot	54.83±0.83	55.44±0.86
5-way 5-shot	74.03±0.68	76.29±0.59

Table 3.2: Ablation study of effect of subspace orthogonalization stimulation (Equation (3.7)) using a ResNet-10 backbone on mini-ImageNet. 1-shot: $\lambda_2 = 10^{-3}$, 5-shot: $\lambda_2 = 10^{-2}$

3.4.2 Ablation study

In order to evaluate the effect of adding the subspace orthogonalization stimulating term to the loss function discussed in Section 3.3.3, we conduct an experiment without it. The results, comparing a ResNet-10 model trained with subspace orthogonalization and without, are shown in Table 3.2. All ablation experiments use ResNet-10 as a backbone.

Under all settings considered, subspace orthogonalization gives a classification accuracy improvement (up to 2%). Note that, even without subspace orthogonalization, our proposed method is still competitive with all other methods in Table 3.1.

3.5 Related work

In addition to the reproduced metric (meta-)learning based few-shot methods (Snell et al., 2017; Vinyals et al., 2016; Sung et al., 2018; Bertinetto et al., 2019), there is a large body of work on few-shot learning and metric (meta-)learning. We discuss work that is more closely related to regression networks in particular.

Our approach shows similarities to the linear regression classification (LRC) method (Naseem et al., 2010), where each class is represented by the vector subspace spanned by its examples. LRC was developed for face recognition, where only a few examples are available, however it relies on a linear embedding. Our approach also uses few examples, but it incorporates neural networks in order to nonlinearly embed examples and we couple this with episodic training to handle the meta-learning few-shot scenario.

Simon et al. (2019) have explored affine subspace representations for few-shot learning. In contrast to our closed-form linear regression approach, they make use of a truncated singular value decomposition (SVD) of the support example matrix. Affine subspaces cannot be constructed with 1-shot learning, a key few-shot learning problem. In contrast, our closed-form linear regression approach relies on vector subspaces, which can be constructed with 1-shot learning.

Bertinetto et al. (2019) propose to use regularized linear regression as a classifier on top of the embedding function. Doing so, they directly approach a classification problem with a regression method, but they show competitive results. To achieve this, they introduce learnable scalars that scale and shift the regression outputs for them to be used in the cross-entropy loss. Regression networks rely on the same closed-form solver for linear regression to compute the transformation matrices, but are inherently designed for classification problems because of their similarity to the LRC method (Naseem et al., 2010).

3.6 Conclusion

We have proposed regression networks for meta-learning few-shot classification. The method assumes that for any embedded point we can regress the closest approximation in every class representation and use the error as a distance measure. Classes are represented by their embedded vector subspaces, which are spanned by their examples. The approach produces better results than other state-of-the-art metric-learning based methods, when rich class representations can be formed with multiple shots. Stimulating subspace orthogonality consistently improves performance. A direction for future work is to study the effect of using a low-rank approximation of the class subspace. Overall, the simplicity and effectiveness of regression networks makes it a promising approach for metric-based few-shot classification.

4 Self-Supervised Prototypical Transfer Learning for Few-Shot Classification

4.1 Preface

Summary: Most approaches in few-shot learning rely on costly annotated data related to the goal task domain during (pre-)training. Recently, unsupervised meta-learning methods have exchanged the annotation requirement for a reduction in few-shot classification performance. Simultaneously, in settings with realistic domain shift, common transfer learning has been shown to outperform supervised meta-learning. Building on these insights and on advances in self-supervised learning, we propose a transfer learning approach which constructs a metric embedding that clusters unlabeled prototypical samples and their augmentations closely together. This pre-trained embedding is a starting point for few-shot classification by summarizing class clusters and fine-tuning. We demonstrate that our self-supervised prototypical transfer learning approach ProtoTransfer outperforms state-of-the-art unsupervised meta-learning methods on few-shot tasks from the mini-ImageNet dataset. In few-shot experiments with domain shift, our approach even has comparable performance to supervised methods, but requires orders of magnitude fewer labels.

This chapter is an edited version of Medina et al. (2020).

Code: https://github.com/indy-lab/ProtoTransfer

Co-authors: Carlos Medina (CM), Matthias Grossglauser (MG).

Contributions:
AD: Conceptualization, Methodology, Visualization, Investigation, Writing - Original Draft, Supervision.
CM: Conceptualization, Methodology, Software, Visualization, Investigation, Writing - Original Draft.
MG: Supervision, Writing - Review & Editing, Project Administration.

4.2 Introduction

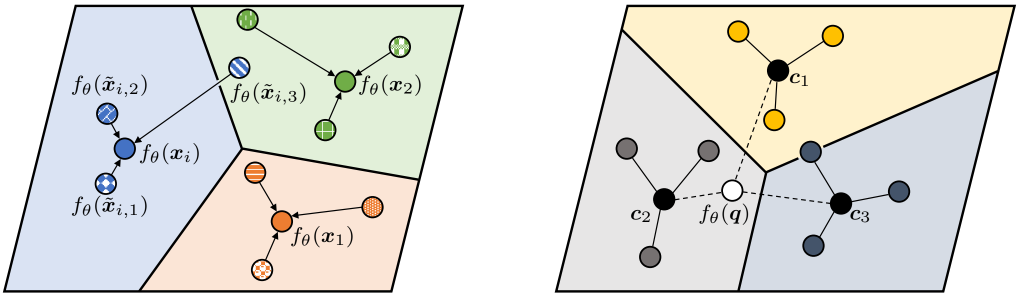
Few-shot classification (Fei-Fei et al., 2006) is a learning task in which a classifier must adapt to distinguish novel classes not seen during training, given only a few examples (shots) of these classes. Meta-learning (Finn et al., 2017; Ren et al., 2018) is a popular approach for few-shot classification by mimicking the test setting during training through so-called episodes of learning with few examples from the training classes. However, several works (W.-Y. Chen et al., 2019; Guo et al., 2019) show that common (non-episodical) transfer learning outperforms meta-learning methods on the realistic cross-domain setting, where training and novel classes come from different distributions.

Nevertheless, most few-shot classification methods still require much annotated data for pre-training. Recently, several unsupervised meta-learning approaches, constructing episodes via pseudo-labeling (Hsu et al., 2019; Ji et al., 2019) or image augmentations (Khodadadeh et al., 2019; Antoniou and Storkey, 2019; Qin et al., 2020), have addressed this problem. To our knowledge, unsupervised non-episodical techniques for transfer learning to few-shot tasks have not yet been explored.

Our approach ProtoTransfer performs self-supervised pre-training on an unlabeled training domain and can transfer to few-shot target domain tasks. During pre-training, we minimize a pairwise distance loss in order to learn an embedding that clusters noisy transformations of the same image around the original image. Our pre-training loss can be seen as a self-supervised version of the prototypical loss in Snell et al. (2017) in line with contrastive learning, which has driven recent advances in self-supervised representation learning (Ye et al., 2019; T. Chen et al., 2020; He et al., 2019). In the few-shot target task, in line with pre-training, we summarize class information in class prototypes for nearest neighbor inference similar to ProtoNet (Snell et al., 2017) and we support fine-tuning to improve performance when multiple examples are available per class.

We highlight our main contributions and results:

1. We show that our approach outperforms state-of-the-art unsupervised meta-learning methods by 5% to 8% on mini-ImageNet few-shot classification tasks and has competitive performance on Omniglot.
2. Compared to the fully supervised setting, our approach achieves competitive performance on mini-ImageNet and multiple datasets from the cross-domain transfer learning CDFSL benchmark, with the benefit of not requiring labels during training.
3. In an ablation study and cross-domain experiments we show that using a larger number of equivalent training classes than commonly possible with episodical meta-learning, and parametric fine-tuning are key to obtaining performance matching supervised approaches.


$f_\theta(\tilde{\boldsymbol{x}}_{i,2})$
$f_\theta(\tilde{\boldsymbol{x}}_{i,3})$
$f_\theta(\boldsymbol{x}_2)$
$f_\theta(\boldsymbol{x}_i)$
$f_\theta(\tilde{\boldsymbol{x}}_{i,1})$
$f_\theta(\boldsymbol{x}_1)$
$\boldsymbol{c}_1$
$\boldsymbol{c}_2$
$f_\theta(\boldsymbol{q})$
$\boldsymbol{c}_3$

(a) Self-Supervised Prototypical Pre-Training (b) Prototypical Fine-Tuning & Inference

Figure 4.1: Self-Supervised Prototypical Transfer Learning. (a): In the embedding, original images $\boldsymbol{x}_i$ serve as class prototypes around which their Q augmented views $\tilde{\boldsymbol{x}}_{i,q}$ should cluster. (b): Prototypes $\boldsymbol{c}_n$ are the means of embedded support examples for each class n and initialize a final linear layer for fine-tuning. An embedded query point $\boldsymbol{q}$ is classified via a softmax over the fine-tuned linear layer.

4.3 A Self-Supervised Prototypical Transfer Learning Algorithm

Section 4.3.1 introduces the few-shot classification setting and relevant terminology. Further, we describe ProtoTransfer's pre-training stage, ProtoCLR, in Section 4.3.2 and its fine-tuning stage, ProtoTune, in Section 4.3.3. Figure 4.1 illustrates the procedure.

4.3.1 Preliminaries

The goal of few-shot classification is to predict classes for a set of unlabeled points (the *query set*) given a small set of labeled examples (the *support set*) from the same classes. Few-shot classification approaches commonly consist of two subsequent learning phases, each using its own set of classes.

The first learning phase utilizes samples from N_b base (training) classes contained within a training set $D_b = \{(\boldsymbol{x}, y)\} \subset I \times Y_b$, where $\boldsymbol{x} \in I$ is a sample with label y in label set Y_b. An important aspect of our specific unsupervised learning setting is that the first phase has no access to the per-sample label information, the distribution of classes, nor the size of the label set Y_b, for pre-training. This first phase serves as a preparation for the actual few-shot learning in the target domain, i.e. the second learning phase. This second supervised learning phase contains N_n novel (testing) classes as $D_n = \{(\boldsymbol{x}, y)\} \subset I \times Y_n$, where only few examples for each of the classes in Y_n are available. Concretely, an N_n-way K-shot classification task consists of K labeled examples for each of the N_n novel classes. In the few-shot learning literature a task is also commonly referred to as an episode.

Algorithm 2 Self-Supervised Prototypical Pre-Training (ProtoCLR)

```
1:  input: batch size $N$, augmentations size $Q$, embedding function $f_\theta$, set of random
    transformations $\mathcal{T}$, step size $\alpha$, distance function $d[\cdot,\cdot]$
2:  Randomly initialize $\theta$
3:  while not done do
4:      Sample minibatch $\{\boldsymbol{x}_i\}_{i=1}^{N}$
5:      for all $i \in \{1,\ldots,N\}$ do
6:          for all $q \in \{1,\ldots,Q\}$ do
7:              draw a random transformation $t \sim \mathcal{T}$
8:              $\tilde{\boldsymbol{x}}_{i,q} = t(\boldsymbol{x}_i)$
9:          end for
10:     end for
11:     let $\ell(i,q) = -\log \frac{\exp(-d[f(\tilde{\boldsymbol{x}}_{i,q}), f(\boldsymbol{x}_i)])}{\sum_{k=1}^{N} \exp(-d[f(\tilde{\boldsymbol{x}}_{i,q}), f(\boldsymbol{x}_k)])}$
12:
13:     $\mathcal{L} = \frac{1}{NQ} \sum_{i=1}^{N} \sum_{q=1}^{Q} \ell(i,q)$
14:     $\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}$
15: end while
16: return embedding function $f_\theta(\cdot)$
```

4.3.2 Self-Supervised Prototypical Pre-Training: ProtoCLR

Similar to the few-shot target tasks, we frame every ProtoCLR pre-training learning step as an N-way 1-shot classification task optimized by a contrastive loss function as described below. In this, we draw inspiration from recent progress in unsupervised meta-learning (Khodadadeh et al., 2019) and self-supervised visual contrastive learning of representations (T. Chen et al., 2020; Ye et al., 2019).

Algorithm 2 details ProtoCLR and it comprises the following parts:

- Batch generation (Algorithm 2 lines 4-10): Each mini-batch contains N random samples $\{\mathbf{x}_i\}_{i=1\ldots N}$ from the training set. As our self-supervised setting does not assume any knowledge about the base class labels Y_b, we treat each sample as it's own class. Thus, each sample $\boldsymbol{x}_i$ serves as a 1-shot support sample and class prototype. For each prototype $\boldsymbol{x}_i$, Q different randomly augmented versions $\tilde{\boldsymbol{x}}_{i,q}$ are used as query samples.
- Contrastive prototypical loss optimization (Algorithm 2 lines 11-13): The pre-training loss encourages clustering of augmented query samples $\{\tilde{\boldsymbol{x}}_{i,q}\}$ around their prototype $\boldsymbol{x}_i$ in the embedding space through a distance metric $d[\cdot,\cdot]$. The softmax cross-entropy loss over N classes is minimized with respect to the embedding parameters θ with mini-batch stochastic gradient descent (SGD).

Commonly, unsupervised pre-training approaches for few-shot classification (Hsu et al., 2019; Khodadadeh et al., 2019; Antoniou and Storkey, 2019; Qin et al., 2020; Ji et al., 2019) rely on meta-learning. Thus, they are required to create small artificial N-way (K-shot) tasks identical to the downstream few-shot classification tasks. Our approach does not use meta-learning and can use any batch size N. Larger batch sizes have been shown to help self-supervised representation learning (T. Chen et al., 2020) and supervised pre-training for few-shot classification (Snell et al., 2017). We also find that larger batches yield a significant performance improvement for our approach (see Section 4.4.3). To generate the query examples, we use image augmentations similar to (T. Chen et al., 2020) and adjust them for every dataset. The exact transformations are listed in Appendix B.1.3. Following Snell et al. (2017), we use the Euclidean distance, but our method is generic and works with any metric.

4.3.3 Supervised Prototypical Fine-Tuning: ProtoTune

After pre-training the metric embedding $f_\theta(\cdot)$, we address the target task of few-shot classification. For this, we extend the prototypical nearest-neighbor classifier ProtoNet (Snell et al., 2017) with prototypical fine-tuning of a final classification layer, which we refer to as ProtoTune. First, the class prototypes $\boldsymbol{c}_n$ are computed as the mean of the class samples in the support set S of the few-shot task:

$$\boldsymbol{c}_n = \frac{1}{|S_n|} \sum_{(\boldsymbol{x}_i, y_i = n) \in S} f_\theta(\boldsymbol{x}_i).$$

ProtoNet uses non-parametric nearest-neighbor classification with respect to $\boldsymbol{c}_n$ and can be interpreted as a linear classifier applied to a learned representation $f_\theta(\mathbf{x})$. Following the derivation in Snell et al. (2017), we initialize a final linear layer with weights $\mathbf{W}_n = 2\mathbf{c}_n$ and biases $b_n = -||\mathbf{c}_n||^2$. Then, this final layer is fine-tuned with a softmax cross-entropy loss on samples from S, while keeping the embedding function parameters θ fixed. Triantafillou et al. (2020) proposed a similar fine-tuning approach with prototypical initialization, but their approach always fine-tunes all model parameters.

4.4 Experiments

We carry out several experiments to benchmark and analyze ProtoTransfer. In Section 4.4.1, we conduct in-domain classification experiments on the Omniglot (B. Lake et al., 2011) and mini-ImageNet (Vinyals et al., 2016) benchmarks to compare to state-of-the-art unsupervised few-shot learning approaches and methods with supervised pre-training. In Section 4.4.2, we test our method on a more challenging cross-domain few-shot learning benchmark (Guo et al., 2019). Section 4.4.3 contains an ablation study showing how the different components of ProtoTransfer contribute to its performance. In Section 4.4.4, we study how pre-training with varying class diversities affects performance. In Section 4.4.5, we give insight in generalization from training classes to novel classes from both unsupervised and supervised perspectives. Experimental details can be found in Appendix B.1 and code is made available[I].

4.4.1 In-Domain Few-shot Classification: Omniglot and mini-ImageNet

For our in-domain experiments, where the disjoint training class set and novel class set come from the same distribution, we used the popular few-shot datasets Omniglot (B. Lake et al., 2011) and mini-ImageNet (Vinyals et al., 2016). For comparability we use the Conv-4 architecture proposed in Vinyals et al. (2016). Specifics on the datasets, architecture and optmization can be found in Appendices B.1.1 and B.1.2. We apply limited hyperparameter tuning, as suggested in Oliver et al. (2018), and use a batch size of $N = 50$ and number of query augmentations $Q = 3$ for all datasets.

In Table 4.1, we report few-shot accuracies on the mini-ImageNet and Omniglot benchmarks. We compare to unsupervised clustering based methods CACTUs (Hsu et al., 2019) and UFLST (Ji et al., 2019) as well as the augmentation based methods UMTRA (Khodadadeh et al., 2019), AAL (Antoniou and Storkey, 2019) and ULDA (Qin et al., 2020). More details on how these approaches compare to ours can be found in Section 4.5. Pre+Linear represents classical supervised transfer learning, where a deep neural network classifier is (pre)trained on the training classes and then only the last *linear* layer is fine-tuned on the novel classes. On mini-ImageNet, ProtoTransfer outperforms all other state-of-the-art unsupervised pre-training approaches by at least 5% up to 8% and mostly outperforms the supervised meta-learning method MAML (Finn et al., 2017), while requiring orders of magnitude fewer labels (NK vs $38400 + NK$). On Omniglot, ProtoTransfer shows competitive performance with most unsupervised meta-learning approaches.

4.4.2 Cross-domain Few-Shot Classification: CDFSL benchmark

For our cross-domain experiments, where training and novel classes come from different distributions, we turn to the CDFSL benchmark (Guo et al., 2019). This benchmark specifically tests how well methods trained on mini-ImageNet can transfer to few-shot tasks with only limited similarity to mini-ImageNet. In order of decreasing similarity, the four datasets are plant disease images from CropDiseases (Mohanty et al., 2016), satellite images from EuroSAT (Helber et al., 2019), dermatological images from ISIC2018 (Tschandl et al., 2018; Codella et al., 2019) and grayscale chest X-ray images from ChestX (Wang et al., 2017). Following Guo et al. (2019), we use a ResNet-10 neural network architecture. As there is no validation data available for the target tasks in CDFSL, we keep the same ProtoTransfer hyperparameters $N = 50$, $Q = 3$ as used in the mini-ImageNet experiments. Experimental details are listed in Appendices B.1.1 and

---

[I] Our code and pre-trained models are available at `https://www.github.com/indy-lab/ProtoTransfer`

Table 4.1: Accuracy (%) of unsupervised pre-training methods on N-way K-shot classification tasks on Omniglot and mini-Imagenet on a Conv-4 architecture. For detailed results, see Tables B.2 and B.3 in the Appendix. Results style: **best** and second best.

Method (N,K)	**(5,1)**	**(5,5)**	**(20,1)**	**(20,5)**	**(5,1)**	**(5,5)**	**(5,20)**	**(5,50)**
	Omniglot				**mini-ImageNet**			
Training (scratch)	52.50	74.78	24.91	47.62	27.59	38.48	51.53	59.63
CACTUs-MAML	68.84	87.78	48.09	73.36	39.90	53.97	63.84	69.64
CACTUs-ProtoNet	68.12	83.58	47.75	66.27	39.18	53.36	61.54	63.55
UMTRA	83.80	95.43	74.25	92.12	39.93	50.73	61.11	67.15
AAL-ProtoNet	84.66	89.14	68.79	74.28	37.67	40.29	-	-
AAL-MAML++	88.40	97.96	70.21	88.32	34.57	49.18	-	-
UFLST	**97.03**	**99.19**	**91.28**	**97.37**	33.77	45.03	53.35	56.72
ULDA-ProtoNet	-	-	-	-	40.63	55.41	63.16	65.20
ULDA-MetaOptNet	-	-	-	-	40.71	54.49	63.58	67.65
ProtoTransfer (ours)	88.00	96.48	72.27	89.08	**45.67**	**62.99**	**72.34**	**77.22**
Supervised training								
MAML	94.46	98.83	84.60	96.29	46.81	62.13	71.03	75.54
ProtoNet	97.70	99.28	94.40	98.39	46.44	66.33	76.73	78.91
Pre+Linear	94.30	99.08	86.05	97.11	43.87	63.01	75.46	80.17

B.1.2.

For comparison to unsupervised meta-learning, we include our results on UMTRA-ProtoNet and its fine-tuned version UMTRA-ProtoTune (Khodadadeh et al., 2019). Both use our augmentations instead of those from (Khodadadeh et al., 2019). For further comparison, we include ProtoNet (Snell et al., 2017) for supervised few-shot learning and Pre+Mean-Centroid and Pre+Linear as the best-on-average performing transfer learning approaches from Guo et al. (2019). As the CDFSL benchmark presents a large domain shift with respect to mini-ImageNet, all model parameters are fine-tuned in ProtoTransfer during the few-shot fine-tuning phase with ProtoTune.

We report results on the CDFSL benchmark in Table 4.2. ProtoTransfer consistently outperforms its meta-learned counterparts by at least 0.7% up to 19% and performs mostly on par with the supervised transfer learning approaches. Comparing the results of UMTRA-ProtoNet and UMTRA-ProtoTune, starting from 5 shots, parametric fine-tuning gives improvements ranging from 1% to 13%. Notably, on the dataset with the largest domain shift (ChestX), ProtoTransfer outperforms all other approaches.

Table 4.2: Accuracy (%) of methods on N-way K-shot (N,K) classification tasks of the CDFSL benchmark (Guo et al., 2019). Both our results on methods with unsupervised pre-training (UnSup) and results on methods with supervised pre-training from CDFSL are listed. All models are trained on mini-ImageNet with ResNet-10. For detailed results, see Appendix Table B.4. Results style: **best** and second best.

Method	**UnSup**	**(5,5)**	**(5,20)**	**(5,50)**	**(5,5)**	**(5,20)**	**(5,50)**
			ChestX			**ISIC**	
ProtoNet		24.05	28.21	29.32	39.57	49.50	51.99
Pre+Mean-Centroid		26.31	30.41	34.68	47.16	56.40	61.57
Pre+Linear		25.97	31.32	35.49	**48.11**	**59.31**	**66.48**
UMTRA-ProtoNet	✓	24.94	28.04	29.88	39.21	44.62	46.48
UMTRA-ProtoTune	✓	25.00	30.41	35.63	38.47	51.60	60.12
ProtoTransfer (ours)	✓	**26.71**	**33.82**	**39.35**	45.19	59.07	66.15
			EuroSat			**CropDiseases**	
ProtoNet		73.29	82.27	80.48	79.72	88.15	90.81
Pre+Mean-Centroid		**82.21**	87.62	88.24	87.61	93.87	94.77
Pre+Linear		79.08	**87.64**	**91.34**	**89.25**	**95.51**	**97.68**
UMTRA-ProtoNet	✓	74.91	80.42	82.24	79.81	86.84	88.44
UMTRA-ProtoTune	✓	68.11	81.56	85.05	82.67	92.04	95.46
ProtoTransfer (ours)	✓	75.62	86.80	90.46	86.53	95.06	97.01

4.4.3 Ablation Study: Batch Size, Number of Queries, and Fine-Tuning

We conduct an ablation study of ProtoTransfer's components to see how they contribute to its performance. Starting from ProtoTransfer we successively remove components to arrive at the equivalent UMTRA-ProtoNet which shows similar performance to the original UMTRA approach (Khodadadeh et al., 2019) on mini-ImageNet. As a reference, we provide results of a ProtoNet classifier on top of a fixed randomly initialized network.

Table 4.3 shows that increasing the batch size from $N = 5$ for UMTRA-ProtoNet to 50 for ProtoCLR-ProtoNet, keeping everything else equal, is crucial to our approach and yields a 5% to 9% performance improvement. Importantly, UMTRA-ProtoNet uses our augmentations instead of those from (Khodadadeh et al., 2019). Thus, this improvement cannot be attributed to using different augmentations than UMTRA. Increasing the training query number to $Q = 3$ gives better gradient information and yields a relatively small but consistent performance improvement. Fine-tuning in the target domain does not always give a net improvement. Generally, when many shots are available, fine-tuning gives a significant boost in performance as exemplified by ProtoCLR-ProtoTune and UMTRA-MAML in the 50-shot case. Interestingly, our approach reaches competitive performance in the few-shot regime even before fine-tuning.

Table 4.3: Accuracy (%) of methods on N-way K-shot (N, K) classification tasks on mini-ImageNet with a Conv-4 architecture for different training image batch sizes, number of training queries (Q) and optional finetuning on target tasks (FT). UMTRA-MAML results are taken from Khodadadeh et al. (2019), where UMTRA uses AutoAugment (Cubuk et al., 2019) augmentations. For detailed results see Table B.5 in the Appendix. Results style: **best** and second best.

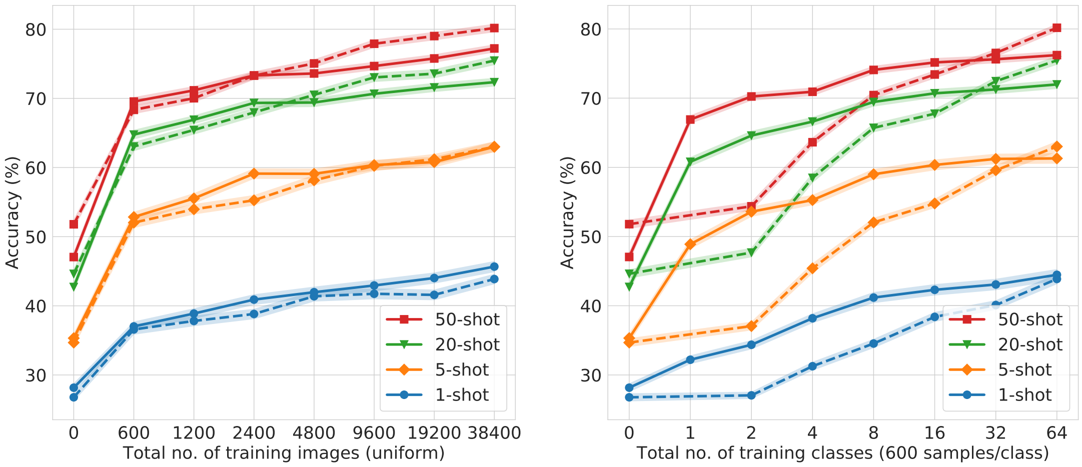
Training	**Testing**	**batch size**	**Q**	**FT**	**(5,1)**	**(5,5)**	**(5,20)**	**(5,50)**
n.a.	ProtoNet	n.a.	n.a.	no	27.05	34.12	39.68	41.40
UMTRA	MAML	$N(=5)$	1	yes	39.93	50.73	61.11	67.15
UMTRA	ProtoNet	$N(=5)$	1	no	39.17	53.78	62.41	64.40
ProtoCLR	ProtoNet	50	1	no	44.53	62.88	70.86	73.93
ProtoCLR	ProtoNet	50	3	no	44.89	**63.35**	72.27	74.31
ProtoCLR	ProtoTune	50	3	yes	**45.67**	62.99	**72.34**	**77.22**

4.4.4 Number of Training Classes and Samples

While ProtoTransfer already does not require any labels during pre-training, for some applications, e.g. rare medical conditions, even the collection of sufficiently similar data might be difficult. Thus, we test our approach when reducing the total number of available training images under the controlled setting of mini-ImageNet. Moreover, not all training datasets will have such a diverse set of classes to learn from as the different animals, vehicles and objects in mini-ImageNet. Therefore, we also test the effect of reducing the number of training classes and thereby the class diversity. To contrast the effects of reducing the number of classes or reducing the number of samples, we either remove whole classes from the mini-ImageNet training set or remove the corresponding amount of samples randomly from all classes. The number of samples are decreased in multiples of 600 as each mini-ImageNet class contains exactly 600 samples. We compare the mini-ImageNet few-shot classification accuracies of ProtoTransfer to the popular supervised transfer learning baseline Pre+Linear in Figure 4.2.

As expected, when uniformily reducing the number of images from all classes (Figure 4.2a), the few-shot classification accuracy is reduced as well. The performance of ProtoTransfer and the supervised baseline closely match in this case. When reducing the number of training classes in Figure 4.2b, ProtoTransfer consistently and significantly outperforms the supervised baseline when the number of mini-ImageNet training classes drops below 16. For example in the 20-shot case with only two training classes, ProtoTransfer outperforms the supervised baseline by a large margin of 16.9% (64.59% vs 47.68%). Comparing ProtoTransfer in Figures 4.2a and 4.2b, there is only a small difference between reducing images randomly from all classes or taking entire classes away. In contrast, the supervised baseline performance suffers substantially from having fewer classes.

To validate these in-domain observations in a cross-domain setting, following Devos and



(a) Varying number of training images. (b) Varying number of training classes.

Figure 4.2: 5-way K-shot accuracies with 95% confidence intervals on mini-ImageNet as a function of training images and classes. Methods: ProtoTransfer (——), transfer learning baseline Pre+Linear (- - -). Note the logarithmic scale. Detailed results available in Table B.6 in the appendix.

Grossglauser (2020), we compare few-shot classification performance when training on CUB (Welinder et al., 2010; Wah et al., 2011) and testing on mini-ImageNet (Vinyals et al., 2016). CUB consists of 200 classes of birds, while only three of the 64 mini-ImageNet training classes are birds (see B.1.1, B.1.2 for details on CUB). Thus CUB possesses a lower class diversity than mini-ImageNet. Table 4.4 confirms our previous observation numerically and shows that ProtoTransfer has a superior transfer accuracy of 2% to 4% over the supervised approach when limited diversity is available in the training classes.

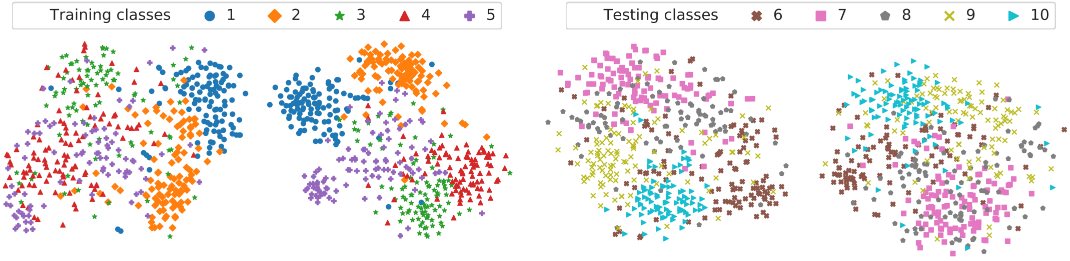
We conjecture that this difference is due to the fact that our self-supervised approach does not make a difference between samples coming from the same or different (latent) classes during training. Thus, we expect it to learn discriminative features despite a low training class diversity. In contrast, the supervised case forces multiple images with rich features into the same classes. We thus expect the generalization gap between tasks coming from training classes and testing classes to be smaller with self-supervision. We provide evidence to support this conjecture in Section 4.4.5.

4.4.5 Task Generalization Gap

To compare the generalization of ProtoCLR with its supervised embedding learning counterpart ProtoNet (Snell et al., 2017), we visualize the learned embedding spaces with t-SNE (Maaten and G. Hinton, 2008) in Figure 4.3. We compare both methods on samples from 5 random classes from the training and testing sets of mini-ImageNet. In Figures 4.3a and 4.3b we observe that, for the same training classes, ProtoNet shows

Table 4.4: Accuracy (%) on N-way K-shot (N, K) classification tasks on mini-ImageNet for methods trained on the CUB training set (5885 images) with a Conv-4 architecture. All results indicate 95% confidence intervals over 600 randomly generated test episodes. Results style: **best** and second best.

Training	**Testing**	**(5,1)**	**(5,5)**	**(5,20)**	**(5,50)**
ProtoCLR	ProtoNet	34.56 ± 0.61	**52.76** ± 0.63	62.76 ± 0.59	66.01 ± 0.55
ProtoCLR	ProtoTune	**35.37** ± 0.63	52.38 ± 0.66	**63.82** ± 0.59	**68.95** ± 0.57
Pre(training)	Linear	33.10 ± 0.60	47.01 ± 0.65	59.94 ± 0.62	65.75 ± 0.63



(a) ProtoCLR training (b) ProtoNet training (c) ProtoCLR testing (d) ProtoNet testing

Figure 4.3: t-SNE plots of trained embeddings on 5 classes from the training and testing sets of mini-ImageNet. Trained embeddings considered are self-supervised ProtoCLR and supervised 20-way 5-shot ProtoNet. For details on the depicted classes, please refer to Appendix B.1.4.

more structure. Comparing all subfigures in Figure 4.3, ProtoCLR shows more closely related embeddings in Figures 4.3a and 4.3c than ProtoNet in Figures 4.3b and 4.3d.

These visual observations are supported numerically in Table 4.5. Self-supervised embedding approaches, such as UMTRA and our ProtoCLR approach, show a much smaller task generalization gap than supervised ProtoNet. ProtoCLR shows virtually no classification performance drop. However, supervised ProtoNet suffers a significant accuracy reduction of 6% to 12%.

4.5 Related Work

Unsupervised meta-learning: Both CACTUs (Hsu et al., 2019) and UFLST (Ji et al., 2019) alternate between clustering for support and query set generation and employing standard meta-learning. In contrast, our method unifies self-supervised clustering and inference in a single model. Khodadadeh et al. (2019) propose an unsupervised model-agnostic meta-learning approach (UMTRA), where artifical N-way 1-shot tasks are generated by randomly sampling N support examples from the training set and generating N corresponding queries by augmentation. Antoniou and Storkey (2019) (AAL) generalize

Table 4.5: Accuracy (%) of N-way K-shot (N,K) classification tasks from the training and testing split of mini-ImageNet. Following Snell et al. (2017), ProtoNet is trained with 30-way 1-shot for 1-shot tasks and 20-way K-shot otherwise. All results use a Conv-4 architecture. All results show 95% confidence intervals over 600 randomly generated episodes.

Training	**Testing**	**Data**	**(5,1)**	**(5,5)**	**(5,20)**	**(5,50)**
ProtoNet	ProtoNet	Train	53.74 ± 0.95	79.09 ± 0.69	85.53 ± 0.53	86.62 ± 0.48
ProtoNet	ProtoNet	Val	46.62 ± 0.82	67.34 ± 0.69	76.44 ± 0.57	79.00 ± 0.53
ProtoNet	ProtoNet	Test	46.44 ± 0.78	66.33 ± 0.68	76.73 ± 0.54	78.91 ± 0.57
UMTRA	ProtoNet	Train	41.03 ± 0.79	56.43 ± 0.78	64.48 ± 0.71	66.28 ± 0.66
UMTRA	ProtoNet	Test	38.92 ± 0.69	53.37 ± 0.68	61.69 ± 0.66	65.12 ± 0.59
ProtoCLR	ProtoNet	Train	45.33 ± 0.63	63.47 ± 0.58	71.51 ± 0.51	73.99 ± 0.49
ProtoCLR	ProtoNet	Test	44.89 ± 0.58	63.35 ± 0.54	72.27 ± 0.45	74.31 ± 0.45

this approach to more support shots by randomly grouping augmented images into classes for classification tasks. ULDA (Qin et al., 2020) induce a distribution shift between the support and query set by applying different types of augmentations to each. In contrast, ProtoTransfer uses a single un-augmented support sample, similar to Khodadadeh et al. (2019), but extends to several query samples for better gradient signals and steps away from artificial few-shot task sampling by using larger batch sizes, which is key to learning stronger embeddings.

Supervised meta-learning aided by self-supervision: Several works have proposed to use a self-supervised loss either alongside supervised meta-learning episodes (Gidaris et al., 2019; S. Liu et al., 2019) or to initialize a model prior to supervised meta-learning on the source domain (D. Chen et al., 2019; Su et al., 2019). In contrast, we do not require any labels during training.

Fine-tuning for few-shot classification: W.-Y. Chen et al. (2019) show that adaptation on the target task is key for good cross-domain few-shot classification performance. Similar to ProtoTune, Triantafillou et al. (2020) also initialize a final layer with prototypes after supervised meta-learning, but always fine-tune all parameters of the model.

Contrastive loss learning: Contrastive losses have fueled recent progress in learning strong embedding functions (Ye et al., 2019; T. Chen et al., 2020; He et al., 2019; Tian et al., 2020; Junnan Li et al., 2020). Most similar to our approach is Ye et al. (2019). They propose a per-batch contrastive loss that minimizes the distance between an image and an augmented version of it. Different to us, they do not generalize to using multiple augmented query images per prototype and use 2 extra fully connected layers during training. Concurrently, Junnan Li et al. (2020) also use a prototype-based contrastive loss. They compute the prototypes as centroids after clustering augmented images via k-Means. They also separate learning and clustering procedures, which ProtoTransfer achieves in a single procedure.

4.6 Conclusion

In this work, we proposed ProtoTransfer for few-shot classification. ProtoTransfer performs transfer learning from an unlabeled source domain to a target domain with only a few labeled examples. Our experiments show that on mini-ImageNet it outperforms all prior unsupervised few-shot learning approaches by a large margin. On a more challenging cross-domain few-shot classification benchmark, ProtoTransfer shows similar performance to fully supervised approaches. Our ablation studies show that large batch sizes are crucial to learning good representations for downstream few-shot classification tasks and that parametric fine-tuning on target tasks can significantly boost performance.

5 A meta-learning approach for genomic survival analysis

5.1 Preface

Summary: RNA sequencing has emerged as a promising approach in cancer prognosis as sequencing data becomes more easily and affordably accessible. It is challenging in biomedical settings to build good predictive models since commonly the sample size is limited and the number of features is high. To address these limitations, we propose a meta-learning framework based on neural networks for survival analysis and evaluate it in a genomic cancer research setting. We demonstrate that, compared to regular transfer-learning, meta-learning is a significantly more effective paradigm in this setting. We show that our genomic framework achieves competitive performance with few samples compared to learning from scratch with a significantly larger number of samples. Finally, we demonstrate that the framework implicitly prioritizes genes based on their contribution to survival prediction and allows to identify important pathways in cancer.

This chapter is an edited version of Qiu et al. (2020). It is reproduced with permission of Springer Nature as well as its Creative Commons Attribution 4.0 International License: https://www.nature.com/articles/s41467-020-20167-3

Code: https://github.com/gevaertlab/metalearning_survival

Co-authors: Yeping Lina Qiu (YLQ), Hong Zheng (HZ), Heather Selby (HS), and Olivier Gevaert (OG)

Contributions:
AD: Conceptualization, Methodology, Software, Writing - Original Draft
YLQ: Conceptualization, Methodology, Software, Investigation, Visualization, Writing - Original draft
HZ: Investigation, Writing - Original Draft
HS: Data curation, Writing - Review & Editing.
OG: Supervision, Conceptualization, Writing - Review & Editing, Project Administration.

5.2 Introduction

Cancer is a leading cause of death in the world. Accurate prediction of its survival outcome has been an interesting and challenging problem in cancer research over the past decades. Quantitative methods have been developed to model the relationship between multiple explanatory variables and survival outcome, including fully parametric models (Hosmer et al., 2008; J. P. Klein and Moeschberger, 2006) and semi-parametric models such as the Cox proportional hazards model (Cox, 1972). The Cox model makes a parametric assumption about how the predictors affect the hazard function, but makes no assumption about the baseline hazard function itself (Harrell et al., 1982). In most real world scenarios, the form of the true hazard function is either unknown or too complex to model, making the Cox model the most popular method in survival analysis (Kleinbaum and M. Klein, 2012).

In clinical practice, historically, survival analysis has relied on low-dimensional patient characteristics such as age, sex, and other clinical features in combination with histopathological evaluations such as stage and grade (Louis et al., 2016). With advances in high-throughput sequencing technology, a greater amount of high-dimensional genomic data is now available and more molecular biomarkers can be discovered to determine survival and improve treatment. With the cost of RNA sequencing coming down significantly, from an average of \$100M per genome in 2001 to \$1k per genome in 2015 (S. T. Park and J. Kim, 2016), it is becoming more feasible to use this technology to prognosticate. Such genomic data often has tens of thousands of variables which requires the development of new algorithms that work well with data of high dimensionality.

To address these challenges, several implementations of regularized Cox models have been proposed (Goeman, 2010; M. Y. Park and Hastie, 2007; Wong et al., 2018). A regularized model adds a model complexity penalty to the Cox partial likelihood to reduce the chance of overfitting. More recently, the increasing modeling power of deep learning networks has aided in developing suitable survival analysis platforms for high dimensional feature spaces. For example, autoencoder architectures have been employed to extract features from genomic data for liver cancer prognosis prediction (Chaudhary et al., 2018). The Cox model has also been integrated in a neural network setting to allow greater modeling flexibility (Cheerla and Gevaert, 2019; Ching et al., 2018; Luck et al., 2017; Yousefi et al., 2017).

In studying a specific rare cancer's survival outcome, one interesting problem is whether it is possible to make use of the abundant data that is available for more common relevant cancers and leverage that information to improve the survival prediction. This problem is commonly approached with transfer-learning (Pratt, 1993), where a model which has been trained on a single task (e.g., 1 or more abundant cancers) is used to fine-tune on a related target task (rare cancer). In survival analysis, transfer-learning has shown to significantly improve prediction performance (Y. Li et al., 2016). Deep neural networks used to analyze biomedical imaging data can also take advantage of information transfer from data in other settings. For example, multiple studies show that convolutional neural networks pretrained on ImageNet data can be used to build performant survival models with histology images (Deng et al., 2009; Mobadersany et al., 2018).

In this context, meta-learning is an area in deep learning research that has gained much attention in recent years which addresses the problem of “learning to learn” (Finn et al., 2017; Vilalta and Drissi, 2002). A meta-learning model explicitly learns to adapt to new tasks quickly and efficiently, usually with a limited exposure to the new task environment. Such a framework may potentially adapt better than the traditional transfer-learning setting where there is no explicit adaptation incorporated in the pre-training algorithm. This problem setting with limited exposure to a new task is also known as few-shot learning: learn to generalize well, given very few examples (called *shots*) of a new task (Y. Li et al., 2016). Recent advances in meta-learning have shown that, compared to transfer-learning, it is a more effective approach to few-shot classification (X. Chen et al., 2017; Devos and Grossglauser, 2020), regression (Finn et al., 2017), and reinforcement learning (Duan et al., 2016; Finn et al., 2017). In this study, we propose a meta-learning framework based on neural networks for survival analysis applied in a cancer research setting. Specifically, for the application of predicting survival outcome, we demonstrate that our method is a preferable choice compared to regular transfer-learning pre-training and other competing methods on three cancer datasets when the number of training samples from the specific target cancer is very limited. Finally, we demonstrate that the meta-learning model implicitly prioritizes genes based on their contribution to survival prediction and allows us to uncover biological pathways associated with cancer survival outcome.

5.3 Methods

5.3.1 Datasets

We use the RNA sequencing data from The Cancer Genome Atlas (TCGA) pan-cancer datasets (Tomczak et al., 2015). We remove the genes with NA values and normalized the data by log transformation and z-score transformation. The feature dimension is 17176 genes after preprocessing. The data contains 9707 samples from 33 cancer types. The outcome is the length of survival time in months. 78% of the patients are censored, which means that the subject leaves the study before an event occurs or the study terminates before an event occurs to the subject.

5.3.2 Survival prediction model

To describe the effect of categorical or quantitative variables on survival time, several approaches are commonly considered (Ching et al., 2018). The most popular method is the Cox-PH model (Cox, 1972), which is a semi-parametric proportional hazards model, where the patient hazards depend linearly on the patient features and the relative risks of the patients are expressed in the hazard ratios. Survival trees and random survival forests are an attractive alternative approach to the Cox models (Ishwaran et al., 2008). They are an extension of classification and regression trees and random forests for time-to-event data, and are fully non-parametric and flexible. Artificial Neural networks (ANNs) based models have also been used to predict survival, but the survival time is often converted to a binary variable or discrete variables and the prediction is framed as a classification problem (Petalidis et al., 2008; Chi et al., 2007). To overcome the potential loss of accuracy in the previous methods, ANNs based on proportional hazards are recently developed. It is shown that when applied to high dimensional RNA-seq data, the neural network extension of the Cox model achieves better performance than the Cox-PH (including Ridge and LASSO regularization), random survival trees, and other ANN based models (Ching et al., 2018). It can directly integrate the meta-learning optimization algorithm and is therefore the most suitable choice of model structure in our framework.

5.3.3 Meta-learning

Our proposed survival prediction framework is based on a neural network extension of the Cox regression model that relies on semi-parametric modeling by using a Cox loss (Ching et al., 2018). The model consists of two modules: the feature extraction network and the Cox loss module (Figure 5.1). We use a neural network with two hidden layers to extract features from the RNA sequencing data input, which yields a lower dimensional feature vector for each patient. The features are then fed to the Cox loss module, which performs survival prediction by doing a Cox regression with the features as linear predictors of the hazard (Cox, 1972). The parameters of the Cox loss module β are optimized by minimizing the negative of the partial log-likelihood:

$$\mathcal{L}(\beta) = -\sum_{y_i=uncensored}\left[\mathbf{z}_i\beta - \log\left(\sum_{y_j\geq y_i} e^{\mathbf{z}_j\beta}\right)\right] \tag{5.1}$$

where y_i is the survival length for patient i, $\mathbf{z}_i$ contains the extracted features for patient i, and β is the coefficient weight vector between the features and the output. Since $\mathbf{z}_i$ is the output of the feature extraction module, it can be further represented by:

$$\mathbf{z}_i = f_\varphi\left(\mathbf{x}_i\right) \tag{5.2}$$

where $\mathbf{x}_i$ is the input predictor of patient i, f denotes a nonlinear mapping that the neural network learns to extract features form the predictor, and φ denotes the model parameters including the weights and biases of each neural network layer. The feature extraction module parameters φ and Cox loss module parameters β are jointly trained in the model. For convenience in the following discussion we denote the combined parameters as θ.

The optimization of parameters θ consists of two stages: a meta-learning stage, and a final learning stage. The meta-learning stage is the key process, where the model aims to learn a suitable parameter initialization for the final learning stage, so that during final learning the model can adapt efficiently to previously unseen tasks with a few training samples (Nichol et al., 2018). In order to reach the desired intermediate state, a first order gradient-based meta-learning algorithm is used to train the network during the meta-learning stage (Finn et al., 2017; Nichol et al., 2018).

Specifically, at the beginning of meta-learning training, the model is randomly initialized with parameter θ. Consider that the training samples for the meta-learning stage consist of n tasks T_τ, $\tau = 1, 2 \ldots n$. A task is defined as a common learning theme shared by a subgroup of samples. Concretely, these samples come from a distribution on which we want to carry out a classification task, regression task or reinforcement learning task. The algorithm continues by sampling a task T_τ and using samples of T_τ to update the inner-learner. The inner-learner learns T_τ by taking k steps of stochastic gradient descent (SGD) and updating the parameters to θ_τ^k:

$$\begin{aligned}
&\theta_\tau^0 = \theta \\
&\theta_\tau^1 \leftarrow \theta_\tau^0 - \alpha \mathcal{L}'_{\tau,0}\left(\theta_\tau^0\right) \\
&\theta_\tau^2 \leftarrow \theta_\tau^1 - \alpha \mathcal{L}'_{\tau.1}\left(\theta_\tau^1\right) \\
&\ldots \\
&\theta_\tau^k \leftarrow \theta_\tau^{k-1} - \alpha \mathcal{L}'_{\tau,k-1}\left(\theta_\tau^{k-1}\right)
\end{aligned} \tag{5.3}$$

where θ_τ^k is the model parameter at step k for learning task τ, $\mathcal{L}_{\tau,k-1}$ is the loss computed on the k$^{\text{th}}$ minibatch sampled from task τ, $\mathcal{L}_{\tau,1}$ is the loss computed on the second minibatch sampled from task τ and so on. The 'prime' symbol denotes differentiation, and α is the inner learner step size. Note that this learning process is the separate for all tasks, starting from the same initialization θ.

After an arbitrary m ($< n$) number of tasks are independently learnt by the above k-step SGD process, and obtaining θ_τ^k, τ =1, 2, … m, we make one update across all these m tasks with the meta-learner to get a better initialization θ:

$$\theta \leftarrow \theta + \gamma \frac{1}{m} \sum_{\tau=1}^{m} (\theta_\tau^k - \theta) \tag{5.4}$$

where γ is the learning step of the meta-learner. The term $\frac{1}{m}\sum_{\tau=1}^{m}(\theta_\tau^k - \theta)$ can be considered as a gradient, so that for example a popular optimization algorithm such as Adam (Kingma and J. L. Ba, 2015) can be used by the meta-learner to self-adjust learning rates for each parameter. The entire process of inner-learner update and meta-learner update is repeated until a chosen maximum number of meta-learning epochs is reached. This algorithm is shown to encourage the gradients of different minibatches of a given task to align in the same direction, thereby improving generalization and efficient learning later on (Nichol et al., 2018).

In the final learning stage, the model is provided with a few-sample dataset of a new task. First, the model is initialized with the meta-learnt parameters θ, which are then fine-tuned with the new task training data to $\theta_\tau^{k'}$ and finally the fine-tuned model is evaluated with testing data from the new task. The training procedure of final learning does not require a special algorithm, and can be conducted with regular mini-batch stochastic gradient descent. This final learning stage is equal to the inner-learning loop for a single task in Equation (5.3) without any outer loop.

Algorithm 3 summarizes the complete procedure.

Algorithm 3 Meta-Learning for Few-Shot Survival Prediction

```
Initialize randomly $\theta = \{\phi, \beta\}$, the feature extractor and Cox model parameters, respectively
Let the (inner) survival loss function be defined as in Equation (5.1): $\mathcal{L} = -\sum_{y_i=uncensored}\left[f_\varphi(\mathbf{x}_i)\beta - \log\left(\sum_{y_j \geq y_i} e^{f_\varphi(\mathbf{x}_j)\beta}\right)\right]$
for $i = 0$ to $n$ do
    for m randomly sampled tasks $T_\tau$ do
        Compute $\theta_\tau^k$, denoting $k$ update steps with $\mathcal{L}$, as in Equation (5.3)
    end for
    Update $\theta \leftarrow \theta + \gamma \frac{1}{m}\sum_{\tau=1}^{m}(\theta_\tau^k - \theta)$
    $i \leftarrow i + m$
end for
return $\theta$
```

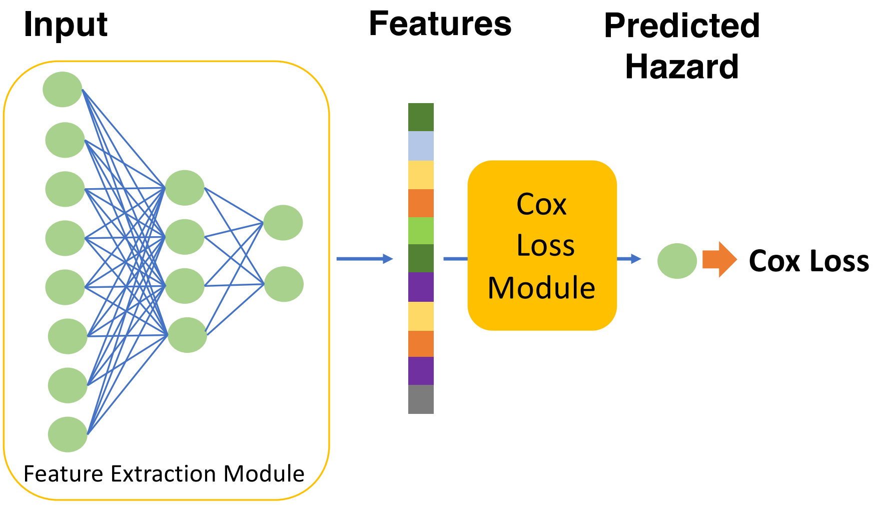


Figure 5.1: Schematic showing the survival prediction model architecture.

5.3.4 Experimental setup

In order to assess the meta-learning method's performance, we compare it with several alternative training schemes based on the same neural network architecture: regular *pre-training*, *combined learning*, and *direct learning*. First, meta-learning initially learns general knowledge from a dataset containing tasks that are relevant but not directly related to the target, and then learns task-specific knowledge from a very small target task dataset. We define the first dataset as the "multi-task training data", and the second as the "target task training data". Secondly, regular pre-training also has a two-stage learning process on the same datasets, but unlike meta-learning without explicitly focusing on learning to reach an initialization that is easy to adapt to new tasks. Thirdly, combined learning does not involve a two-stage learning process, but also leverages knowledge from the relevant tasks by combining the multi-task training data and the target task training data together in one dataset to train a prediction model. Direct learning on the other hand, only uses the target task training samples. To illustrate the effectiveness of the methods with few samples, we consider three cases of direct learning: a large sample size, a medium sample size, and a small sample size which is the same size as the "target task training data" used for the other methods (i.e. regular pre-training, combined learning and meta-learning) (Figure 5.2).

In our experiments, the "multi-task training data" is the pan-cancer RNA sequencing data containing samples from any cancer sites except one cancer site that we define as the target cancer site. The associated target cancer data is considered as the "target

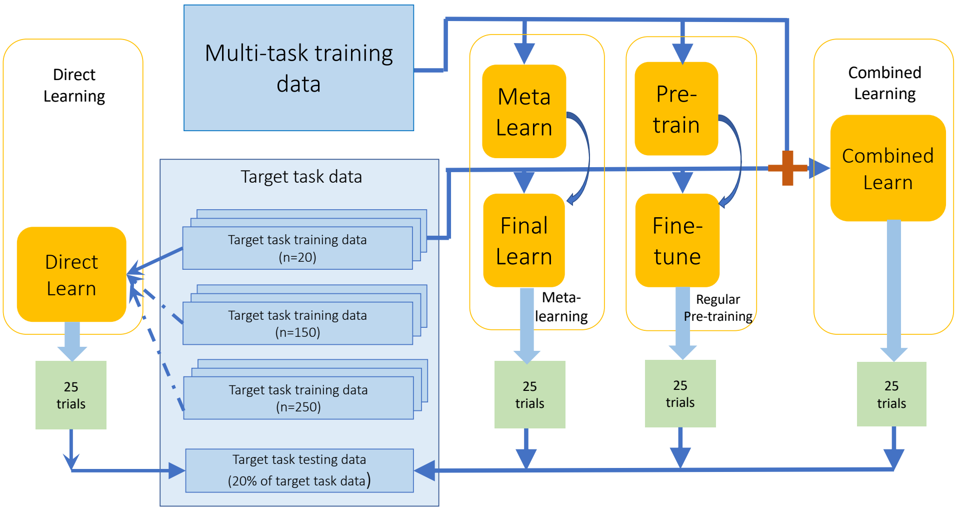


Figure 5.2: Data flow for meta-learning, regular pre-training, and combined learning frameworks

task data". This target task dataset is split into training data and testing data, stratified by disease sub-type and censoring status. For meta-learning, regular pre-training, and combined learning we will not use all of the training set for the target task, as we want to assess the performances when the algorithm is exposed to only a small number of target task training samples. Therefore, we will randomly draw 20 samples from the training dataset as one "target task training data". We choose a small sample size of 20 because it is a possible case in real life situations where the target task is the study of rare diseases (Hee et al., 2017), or where new technologies are used to produce data which only have the capacity to produce a small sample. For direct learning, we randomly draw three different sizes from the training datasets to form training data, 20 for the small size, 150 for the medium size, and 250 for the large size. All methods are evaluated on the common testing data of the target task.

Finally, as a linear baseline, we use the combined learning training sample (multi-task training data and target task training data) to train a linear cox regression model. We conduct 25 experiment trials for each method, where each trial is trained with a randomly drawn "target task training dataset" as described above.

5.3.5 Evaluation

We evaluate the survival prediction model performances with two commonly used evaluation metrics: the concordance index (C-index) (Harrell et al., 1982) and the integrated Brier score (IBS) (Brier, 1950). Fistly, the C-index is a standard performance measure of a model's predictive ability in survival analysis. It is calculated by dividing the number of all pairs of subjects whose predicted survival times are correctly ordered, by the number of pairs of subjects that can possibly be ordered. A pair cannot be ordered if the earlier time in the pair is censored or both events in the pair are censored. A C-index value of 1.0 indicates perfect prediction where all the predicted pairs are correctly ordered, and a value of 0.5 indicates random prediction. Secondly, the integrated Brier score is used to evaluate the error of survival prediction and is represented by the mean squared differences between observed survival status and the predicted survival probability at a given time point. The IBS provides an overall calculation of the model performance at all available times. An IBS value of 0 indicates perfect prediction, while 1 indicates entirely inaccurate prediction.

We select target cancer sites from TCGA with the following two inclusion criteria: (1) a minimum of 450 samples, providing enough training samples for different benchmarking training schemes and (2) a minimum of 30% non-censoring samples, enabling more accurate evaluation than more heavily censored cohorts. This results in the following cancers: glioma, including glioblastoma (GBM) and low-grade-glioma (LGG); non-small cell lung cancer, including lung adenocarcinoma (LUAD) and lung squamous cell carcinoma (LUSC), and head-neck squamous cell carcinoma (HNSC). These three types of cancers are also of clinical interest, because gliomas are the most common type of malignant brain tumor, and lung cancer is the deadliest cancer in the world (Ceccarelli et al., 2016; Herbst and Lippman, 2007). HNSC, on the other hand, is a less widely studied type of cancer, which nonetheless attracted growing attention in the recent decade since the release of the publicly available largest dataset in HNSC by TCGA (Brennan et al., 2017; Tonella et al., 2017).

In addition, to further validate the model in the small sample training setting, we select an additional rare cancer cohort, mesothelioma (MESO), with less than 90 samples in total. Due to the small sample size, we do not compare to the medium or large sample direct learning, but only compare to the small sample direct learning.

Finally, we use a fully independent testing cohort to validate the model. We use a non-small cell lung cancer cohort consisting of 129 patient samples collected from the Stanford University School of Medicine and Palo Alto Veterans Affairs Healthcare System (Bakr et al., 2018). The data is available at National Center for Biotechnology Information Gene Expression Omnibus (NCBI GEO) (Barrett et al., 2009). For the meta-learning, regular pre-training and combined learning methods, we use the same meta-learnt and pre-trained models that are trained with TCGA data for testing the non-small cell lung cancer. The final training and testing is done on the independent dataset. Due to the small sample size, we also only include small sample direct learning for comparison.

For the three large target cancer cohorts, 20% of the target data is used for testing, and we evaluate the C-index and IBS in 25 experimental trials for each method. For the small cancer cohort and independent data cohort, 50% of the data is used for testing, and we conduct 10 trials for each method due to limited training samples for sampling.

5.3.6 Hyper-parameter selection

To avoid overfitting, we do not conduct a separate hyper-parameter search for each of the cancer datasets. Instead, we search for hyper-parameters on one type of cancer and apply the chosen parameters to all experiments. We select the largest cancer cohort, glioma, and use 5-fold cross-validation for hyper-parameter selection. For each given set of hyper-parameters, we average the results from five validation sets (each is 20% of training data). Since there is similarity in the algorithm between methods (combined learning, direct learning, and regular pre-training), we share hyper-parameters between experiments when it makes sense, as detailed below.

All methods use the same neural network architecture with two hidden fully connected layers of size 6000 and 2000, and an output fully connected feature layer of size 200. Each layer uses the ReLU activation function (Nair and G. E. Hinton, 2010). Initially we experiment with 4 different structures: 1 or 2 hidden layers with feature size of 200 or 50, respectively. We chose the optimal structure detailed before and use it as the architecture for all methods in our subsequent discussion.

For the regular pre-training model, we search for hyper-parameters for the pre-training stage and fine-tuning stage separately. For both stages, we test the mini-batch gradient descent and Adam optimizers, and determine learning rates with grid search on a grid of [.1, .05, .01, .005, .001] for SGD and a grid of [.001, .0005, .0001,.00005, .00001] for Adam. We test batch sizes of 50, 100, 200, and 800 for pre-training. The selected parameters for the pre-train stage are: an SGD optimizer with learning rate of .001, L2 regularization scale of 0.1 and batch size of 800. The selected parameters for the fine-tune stage are: an SGD optimizer with learning rate of .001, L2 regularization scale of 0.1, and batch size of 20 which is the size of each target cancer training dataset. For the combined learning model and direct learning model, since the algorithm is very similar to the regular pre-training model's pre-train stage, we use the same parameters selected for the pre-train. The batch sizes for direct learning is half of the size of training data.

For the meta-learning model, we search for hyper-parameters for the meta-learning only. For the final learning stage, we use the same hyper-parameters as in the fine-tune stage of the regular pre-training model, as both methods can use similar algorithms in the last stage of training. From our previous discussion, in the meta-learning stage an SGD

Hyper-parameter	Value
Task-level optimizer	SGD
Task-level learning rate	0.01
Task-level gradient steps	5
Task-level Batch size	100
Meta-level optimizer	ADAM
Meta-level learning rate	0.0001
Meta-level tasks batch size	10
L2 regularization scale	0.1

Table 5.1: Selected hyper-parameters for meta-learning's meta-learning stage

Method	Glioma	Lung cancer	HNSC
Direct (250 samples)	**0.24 ± 0.02**	0.19 ± 0.01	0.20 ± 0.01
Direct (150 samples)	0.25 ± 0.01	0.19 ± 0.01	0.21 ± 0.01
Direct	0.30 ± 0.02	0.24 ± 0.02	0.30 ± 0.02
Combined	0.29 ± 0.02	0.21 ± 0.02	0.26 ± 0.02
Pre-training	0.31 ± 0.02	0.23 ± 0.02	0.26 ± 0.02
Meta-learning	0.28 ± 0.01	**0.16 ± 0.01**	**0.16 ± 0.00**

Table 5.2: Integrated Brier scores (IBS) with 95% confidence intervals (n=25 trials) for target cancer survival prediction with 20 samples, unless specified otherwise. Lower value is better. Best performing method in bold.

optimizer and an Adam optimizer are suitable for the inner learner and meta-learner respectively. For the learning rates, we perform grid search on a grid of [.1, .05, .01, .005, .001] for SGD and a grid of [.001, .0005, .0001,.00005, .00001] for Adam. Batch size is searched from [50, 100, 200, 800], the number of tasks for averaging one meta-learner update is searched from [5, 10, 20], and the number of gradient descent steps for the inner-learner is searched from [3, 5, 10, 20]. The selected parameters for the meta-learning stage are shown in Table 5.1.

Finally, in order to evaluate the effect of fluctuations of the meta-learning hyper-parameters, and ensure that our results reflect the average performance over fluctuations, we conduct a series of tests on the validation data where in each experiment we vary one of the five unique meta-learning hyper-parameters from the chosen value by tuning it up or down by one grid, obtaining ten sets of varied hyper-parameters. We do 5-fold cross-validation for each set of varied hyper-parameters and compute the concordance index from the resulting fifty experiments. We also do fifty random experiments using the selected hyper-parameters and compare the average results of varied versus selected hyper-parameters. We conduct a two-sample t-test on the two results, and conclude that the results obtained by varied parameters do not have a significant difference from the results obtained by the chosen parameters (mean concordance index difference of 0.005 with p value 0.50). Therefore, our results are robust with respect to fluctuations of the hyper-parameters and our conclusions are not based on excessive hyper-parameter tuning.

5.3.7 Interpretation of the genes prioritized by the meta-learning model

We apply *risk score backpropagation* (Yousefi et al., 2017) to the meta-learned models to investigate the feature importance of genes for each of the three target cancer sites. For a given sample, each input feature is assigned a risk score by taking the partial derivatives of the risk with respect to the feature. A positive risk score with high absolute value means the feature is important in poor prediction (high risk), and a negative risk score with high absolute value means the feature is important in good prediction (low risk). The features are ranked by the average of risk score across all samples.

Two approaches were adopted for annotating the genes with ranked risk scores generated by the meta-learning model. Firstly, the top 10% high-risk genes (genes with positive risk scores) and the top 10% low-risk genes (genes with negative risk scores) from each cancer type were subjected to gene set over-representation analysis, by comparing the genes against the gene sets annotated with well-defined biological functions and processes. We model the association between the genes and each gene set using a hypergeometric distribution and Fisher's exact test. Secondly, instead of arbitrary thresholding in the first approach, all the genes, together with their ranked risk scores were incorporated in the gene set enrichment analysis with the fgsea R package (Sergushichev, 2016) which calculates a cumulative gene set enrichment statistic value for each gene set. The gene set databases used in this analysis include Kyoto Encyclopedia of Genes and Genomes (KEGG) (Kanehisa and Goto, 2000), The Reactome Pathway Knowledgebase (Croft et al., 2014) and WikiPathways (Slenter et al., 2018).

5.4 Results

5.4.1 Meta-learning outperforms regular pre-training and combined learning

For all of the large target cancer sites, meta-learning achieves similar or better performance than regular pre-training or combined learning (Figure 5.3; Table 5.2). For the glioma cohort, the mean C-index for meta-learning is 0.86 (0.85-0.86 95% CI), compared to 0.84 (0.83-0.85 95% CI) for regular pre-training and 0.81 (0.81-0.82 95% CI) for combined training. For the lung cancer cohort, the mean C-index is 0.65 (0.65-0.66 95% CI) for meta-learning, 0.60 (0.58-0.61 95% CI) for regular pre-training, and 0.62 (0.62-0.63 95% CI) for combined training. For the HNSC cohort, the result is 0.61 (0.59-0.63 95% CI) for meta-learning, 0.59 (0.57-0.61 95% CI) for regular pre-training and 0.62 (0.61-0.64 95% CI) for combined training. Note that, the variance of the meta-learning results across 25

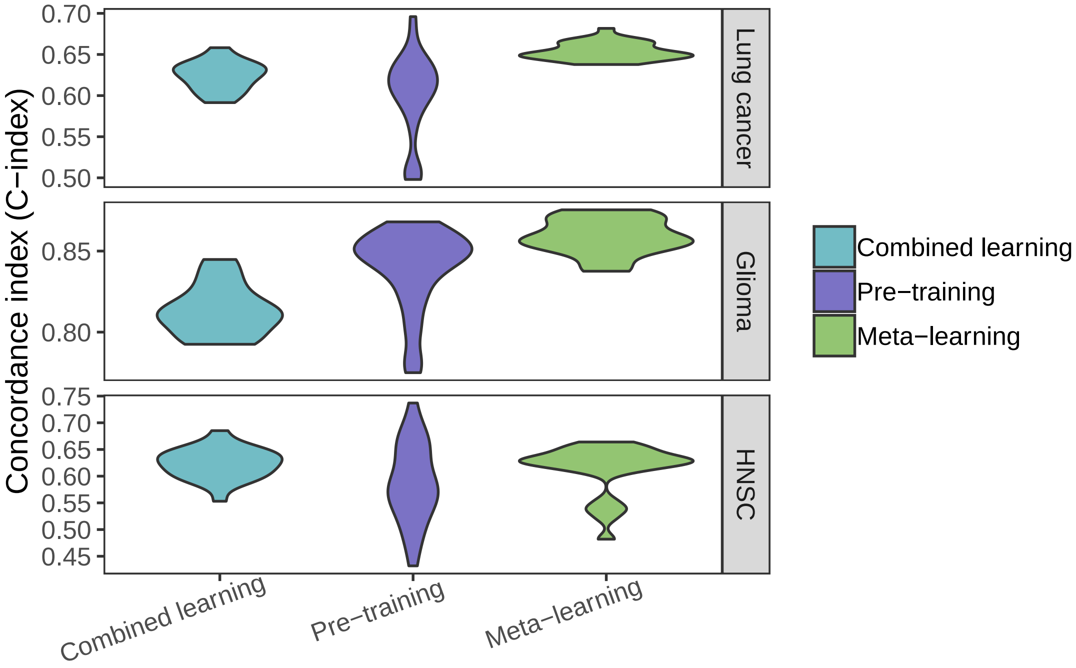


Figure 5.3: C-Index for target cancer survival prediction, comparing combined learning, regular pre-training and meta-learning

random trials also tends to be the smallest, which is most observable for the lung cancer and glioma cohorts. In addition, each of these multi-layer neural networks also shows better performance on average than a linear baseline model. The linear baseline model achieves a C-index of 0.61 for lung cancer (0.60-0.62 95% CI), 0.77 for glioma (0.74-0.80 95% CI), and 0.59 for HNSC (0.58-0.62 95% CI).

5.4.2 Meta-learning achieves competitive predictive performance compared to direct learning

Next, we compare our meta-learning approach with regular direct learning on the target task training samples with different cohort sizes. The performance of direct learning drops significantly when the number of training samples decreases from 250 to 20, which is anticipated because a great amount of information is lost and the model can hardly learn well. However, meta-learning and pre-training can compensate for such lack of information by transferring knowledge from the pan-cancer data explicitly and implicitly, respectively. We show that meta-learning achieves similar or better prediction performance than large-sample direct training in lung cancer and HNSC, and reaches comparable performance with medium-sample direct training in glioma (Figure 5.4; Table 5.2). For the lung cancer cohort, the mean C-index is 0.57 (0.56-0.58 95% CI) for large

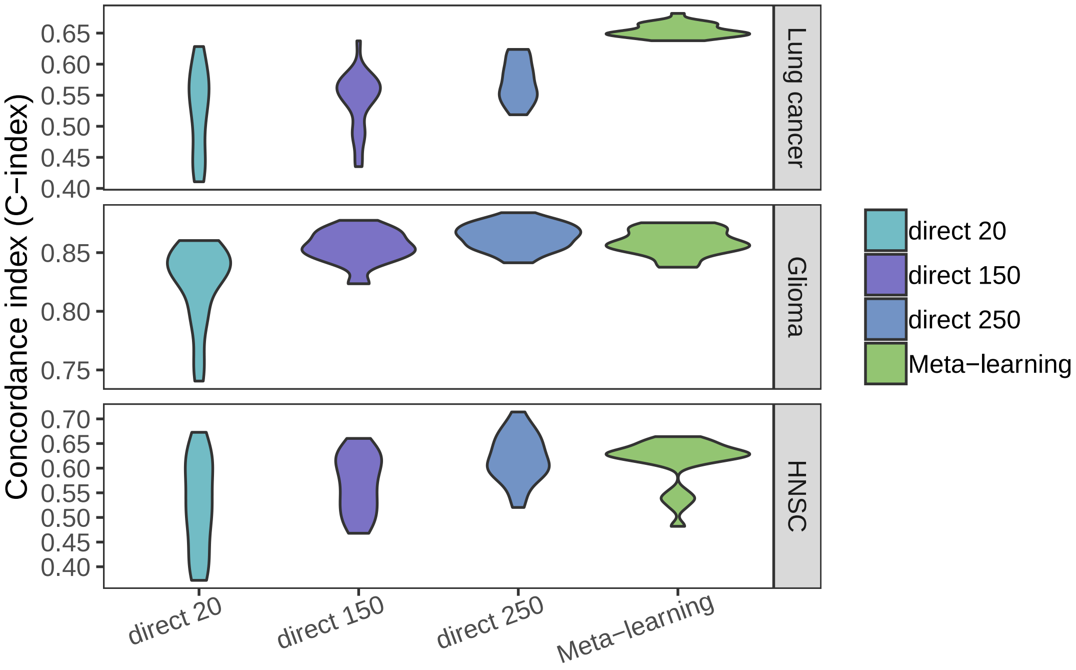


Figure 5.4: C-Index for target cancer survival prediction, comparing direct learning with large (250), medium (150) and small (20) size samples and meta-learning.

sample direct learning, 0.54 (0.52-0.56 95% CI) for medium sample direct learning, 0.53 (0.50-0.55 95% CI) for small sample direct learning, and 0.65 (0.65-0.66 95% CI) for meta-learning. For the glioma cohort, the mean C-indices for large sample, medium sample and small sample direct learning are 0.86 (0.86-0.87 95% CI), 0.85 (0.85-0.86 95% CI), and 0.82 (0.81-0.84 95% CI) respectively, and for meta-learning the mean C-index is 0.86 (0.85-0.86 95% CI). For the HNSC cohort, the mean C-index is 0.62 (0.60-0.64 95% CI) for large sample direct learning, 0.57 (0.54-0.59 95% CI) for medium sample direct learning, 0.53 (0.49-0.56 95% CI) for small sample direct learning, and 0.61 (0.59-0.63 95% CI) for meta-learning. Thus, in all three cancer sites, meta-learning reaches competitive performances as large sample direct learning and can outperform it in certain cases.

5.4.3 Risk score ranked genes are enriched in key cancer pathways

Next, we investigated for each cancer site what genes are most important in the meta-learning model (Figure 5.5, Supplementary Tables 1 - 6). In gliomas, the high-risk genes are associated with viral carcinogenesis (p value 0.002), Herpes simplex infection (p value 0.007), cell cycle (p value 0.03), apoptosis (p value 0.03), DNA damage response (p value 0.04), all of which are also enriched in gene set enrichment analysis with positive enrichment scores (Figure 5.5a). The low-risk genes are associated with HSF1 activation (p value 0.02) which is involved in hypoxia pathway, and tryptophan metabolism (p value 0.04), the latter is also enriched in gene set enrichment analysis with negative enrichment score. Tryptophan catabolism has been increasingly recognized as an important microenvironmental factor in anti-tumor immune responses (Platten et al., 2012) and it is a common therapeutic target in cancer and neurodegeneration diseases (Platten et al., 2019).

In head and neck cancer, the high-risk genes are associated with PTK6 signaling (p value 0.01), which regulates cell cycle and growth, and cytokines and inflammatory response (p value 0.009). The low-risk genes are associated with autophagy (p value 0.02), which is also enriched in gene set enrichment analysis. Other enriched pathways include B cell receptor signaling pathway, cell cycle, and interleukin 1 signaling pathway (Figure 5.5b). Interleukin 1 is an inflammatory cytokine which plays a key role in carcinogenesis and tumor progression (Mantovani et al., 2018).

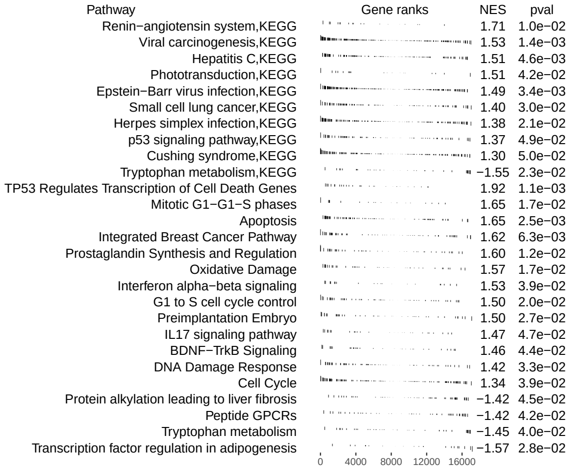
In lung cancer, the top high-risk genes are associated with "non-small cell lung cancer" pathway (p value 0.01), tuberculosis (p value 0.008), Hepatitis B and C virus infection (p value 0.03), and many pathways implicated previously in cancer. These pathways are also enriched in gene set enrichment analysis (Figure 5.5c). Pulmonary tuberculosis has been shown to increase the risk of lung cancer (C.-Y. Wu et al., 2011; Yu et al., 2011). The low-risk genes are associated with energy metabolism (p value 0.03), ferroptosis (p value 0.037) and AMPK signaling pathway (p value 0.046), all related to energy metabolism, particularly lipid metabolism. AMPK signaling pathway activation by an AMPK agonist was shown to suppresses non-small cell lung cancer through inhibition of lipid metabolism (X. Chen et al., 2017). AMPK signaling and energy metabolism are also enriched in gene set enrichment analysis. Other enriched pathways include Notch signaling, interleukin signaling, ErbB signaling, and signaling pathways regulating pluripotency of stem cells.

5.4.4 Validation on the small sample rare cancer cohort

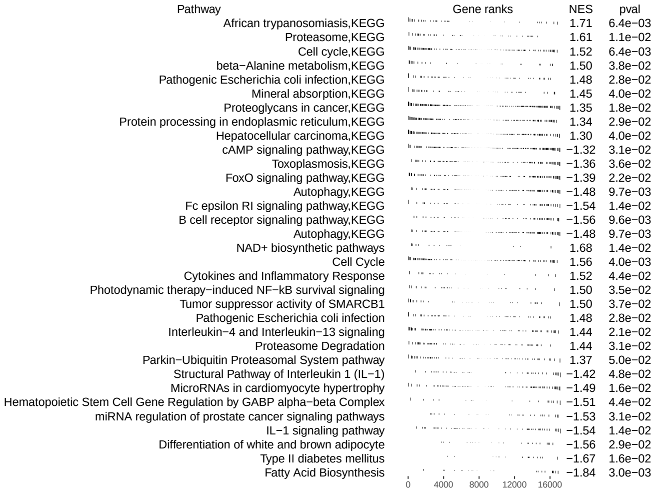
Next we conduct validation on the small sample rare cancer cohort. It is shown that meta-learning achieves similar or better performance than regular pre-training, combined learning, or small sample direct learning (Figure 5.6). The mean C-index for meta-learning is 0.66 (0.63-0.69 95% CI), compared to 0.62 (0.59-0.64 95% CI) for regular pre-training, 0.65 (0.63-0.67 95% CI) for combined training, and 0.60 (0.59-0.62 95% CI) for small sample direct learning.

5.4.5 Validation on the independent lung cancer cohort

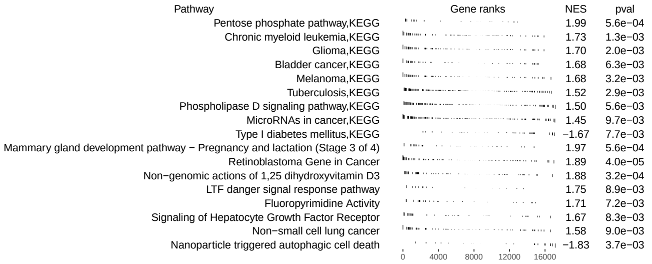
Finally we test on the independent lung cancer cohort. We use the same meta-learnt and pre-trained models that are trained with TCGA data for the lung cancer target



(a) Glioma



(b) Head and neck cancer



(c) Lung cancer

Figure 5.5: Gene set enrichment analysis of the ranked gene list in (a) Glioma (b) Head and neck cancer (c) Lung cancer. The gene set databases used in this analysis included Kyoto Encyclopedia of Genes and Genomes (KEGG) and WikiPathways. Pval, enrichment p-value; NES, normalized enrichment score. In (a) and (b) the enriched pathways with p value below 0.05 were displayed. In (c) the enriched pathways with p value below 0.01 were displayed.

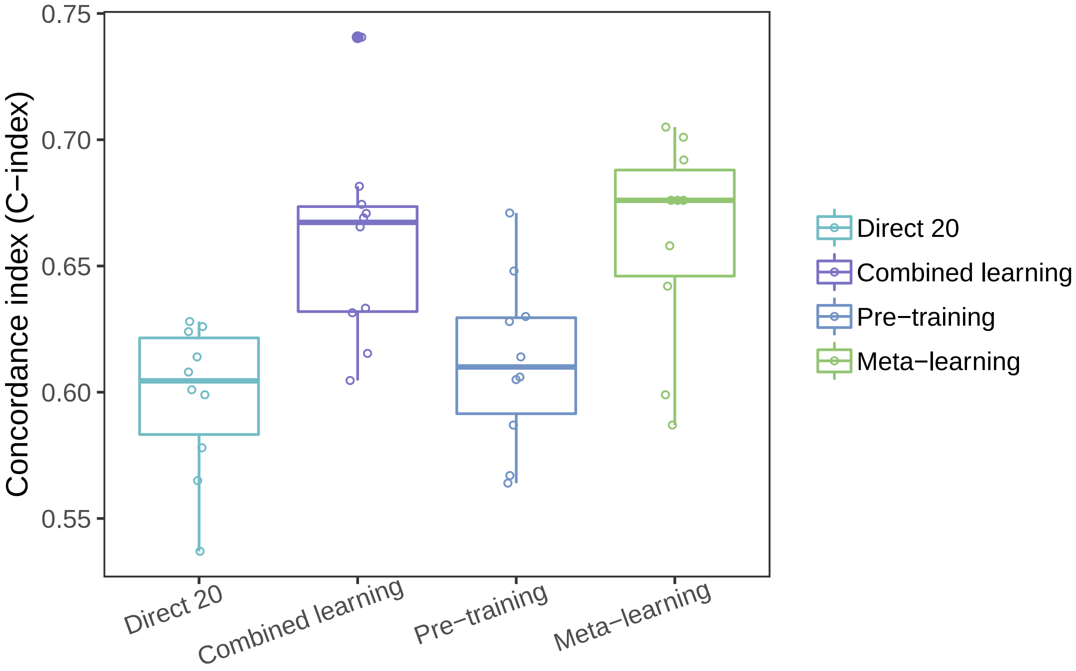


Figure 5.6: C-Index for survival prediction on the mesothelioma cohort, comparing small (20) size sample direct learning, combined learning, regular pre-training and meta-learning.

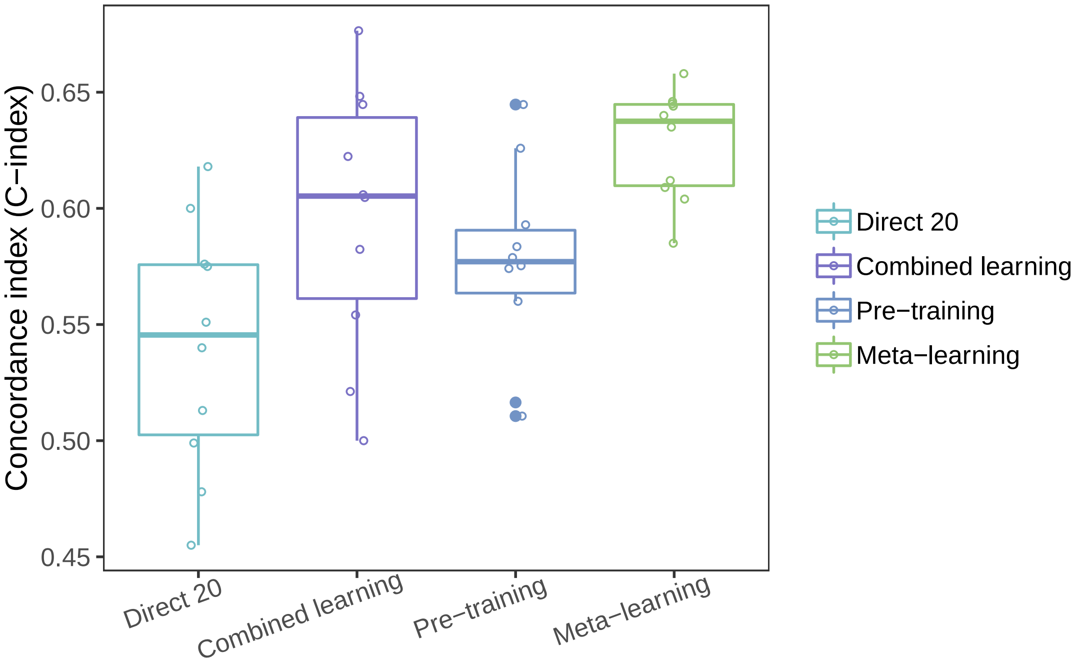


Figure 5.7: C-Index for survival prediction on the independent lung cancer cohort, comparing small (20) size sample direct learning, combined learning, regular pre-training and meta-learning.

site. On this cohort, it is shown that meta-learning has better performance than regular pre-training, combined learning, or small sample direct learning (Figure 5.7). The mean C-index for meta-learning is 0.63 (0.61-0.65 95% CI), compared to 0.58 (0.55-0.61 95% CI) for regular pre-training, 0.59 (0.55-0.64 95% CI) for combined training, and 0.54 (0.50-0.58 95% CI) for small sample direct learning.

5.5 Discussion

Previous studies have shown that when analyzing high-dimensional genomic data, deep learning survival models can achieve comparable or superior performance compared to other methods (e.g., Cox elastic net regression, random survival forests) (D. W. Kim et al., 2019). However, the performance of deep learning is often limited by the relatively small amount of available data (Yousefi et al., 2017). To address this issue, our work investigates different deep learning paradigms to improve the performance of deep survival models, especially in the setting of small size training data.

In previous studies the most common way to build deep survival models is to train neural networks with a large number of target task training samples from scratch, a process we call direct training. Direct training with a large sample size (e.g. $n = 250$) can thus be considered as a baseline. As expected, the performance of direct training drops when the number of training samples decreases (e.g. from $n = 150$ to $n = 20$). On the other hand, combined learning, regular pre-training, and meta-learning all leverage additional data from other sources, thereby enabling them to achieve better performances when the training sample size is small. We use a small number of target cancer site training samples (e.g. $n = 20$) with these methods and investigate their performance.

When only small (task) sample sizes are available for meta-training, a Bayesian approach to meta-learning is an option (Finn et al., 2018; Yoon et al., 2018). Although Bayesian meta-learning with few(er) training tasks has shown less meta-overfitting (on training tasks) and performance improvements over regular meta-learning in low-dimensional settings, in high-dimensional settings the improvement is marginal (Yoon et al., 2018).

It is important to note that combined learning, regular pre-training and meta-learning are exposed to exactly the same information, but differ only in their algorithms. Combined learning is a one-stage learning process, whereas pre-training and meta-learning are two-stage learning methods. Meta-learning shows better predictive performance than combined or regular pre-training, indicating that it is able to adapt to a new task more effectively due to the improved optimization algorithm targeting the few-sample training environment.

It has been shown that methods which use only target task data (direct learning with different size samples) and methods which use additional information (combined, pre-training and meta-learning) perform differently, and one type of approach may be better than the other on different cancer sites. For example, on glioma, direct learning tends to do better overall; whereas on lung cancer, the other methods outperform direct learning. This may be due to that fact that the amount of information that can be learnt from related data versus from the target data is different for each cancer site. If there is significant information within the target cancer samples alone, then direct training will be more effective than learning from other cancer samples. On all three cancer sites we observe that meta-learning achieves similar or better performance than medium-size direct training, and outperforms large-size direct training in some cases. However, the advantage of meta-learning may not generalize to every cancer site. Certain cancers may have very unique characteristics so that transfer of information from other cancers may not help in prediction regardless of improved adaptivity. For the three cancer sites, the affinity of each target cancer to other types of cancers in the pan-cancer data aids the performance of meta-learning, which efficiently transfers the information from other cancers to the target cancers. On the other hand, some cancers are more dissimilar from other cancer sites which makes information transfer difficult. For example, for another cancer site, kidney cancer, specifically kidney renal clear cell carcinoma (KIRC) and kidney renal papillary cell carcinoma (KIRP), both meta-learning and pre-training do not produce good survival prediction. This can be visualized in Supplementary Fig.

1, comparing the affinity between different target cancers with the rest of the cancers on a t-distributed stochastic neighbor embedding (t-SNE) graph. Therefore, in order for meta-learning to achieve good performance, the related tasks training data need to contain a reasonable amount of transferable information to the target task.

The performance of meta-learning can be explained by the learned learning algorithm at the meta-learning stage where the model learns from related tasks. We further investigate how to optimize meta-learning performance. We examine results from two sampling approaches when forming one task, where we either draw samples only from one cancer, or draw samples from multiple types of cancer. It is a more natural choice to consider each cancer type as a separate task, but we found that the latter leads to improved performance. To explain this improvement, we examine the gradient of the meta-learning loss function. It can be shown that the gradient of the loss function contains a term that encourages the gradients from different minibatches for a given task to align in the same direction (Appendix A). If the two minibatches contain samples from the same type of cancer, their gradient might already be very similar and thus this higher order term would not have a large effect. On the other hand, if the second minibatch contains samples from a different type of cancer than the first, the algorithm will learn something that is common to both of them and thereby help to improve generalization.

From a molecular point of view, certain cancer types are related to each other and there is an inherent presence of inter-dependency. For example, colon and rectal cancers are found to have considerably similar patterns of genomic alteration (Network et al., 2012). In our work, the multi-task training data contains many cancer sites including cancer sites that may have inter-dependencies. We did not focus on modeling the inter-dependencies within the multi-task training data, but on transferring information from the multi-task data to the new tasks. However, handling cross-task relations in meta-learning is an interesting topic that could potentially improve generalization further. Recent work has proposed methods to accommodate the relations between tasks (Yao et al., 2020). This would be worth investigating in future work.

The gene set enrichment analysis results validate our model for prioritizing the genes for survival predication. In the three cancer types investigated, the resulting gene lists are enriched in key pathways in cancer including cell cycle regulation, DNA damage response, cell death, interleukin signaling, and NOTCH signaling pathway, etc.

Apart from the well-recognized cancer pathways, our results also reveal potential players affecting cancer development and prognosis, that are not well-studied yet. Viruses have been linked to the carcinogenesis of several cancers, including human papilloma virus in cervical cancer, hepatitis B and C viruses in liver cancer, and Epstein-Barr virus in several lymphomas and nasopharyngeal carcinoma (Martin and Gutkind, 2008). Our results further suggest that viruses might also play a role in glioma and lung cancer, where the high-risk genes are enriched in several viral carcinogenesis pathways. In gliomas, the enriched pathways that are unfavorable for survival include Epstein-Barr virus and herpes simplex infection. In lung cancer, both hepatitis B and C virus infection pathways are enriched. This suggests that that hepatotropic viruses may affect the respiratory system, including the association with lung cancer. For example, hepatitis B virus infection has been associated with poor prognosis in patients with advanced non-small cell lung cancer (Peng et al., 2015). The role of Epstein-Barr virus in gliomagenesis have also been studied but the results remain inconclusive (Akhtar et al., 2018). Whether these viruses do play a role in carcinogenesis and further affect cancer prognosis, or the association we observed reflects an abnormal immune system that is unfavorable for the survival of cancer patients remains to be investigated.

While it would be interesting to decipher the mechanisms of viral infections in tumorigenesis in these cancers, it is difficult to establish a direct link using genomic analysis from currently available data. More well-designed experiments are needed. That being said, the field of viral infections in tumorigenesis is under active research now and there are several interesting studies that suggest a wider role of viral infections in cancer. The recent study of viral landscape in cancer by Pan-Cancer Analysis of Whole Genomes (PCAWG) Consortium (Zapatka et al., 2020) examined whole genome and whole transcriptome sequencing data from 2,658 cancers across 38 tumor types. Apart from the well-known viral etiology in cancer (HPV in cervical cancer and head and neck cancer, HBV in liver cancer, and EBV in gastric cancer), the study also found frequent appearance of herpesviruses (EBV and HHV-6B) in brain cancers. A 2018 study in Neuron (Rizzo, 2020) found frequent presence of herpesviruses in the brain tissues. Although this study is designed to study Alzheimer's disease, the fact that herpesviruses are frequently found in brain tissues warrants further research of the role of herpesviruses in not only neurodegenerative diseases, but also cancer. A 2018 study in Cancer Research (Varn et al., 2018) found virus infection shapes the tumor immune microenvironment and genetic architecture of 6 virus-associated tumor types. They found that EBV infection was associated with decreased receptor diversity in multiple cancers. The altered immune profile in the tumor microenvironment may affect tumor progression and patient survival, but more study is needed to confirm it.

As for the enriched pathways that are favorable for cancer survival, we identified pathways related to metabolism, in particular, lipid metabolism, in all the three cancer types investigated. In glioma, the top enriched pathway favorable for cancer survival is adipogenesis regulation. In head and neck cancer, differentiation of adopocyte and fatty acid biosynthesis are top enriched favorable pathways. In lung cancer, ferroptosis and AMPK signaling pathway are both related to energy metabolism. Ferroptosis is a process driven by accumulated iron-dependent lipid ROS that leads to cell death. Small molecules-induced ferroptosis has a strong inhibition of tumor growth and enhances the sensitivity of chemotherapeutic drugs, especially in drug resistance (Lu et al., 2018). AMPK plays a central role in the control of cell growth, proliferation and autophagy through the regulation of mTOR activity and lipid metabolism (X. Chen et al., 2017;

Han et al., 2013). The link between cancer and metabolism is worth investigating in future studies.

To conclude, in survival analyses one problem that researchers have encountered is the insufficient amount of training samples for machine learning algorithms to achieve good performances. We address this problem by adapting a meta-learning approach which learns effectively with only a small number of target task training samples. We show that the meta-learning framework is able to achieve similar performance as learning from a significantly larger number of samples by using an efficient knowledge transfer. Moreover, in the context of limited training sample exposure, we demonstrate that this framework achieves superior predictive performance over both regular pre-training and combined learning methods on two types of target cancer sites. Finally, we show that meta-learning models are interpretable and can be used to investigate biological phenomena associated with cancer survival outcome.

The problem of small data size may be a limiting factor in many biomedical analyses, especially when studies are conducted with data that is expensive to produce, or in the case of multi-modal data (Cheerla and Gevaert, 2019). Our work shows the promise of meta-learning for biomedical applications to alleviate the problem of limited data. In future work, we intend to extend this approach to analysis with medical imaging data, such as histopathology data and radiology data, for building predictive models on multi-modal data with limited sets of patients.

5.6 Data Availability

All data used in this manuscript are publicly available. The TCGA Gene expression data is version 2 of the adjusted pan-cancer gene expression data obtained from Synapse: https://www.synapse.org/#!Synapse:syn4976369.2. The independent lung cancer data can be obtained from: https://wiki.cancerimagingarchive.net/display/Public/NSCLC+Radiogenomics. The databases used in gene set enrichment analysis are publicly available: Kyoto Encyclopedia of Genes and Genomes (KEGG) at https://www.genome.jp/kegg/; the Reactome Pathway Knowledgebase at https://reactome.org/download-data; and WikiPathways at https://www.wikipathways.org/index.php/Download_Pathways. The remaining data are available within the Article, Supplementary Information or available from the authors upon request.

6 Model-Agnostic Learning to Meta-Learn

6.1 Preface

Summary: We propose a learning algorithm that enables a model to quickly exploit commonalities among related tasks from an unseen task distribution, before quickly adapting to specific tasks from that same distribution. We investigate how learning with different task distributions can first improve adaptability by meta-finetuning on related tasks before improving goal task generalization with finetuning. Synthetic regression experiments validate the intuition that learning to meta-learn improves adaptability and consecutively generalization. Experiments on more complex image classification, continual regression, and reinforcement learning tasks demonstrate that learning to meta-learn generally improves task-specific adaptation. The methodology, setup, and hypotheses in this proposal were positively evaluated by peer review before conclusive experiments were carried out.

This chapter is an edited version of Devos and Dandi (2021).

Co-author: Yatin Dandi (YD)

Contributions:
AD: Conceptualization, Methodology, Software, Visualization, Investigation, Writing - Original Draft
YD: Conceptualization, Methodology, Software, Visualization, Investigation, Writing - Original Draft

6.2 Introduction

Recent years have seen encouraging developments in meta-learning-based approaches for deep neural networks and their successful application to various domains (Finn et al., 2017; Rajeswaran et al., 2019; Nichol et al., 2018). These approaches typically assume a distribution over tasks and aim to exploit the shared properties across tasks to learn a model that can adapt to unknown tasks from this distribution using only a few training data points. Unfortunately, their adaptive capabilities do not generalize well to unseen tasks from related but different task distributions (W.-Y. Chen et al., 2019).

A number of recent works have proposed addressing the presence of different sets of related tasks by explicitly factoring in the heterogeneous nature of the task distribution in the design of the architecture and update rule (Requeima et al., 2019; Yao et al., 2020; Yao et al., 2019; Vuorio et al., 2019). However, these approaches still assume a fixed task distribution, such as tasks sampled from a fixed set of families of functions or a multi-modal distribution arising out of a fixed set of task datasets. We argue that generalizing across unknown datasets and task distributions is a fundamentally more difficult problem than fixed distribution meta-learning. With a new task distribution or dataset, it is unrealistic to expect the model to quickly adapt to any arbitrary task from such a distribution. Instead, we expect the model to quickly learn to adapt to any task from the new task distribution after being exposed to only a few tasks of it. This generalizes the notion of few-shot learning to *few-task* (few-shot) learning. Changes in task distributions might also arise due to natural or artificial transformations of the data. With different task distributions arising from a "distribution over task distributions", it is not only desirable to "quickly adapt" to unseen tasks but also "quickly learn to adapt" to unseen task distributions.

We propose a general framework for adapting to unseen task distributions by "learning to meta-learn" on different task distributions during training. Thus, the heterogeneity of tasks in our approach is not fixed but flexibly modeled through hierarchical sampling from a distribution over task distributions. Similar to MAML (Finn et al., 2017), we propose a general framework to learn a suitable initialization for a single set of parameters. Unlike MAML, which only trains a model to quickly adapt the parameters on a new task using few task-specific gradient steps, our model is also trained to quickly adapt its initialization to a new task distribution using few meta gradient steps on this unseen task distribution (see Figure 6.1). We hypothesize that our approach would allow models to transfer learn capabilities across datasets in supervised and unsupervised learning, and new environments in reinforcement learning as well as quickly adapt to unseen augmentations and distortions at test time.

6.3 Related Work

Model-Agnostic Meta-Learning (MAML) by Finn et al. (2017) is a seminal work in few-shot meta-learning which seeks a common model initialization that allows the model to perform well on any goal task from the training task distribution with few gradient steps (and samples). Multimodal MAML (MMAML) by Vuorio et al. (2019) extends MAML with the capability to identify tasks sampled from a multimodal task distribution and adapt quickly through gradient updates. Yao et al. (2019) proposed the hierarchically structured meta-learning (HSML) algorithm that explicitly tailors the transferable knowledge to different clusters of tasks. Automated Relational Meta-Learning (ARML) by Yao et al. (2020) extracts the cross-task relations and constructs a meta-knowledge graph. When a new task arrives, it can quickly find the most relevant structure and tailor the learned structure knowledge to the meta-learner. Still, MMAML, HSML, and ARML only learn to learn from a *fixed* task distribution. Unlike our approach, they are not expected to generalize to new task distributions.

Research on improving meta-learning algorithms is vast, and we will highlight work most related to our approach. Following MAML, Z. Li et al. (2017) and Antoniou et al. (2019) learn the inner learning rate in the outer loop to improve performance, while reducing the hyperparameter tuning requirement. Finn et al. (2017) proposed a first-order approximation of MAML (fo-MAML) to scale to larger models, which was subsequently improved upon by Nichol et al. (2018) with a first-order method called Reptile. Reptile can naturally be extended to our proposed approach, making it scalable and efficient. Yinbo Chen et al. (2020) found that with increasingly deep architectures, common pre-training and transfer learning can outperform meta-learning from scratch in the visual classification domain. Based on this, they proposed to combine regular pre-training with subsequent meta-learning, which empirically gives a further performance improvement. Raghu et al. (2020) found that feature reuse is a dominant factor in MAML, and proposed a variant called Almost no Inner Loop (ANIL) which learns to only fine-tune the last layer linear classifier. On the contrary, Oh et al. (2021) came to the opposite conclusion that fast learning is crucial, and proposed a Body Only update in Inner Loop (BOIL) algorithm. These works motivate several of our research questions for the experiments.

6.4 Methodology

We aim to train models that can quickly change their adaptability before rapid adaptation. We formalize this setting as few-task few-shot learning. In this section, we will clarify the problem setup and we will formalize this learning to meta-learn problem setting for supervised learning, but it can easily be generalized to unsupervised and reinforcement learning (RL).

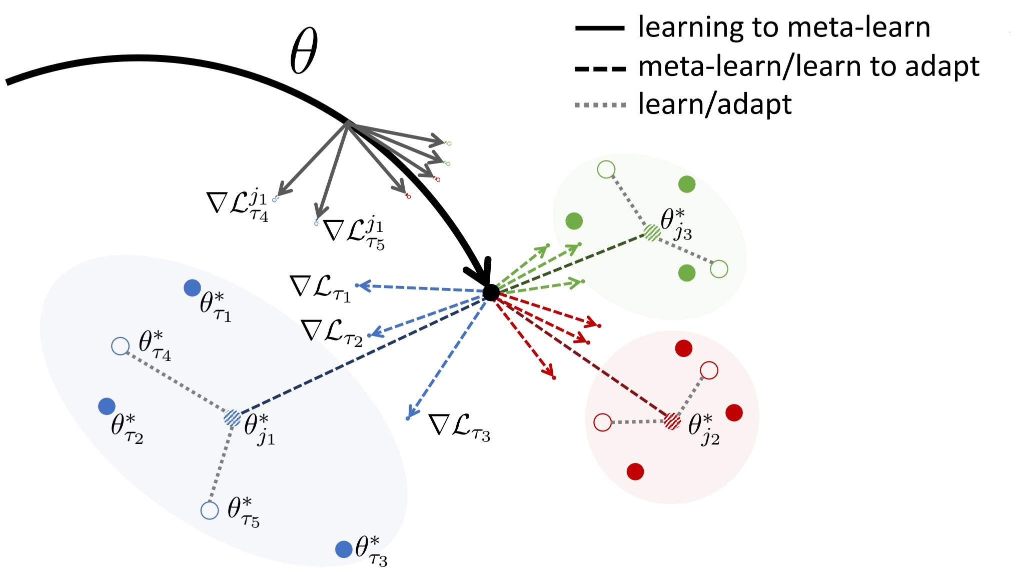


Figure 6.1: Illustration of our model-agnostic learning to meta-learn algorithm (MALTML), which optimizes for a representation θ that can quickly adapt to new task distributions j and consecutively to their tasks τ. Illustrated with single gradient steps for the meta-learning and learn/adapt phases.

6.4.1 Learning to Meta-Learn Problem Setup

In our learning to meta-learn scenario, we consider a distribution $p(j)$ over task distributions such as families of related functions or datasets with similarities. Our goal is to allow the model to adapt to unseen task distributions as well as specific tasks within such distributions. In the L-task K-shot setting, the model is trained to meta-learn a new task distribution j_d from only L tasks with only K examples each for task-learning and Q examples for meta-learning, before learning a single goal task $\mathcal{T}_i$ drawn from $p_{j_d}(\mathcal{T})$ from only K samples. The model f is then improved by considering how the test error on new (validation) data from $\mathcal{T}_i$ changes with respect to the original parameters. This test error on the final goal tasks serves as the training error of the learning to meta-learn process. At the end of training, new families are sampled from $p(j_d)$ and the model's learning to meta-learn performance is measured by the model's performance after meta-finetuning on L tasks with $K + Q$ examples and finetuning on one or multiple goal tasks from the same family with K examples.

6.4.2 A Model-Agnostic Learning to Meta-Learn Algorithm

We propose a method that can learn the parameters of any model via learning to meta-learn in such a way as to prepare that model to first quickly change its adaptation capability (initialization) and consecutively adapt quickly to a goal task. The intuition behind this approach is that some internal representations are more transferable to meta-learn with. For example, a neural network could learn features that are broadly applicable to all tasks in the distribution over task distributions $p(j)$ and can then specialize to an individual task distribution $p_{j_d}(\mathcal{T})$, rather than to a single task distribution or task. We assume no specific form of the model, other than that it is parametrized by some parameter set θ, and that the loss functions are sufficiently smooth in θ such that we can employ gradient-based learning techniques.

Formally, we consider a model represented by a parametrized function f_θ with parameters θ. When adjusting the adaptability to a new task *family* j_d, the model's parameters θ become θ'_{j_d}, and consecutively when adapting (finetuning) to a new task $\mathcal{T}_c \sim p_{j_d}(\mathcal{T})$ the model's parameters θ'_{j_d} become θ'_c. In our approach, the updated parameter vector θ'_{j_d} is obtained by few (inner) *meta*-learning steps on few tasks from dataset j_d, also called *meta-finetuning*. Each meta-finetuning step is taken across few task-finetuning steps. A single ($r = 1$) *task* gradient step with step size α is:

$$\theta'_{\mathcal{T}_i} = U^{r=1,\alpha}_{\mathcal{T}_i}(\theta) = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta). \tag{6.1}$$

Then, the family-specific meta-finetuning update, with one ($r = 1$) task-level parameter update as in Equation (6.1) and one ($m = 1$) meta-level update across tasks from j_d with step size β, is:

$$\theta'_{j_d} = V^{m=1,\beta}_{j_d}(\theta) = \theta - \beta \nabla_\theta \left(\sum_{\mathcal{T}_i \sim p_{j_d}(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{U^{r=1,\alpha}_{\mathcal{T}_i}(\theta)}) \right). \tag{6.2}$$

Consecutively, the updated task-specific parameter vector θ'_c is obtained by taking few gradient descent steps, with learning rate γ, on tasks $\mathcal{T}_c \sim p_{j_d}(\mathcal{T})$, starting from θ'_{j_d}. For example, using Equation (6.1), with 1 gradient step: $\theta'_c = U^{r=1,\gamma}_{\mathcal{T}_c}(\theta'_{j_d})$. Note that using the same r for meta-finetuning and goal task finetuning is a natural choice, but can be deviated from. The step-sizes α, β, and γ may be fixed as hyperparameters or learned on the outer learning loop (see below).

Finally, the global model parameters θ are optimized to perform well after this two-step (meta-learn, then learn) process. Specifically, the model parameters are trained by optimizing the performance of every $f_{\theta'_c}$ on its task $\mathcal{T}_c$. This is done across task distributions sampled from $p(j)$ and tasks sampled from them ($\mathcal{T}_c \sim p_{j_d}(\mathcal{T})$). Concretely, the *learning-to-meta-learn* objective is:

$$\min_{\theta} \sum_{j_d \sim p(j)} \sum_{\mathcal{T}_c \sim p_{j_d}(\mathcal{T})} \mathcal{L}_{\mathcal{T}_c}(f_{\theta'_c}) = \min_{\theta} \sum_{j_d \sim p(j)} \sum_{\mathcal{T}_c \sim p_{j_d}(\mathcal{T})} \mathcal{L}_{\mathcal{T}_c}\left(U^{r,\gamma}_{\mathcal{T}_c}(V^{m,\beta}_{j_d}(\theta))\right)$$

Note that the learning to meta-learn optimization is performed over the model parameters θ, whereas the learning to meta-learn objective itself is computed using the updated model parameters θ'_c.

The outer optimization across task distributions is performed via stochastic gradient descent (SGD), such that the model parameters θ are updated as follows:

$$\theta \leftarrow \theta - \eta \nabla_{\theta} \sum_{j_d \sim p(j)} \sum_{\mathcal{T}_c \sim p_{j_d}(\mathcal{T})} \mathcal{L}_{\mathcal{T}_c}(f_{\theta'_c}) \tag{6.3}$$

where η is the outer step size. The full algorithm, in the general case, is outlined in Algorithm 4.

The MALTML outer gradient update yields a third-order gradient with respect to θ. To make MALTML computationally usable for high-dimensional models, we propose a first-order approximation. Concretely, following Nichol et al. (2018), for every family j_d we employ multiple *first-order* meta-learning updates $\theta'_{j_d} = \widetilde{V}^{m>1}_{j_d}(\theta)$, before updating the model parameters with $\theta \leftarrow \theta + \eta \sum_{j_d}(\theta'_{j_d} - \theta)$. Note that in this case, due to the nature of our two-level first-order approximation, goal task finetuning is not required anymore.

6.5 Experimental Protocol

The goal of our experimentals is to get conclusive results in different learning domains on whether MALTML can enable a quick and significant change of adaptability to goal tasks from new task distributions. Moreover, we wish to examine whether fast learning (to meta-learn) (Oh et al., 2021) or feature reuse (Raghu et al., 2020) is the dominant factor in the performance.

All the learning to meta-learn problems we consider require some form of change in adaptability and subsequent adaptation to new tasks at test time. When possible, we will compare our results to an oracle that receives the identities of the family and goal task as an additional input or a MAML oracle that is able to meta-finetune on a large number of tasks from the new task distribution, as upper bounds on the performance of the models.

Algorithm 4 Model-Agnostic Learning to Meta-Learn

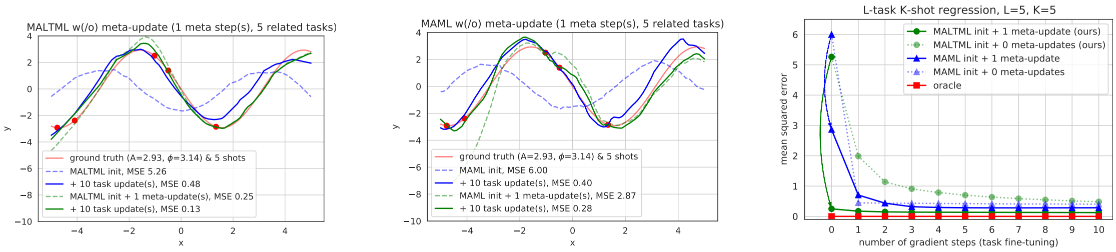
```
Require: $p(j)$: distribution over task distributions parameter $j$
Require: $\alpha$, $\beta$, $\gamma$, $\eta$: step size hyperparameters
randomly initialize $\theta$
while not done do
    Sample batch of task distributions $j_d \sim p(j)$
    for all $j_d$ do
        Sample $L$ tasks $\mathcal{T}_i \sim p_{j_d}(\mathcal{T})$
        for all $\mathcal{T}_i$ do
            Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to $K$ examples
            Compute adapted parameters with gradient descent: $\theta'_i = \theta - \alpha\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
        end for
        Update $\theta'_{j_d} \leftarrow \theta - \beta\nabla_\theta \sum_{\mathcal{T}_i} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i})$ using $Q$ examples per task
        Sample batch of tasks $\mathcal{T}_c \sim p_{j_d}(\mathcal{T})$
        for all $\mathcal{T}_c$ do
            Evaluate $\nabla_{\theta'_{j_d}} \mathcal{L}_{\mathcal{T}_c}(f_{\theta'_{j_d}})$ with respect to $K$ examples
            Compute adapted parameters with gradient descent: $\theta'_c = \theta'_{j_d} - \gamma\nabla_{\theta'_{j_d}} \mathcal{L}_{\mathcal{T}_c}(f_{\theta'_{j_d}})$
        end for
    end for
    Update $\theta \leftarrow \theta - \eta\nabla_\theta \sum_{j_d \sim p(j)} \sum_{\mathcal{T}_c \sim p_{j_d}(\mathcal{T})} \mathcal{L}_{\mathcal{T}_c}(f_{\theta'_c})$
end while
```

Regarding model architecture and optimization, we will follow Finn et al. (2017). We will use insights from Antoniou et al. (2019) to stabilize training where applicable and follow its hierarchical hyperparameter search methodology. We will carry out the experiments in PyTorch (Paszke et al., 2019), using the torchmeta package (Deleu et al., 2019).

6.5.1 Illustrative preliminary experiment: regression

We start with a toy regression problem which illustrates the experimental protocol of few-task few-shot learning and the basic principles of MALTML.

Each task distribution (family) consists of sinusoid regression tasks with a specific phase. Thus, $p(j)$ is continuous, where the phase j varies uniformly within $[0, \pi]$. Each task involves regressing from the input to the output of a sine wave, where the amplitude is varied between tasks. Thus, $p_{j_d}(\mathcal{T})$ is continuous, where the amplitude varies within $[0.1, 5.0]$ and the input and output both have a dimensionality of 1. During training and testing, datapoints x are sampled uniformily from $[-5, 5]$. The loss is the mean squared error between the prediction $f(x)$ and the true value. The model is a neural network with 2 hidden layers of size 40 with ReLU nonlinearities. When training with MALTML, we use single gradient updates with fixed step sizes of $\alpha = 0.001$, $\beta = 0.01$, $\gamma = 0.001$, and use Adam (Kingma and J. Ba, 2015) with an initial learning rate of $\eta = 0.001$. The baselines are also trained with Adam, and an inner learning rate of $\alpha = 0.001$ for MAML. For the 5-task 5-shot regression experiment, we train for 70,000 outer steps with a family batch size of 10, $Q = 5$, and 2 validation tasks per family.



(a) MALTML adaptation (b) MAML adaptation (c) test error

Figure 6.2: Qualitative and quantitative effects of the meta-update and task fine-tuning. For (c), the meta-update improvement is indicated by the arrows at 0 steps.

For this preliminary experiment we only contrast with a MAML baseline, which disregards the family structure, and an oracle receiving the true amplitude and phase of the goal task as additional input. In general, we intend to compare to the other oracles described before and another baseline: pretraining on all tasks, which in this case involves training a model to regress random sinusoid functions.

The toy results in Figures 6.2a and 6.2c show that the learned MALTML model is able to quickly change its adaptability to the new family's phase on 5 related 5-shot tasks. Due to this, it reaches a better fit than MAML, which benefits less from the meta-adaptation of its initialization (Figures 6.2b and 6.2c).

6.5.2 Specifics of main experiments

Besides the toy experiment in section 6.5.1, we propose to test the effectiveness of MALTML for:

Classification. We propose to apply our method to modified versions of the Omniglot (B. M. Lake et al., 2015) and ImageNet (Russakovsky et al., 2015b) datasets. For ImageNet, we will propose a few-task few-shot dataset, making use of its hierarchical structure to generate a sufficient amount of training families. Given the finite number of alphabets in Omniglot, which will serve as families, we will use data augmentations similar to the ones used in Khodadadeh et al. (2019) to generate a large number of training families. To arrive at a realistic setting where the (imposed) hierarchical structure from the supervised case is lacking, we will use a hierarchically augmented version of unsupervised meta-learning (Khodadadeh et al., 2019). Specifically, a subset of data augmentation parameters will be sampled per family, before applying them randomly (with the remaining subset) on samples to generate tasks from each family. Note that in this case, on a (family) meta-level the method needs to be sensitive to augmentations, whereas on a task-level it should aim to be invariant to them.

2D Navigation. We propose to evaluate MALTML on a set of families of RL tasks where a point agent must move to different goal positions in 2D, while being given related tasks from the same family. Every family constitutes of a random crop of the unit square, and every task is randomly chosen from within that rectangle. The crops are bounded by 25% to 75% of the original unit length.

Continual Regression We propose to evaluate a continual learning extension of MALTML on incremental sine wave learning as described in Javed and White (2019). Different families of continual learning prediction problems will correspond to different frequencies of the sine waves. The inner meta-learning objective for this setting will be replaced by the Online-aware Meta-Learning objective from Javed and White (2019).

Continual Reinforcement Learning Besides the sine waves continual regression problem, we will aim to evaluate a more challenging and real-world setting of continuous control inspired by Kaplanis et al. (2020).

6.6 Future Work

Based on the results of our main experiments, future work could involve a multi-task setting corresponding to meta-learning on different categories of tasks such as classification, segmentation, and depth estimation on a single dataset or a set of related datasets. This could also involve augmenting our objective for enforcing cross-task (distribution) consistency (Zamir et al., 2020).

6.7 Results

6.7.1 Experimental Setup and Hyperparameters

For the tasks included in Finn et al. (2017), i.e., Omniglot classification, 2D Navigation and Half-Cheetah goal velocity tasks, we use the same topmost optimizer and learning rate, fine-tuning step size, number of gradient steps, and number of tasks for the meta-step for the MAML baselines and the inner task specific updates for MATML. For the other baselines and tasks, we borrow the architectures and hyperparameters from works that compare with MAML's original setup, i.e., Raghu et al. (2020) for ANIL and Oh et al. (2021) for BOIL and tiered-ImageNet. The used architectures and hyperparameters are further described in Appendix D.1 and D.2. For the inner meta-step size, we performed a grid search over the set of values $\beta \in \{2, 1, 0.5, 0.1, 0.05, 0.01, 0.001, 0.0001\}$. Due to computational constraints on our Tesla V100-SXM2 32GB GPU, for tiered-ImageNet and continual regression, we sample only one family per outer update. Other details specific to the tasks are discussed in the corresponding sections and the Appendix.

6.7.2 Classification

Omniglot

Although the original Omniglot dataset (B. M. Lake et al., 2015) provided in characters from separated alphabets for training and testing, it did not provide in a pre-defined validation set as is common in few-shot learning (Vinyals et al., 2016). In Vinyals et al. (2016) different sets of characters for training, validation and testing are sampled, disregarding the alphabet structure. Needing to separate alphabets across these sets for learning to meta-learn, we created new sets. The details of the adapted dataset are described in Appendix D.1. Since we use a different dataset organization than most other literature, we reproduce results for the baselines as well. Specifically, we reproduce results for MAML (Finn et al., 2017), ANIL (Raghu et al., 2020) and BOIL (Oh et al., 2021) for an honest comparison of rapid learning and feature reuse. Our equivalent learning to meta-learn methods are MALTML, MALTML-ANIL, and MALTML-BOIL, respectively. Appendix Table D.1 lists the hyperparameters used for Omniglot. Table 6.1 shows the results on Omniglot.

Table 6.1: Accuracies of 5-way (5-task) Few-shot learning on 16 held-out Omniglot alphabets, before and after adaptation to the test families. Includes 95% confidence intervals across 600 test tasks from the 16 test alphabets.

Model	**1-shot**	**5-task 1-shot**	**5-shot**	**5-task 5-shot**
MAML (Finn et al., 2017)	97.86 ± 0.29%	98.04 ± 0.30%	99.32 ± 0.17%	99.26 ± 0.19%
ANIL (Raghu et al., 2020)	96.50 ± 0.41%	96.86 ± 0.38%	98.62 ± 0.27%	98.60 ± 0.26%
BOIL (Oh et al., 2021)	97.39 ± 0.35%	97.57 ± 0.32%	99.10 ± 0.22%	99.19 ± 0.18%
MALTML (ours)	95.82 ± 0.41%	96.49 ± 0.35%	98.65 ± 0.21%	98.77 ± 0.19%
MALTML-ANIL (full, ours)	91.30 ± 0.69%	92.73 ± 0.65%	94.92 ± 0.56%	95.12 ± 0.58%
MALTML-BOIL (full, ours)	97.04 ± 0.35%	97.28 ± 0.32%	98.85 ± 0.23%	99.20 ± 0.17%

Tiered-ImageNet

In a more challenging setting, we evaluate MATLML and the baselines on the Tiered-Imagenet dataset with the families corresponding to different categories of classes, as defined in Ren et al. (2018). Following Ren et al. (2018), the 34 categories are divided into 20 training, 6 validation, and 8 test categories. We visualize a selection of categories (families) and their tasks in Appendix D.2. Table 6.2 shows the results on tiered-ImageNet.

Table 6.2: 5-way Few-shot and 5-way Few-task Few-shot classification on held-out tiered-ImageNet families, before and after adaptation to the test families. The ± shows 95% confidence intervals over families. Here MALTML-Reptile refers to the model utilizing the first-order approximation for the inner meta-steps. MALTML-ANIL (partial) and MALTML-ANIL (full) refer to the models updating only the head in task specific adaptation and task as well as family specific adaptation, respectively. Similarly, MALTML-BOIL (partial) and MALTML-BOIL (full) refer to updating only the body in the corresponding settings. The size of the confidence intervals is relatively large due to the limited number of 8 test families.

Model	**1-shot**	**4-task 1-shot**	**5-shot**	**2-task 5-shot**
MAML (Finn et al., 2017)	$46.3 \pm 2.4\%$	$46.5 \pm 2.8\%$	$62.1 \pm 2.6\%$	$62.6 \pm 3.2\%$
ANIL (Raghu et al., 2020)	$47.2 \pm 1.3\%$	$46.4 \pm 1.9\%$	$62.8 \pm 2.2\%$	$63.1 \pm 2.4\%$
BOIL (Oh et al., 2021)	$48.4 \pm 1.2\%$	$48.9 \pm 2.4\%$	$65.7 \pm 2.4\%$	$66.4 \pm 3.1\%$
MALTML (ours)	$45.4 \pm 1.6\%$	$47.2 \pm 3.2\%$	$60.8 \pm 3.4\%$	$63.9 \pm 4.1\%$
MALTML-Reptile (ours)	$32.9 \pm 2.4\%$	$33.6 \pm 2.9\%$	$48.3 \pm 3.2\%$	$49.1 \pm 3.5\%$
MALTML-ANIL (partial, ours)	$34.4 \pm 1.4\%$	$38.3 \pm 2.4\%$	$51.5 \pm 1.7\%$	$54.7 \pm 2.2\%$
MALTML-ANIL (full, ours)	$34.1 \pm 1.9\%$	$39.5 \pm 2.6\%$	$52.3 \pm 2.0\%$	$52.8 \pm 2.9\%$
MALTML-BOIL (partial, ours)	$37.6 \pm 1.8\%$	$39.2 \pm 2.5\%$	$55.4 \pm 2.3\%$	$57.3 \pm 3.7\%$
MALTML-BOIL (full, ours)	$37.3 \pm 2.0\%$	$38.8 \pm 3.1\%$	$56.1 \pm 2.4\%$	$58.8 \pm 3.8\%$

6.7.3 2D Navigation

Inspired by the 2D navigation problem in Finn et al. (2017), we evaluate an equivalent family-tuning before task-tuning setting. Specifically, before goal task fine-tuning, we give the agent few related tasks to meta-tune on. All tasks are sampled from a box region of $x \in (-.5, .5)$ and $y \in (-.5, .5)$. A family with lower bound a and upper bound b constitutes of tasks with goals within the area $x \in (a, b)$ and $y \in (a, b)$. a and b are randomly sampled from $[-0.5, 0.5]$. We use the same hyperparameters as in Finn et al. (2017), a meta-learning rate of 0.01, an outer training batch size of 1, and 20 support tasks and 20 validation tasks. To allow large changes in the policy due to meta updates, we use policy gradient for both inner task specfic updates and inner meta-steps, while the topmost updates are obtained using TRPO (Schulman et al., 2015). Appendix D.3 provides illustrations. Table 6.3 shows the results on the 2D navigation problem.

Table 6.3: Average return for (meta-)adapting the 2D Navigation RL policy. Higher return is better, with 0 being the maximal reward.

Model	**init(ialization)**	**1 task update**	**1 meta-update init**	**1 task update**
MAML (Finn et al., 2017)	-12.6	-11.7	-13.6	-11.3
MALTML (ours)	-38.4	-27.7	-13.3	-10.2

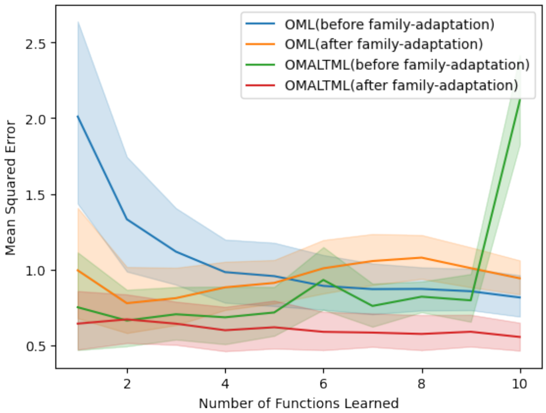


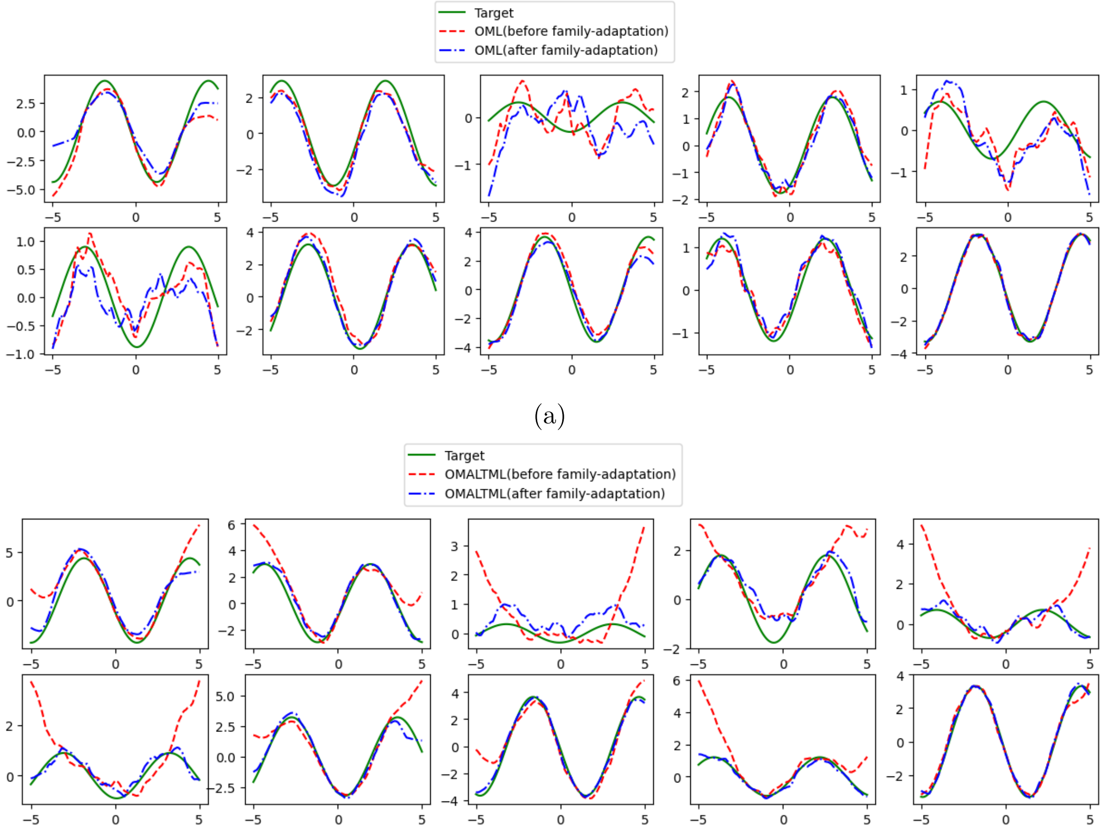
Figure 6.3: Mean squared error for OMALTML and OML averaged over 50 sinusoid families (frequencies) with 95% confidence interval drawn by 1,000 bootstraps. The x-axis denotes the number of functions in a given trajectory having being utilized to provide task(trajectory) specific updates to the model.

6.7.4 Continual Regression

Following Javed and White (2019), we extend the few-shot regression task to continual learning over trajectories of tasks consisting of sequences of sinusoid functions of randomly sampled frequencies, amplitudes, and phases. The families (task distributions) are constructed by using the same randomly sampled frequency for each task (trajectory) within a family. Further details about the architecture, hyperparameters, and the sampling procedure are provided in Appendix D.4. We refer to the online extensions of MALTML and MAML as OMALTML and OML, respectively. To ensure fair comparison, we utilize the same architecture and inner loop updates as Javed and White (2019), who split the training network into representation learning and prediction learning modules, with only the prediction learning modules being updated in the inner loop. Figures 6.3, 6.4a, and 6.4b demonstrate the effectiveness of OMALTML in adapting to unseen families of trajectories to improve the subsequent task specific adaptation on the given family.

6.7.5 Continual Reinforcement Learning

We evaluate MALTML's adaptation capabilities on continual Reinforcement Learning through the half-cheetah locomotion task with oscillating gravity recently introduced by Kaplanis et al. (2020). Each family is constructing by adding an oscillating function to a baseline gravity value of -12. The oscillating function corresponds to a sinusoid function with a phase, frequency and amplitude sampled uniformly from the ranges $[0, 2.0]$, $[0.1, 5.0]$, $[0, \pi]$ which describes the evolution of the environment's gravity with time. While obtaining the gravity's value, the timesteps are divided by the horizon (set to 200), to scale the input range to $[0, 1]$. Within each family, different tasks correspond to different goal velocities. Following Kaplanis et al. (2020), the gravity's value is appended



(a)

(b)

Figure 6.4: Few-Task Few-shot adaptation in a continual regression task for a randomly sampled trajectory before and after adaptation to another trajectory from the same family: (a) MAML (b) MALTML. Note that MALTML's ability to adapt to the trajectory signficantly improves due to the meta-step while MAML's output function demonstrates negligible change.

to the input state at each timestep. Table 6.4 shows the continual RL results.

Table 6.4: Continual Reinforcement Learning: average return for Half-Cheetah with Oscillating Gravity RL policy. Higher return is better.

Model	**init(ialization)**	**1 task update**	**1 meta-update init**	**1 task update**
MAML (Finn et al., 2017)	-145.5	-74.8	-113.1	-70.0
MALTML (ours)	-156.9	-87.8	-113.9	-62.8

6.8 Findings

6.8.1 Rapid Meta-Learning

MALTML

As demonstrated through the results in Figure 6.3, and Tables 6.3,6.4, MALTML successfully learns to adapt to new task distributions using a meta step on a few tasks for continual regression. However, the improvements due to test time adaptation to new families is significantly limited for classification tasks (Tables 6.1, 6.2). We hypothesize that this is primarily due to the limited number of training families in realistic image classification datasets unlike the continuous distribution over families available for continual regression. Moreover, we found that the magnitude of improvement due to the meta-step is quite sensitive to hyperparameters such as the step size used for the meta-step during training.

MAML

Through the results in Tables 6.1,6.2,6.3,6.4, and Figure 6.3, we observe that unlike MALTML, MAML is unable to leverage the meta-steps on unseen tasks to improve adaptability on test task distributions. Thus training MALTML to quickly adapt through meta-steps is beneficial for adaptation to unseen task distributions.

6.8.2 Rapid (Meta-)Learning is More Important than Feature Reuse

The Omniglot results in Table 6.1 and the tiered-ImageNet results in Table 6.2 show that *rapid learning* is a dominant factor in achieving good classification accuracy. Specifically, for learning to meta-learn on Omniglot, there is a clear gap in performance of 2% to 3% between the setting where only the features are trained to rapidly meta-learn and learn (MALTML-BOIL) and the setting where only the classifier is trained to rapidly meta-learn and learn (MALTML-ANIL). Note however, that although *rapid meta-learning* (using few related tasks) does seem to bring an overall performance improvement, it is not significant on Omniglot. For meta-learning, the performance improvement of BOIL over ANIL is not significant, but still consistently present, in agreement with the experiments in Oh et al. (2021).

6.8.3 Overfitting

Through the tiered-ImageNet results in Table 6.2, we observe that even though MALTML achieves significant improvement through the meta-step, the mean accuracy on test families can still be significantly lower than training task distribution top accuracies. We hypothesize that this occurs due to the model overfitting on the small number of training families (20 for tiered-ImageNet). For a continuous distribution over families, such as in continual regression, MALTML obtains significant improvement over the baselines in adapting to unseen families.

6.8.4 Reptile

As shown in Table 6.2, using the first order Reptile (Nichol et al., 2018) approximation for the inner meta-steps leads to a significant drop in the performance. This suggests a need to design more effective first order approaches for the few-task few-shot learning task.

6.9 Documented Modifications

1. Instead of creating a custom hierarchical dataset from all the classes defined in Imagenet, we directly utilized the hierarchy introduced in tiered-ImageNet (Ren et al., 2018) based on 34 categories defined on 608 classes (779,165 images). This was done to ensure computational feasibility of the experiments and the absence of any baselines for meta-learning on the full Imagenet dataset.

2. Contrary to the originally proposed oracle baselines, we primarily used MAML as a baseline, since we found that the number of families and the number of tasks within each family were insufficient to obtain reliable family specific or adaptation based oracles. Thus, following the literature, e.g., (Raghu et al., 2020; Oh et al., 2021), we don't include the oracles, and focus on contrasting our approach with real-world baselines.

3. We adjusted the 2D navigation task to have families defined by a single upper and lower bound. This eased the sampling of families and tasks.

4. Self-Supervised Learning: upon deeper investigation based on the proposal, it has been determined that few-shot self-supervised learning using augmentations is not much closer to a real-life setting than the fully supervised case. Namely, its

performance is very dependent on the augmentations used. These augmentations can only be tuned on a validation set with labels. To get to an adequate set of augmentations, one still needs a validation set consisting of (many) labeled validation families and tasks. This can be seen, e.g., in Khodadadeh et al. (2019), where the augmentations are tuned on the Omniglot test set. Hence, we have not conducted this set of experiments.

7 Conclusion

In this thesis, we have proposed few-shot learning approaches and evaluations for machine learning model adaptation with improved effectiveness and efficiency. Specifically, we have presented solutions for: more effective few-shot classification with simple but expressive inductive biases, more efficient pre-training with self-supervised learning, more effective transfer with meta-learning in a biomedical regression setting, and more effective adaptation to new task distributions by introducing a learning to meta-learn paradigm. Still, the problem of increasing the effectiveness and efficiency of machine learning model adaptation is far from being solved and will require a substantial effort in a number of directions to make significant progress. We outline a couple of such potential research directions below.

Direction 1: enhanced data utilization Although this work has mostly focused on methodological improvements and applications, improving the utilization of data could bring significant leaps in adaptation performance as well. On that front, several approaches have shown great promise. Instead of considering the (pre)train and test data in isolation, one can generate more few-shot data or reuse features from high-quality training classes (Antoniou et al., 2018; S. Yang et al., 2021). Meta-data (e.g., class names), also a key ingredient for zero-shot learning, is commonly available as well (A. Li et al., 2019; H. Yang et al., 2022). In addition, it is also realistic to assume that there is some unlabeled target data and distractors available (Ren et al., 2018). Beyond the available data mentioned above, few-shot active learning and uncertainty quantification together with informed further data collection can further boost performance. Especially since in the few-shot setting, every (good) extra sample can benefit performance significantly. With advances in zero-shot text-to-image generation with models such as CLIP (Radford et al., 2021) and DALLE (Ramesh et al., 2021), sampling data directly based on semantic (textual) information becomes possible as well. Combining all these different types of data builds on multiple areas of machine learning but will be crucial to squeeze maximum performance out of adapting machine learning models with small data.

Direction 2: few-shot generalization to the tail and beyond The benchmark datasets used in this work generally contain an equal number of samples per (new) concept or task. In practice, imbalanced data settings, where the number of samples per concept follows a power law distribution with abundant data at the head and few-shot data for most of the long tail, are more realistic (Lee et al., 2020; Ochal et al., 2023). A simple approach is to transfer learn between the head and the tail. This requires making an arbitrary cut between head and tail data, and if the head is chosen too small, generalization to tail data is not guaranteed. Self-supervised learning has been shown to be more robust to dataset imbalance (H. Liu et al., 2021), and could be a key part of developing approaches that transfer knowledge from head to tail classes more effectively.

Another more realistic dataset consideration is domain-shift, where the training data distributions and testing data distributions are shifted. These shifts can occur naturally because of, e.g., different sensors or environmental conditions. The related field of domain adaptation mostly assumes access to test samples from a single test distribution (task) at training time, which is more relaxed than the objective of few-shot learning (Farahani et al., 2021). In Chapters 3 and 4 the domain shift robustness of our proposed approaches has been studied. Most approaches targeting this setting explicitly either try to learn a single model that is robust to domain shift (Arjovsky et al., 2019) or meta-learn to adapt using unlabeled target domain samples (M. Zhang et al., 2021). These methods are highly specialized, and not always practical due to their requirement of seeing multiple domains during training. Alternatively, quantifying domain shift at test time and developing recommendations or (meta-)models on how to best transfer-learn with off-the-shelf pre-trained models with that information would be very valuable.

Direction 3: improving effective few-shot adaptation of foundation models AI has recently been undergoing a paradigm shift with the introduction of very large models trained across a broad range of data, which are called "foundation models" (Bommasani et al., 2021). These models are generally trained using self-supervised learning techniques such as the ones introduced in Chapter 4, and adapt well to downstream tasks. Examples are the large language model GPT-3 (Brown et al., 2020) and text-to-image model DALL-E (Ramesh et al., 2021). From this family, especially text-based large language models allow for a broad range of applications, including creative assistants such as ChatGPT (Ouyang et al., 2022). These models have shown emergent few-shot learning behavior through in-context learning (Brown et al., 2020), and can have their performance further boosted through efficient fine-tuning (Hu et al., 2022). Since in-context few-shot learning can be seen as a result of these models doing black-box meta-learning during learning, meta-tuning the model's parameters across a collection of a few related tasks (similar to MALTML of Chapter 6) aids the in-context learning few-shot performance (Yanda Chen et al., 2022). A promising research direction is understanding in-context learning better and how it can potentially be combined with fine-tuning, a procedure that remains quite sensitive to hyperparameter settings.

A Appendix for chapter 3

A.1 Algorithm

Algorithm 5 Regression Networks

```
Require: Training set with task batches of $N_T$ $N$-way $K$-shot episodes/tasks $\{\mathcal{T}_1, \dots, \mathcal{T}_{N_T}\}$, with every $\mathcal{T}_i$ containing sets $\mathcal{S}_{\mathcal{T}_i} = \{(\mathbf{x}_{i11}, y_{i11}), \dots, (\mathbf{x}_{iNK}, y_{iNK})\}$ and $\mathcal{Q}_{\mathcal{T}_i} = \{(\mathbf{x}_{i1(K+1)}, y_{i1(K+1)}), \dots, (\mathbf{x}_{iN(K+Q)}, y_{iN(K+Q)})\}$. $\mathcal{S}_{\mathcal{T}_i n}$ and $\mathcal{Q}_{\mathcal{T}_i n}$ denote the class $n$ subsets of support and query sets, respectively, of episode $\mathcal{T}_i$.
Require: $\alpha, \lambda_1, \lambda_2$: step size, conditioning and orthogonalization parameters
Randomly initialize $\phi$
while not done do
  for each batch do {Sample batch of tasks}
    for $i = 1$ to $N_T$ do {Select task}
      for n in $\{1, \dots, N\}$ do {Select class}
        $\mathbf{S}_n \leftarrow f_\phi(\mathcal{S}_{\mathcal{T}_i n})$ {Embed class support set subspace $\mathbf{S}_n \in \mathbb{R}^{M \times K}$}
        $\mathbf{P}_n \leftarrow \mathbf{S}_n \left(\mathbf{S}_n^T \mathbf{S}_n + \lambda_1 \mathbf{I}\right)^{-1} \mathbf{S}_n^T$ {Compute transformation matrix}
      end for
      $\mathcal{L}_{\mathcal{T}_i} \leftarrow 0$ {Initialize episode loss}
      for $n$ in $\{1, \dots, N\}$ do
        for all $(\mathbf{x}, y = n)$ in $\mathcal{Q}_{\mathcal{T}_i n}$ do
          $\mathcal{L}_{\mathcal{T}_i} \leftarrow \mathcal{L}_{\mathcal{T}_i} + \frac{1}{NQ}\left[\|\mathbf{x} - \mathbf{P}_n \mathbf{x}\|_2 + \log\left(\sum_{n'} \exp(-\|\mathbf{x} - \mathbf{P}_{n'} \mathbf{x}\|_2)\right)\right] + \lambda_2 \sum_{i \neq j}^{N} \frac{\|\mathbf{S}_i^T \mathbf{S}_j\|_F^2}{\|\mathbf{S}_i\|_F^2 \|\mathbf{S}_j\|_F^2}$
        end for
      end for
      $\phi \leftarrow \phi - \alpha \nabla_\phi \sum_i \mathcal{L}_{\mathcal{T}_i}$ {Update embedding parameters $\phi$ with gradient descent}
    end for
  end for
end while
```

A.2 Experimental details

A.2.1 Experimental Setup and Datasets

The mini-Imagenet dataset proposed by (Vinyals et al., 2016) contains 100 classes, with 600 84×84 images per class sampled from the larger ImageNet dataset (Deng et al., 2009). Following Ravi and Larochelle (2017), 64 classes are isolated for the training set and, from the remaining classes, the validation and test sets of 16 and 20 classes, respectively, are constructed. We use exactly the same train/validation/test split of classes as the one suggested by Ravi and Larochelle (2017). We implement regression networks using the automatic-differentation framework PyTorch (Paszke et al., 2017).

Many training optimizations exist, including using more classes in the training episodes than in the testing episodes (Snell et al., 2017) or pre-training the feature extractor with all training classes and a linear output layer (Qiao et al., 2018). However, we train all methods from scratch and construct our training and testing episodes to have the same number of classes N and shots K because we are interested in the relative performance of the methods. For RegressionNet, the conditioning parameter used to ensure a fully invertible matrix in Equation (3.5) is set to $\lambda_1 = 10^{-3}$. For 1-shot and 5-shot learning, $\lambda_2 = 10^{-3}$ and $\lambda_2 = 10^{-2}$ are used, respectively.

A.2.2 Architectures and training

Despite the modification of some implementation details of the methods with respect to the original papers, these settings ensure a fair comparison and W.-Y. Chen et al. (2019) report a maximal drop in classification performance of 2% with respect to the original reported performance of each method. For MAML, we reuse the results from W.-Y. Chen et al. (2019).

The *Conv-4* backbone is composed of four convolutional blocks with an input size of 84×84 as in Snell et al. (2017). Each block comprises a 64-filter 3×3 convolution with a padding of 1 and a stride of 1, a batch normalization layer, a ReLU nonlinearity and a 2×2 max-pooling layer.

The *ResNet-10* backbone has an input size of 224×224 and is a simplified version of ResNet-18 in He et al. (2016), by using only one residual building block in each layer.

All methods are trained with a random parameter initialization and use the Adam optimizer (Kingma and J. L. Ba, 2015) with an initial learning rate of 10^{-3}. During the training stage, data augmentation is done in the form of: random crop, color jitter, and left-right flip. For MatchingNet, a rather sophisticated long-short term memory (LSTM)-based full context embedding (FCE) classification layer without fine-tuning is used over the support set, and the cosine similarity metric is multiplied by a constant factor of 100. In RelationNet, the L2 norm is replaced with a softmax layer to help the training procedure (W.-Y. Chen et al., 2019). The relation module is composed of 2 convolutional blocks (same as in Conv-4), followed by two fully connected layers of size 8 and 1.

In all settings, we train for 60,000 episodes and use the held-out validation set to select the best model during training. To construct an N-way episode, we sample $N = 5$ classes from the training set of classes. From each sampled class, we sample K examples to construct the support set of an episode and $Q = 16$ examples for the query set. During testing, we average the results over 600 episodes of exactly the same form, but this time they are sampled from the test set classes.

A.3 Extra experiments: mini-ImageNet and CUB

We provide additional results for some considered methods on the effect of increasing the feature embedding backbone depth even further and increasing the number of shots to 10. Also, next to the already considered mini-ImageNet few-shot dataset, we also evaluate performance on CUB. Note that for RegressionNet we do not use the subspace orthogonalization regularizer from Section 3.3.3. Tables A.1 to A.6 show the results on this.

As expected, performance of the few-shot classification methods generally improves when a deeper backbone is used. As can be seen, the relative rank in terms of classification accuracy of the distance-metric based methods shifts significantly when going from the Conv-4 embedding architecture to a deeper ResNet-10 or ResNet-34. For example in Table A.2, RelationNet performs best in the Conv-4 setting, but worst when using ResNet-34 and RegressionNet takes first place. Generally, across datasets and number of shots, regression networks outperform the other considered methods when the backbone is relatively deep (ResNet-10 and deeper).

Since regression networks are expected to build better subspace representations when more support examples are available per class, we investigate the effect of the number of shots. As expected, when increasing the number of shots K per class, the classification accuracies increase for almost all methods. Even for the smaller Conv-4 backbone, regression networks outperform all other methods on both datasets when 10 shots are used.

Comparing the classification results on the CUB and mini-ImageNet datasets, the accuracies are consistently higher for CUB under the same setting (backbone, shots, method). This can be explained by the difference in divergence of classes; this difference is higher in mini-ImageNet than in CUB (W.-Y. Chen et al., 2019). Appendix A.4 discusses extra experiments on the transferrability of different methods under domain-shift and

	Conv-4	ResNet-10	ResNet-34
MatchingNet	61.16 ± 0.89	71.29 ± 0.90	71.44 ± 0.96
ProtoNet	51.31 ± 0.91	70.13 ± 0.94	72.03 ± 0.91
RelationNet	**62.45** ± 0.98	68.65 ± 0.91	66.20 ± 0.99
RegressionNet (ours)	59.05 ± 0.90	**72.92** ± 0.90	**73.74**± 0.88

Table A.1: CUB 5-way 1-shot classification accuracies with 95% confidence intervals for different methods and backbones. For each backbone, the best method is highlighted.

	Conv-4	ResNet-10	ResNet-34
MatchingNet	76.74 ± 0.67	83.75 ± 0.60	85.96 ± 0.52
ProtoNet	76.14 ± 0.68	85.70 ± 0.52	88.49 ± 0.46
RelationNet	**78.98** ± 0.63	82.67 ± 0.61	84.13 ± 0.57
RegressionNet (ours)	78.48 ± 0.65	**87.45** ± 0.48	**89.30** ± 0.45

Table A.2: CUB 5-way 5-shot classification accuracies with 95% confidence intervals for different methods and backbones. For each backbone, the best method is highlighted.

further reports on the difference in divergence between classes in CUB and mini-ImageNet.

A.4 Domain Shift: mini-ImageNet to CUB and CUB to mini-ImageNet

We provide additional results on how some of the considered few-shot classification methods perform when the test set is increasingly different from the train set. Note that for RegressionNet we do not use the subspace orthogonalization regularizer from Section 3.3.3. In this section we perform experiments by permuting the training and testing domain from one dataset to another (denoted as *train* → *test*). The mini-ImageNet → CUB setting, introduced by W.-Y. Chen et al. (2019), represents a coarse-grained

	Conv-4	ResNet-10	ResNet-34
MatchingNet	79.27 ± 0.61	85.38 ± 0.54	89.03 ± 0.46
ProtoNet	81.77 ± 0.57	87.61 ± 0.44	90.35 ± 0.40
RelationNet	82.66 ± 0.56	84.96 ± 0.53	87.29 ± 0.46
RegressionNet (ours)	**83.02** ± 0.53	**89.28** ± 0.41	**91.46** ± 0.36

Table A.3: CUB 5-way 10-shot classification accuracies with 95% confidence intervals for different methods and backbones. For each backbone, the best method is highlighted.

	Conv-4	ResNet-10	ResNet-34
MatchingNet	48.14 ± 0.78	54.49 ± 0.81	53.20 ± 0.78
ProtoNet	44.42 ± 0.84	51.98 ± 0.84	53.90 ± 0.83
RelationNet	**49.31** ± 0.85	52.19 ± 0.83	52.74 ± 0.83
RegressionNet (ours)	47.99 ± 0.80	**54.83** ± 0.83	**53.93** ± 0.81

Table A.4: miniImageNet 5-way 1-shot classification accuracies with 95% confidence intervals for different methods and backbones. For each backbone, the best method is highlighted.

	Conv-4	ResNet-10	ResNet-34
MatchingNet	63.28 ± 0.68	69.14 ± 0.69	70.36 ± 0.70
ProtoNet	65.15 ± 0.67	73.77 ± 0.64	75.14 ± 0.65
RelationNet	65.33 ± 0.70	69.97 ± 0.68	71.38 ± 0.68
RegressionNet (ours)	**66.41** ± 0.66	**74.03** ± 0.68	**75.62** ± 0.61

Table A.5: miniImageNet 5-way 5-shot classification accuracies with 95% confidence intervals for different methods and backbones. For each backbone, the best method is highlighted.

	Conv-4	ResNet-10	ResNet-34
MatchingNet	67.47 ± 0.64	74.63 ± 0.62	73.88 ± 0.65
ProtoNet	72.01 ± 0.67	78.92 ± 0.54	79.62 ± 0.55
RelationNet	70.27 ± 0.63	75.69 ± 0.61	75.40 ± 0.62
RegressionNet (ours)	**72.69** ± 0.61	**80.08** ± 0.57	**80.42** ± 0.53

Table A.6: miniImageNet 5-way 10-shot classification accuracies with 95% confidence intervals for different methods and backbones. For each backbone, the best method is highlighted.

training set test set	CUB CUB	mini-ImageNet mini-ImageNet	mini-ImageNet CUB	CUB mini-ImageNet
MatchingNet	83.75 ± 0.60	69.14 ± 0.69	52.59 ± 0.71	48.95 ± 0.67
ProtoNet	85.70 ± 0.52	73.77 ± 0.64	59.22 ± 0.74	53.58 ± 0.73
RelationNet	82.67 ± 0.61	69.97 ± 0.68	54.36 ± 0.71	45.27 ± 0.66
RegressionNet (ours)	**87.45** ± 0.48	**74.03** ± 0.68	**62.71** ± 0.71	**56.66** ± 0.68

Table A.7: Identical domain and domain shift accuracies for 5-way 5-shot classification using a ResNet-10 backbone and mini-ImageNet and CUB datasets. The best-performing method is highlighted.

train set to fine-grained test set domain-shift. Conversely, we propose to use the inverse scenario CUB → mini-ImageNet as well.

In mini-ImageNet → CUB, all 100 classes from mini-ImageNet make up the training set and the same 50 validation and 50 test classes from CUB are used. Conversely, in CUB → mini-ImageNet, all 200 classes from CUB make up the training set and the same 16 validation and 20 test classes from mini-ImageNet are used. To illustrate the domain shift, we can look at the mini-ImageNet → CUB setting: the 200 classes of CUB are all types of birds, whereas the 64 classes in the training set of mini-ImageNet only contain 3 types of birds, which are not to be found in CUB. Evaluating the domain-shift scenario enables us to understand better which method is more general.

Table A.7 presents our results in the context of existing state-of-the-art distance-metric learning based methods. All experiments in this setting are conducted with a ResNet-10 backbone on a 5-way 5-shot problem. The intra-domain difference between classes is higher in mini-ImageNet than in CUB (W.-Y. Chen et al., 2019), which is reflected by a drop of the test accuracies of all methods. It can also be seen that when the domain difference between the training and test stage classes increases (left to right in Table A.7) the performance of all few-shot algorithms lowers.

As expected, the classification accuracy lowers when going from the mini-ImageNet → CUB to the CUB → mini-ImageNet setting. Interestingly though, it does not decrease substantially. This shows that the different feature embeddings generalize well, at least across these two different datasets. Compared to other distance-metric learning based methods, regression networks show significantly better performance in both the mini-ImageNet → CUB setting and CUB → mini-ImageNet setting.

B Appendix for chapter 4

B.1 Experimental Details

B.1.1 Datasets

In-domain datasets

For our in-domain experiments we used the popular few-shot datasets Omniglot (B. Lake et al., 2011) and mini-ImageNet (Vinyals et al., 2016).

Omniglot consists of 1623 handwritten characters from 50 alphabets and 20 examples per character. Identical to Vinyals et al. (2016), the grayscale images are resized to 28x28. Following Santoro et al. (2016), we use 1200 characters for training and 423 for testing.

Mini-ImageNet is a subset of the ILSVRC-12 dataset (Russakovsky et al., 2015a), which contains 60,000 color images that we resized to 84x84. For comparability, we use the splits introduced by Ravi and Larochelle (2017) over 100 classes with 600 images each. 64 classes are used for pre-training and 20 for testing. We only use the 16 validation set classes for limited hyperparameter tuning of batch size N, number of queries Q and the augmentation strengths.

Cross-domain datasets

We evaluate all cross-domain experiments the CDFSL-benchmark (Guo et al., 2019). It comprises four datasets with decreasing similarity to mini-ImageNet. In order of similarity, they are plant disease images from CropDiseases (Mohanty et al., 2016), satellite images from EuroSAT (Helber et al., 2019), dermatological images from ISIC2018 (Tschandl et al., 2018; Codella et al., 2019) and grayscale chest x-ray images from ChestX (Wang et al., 2017).

Table B.1: ProtoTransfer hyperparameter summary.

	in-domain		cross-domain	
Hyperparameter	Omniglot	mini-ImageNet	mini-ImageNet	CUB
Model architecture	Conv-4	Conv-4	ResNet-10	Conv-4
Image input size	28 × 28	84 × 84	224 × 224	84 × 84
Optimizer	Adam	Adam	Adam	Adam
Learning rate	0.001	0.001	0.001	0.001
Learning rate decay factor	0.5	0.5	/	0.5
Learning rate decay period	25,000	25,000	/	25,000
Support examples	1	1	1	1
Augmented queries (Q)	3	3	3	3
Training batch size (N)	50	50	50	50
Augmentation appendix	B.1.3	B.1.3	B.1.3	B.1.3
Fine-tuning optimizer	Adam	Adam	Adam	Adam
Fine-tuning learning rate	0.001	0.001	0.001	0.001
Fine-tuning batch size	5	5	5	5
Fine-tuning epochs	15	15	15	15
Fine-tune last layer	✓	✓	✓	✓
Fine-tune backbone			✓	

Caltech-UCSD Birds-200-2011 (CUB) dataset

We use the Caltech-UCSD Birds-200-2011 (CUB) dataset (Welinder et al., 2010; Wah et al., 2011) in our ablation studies. It is composed of 11,788 images from 200 different bird species. We follow the splits proposed by Hilliard et al. (2018) with 100 training, 50 validation and 50 test classes. We do not use the validation set classes.

B.1.2 Architecture and Optimization Parameters

In the following, we describe the experimental details for the individual experiments. We deliberately stay close to the parameters reported in prior work and do not perform an extensive hyperparameter search for our specific setup, as this can easily lead to performance overestimation compared to simpler approaches (Oliver et al., 2018)). Table B.1 summarizes the hyperparameters we used for ProtoTransfer.

In-Domain Experiments

Our mini-ImageNet and Omniglot experiments use the Conv-4 architecture proposed in Vinyals et al. (2016) for comparability. Its four convolutional blocks each apply a 64-filters

3x3 convolution, batch normalization, a ReLU nonlinearity and 2x2 max-pooling. The pre-training mostly mirrors Snell et al. (2017) and uses Adam (Kingma and J. L. Ba, 2015) with an initial learning rate of 0.001, which is multiplied by a factor of 0.5 every 25000 iterations. We use a batch size of 50. We do not use the validation set to select the best training epoch. Instead training stops after 20.000 iterations without improvement in training accuracy.

Cross-Domain Experiments

Our experiments on the CDFSL-Challenge are based on the code provided by Guo et al. (2019). Following Guo et al. (2019), we use a ResNet10 architecture that is pre-trained on mini-Imagenet images of size 224x224 for 400 epochs with Adam (Kingma and J. L. Ba, 2015) and the default learning rate of 0.001 for best comparability with the results reported in Guo et al. (2019). The batch size for self-supervised pre-training is 50. We do not use a validation set.

Caltech-UCSD Birds-200-2011 (CUB) Experiments

The CUB training is identical in terms of architecture (Conv-4) and optimization to the setup for our in-domain experiments.

Prototypical Fine-Tuning

During the fine-tuning stage we add a fully connected classification layer after the embedding function and initialize as described in Section 4.3.3. We split the support examples into batches of 5 images each and perform 15 fine-tuning epochs with Adam (Kingma and J. L. Ba, 2015) and an initial learning rate of 0.001. For target datasets mini-ImageNet and Omniglot only the last fully connected layer is optimized, while for the CDFSL benchmark experiments the embedding network is adapted as well.

B.1.3 Augmentations

CDFSL transforms

For the CDFSL-benchmark (Guo et al., 2019) experiments we employ the same augmentations as T. Chen et al. (2020), as these have proven to work well for ImageNet (Russakovsky et al., 2015a) images of size 224x224. They are as follows:

1. Random crop and resize: `scale` $\in [0.08, 1.0]$, `aspect ratio` $\in [3/4, 4/3]$, Bilinear filter with `interpolation = 2`

2. Random horizontal flip
3. Random ($p = 0.8$) color jitter: `brightness = contrast = saturation = 0.8`, `hue=0.2`
4. Random ($p = 0.2$) grayscale
5. Gaussian blur, random radius $\sigma \in [0.1, 0.2]$

mini-ImageNet & CUB transforms

For the mini-Imagenet and CUB experiments we used lighter versions of the T. Chen et al. (2020) augmentations, namely no Gaussian blur, lower color jitter strengths and smaller rescaling and cropping ranges. They are as follows:

1. * Random crop and resize: `scale` $\in [\mathbf{0.5}, 1.0]$, `aspect ratio` $\in [3/4, 4/3]$, Bilinear filter with `interpolation = 2`
2. Random horizontal flip
3. * Random vertical flip
4. * Random ($p = 0.8$) color jitter: `brightness = contrast = saturation = 0.4`, `hue=0.2`
5. Random ($p = 0.2$) grayscale

Omniglot transforms

For Omniglot we use a set of custom augmentations, namely random resizing and cropping, horizontal and vertical flipping, Image-Pixel Dropout (Krizhevsky et al., 2012) and Cutout (DeVries and Taylor, 2017). They are as follows:

1. Resize to a size of 28x28 pixels
2. Random and resize: `scale` $\in [0.6, 1.0]$, `aspect ratio` $\in [3/4, 4/3]$, Bilinear filter with `interpolation = 2`
3. Random horizontal flip
4. Random vertical flip
5. Random ($p = 0.3$) dropout
6. Random erasing of a rectangular region in an image (Zhong et al., 2020), setting pixel values to `0`: `scale` $\in [0.02, 0.33]$, `aspect ratio` $\in [0.3, 3.3]$

B.1.4 Classes for t-SNE Plots

The classes in the t-SNE plots are a random subset of classes from the mini-ImageNet base classes (classes 1-5) and the mini-ImageNet novel classes (classes 6-10). Their corresponding labels are the following:

1. n02687172 aircraft carrier
2. n04251144 snorkel
3. n02823428 beer bottle
4. n03676483 lipstick
5. n03400231 frying pan
6. n03272010 electric guitar
7. n07613480 trifle
8. n03775546 mixing bowl
9. n03127925 crate
10. n04146614 school bus

Each of the t-SNE plots in Figure 4.3 shows 500 randomly selected embedded images from within those classes.

B.1.5 Results With Full Confidence Intervals & References

Table B.2: Accuracy (%) of methods on N-way K-shot classification tasks on Omniglot and a Conv-4 architecture. All results are reported with 95% confidence intervals over 600 randomly generated test episodes. Results style: **best** and <u>second best</u>.

Method (N,K)	**(5,1)**	**(5,5)**	**(20,1)**	**(20,5)**
	Omniglot			
Training (scratch)	52.50 ± 0.84	74.78 ± 0.69	24.91 ± 0.33	47.62 ± 0.44
CACTUs-MAML[I]	68.84 ± 0.80	87.78 ± 0.50	48.09 ± 0.41	73.36 ± 0.34
CACTUs-ProtoNet[I]	68.12 ± 0.84	83.58 ± 0.61	47.75 ± 0.43	66.27 ± 0.37
UMTRA[II]	83.80 ± -	95.43 ± -	<u>74.25</u> ± -	<u>92.12</u> ± -
AAL-ProtoNet[III]	84.66 ± 0.70	89.14 ± 0.27	68.79 ± 1.03	74.28 ± 0.46
AAL-MAML++[III]	<u>88.40</u> ± 0.75	<u>97.96</u> ± 0.32	70.21 ± 0.86	88.32 ± 1.22
UFLST[IV]	**97.03** ± -	**99.19** ± -	**91.28** ± -	**97.37** ± -
ProtoTransfer (ours)	88.00 ± 0.64	96.48 ± 0.26	72.27 ± 0.47	89.08 ± 0.23
Supervised training				
MAML[I]	94.46 ± 0.35	98.83 ± 0.12	84.60 ± 0.32	96.29 ± 0.13
ProtoNet	97.70 ± 0.29	99.28 ± 0.10	94.40 ± 0.23	98.39 ± 0.08
Pre+Linear	94.30 ± 0.43	99.08 ± 0.10	86.05 ± 0.34	97.11 ± 0.11

[I] Hsu et al. (2019)

[II] Khodadadeh et al. (2019)

[III] Antoniou and Storkey (2019)

[IV] Ji et al. (2019)

Table B.3: Accuracy (%) of methods on N-way K-shot classification tasks mini-Imagenet and a Conv-4 architecture. All results are reported with 95% confidence intervals over 600 randomly generated test episodes. Results style: **best** and second best.

Method (N,K)	**(5,1)**	**(5,5)**	**(5,20)**	**(5,50)**
	Mini-ImageNet			
Training (scratch)	27.59 ± 0.59	38.48 ± 0.66	51.53 ± 0.72	59.63 ± 0.74
CACTUs-MAML[I]	39.90 ± 0.74	53.97 ± 0.70	63.84 ± 0.70	69.64 ± 0.63
CACTUs-ProtoNet[I]	39.18 ± 0.71	53.36 ± 0.70	61.54 ± 0.68	63.55 ± 0.64
UMTRA[II]	39.93 ± -	50.73 ± -	61.11 ± -	67.15 ± -
AAL-ProtoNet[III]	37.67 ± 0.39	40.29 ± 0.68	-	-
AAL-MAML++[III]	34.57 ± 0.74	49.18 ± 0.47	-	-
UFLST[IV]	33.77 ± 0.70	45.03 ± 0.73	53.35 ± 0.59	56.72 ± 0.67
ULDA-ProtoNet[V]	40.63 ± 0.61	55.41 ± 0.57	63.16 ± 0.51	65.20 ± 0.50
ULDA-MetaOptNet[V]	40.71 ± 0.62	54.49 ± 0.58	63.58 ± 0.51	67.65 ± 0.48
ProtoTransfer (ours)	**45.67** ± 0.79	**62.99** ± 0.75	**72.34** ± 0.58	**77.22** ± 0.52
Supervised training				
MAML[I]	46.81 ± 0.77	62.13 ± 0.72	71.03 ± 0.69	75.54 ± 0.62
ProtoNet	46.44 ± 0.78	66.33 ± 0.68	76.73 ± 0.54	78.91 ± 0.57
Pre+Linear	43.87 ± 0.69	63.01 ± 0.71	75.46 ± 0.58	80.17 ± 0.51

[I] Hsu et al. (2019)

[II] Khodadadeh et al. (2019)

[III] Antoniou and Storkey (2019)

[IV] Ji et al. (2019)

[V] Qin et al. (2020)

Table B.4: Accuracy (%) of methods on N-way K-shot classification tasks of the CDFSL benchmark (Guo et al., 2019). All models are trained on mini-ImageNet with ResNet-10. All results are reported with 95% confidence intervals over 600 randomly generated test episodes. Results style: **best** and second best.

Method	UnSup	(5,5)	(5,20)	(5,50)	(5,5)	(5,20)	(5,50)
			ChestX			**ISIC**	
ProtoNet[*]		24.05 ± 1.01	28.21 ± 1.15	29.32 ± 1.12	39.57 ± 0.57	49.50 ± 0.55	51.99 ± 0.52
Pre+Mean-Centroid[*]		26.31 ± 0.42	30.41 ± 0.46	34.68 ± 0.46	47.16 ± 0.54	56.40 ± 0.53	61.57 ± 0.66
Pre+Linear[*]		25.97 ± 0.41	31.32 ± 0.45	35.49 ± 0.45	**48.11** ± 0.64	**59.31** ± 0.48	**66.48** ± 0.56
UMTRA-ProtoNet	✓	24.94 ± 0.43	28.04 ± 0.44	29.88 ± 0.43	39.21 ± 0.53	44.62 ± 0.49	46.48 ± 0.47
UMTRA-ProtoTune	✓	25.00 ± 0.43	30.41 ± 0.44	35.63 ± 0.48	38.47 ± 0.55	51.60 ± 0.54	60.12 ± 0.50
ProtoTransfer (ours)	✓	**26.71** ± 0.46	**33.82** ± 0.48	**39.35** ± 0.50	45.19 ± 0.56	59.07 ± 0.55	66.15 ± 0.57
			EuroSat			**CropDiseases**	
ProtoNet[*]		73.29 ± 0.71	82.27 ± 0.57	80.48 ± 0.57	79.72 ± 0.67	88.15 ± 0.51	90.81 ± 0.43
Pre+Mean-Centroid[*]		**82.21** ± 0.49	87.62 ± 0.34	88.24 ± 0.29	87.61 ± 0.47	93.87 ± 0.68	94.77 ± 0.34
Pre+Linear[*]		79.08 ± 0.61	**87.64** ± 0.47	**91.34** ± 0.37	**89.25** ± 0.51	**95.51** ± 0.31	**97.68** ± 0.21
UMTRA-ProtoNet	✓	74.91 ± 0.72	80.42 ± 0.66	82.24 ± 0.61	79.81 ± 0.65	86.84 ± 0.50	88.44 ± 0.46
UMTRA-ProtoTune	✓	68.11 ± 0.70	81.56 ± 0.54	85.05 ± 0.50	82.67 ± 0.60	92.04 ± 0.43	95.46 ± 0.31
ProtoTransfer (ours)	✓	75.62 ± 0.67	86.80 ± 0.42	90.46 ± 0.37	86.53 ± 0.56	95.06 ± 0.32	97.01 ± 0.26

[*] Results from Guo et al. (2019)

Table B.5: Accuracy (%) of methods on N-way K-shot (N, K) classification tasks on mini-ImageNet with a Conv-4 architecture for different training image batch sizes, number of training queries (Q) and optional finetuning on target tasks (FT). UMTRA-MAML results are taken from Khodadadeh et al. (2019), where UMTRA uses AutoAugment (Cubuk et al., 2019) augmentations. All results are reported with 95% confidence intervals over 600 randomly generated test episodes. Results style: **best** and second best.

Training	Testing	batch size	Q	FT	(5,1)	(5,5)	(5,20)	(5,50)
n.a.	ProtoNet	n.a.	n.a.	no	27.05 ± 0.56	34.12 ± 0.59	39.68 ± 0.59	41.40 ± 0.59
UMTRA[*]	MAML	5	1	yes	39.93 ± -	50.73 ± -	61.11 ± -	67.15 ± -
UMTRA	ProtoNet	5	1	no	39.17 ± 0.53	53.78 ± 0.53	62.41 ± 0.49	64.40 ± 0.46
ProtoCLR	ProtoNet	50	1	no	44.53 ± 0.60	62.88 ± 0.54	70.86 ± 0.48	73.93 ± 0.44
ProtoCLR	ProtoNet	50	3	no	44.89 ± 0.58	**63.35** ± 0.54	72.27 ± 0.45	74.31 ± 0.45
ProtoCLR	ProtoNet	50	5	no	45.00 ± 0.57	63.17 ± 0.55	71.70 ± 0.48	73.98 ± 0.44
ProtoCLR	ProtoNet	50	10	no	44.98 ± 0.58	62.56 ± 0.53	70.78 ± 0.48	73.69 ± 0.44
ProtoCLR	ProtoTune	50	3	yes	**45.67** ± 0.76	62.99 ± 0.75	**72.34** ± 0.58	**77.22** ± 0.52

[*] Khodadadeh et al. (2019)

Table B.6: Accuracy (%) of methods on N-way K-shot classification tasks on Mini-ImageNet with a Conv-4 architecture when reducing the number of pre-training classes or images. All results are reported with 95% confidence intervals over 600 randomly generated test episodes. Results style: **best** and second best.

Method	**# images**	**# classes**	**(5,1)**	**(5,5)**	**(5,20)**	**(5,50)**
Random+ProtoTune[*]	0	0	28.16 ± 0.56	35.32 ± 0.60	42.72 ± 0.63	47.05 ± 0.61
Random+Linear	0	0	26.77 ± 0.52	34.68 ± 0.58	44.62 ± 0.61	51.79 ± 0.61
ProtoTransfer	600	1	32.20 ± 0.57	48.89 ± 0.64	60.80 ± 0.62	66.91 ± 0.56
ProtoTransfer	600	≤ 64	37.02 ± 0.65	52.84 ± 0.72	64.76 ± 0.63	69.54 ± 0.58
Pre+Linear	600	≤ 64	36.58 ± 0.69	52.03 ± 0.72	63.04 ± 0.68	68.34 ± 0.64
ProtoTransfer	1200	2	34.15 ± 0.61	53.59 ± 0.68	64.59 ± 0.63	70.24 ± 0.54
ProtoTransfer	1200	≤ 64	38.88 ± 0.70	55.53 ± 0.69	66.91 ± 0.57	71.16 ± 0.56
Pre+Linear	1200	2	27.05 ± 0.46	37.06 ± 0.57	47.68 ± 0.62	54.37 ± 0.59
Pre+Linear	1200	≤ 64	37.81 ± 0.70	53.96 ± 0.69	65.43 ± 0.68	70.02 ± 0.59
ProtoTransfer	2400	4	37.96 ± 0.64	55.27 ± 0.69	66.61 ± 0.60	70.92 ± 0.55
ProtoTransfer	2400	≤ 64	40.90 ± 0.71	59.12 ± 0.71	69.34 ± 0.60	73.32 ± 0.55
Pre+Linear	2400	4	31.26 ± 0.57	45.41 ± 0.65	58.48 ± 0.65	63.63 ± 0.61
Pre+Linear	2400	≤ 64	38.82 ± 0.69	55.26 ± 0.70	67.96 ± 0.64	73.29 ± 0.58
ProtoTransfer	4800	8	40.74 ± 0.73	59.00 ± 0.71	69.45 ± 0.61	74.08 ± 0.53
ProtoTransfer	4800	≤ 64	41.97 ± 0.74	59.09 ± 0.71	69.40 ± 0.61	73.60 ± 0.56
Pre+Linear	4800	8	34.54 ± 0.60	52.04 ± 0.69	65.71 ± 0.59	70.44 ± 0.53
Pre+Linear	4800	≤ 64	41.38 ± 0.70	58.15 ± 0.73	70.51 ± 0.63	75.05 ± 0.56
ProtoTransfer	9600	16	42.04 ± 0.76	60.35 ± 0.72	70.70 ± 0.58	75.16 ± 0.57
ProtoTransfer	9600	≤ 64	42.94 ± 0.78	60.36 ± 0.72	70.66 ± 0.59	74.67 ± 0.55
Pre+Linear	9600	16	38.39 ± 0.65	54.78 ± 0.67	67.75 ± 0.60	73.42 ± 0.52
Pre+Linear	9600	≤ 64	41.74 ± 0.73	60.24 ± 0.68	73.03 ± 0.61	77.90 ± 0.53
ProtoTransfer	19200	32	43.88 ± 0.76	61.22 ± 0.69	71.26 ± 0.59	75.62 ± 0.52
ProtoTransfer	19200	≤ 64	44.02 ± 0.74	60.78 ± 0.72	71.58 ± 0.56	75.77 ± 0.52
Pre+Linear	19200	32	40.10 ± 0.63	59.58 ± 0.65	72.45 ± 0.56	76.53 ± 0.52
Pre+Linear	19200	≤ 64	41.58 ± 0.71	61.20 ± 0.66	73.57 ± 0.56	79.01 ± 0.51
ProtoTransfer	38400	64	45.67 ± 0.76	62.99 ± 0.75	72.34 ± 0.58	77.22 ± 0.52
Pre+Linear	38400	64	43.87 ± 0.69	63.01 ± 0.71	75.46 ± 0.58	80.17 ± 0.51

[*] Trained for 100 epochs instead of the default 15 epochs for ProtoTune, since training a classifier on top of a fixed randomly initialized network is expected to require more fine-tuning than starting from a pre-trained network.

Table B.7: Accuracy (%) on N-way K-shot classification tasks on Mini-ImageNet for methods trained on the CUB training set (5885 images) with a Conv-4 architecture. All results are reported with 95% confidence intervals over 600 randomly generated test episodes. Results style: **best** and second best.

Training	**Testing**	**(5,1)**	**(5,5)**	**(5,20)**	**(5,50)**
ProtoCLR	ProtoNet	34.56 ± 0.61	**52.76** ± 0.63	62.76 ± 0.59	66.01 ± 0.55
ProtoCLR	ProtoTransfer	**35.37** ± 0.63	52.38 ± 0.66	**63.82** ± 0.59	**68.95** ± 0.57
Pre(training)	Linear	33.10 ± 0.60	47.01 ± 0.65	59.94 ± 0.62	65.75 ± 0.63

C Appendix for chapter 5

We compute the gradient of the meta-learning loss function. Suppose that, for each task T_τ, the inner-learner takes 2 steps of stochastic gradient descent and updates the parameters to θ_τ^2. From Equation (5.3), we can write the gradient using a Taylor expansion:

$$g = \frac{\theta_\tau^0 - \theta_\tau^2}{\alpha} = \mathcal{L}'_{\tau,0}\left(\theta_\tau^0\right) + \mathcal{L}'_{\tau,1}\left(\theta_\tau^0\right) - \alpha\mathcal{L}''_{\tau,1}\left(\theta_\tau^0\right)\mathcal{L}'_{\tau,0}\left(\theta_\tau^0\right) + O\left(\alpha^2\right) \tag{C.1}$$

$\mathcal{L}_{\tau,0}$ is the loss computed on the first minibatch sampled from task τ, and $\mathcal{L}_{\tau,1}$ is the loss computed on the second minibatch sampled from task τ. The expectation of the first two terms in Equation (C.1) corresponds to the gradient of expected loss, and the expectation of the third term can be written as:

$$\mathbb{E}_{\tau,0,1}\left[\mathcal{L}''_1(\theta)\mathcal{L}'_0(\theta)\right] = \frac{1}{2}\mathbb{E}_{\tau,0,1}\left[\frac{\partial}{\partial\theta}\left(\mathcal{L}'_1(\theta)\cdot\mathcal{L}'_0(\theta)\right)\right]. \tag{C.2}$$

This term increases the inner product of the gradient of the first minibatch and the gradient of the second minibatch, which means it encourages the gradients from different minibatches for a given task to align in the same direction.

D Appendix for chapter 6

D.1 Omniglot

Omniglot consists of 1623 handwritten characters from 50 alphabets and 20 examples per character. Identical to Vinyals et al. (2016), the grayscale images are resized to 28x28. However, to not have overlapping alphabets between training and testing sets, we resample the characters and alphabets according to Section D.1.2.

D.1.1 Hyperparameters

Table D.1 provides the hyperparamters used for Omniglot training and testing.

D.1.2 training (30), validation (4), test (16) alphabets

training *Anglo-Saxon_Futhorc, Armenian, Atlantean, Aurek-Besh, Balinese, Bengali, Braille, Burmese_(Myanmar), Cyrillic, Early_Aramaic, Ge_ez, Grantha, Gujarati, Inuktitut_(Canadian_Aboriginal_Syllabics), Japanese_(hiragana), Japanese_(katakana), Kannada, Keble, Korean, Latin, Malay_(Jawi_-_Arabic), Malayalam, Manipuri, Mkhedruli_(Georgian), Ojibwe_(Canadian_Aboriginal_Syllabics), Sanskrit, Sylheti, Syriac_(Estrangelo), Tagalog, Tifinagh*

validation *Asomtavruli_(Georgian), Futurama, Oriya, ULOG*

testing *Alphabet_of_the_Magi, Angelic, Arcadian, Atemayar_Qelisayer, Avesta, Blackfoot_(Canadian_Aboriginal_Syllabics), Glagolitic, Greek, Gurmukhi, Hebrew, Mongolian, N_Ko, Old_Church_Slavonic_(Cyrillic), Syriac_(Serto), Tengwar, Tibetan*

Table D.1: Omniglot hyperparameter summary.

	Few-shot			Few-task Few-shot		
Hyperparameter	MAML	ANIL	BOIL	MALTML	MALTML-ANIL	MALTML-BOIL
Model architecture	Conv-4	Conv-4	Conv-4	Conv-4	Conv-4	Conv-4
Image input size	28×28	28×28	28×28	28×28	28×28	28×28
Outer optimizer	Adam	Adam	Adam	Adam	Adam	Adam
Outer step size	0.001	0.001	0.001	0.001	0.001	0.001
Query examples/task	15	15	15	15	15	15
Support Tasks/family	n.a.	n.a.	n.a.	5	5	5
Query Tasks/family	n.a.	n.a.	n.a.	15	15	15
Training batch size	16 tasks	16 tasks	16 tasks	4 families	4 families	4 families
Meta-tuning optimizer	n.a.	n.a.	n.a.	SGD	SGD	SGD
Meta-tuning step size	n.a.	n.a.	n.a.	0.4	0.1	0.4
Meta-tuning steps	n.a.	n.a.	n.a.	1	1	1
Meta-tune last layer	n.a.	n.a.	n.a.	✓	✓	
Meta-tune backbone	n.a.	n.a.	n.a.	✓		✓
Fine-tuning optimizer	SGD	SGD	SGD	SGD	SGD	SGD
Fine-tuning step size	0.1	0.1	0.1	0.4	0.1	0.4
Fine-tuning steps	1	5	1	1	5	1
Fine-tune last layer	✓	✓		✓	✓	
Fine-tune backbone	✓		✓	✓		✓

Table D.2: Tiered-ImageNet hyperparameter summary.

	Few-shot			Few-task Few-shot		
Hyperparameter	MAML	MALTML	MALTML-Reptile	MALTML	MALTML-ANIL (partial)	MALTML-ANIL (full)
Model architecture	Conv-4	Conv-4	Conv-4	Conv-4	Conv-4	Conv-4
Image input size	84 × 84	84 × 84	84 × 84	84 × 84	84 × 84	84 × 84
Outer optimizer	Adam	Adam	Adam	Adam	Adam	Adam
Outer step size	0.001	0.001	0.001	0.001	0.001	0.001
Query examples/task	15	15	15	15	15	15
Support Tasks/family *(shots)*	n.a.	4(1), 2(5)	4(1), 2(5)	4(1), 2(5)	4(1), 2(5)	4(1), 2(5)
Query Tasks/family *num(shot)*	n.a.	8(1), 4(5)	8(1), 4(5)	8(1), 4(5)	8(1), 4(5)	8(1), 4(5)
Training batch size *num(shot)*	8(1), 4(5)	1 family	1 family	1 family	1 family	1 family
Meta-tuning optimizer	n.a.	SGD	SGD	SGD	SGD	SGD
Meta-tuning step size	n.a.	0.05	0.05	0.05	0.05	0.05
Meta-tuning steps	n.a.	1	1	1	1	1
Meta-tune entire network	✓	✓	✓	✓	×	✓
Fine-tuning optimizer	SGD	SGD	SGD	SGD	SGD	SGD
Fine-tuning step size	0.5	0.5	0.5	0.5	0.5	0.5
Fine-tuning steps	1	1	1	1	1	1
Fine-tune entire network	✓	✓	✓	✓	×	×

D.2 Tiered-ImageNet

D.2.1 Hyperparameters

Table D.2 provides the hyperparameters used for tiered-ImageNet training and testing.

D.2.2 training (20), validation (6), test (8) categories

training *'game equipment', 'electronic equipment', 'snake, serpent, ophidian', 'tool', 'establishment', 'passerine, passeriform bird', 'aquatic bird', 'primate', 'garment', 'terrier', 'saurian', 'ungulate, hoofed mammal', 'feline, felid', 'restraint, constraint', 'building, edifice', 'musical instrument, instrument', 'instrument', 'protective covering, protective cover, protect', 'hound, hound dog', 'craft'*

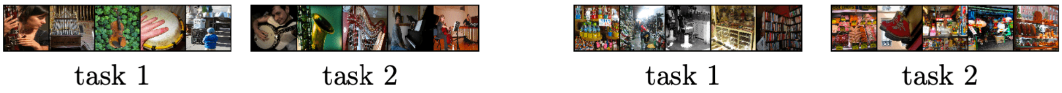
validation *'motor vehicle, automotive vehicle', 'furnishing', 'machine', 'durables, durable goods, consumer durables', 'mechanism', 'sporting dog, gun dog'*

testing *'working dog', 'aquatic vertebrate', 'vessel', 'geological formation, formation', 'obstruction, obstructor, obstructer, impedimen', 'solid', 'substance', 'insect'*

D.2.3 Family and task visualization

(a) family: 'instrument'

(b) family: 'saurian'



(c) family: 'musical instrument, instrument'

(d) family: 'establishment'

Figure D.1: Visualization of the tiered-ImageNet family distribution

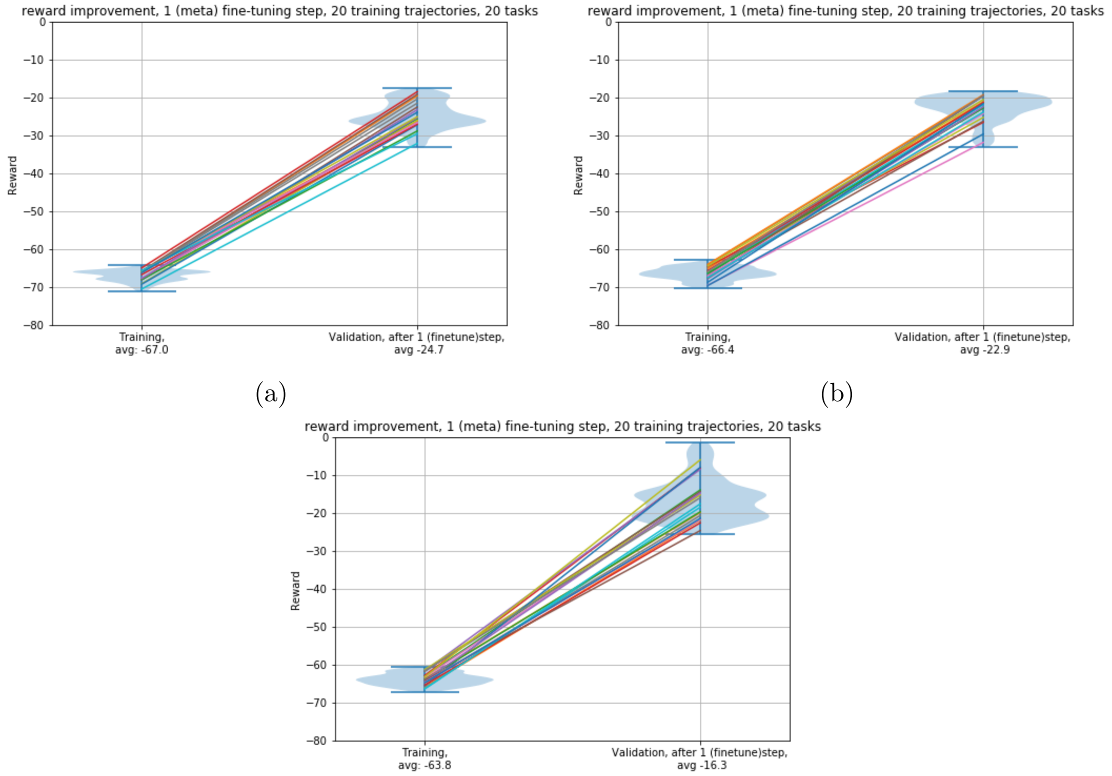
D.3 2D navigation

Figure D.2 shows the results for 2D navigation.

D.4 Continual Regression

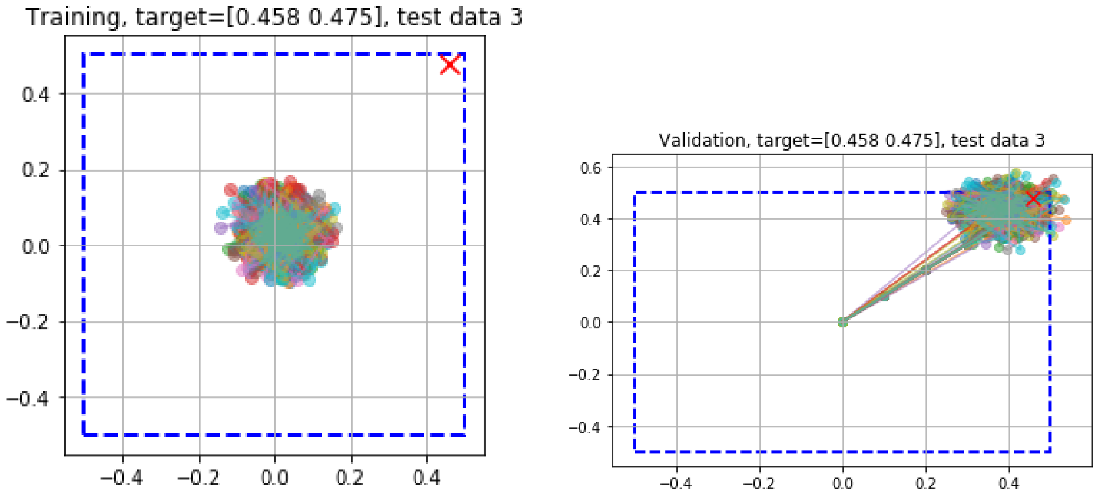
D.4.1 Hyperparameters

We borrow the architecture and inner loop hyperparameters from Javed and White (2019). For the inner meta-steps, we use a step size of 0.2. Due to computational constraints, we use only one trajectory each as query and support task and sample one family for each outer update. For each trajectory, we use the same number of tasks (10) and minibatches (40) per task as Javed and White (2019). The frequencies, amplitudes, and phases are sampled uniformly from the ranges $[1.0, 3.0]$,$[0.1, 5.0]$, $[0.0, \pi]$, respectively.



(a) (b) (c)

Figure D.2: Effect on MAML init of (a) task-finetuning (b) meta-update (TRPO) + task-finetuning (c) meta-update (grad) + task-finetuning.



(a) (b)

Figure D.3: Effect on MAML init of (a) 1 gradient meta-tuning step (b) additionally a goal task fine-tuning step.

Bibliography

Abadi, Martin, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. (2016). “Tensorflow: a system for large-scale machine learning.” In: *OSDI*. Vol. 16, pp. 265–283.

Akhtar, Saghir, Semir Vranic, Farhan Sachal Cyprian, and Ala-Eddin Al Moustafa (2018). “Epstein–Barr virus in gliomas: cause, association, or artifact?” In: *Frontiers in oncology* 8, p. 123.

Anonymous (2019). *ICLR 2019 Workshop RML Paper3 AnonReviewer1*. Openreview. URL: https://openreview.net/forum?id=BJx0N2I6IN&noteId=r1gSl4pJFV.

Antoniou, Antreas, Harrison Edwards, and Amos Storkey (2019). “How to train your MAML”. In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HJGven05Y7.

Antoniou, Antreas and Amos Storkey (2019). “Assume, Augment and Learn: Unsupervised Few-Shot Meta-Learning via Random Labels and Data Augmentation”. In: *arXiv preprint arXiv:1902.09884*.

Antoniou, Antreas, Amos Storkey, and Harrison Edwards (2018). *Data Augmentation Generative Adversarial Networks*. arXiv: 1711.04340 `[stat.ML]`.

Arjovsky, Martin, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz (2019). “Invariant risk minimization”. In: *arXiv preprint arXiv:1907.02893*.

Bakr, Shaimaa, Olivier Gevaert, Sebastian Echegaray, Kelsey Ayers, Mu Zhou, Majid Shafiq, Hong Zheng, Jalen Anthony Benson, Weiruo Zhang, Ann NC Leung, et al. (2018). “A radiogenomic dataset of non-small cell lung cancer”. In: *Scientific data* 5.1, pp. 1–9.

Barrett, Tanya, Dennis B Troup, Stephen E Wilhite, Pierre Ledoux, Dmitry Rudnev, Carlos Evangelista, Irene F Kim, Alexandra Soboleva, Maxim Tomashevsky, Kimberly A Marshall, et al. (2009). “NCBI GEO: archive for high-throughput functional genomic data”. In: *Nucleic acids research* 37.suppl_1, pp. D885–D890.

Basri, Ronen and David W Jacobs (2003). “Lambertian reflectance and linear subspaces”. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 2, pp. 218–233.

Bertinetto, Luca, Joao F. Henriques, Philip Torr, and Andrea Vedaldi (2019). “Meta-learning with differentiable closed-form solvers”. In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HyxnZh0ct7.

Bjorck, Ake and Gene H Golub (1973). “Numerical methods for computing angles between linear subspaces”. In: *Mathematics of computation* 27.123, pp. 579–594.

Bommasani, Rishi, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. (2021). “On the opportunities and risks of foundation models”. In: *arXiv preprint arXiv:2108.07258*.

Brand, Amy, Liz Allen, Micah Altman, Marjorie Hlava, and Jo Scott (2015). “Beyond authorship: attribution, contribution, collaboration, and credit”. In: *Learned Publishing* 28.2, pp. 151–155.

Brennan, K, JL Koenig, AJ Gentles, JB Sunwoo, and O Gevaert (2017). “Identification of an atypical etiological head and neck squamous carcinoma subtype featuring the CpG island methylator phenotype”. In: *EBioMedicine* 17, pp. 223–236.

Brier, Glenn W (1950). “Verification of forecasts expressed in terms of probability”. In: *Monthly weather review* 78.1, pp. 1–3.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. (2020). “Language models are few-shot learners”. In: *Advances in neural information processing systems* 33, pp. 1877–1901.

Ceccarelli, Michele, Floris P Barthel, Tathiane M Malta, Thais S Sabedot, Sofie R Salama, Bradley A Murray, Olena Morozova, Yulia Newton, Amie Radenbaugh, Stefano M Pagnotta, et al. (2016). “Molecular profiling reveals biologically discrete subsets and pathways of progression in diffuse glioma”. In: *Cell* 164.3, pp. 550–563.

Chaudhary, Kumardeep, Olivier B Poirion, Liangqun Lu, and Lana X Garmire (2018). “Deep learning–based multi-omics integration robustly predicts survival in liver cancer”. In: *Clinical Cancer Research* 24.6, pp. 1248–1259.

Cheerla, Anika and Olivier Gevaert (2019). “Deep learning with multimodal representation for pancancer prognosis prediction”. In: *Bioinformatics* 35.14, pp. i446–i454.

Chen, Da, Yuefeng Chen, Yuhong Li, Feng Mao, Yuan He, and Hui Xue (2019). “Self-Supervised Learning For Few-Shot Image Classification”. In: *arXiv preprint arXiv:1911.06045*.

Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton (2020). “A Simple Framework for Contrastive Learning of Visual Representations”. In: *arXiv preprint arXiv:2002.05709*.

Chen, Wei-Yu, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang (2019). “A closer look at few-shot classification”. In: *arXiv preprint arXiv:1904.04232*.

Chen, Xi, Chun Xie, Xing-Xing Fan, Ze-Bo Jiang, Vincent Kam-Wai Wong, Jia-Hui Xu, Xiao-Jun Yao, Liang Liu, and Elaine Lai-Han Leung (2017). “Novel direct AMPK activator suppresses non-small cell lung cancer through inhibition of lipid metabolism”. In: *Oncotarget* 8.56, p. 96089.

Chen, Yanda, Ruiqi Zhong, Sheng Zha, George Karypis, and He He (2022). “Meta-learning via Language Model In-context Tuning”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*. Ed. by Smaranda Muresan, Preslav Nakov, and Aline Villavicencio. Association for Computational Linguistics, pp. 719–730. DOI: 10.18653/V1/2022.ACL-LONG.53. URL: https://doi.org/10.18653/v1/2022.acl-long.53.

Chen, Yinbo, Xiaolong Wang, Zhuang Liu, Huijuan Xu, and Trevor Darrell (2020). "A New Meta-Baseline for Few-Shot Learning". In: *arXiv preprint arXiv:2003.04390*.

Chi, Chih-Lin, W Nick Street, and William H Wolberg (2007). "Application of artificial neural network-based survival analysis on two breast cancer datasets". In: *AMIA Annual Symposium Proceedings*. Vol. 2007. American Medical Informatics Association, p. 130.

Ching, Travers, Xun Zhu, and Lana X Garmire (2018). "Cox-nnet: an artificial neural network method for prognosis prediction of high-throughput omics data". In: *PLoS computational biology* 14.4, e1006076.

Codella, Noel, Veronica Rotemberg, Philipp Tschandl, M Emre Celebi, Stephen Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael Marchetti, et al. (2019). "Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC)". In: *arXiv preprint arXiv:1902.03368*.

Cox, David R (1972). "Regression models and life-tables". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 34.2, pp. 187–202.

Croft, David, Antonio Fabregat Mundo, Robin Haw, Marija Milacic, Joel Weiser, Guanming Wu, Michael Caudy, Phani Garapati, Marc Gillespie, Maulik R Kamdar, et al. (2014). "The Reactome pathway knowledgebase". In: *Nucleic acids research* 42.D1, pp. D472–D477.

Cubuk, Ekin D., Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le (2019). "AutoAugment: Learning Augmentation Strategies From Data". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 113–123.

Deleu, Tristan, Tobias Würfl, Mandana Samiei, Joseph Paul Cohen, and Yoshua Bengio (2019). *Torchmeta: A Meta-Learning library for PyTorch*. Available at: https://github.com/tristandeleu/pytorch-meta. URL: https://arxiv.org/abs/1909.06576.

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.

Devos, Arnout, Sylvain Chatel, and Matthias Grossglauser (May 2019). "[Re] Meta-learning with differentiable closed-form solvers". In: *ReScience C Journal* 5.2. #1. DOI: 10.5281/zenodo.3160540.

Devos, Arnout and Yatin Dandi (Dec. 2021). "Model-Agnostic Learning to Meta-Learn". In: *Proceedings of Machine Learning Research*. Vol. 148. PMLR, pp. 155–175. URL: https://proceedings.mlr.press/v148/devos21a.html.

Devos, Arnout and Matthias Grossglauser (June 2020). "Regression Networks for Meta-Learning Few-Shot Classification". In: *ICML 2020 Workshop on Automated Machine Learning (AutoML)*. URL: https://sites.google.com/view/automl2020/accepted-papers_1.

Devos, Arnout and Julia Wagner (2023). *ETH-EPFL Sciencepreneurship Summer School 2023*. EPFL and ETH Zurich. URL: https://sciencepreneurship.ch/summer-school-2023.

DeVries, Terrance and Graham W Taylor (2017). "Improved Regularization of Convolutional Neural Networks With Cutout". In: *arXiv preprint arXiv:1708.04552*.

Duan, Yan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel (2016). "RL2: Fast reinforcement learning via slow reinforcement learning. arXiv, 2016". In: *URL https://arxiv. org/abs/1611.02779*.

Esteva, Andre, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun (2017). "Dermatologist-level classification of skin cancer with deep neural networks". In: *nature* 542.7639, pp. 115–118.

Farahani, Abolfazl, Sahar Voghoei, Khaled Rasheed, and Hamid R Arabnia (2021). "A brief review of domain adaptation". In: *Advances in data science and information engineering: proceedings from ICDATA 2020 and IKE 2020*, pp. 877–894.

Fei-Fei, Li, Rob Fergus, and Pietro Perona (2006). "One-Shot Learning of Object Categories". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.4, pp. 594–611.

Finn, Chelsea, Pieter Abbeel, and Sergey Levine (2017). "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pp. 1126–1135.

Finn, Chelsea, Kelvin Xu, and Sergey Levine (2018). "Probabilistic model-agnostic meta-learning". In: *Advances in Neural Information Processing Systems*, pp. 9516–9527.

Fitzgibbon, Andrew W and Andrew Zisserman (2003). "Joint manifold distance: a new approach to appearance based clustering". In: *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.* Vol. 1. IEEE, pp. I–I.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani (2001). *The elements of statistical learning*. Vol. 1. 10. Springer series in statistics New York.

Garcia, Victor and Joan Bruna (Nov. 2017). "Few-Shot Learning with Graph Neural Networks". en. In: *arXiv:1711.04043 [cs, stat]*. arXiv: 1711.04043. URL: http://arxiv.org/abs/1711.04043 (visited on 12/03/2018).

Gidaris, Spyros, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord (2019). "Boosting Few-Shot Visual Learning with Self-Supervision". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 8059–8068.

Goeman, Jelle J (2010). "L1 penalized estimation in the Cox proportional hazards model". In: *Biometrical journal* 52.1, pp. 70–84.

Guo, Yunhui, Noel CF Codella, Leonid Karlinsky, John R Smith, Tajana Rosing, and Rogerio Feris (2019). "A New Benchmark for Evaluation of Cross-Domain Few-Shot Learning". In: *arXiv preprint arXiv:1912.07200*.

Han, Dong, Shao-Jun Li, Yan-Ting Zhu, Lu Liu, and Man-Xiang Li (2013). "LKB1 / AMPK / mTOR signaling pathway in non-small-cell lung cancer". In: *Asian Pacific Journal of Cancer Prevention* 14.7, pp. 4033–4039.

Harrell, Frank E, Robert M Califf, David B Pryor, Kerry L Lee, and Robert A Rosati (1982). “Evaluating the yield of medical tests”. In: *Jama* 247.18, pp. 2543–2546.

He, Kaiming, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick (2019). “Momentum Contrast for Unsupervised Visual Representation Learning”. In: *arXiv preprint arXiv:1911.05722*.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034.

– (2016). “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Hee, Siew Wan, Adrian Willis, Catrin Tudur Smith, Simon Day, Frank Miller, Jason Madan, Martin Posch, Sarah Zohar, and Nigel Stallard (2017). “Does the low prevalence affect the sample size of interventional clinical trials of rare diseases? An analysis of data from the aggregate analysis of clinicaltrials. gov”. In: *Orphanet journal of rare diseases* 12.1, p. 44.

Helber, Patrick, Benjamin Bischke, Andreas Dengel, and Damian Borth (2019). “Eurosat: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 12.7, pp. 2217–2226.

Herbst, Roy S and Scott M Lippman (2007). “Molecular signatures of lung cancer–toward personalized therapy”. In: *New England Journal of Medicine* 356.1, pp. 76–78.

Hilliard, Nathan, Lawrence Phillips, Scott Howland, Artm Yankov, Courtney D. Corley, and Nathan O. Hodas (2018). “Few-Shot Learning with Metric-Agnostic Conditional Embeddings”. In: *arXiv preprint arXiv:1802.04376*.

Hosmer, David W, Stanley Lemeshow, and S May (2008). *Applied survival analysis: regression modeling of time to event data (Wiley series in probability and statistics)*.

Hsu, Kyle, Sergey Levine, and Chelsea Finn (2019). “Unsupervised Learning via Meta-Learning”. In: *ICLR 2019 : 7th International Conference on Learning Representations*.

Hu, Edward J, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen (2022). “LoRA: Low-Rank Adaptation of Large Language Models”. In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=nZeVKeeFYf9.

Hunen, Jeroen van (2018). *EPFLinnovators doctoral program*. https://web.archive.org/web/20190308024641/https://www.epfl.ch/education/phd/doctoral-studies-structure/customized-curricula/epflinnovators/. Accessed: 2019. EPFL.

Ishwaran, Hemant, Udaya B Kogalur, Eugene H Blackstone, Michael S Lauer, et al. (2008). “Random survival forests”. In: *The annals of applied statistics* 2.3, pp. 841–860.

Javed, Khurram and Martha White (2019). “Meta-Learning Representations for Continual Learning”. In: *Advances in Neural Information Processing Systems*.

Ji, Zilong, Xiaolong Zou, Tiejun Huang, and Si Wu (2019). “Unsupervised Few-shot Learning via Self-supervised Training”. In: *arXiv preprint arXiv:1912.12178*.

Kanehisa, Minoru and Susumu Goto (2000). “KEGG: kyoto encyclopedia of genes and genomes”. In: *Nucleic acids research* 28.1, pp. 27–30.

Kaplanis, Christos, Claudia Clopath, and Murray Shanahan (2020). *Continual Reinforcement Learning with Multi-Timescale Replay.* arXiv: 2004.07530 `[cs.LG]`.

Khodadadeh, Siavash, Ladislau Boloni, and Mubarak Shah (2019). “Unsupervised Meta-Learning for Few-Shot Image Classification”. In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 10132–10142.

Kim, Dong Wook, Sanghoon Lee, Sunmo Kwon, Woong Nam, In-Ho Cha, and Hyung Jun Kim (2019). “Deep learning-based survival prediction of oral cancer patients”. In: *Scientific reports* 9.1, pp. 1–10.

Kingma, Diederik P. and Jimmy Ba (2015). “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.* Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1412.6980.

Kingma, Diederik P. and Jimmy Lei Ba (2015). “Adam: A Method for Stochastic Optimization”. In: *ICLR 2015 : International Conference on Learning Representations 2015.*

Klein, John P and Melvin L Moeschberger (2006). *Survival analysis: techniques for censored and truncated data.* Springer Science & Business Media.

Kleinbaum, David G and Mitchel Klein (2012). “The Cox proportional hazards model and its characteristics”. In: *Survival analysis.* Springer, pp. 97–159.

Koç, Mehmet and Atalay Barkana (2014). “Application of Linear Regression Classification to low-dimensional datasets”. In: *Neurocomputing* 131, pp. 331–335.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). “Imagenet Classification With Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1097–1105.

Lake, Brenden, Ruslan Salakhutdinov, Jason Gross, and Joshua Tenenbaum (2011). “One shot learning of simple visual concepts”. In: *Proceedings of the Annual Meeting of the Cognitive Science Society.* Vol. 33. 33.

Lake, Brenden M, Ruslan Salakhutdinov, and Joshua B Tenenbaum (2015). “Human-level concept learning through probabilistic program induction”. In: *Science* 350.6266, pp. 1332–1338.

Lee, Hae Beom, Hayeon Lee, Donghyun Na, Saehoon Kim, Minseop Park, Eunho Yang, and Sung Ju Hwang (2020). “Learning to Balance: Bayesian Meta-Learning for Imbalanced and Out-of-distribution Tasks”. In: *International Conference on Learning Representations.* URL: https://openreview.net/forum?id=rkeZIJBYvr.

Li, Aoxue, Tiange Luo, Zhiwu Lu, Tao Xiang, and Liwei Wang (2019). “Large-scale few-shot learning: Knowledge transfer with class hierarchy”. In: *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pp. 7212–7220.

Li, Jingjing, Mengmeng Jing, Ke Lu, Lei Zhu, Yang Yang, and Zi Huang (2019). “From zero-shot learning to cold-start recommendation”. In: *Proceedings of the AAAI conference on artificial intelligence.* Vol. 33. 01, pp. 4189–4196.

Li, Junnan, Pan Zhou, Caiming Xiong, Richard Socher, and Steven CH Hoi (2020). “Prototypical Contrastive Learning of Unsupervised Representations”. In: *arXiv preprint arXiv:2005.04966*.

Li, Ke and Jitendra Malik (2017). “Learning to Optimize”. In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=ry4Vrt5gl.

Li, Yan, Lu Wang, Jie Wang, Jieping Ye, and Chandan K Reddy (2016). “Transfer learning for survival analysis via efficient L2, 1-norm regularized Cox regression”. In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, pp. 231–240.

Li, Zhenguo, Fengwei Zhou, Fei Chen, and Hang Li (2017). “Meta-sgd: Learning to learn quickly for few-shot learning”. In: *arXiv preprint arXiv:1707.09835*.

Licina, Veronika Foldvary (2021). *EPFLinnovators: Arnout Devos*. EPFL. URL: https://actu.epfl.ch/news/epflinnovators-arnout-devos/.

Liu, Hong, Jeff Z HaoChen, Adrien Gaidon, and Tengyu Ma (2021). “Self-supervised learning is more robust to dataset imbalance”. In: *arXiv preprint arXiv:2110.05025*.

Liu, Shikun, Andrew Davison, and Edward Johns (2019). “Self-Supervised Generalisation with Meta-Auxiliary Learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1677–1687.

Louis, David N, Arie Perry, Guido Reifenberger, Andreas Von Deimling, Dominique Figarella-Branger, Webster K Cavenee, Hiroko Ohgaki, Otmar D Wiestler, Paul Kleihues, and David W Ellison (2016). “The 2016 World Health Organization classification of tumors of the central nervous system: a summary”. In: *Acta neuropathologica* 131.6, pp. 803–820.

Lu, Bin, Xiao Bing Chen, Mei Dan Ying, Qiao Jun He, Ji Cao, and Bo Yang (2018). “The role of ferroptosis in cancer development and treatment response”. In: *Frontiers in pharmacology* 8, p. 992.

Luck, Margaux, Tristan Sylvain, Héloı¨se Cardinal, Andrea Lodi, and Yoshua Bengio (2017). “Deep learning for patient-specific kidney graft survival analysis”. In: *arXiv preprint arXiv:1705.10245*.

Maaten, Laurens van der and Geoffrey Hinton (2008). “Visualizing data using t-SNE”. In: *Journal of machine learning research* 9.Nov, pp. 2579–2605.

Manning, Christopher (2020). *Artificial Intelligence Definitions*. https://hai.stanford.edu/sites/default/files/2020-09/AI-Definitions-HAI.pdf.

Mantovani, Alberto, Isabella Barajon, and Cecilia Garlanda (2018). “IL-1 and IL-1 regulatory pathways in cancer progression and therapy”. In: *Immunological reviews* 281.1, pp. 57–61.

Martin, D and JS Gutkind (2008). “Human tumor-associated viruses and new insights into the molecular mechanisms of cancer”. In: *Oncogene* 27.2, S31–S42.

Medina, Carlos, Arnout Devos, and Matthias Grossglauser (June 2020). “Self-supervised prototypical transfer learning for few-shot classification”. In: *ICML 2020 Workshop on Automated Machine Learning (AutoML)*. URL: https://sites.google.com/view/automl2020/accepted-papers_1.

Mhaskar, Hrushikesh, Qianli Liao, and Tomaso Poggio (2016). "Learning functions: when is deep better than shallow". In: *arXiv preprint arXiv:1603.00988*.

Mishra, Nikhil, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel (July 2017). "A Simple Neural Attentive Meta-Learner". en. In: *arXiv:1707.03141 [cs, stat]*. arXiv: 1707.03141. URL: http://arxiv.org/abs/1707.03141 (visited on 12/03/2018).

Mobadersany, Pooya, Safoora Yousefi, Mohamed Amgad, David A Gutman, Jill S Barnholtz-Sloan, José E Velázquez Vega, Daniel J Brat, and Lee AD Cooper (2018). "Predicting cancer outcomes from histology and genomics using convolutional networks". In: *Proceedings of the National Academy of Sciences* 115.13, E2970–E2979.

Mohanty, Sharada P, David P Hughes, and Marcel Salathé (2016). "Using Deep Learning for Image-Based Plant Disease Detection". In: *Frontiers in Plant Science* 7, p. 1419.

Nair, Vinod and Geoffrey E Hinton (2010). "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814.

Naseem, Imran, Roberto Togneri, and Mohammed Bennamoun (2010). "Linear regression for face recognition". In: *IEEE transactions on pattern analysis and machine intelligence* 32.11, pp. 2106–2112.

Network, Cancer Genome Atlas et al. (2012). "Comprehensive molecular characterization of human colon and rectal cancer". In: *Nature* 487.7407, p. 330.

Nichol, Alex, Joshua Achiam, and John Schulman (2018). "On first-order meta-learning algorithms". In: *arXiv preprint arXiv:1803.02999*.

Ochal, Mateusz, Massimiliano Patacchiola, Jose Vazquez, Amos Storkey, and Sen Wang (2023). "Few-shot learning with class imbalance". In: *IEEE Transactions on Artificial Intelligence*.

Oh, Jaehoon, Hyungjun Yoo, ChangHwan Kim, and Se-Young Yun (2021). "{BOIL}: Towards Representation Change for Few-shot Learning". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=umIdUL8rMH.

Oliver, Avital, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow (2018). "Realistic Evaluation of Deep Semi-Supervised Learning Algorithms". In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 3235–3246.

Oreshkin, Boris, Pau Rodrı´guez López, and Alexandre Lacoste (2018). "Tadam: Task dependent adaptive metric for improved few-shot learning". In: *Advances in Neural Information Processing Systems*, pp. 721–731.

Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. (2022). "Training language models to follow instructions with human feedback". In: *Advances in Neural Information Processing Systems* 35, pp. 27730–27744.

Park, Mee Young and Trevor Hastie (2007). "L1-regularization path algorithm for generalized linear models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 69.4, pp. 659–677.

Park, Sang Tae and Jayoung Kim (2016). “Trends in next-generation sequencing and a new era for whole genome sequencing”. In: *International neurourology journal* 20.Suppl 2, S76.

Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer (2017). “Automatic differentiation in pytorch”. In.

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library.* arXiv: `1912.01703 [cs.LG]`.

Peng, Jie-Wen, Dong-Ying Liu, Gui-Nan Lin, JJ Xiao, and Zhong-Jun Xia (2015). “Hepatitis B virus infection is associated with poor prognosis in patients with advanced non small cell lung cancer”. In: *Asian Pac J Cancer Prev* 16.13, pp. 5285–5288.

Petalidis, Lawrence P, Anastasis Oulas, Magnus Backlund, Matthew T Wayland, Lu Liu, Karen Plant, Lisa Happerfield, Tom C Freeman, Panayiota Poirazi, and V Peter Collins (2008). “Improved grading and survival prediction of human astrocytic brain tumors by artificial neural network analysis of gene expression microarray data”. In: *Molecular cancer therapeutics* 7.5, pp. 1013–1024.

Platten, Michael, Ellen AA Nollen, Ute F Röhrig, Francesca Fallarino, and Christiane A Opitz (2019). “Tryptophan metabolism as a common therapeutic target in cancer, neurodegeneration and beyond”. In: *Nature Reviews Drug Discovery* 18.5, pp. 379–401.

Platten, Michael, Wolfgang Wick, and Benoıˆt J Van den Eynde (2012). “Tryptophan catabolism in cancer: beyond IDO and tryptophan depletion”. In: *Cancer research* 72.21, pp. 5435–5440.

Pratt, Lorien Y (1993). “Discriminability-based transfer between neural networks”. In: *Advances in neural information processing systems*, pp. 204–211.

Qiao, Siyuan, Chenxi Liu, Wei Shen, and Alan L Yuille (2018). “Few-shot image recognition by predicting parameters from activations”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7229–7238.

Qin, Tiexin, Wenbin Li, Yinghuan Shi, and Yang Gao (2020). “Unsupervised Few-shot Learning via Distribution Shift-based Augmentation”. In: *arXiv preprint arXiv:2004.05805*.

Qiu, Yeping Lina, Hong Zheng, Arnout Devos, Heather Selby, and Olivier Gevaert (Dec. 2020). “A meta-learning approach for genomic survival analysis”. In: *Nature communications* 11.1, p. 6350. URL: https://www.nature.com/articles/s41467-020-20167-3.

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. (2021). “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PMLR, pp. 8748–8763.

Raghu, Aniruddh, Maithra Raghu, Samy Bengio, and Oriol Vinyals (2020). "Rapid Learning or Feature Reuse? Towards Understanding the Effectiveness of MAML". In: *International Conference on Learning Representations.* URL: https://openreview.net/forum?id=rkgMkCEtPB.

Rajeswaran, Aravind, Chelsea Finn, Sham Kakade, and Sergey Levine (2019). "Meta-Learning with Implicit Gradients". In: *Advances in Neural Information Processing Systems.*

Ramesh, Aditya, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever (2021). "Zero-shot text-to-image generation". In: *International Conference on Machine Learning.* PMLR, pp. 8821–8831.

Ravi, Sachin and Hugo Larochelle (2017). "Optimization as a Model for Few-Shot Learning". In: *ICLR 2017 : International Conference on Learning Representations 2017.*

Ren, Mengye, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel (2018). "Meta-Learning for Semi-Supervised Few-Shot Classification". In: *Proceedings of 6th International Conference on Learning Representations ICLR.*

Requeima, James, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner (2019). "Fast and flexible multi-task classification using conditional neural adaptive processes". In: *Advances in Neural Information Processing Systems* 32.

Rizzo, Roberta (2020). "Controversial role of herpesviruses in Alzheimers disease". In: *PLoS pathogens* 16.6, e1008575.

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, and Michael Bernstein (2015a). "Imagenet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision* 115.3, pp. 211–252.

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei (2015b). "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3, pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

Russell, Stuart J and Peter Norvig (2010). *Artificial intelligence a modern approach.* London.

Santoro, Adam, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy P. Lillicrap (2016). "One-shot Learning with Memory-Augmented Neural Networks". In: *arXiv preprint arXiv:1605.06065.*

Schulman, John, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz (2015). "Trust region policy optimization". In: *International conference on machine learning.* PMLR, pp. 1889–1897.

Sergushichev, Alexey (2016). "An algorithm for fast preranked gene set enrichment analysis using cumulative statistic calculation". In: *BioRxiv*, p. 060012.

Silver, David, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587, pp. 484–489.

Simon, Christian, Piotr Koniusz, and Mehrtash Harandi (2019). *Projective Subspace Networks For Few-Shot Learning.* URL: https://openreview.net/forum?id=rkzfuiA9F7.

Slenter, Denise N, Martina Kutmon, Kristina Hanspers, Anders Riutta, Jacob Windsor, Nuno Nunes, Jonathan Mélius, Elisa Cirillo, Susan L Coort, Daniela Digles, et al. (2018). "WikiPathways: a multifaceted pathway database bridging metabolomics to other omics research". In: *Nucleic acids research* 46.D1, pp. D661–D667.

Snell, Jake, Kevin Swersky, and Richard Zemel (2017). "Prototypical Networks for Few-Shot Learning". In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 4077–4087.

Su, Jong-Chyi, Subhransu Maji, and Bharath Hariharan (2019). "When Does Self-Supervision Improve Few-Shot Learning?" In: *arXiv preprint arXiv:1910.03560.*

Sung, Flood, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales (2018). "Learning to compare: Relation network for few-shot learning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1199–1208.

Tian, Yonglong, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola (2020). "What makes for good views for contrastive learning". In: *arXiv preprint arXiv:2005.10243.*

Tomczak, Katarzyna, Patrycja Czerwińska, and Maciej Wiznerowicz (2015). "The Cancer Genome Atlas (TCGA): an immeasurable source of knowledge". In: *Contemporary oncology* 19.1A, A68.

Tonella, Luca, Marco Giannoccaro, Salvatore Alfieri, Silvana Canevari, and Loris De Cecco (2017). "Gene expression signatures for head and neck cancer patient stratification: are results ready for clinical application?" In: *Current treatment options in oncology* 18.5, p. 32.

Triantafillou, Eleni, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle (2020). "Meta-Dataset: A Dataset of Datasets for Learning to Learn from Few Examples". In: *ICLR 2020 : Eighth International Conference on Learning Representations.*

Tschandl, Philipp, Cliff Rosendahl, and Harald Kittler (2018). "The HAM10000 Dataset, a Large Collection of Multi-Source Dermatoscopic Images of Common Pigmented Skin Lesions". In: *Scientific data* 5, p. 180161.

Varn, Frederick S, Evelien Schaafsma, Yue Wang, and Chao Cheng (2018). "Genomic characterization of six virus-associated cancers identifies changes in the tumor immune microenvironment and altered genetic programs". In: *Cancer research* 78.22, pp. 6413–6423.

Vilalta, Ricardo and Youssef Drissi (2002). “A perspective view and survey of meta-learning”. In: *Artificial intelligence review* 18.2, pp. 77–95.

Vinyals, Oriol, Charles Blundell, Timothy Lillicrap, and Daan Wierstra (2016). “Matching Networks for One Shot Learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 3630–3638.

Vuorio, Risto, Shao-Hua Sun, Hexiang Hu, and Joseph J Lim (2019). “Multimodal Model-Agnostic Meta-Learning via Task-Aware Modulation”. In: *Advances in Neural Information Processing Systems*, pp. 1–12.

Wah, Catherine, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie (2011). “The caltech-ucsd birds-200-2011 dataset”. In.

Wang, Xiaosong, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers (2017). “Chestx-ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2097–2106.

Welinder, P., S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona (2010). *Caltech-UCSD Birds 200.* Tech. rep. CNS-TR-2010-001. California Institute of Technology.

Wong, Kin Yau, Cheng Fan, Maki Tanioka, Joel S Parker, Andrew B Nobel, Donglin Zeng, Dan-Yu Lin, and Charles M Perou (2018). “An integrative boosting approach for predicting survival time with multiple genomics platforms”. In: *bioRxiv*, p. 338145.

Wu, Chen-Yi, Hsiao-Yun Hu, Cheng-Yun Pu, Nicole Huang, Hsi-Che Shen, Chung-Pin Li, and Yiing-Jeng Chou (2011). “Pulmonary tuberculosis increases the risk of lung cancer: a population-based cohort study”. In: *Cancer* 117.3, pp. 618–624.

Xing, Chen, Negar Rostamzadeh, Boris N. Oreshkin, and Pedro O. Pinheiro (2019). “Adaptive Cross-Modal Few-Shot Learning”. In: *CoRR* abs/1902.07104. arXiv: 1902.07104. URL: http://arxiv.org/abs/1902.07104.

Yang, Huayi, Deqing Wang, Zhengyang Zhao, and Xuying Wang (2022). “SSL-DC: Improving Transductive Few-Shot Learning via Self-Supervised Learning and Distribution Calibration”. In: *2022 26th International Conference on Pattern Recognition (ICPR)*, pp. 4892–4898. URL: https://api.semanticscholar.org/CorpusID:254098676.

Yang, Shuo, Lu Liu, and Min Xu (2021). “Free Lunch for Few-shot Learning: Distribution Calibration”. In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=JWOiYxMG92s.

Yao, Huaxiu, Ying Wei, Junzhou Huang, and Zhenhui Li (2019). “Hierarchically Structured Meta-learning”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pp. 1126–1135.

Yao, Huaxiu, Xian Wu, Zhiqiang Tao, Yaliang Li, Bolin Ding, Ruirui Li, and Zhenhui Li (2020). “Automated Relational Meta-learning”. In: *International Conference on Learning Representations (ICLR)*.

Ye, Mang, Xu Zhang, Pong C Yuen, and Shih-Fu Chang (2019). “Unsupervised Embedding Learning via Invariant and Spreading Instance Feature”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6210–6219.

Yoon, Jaesik, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn (2018). “Bayesian model-agnostic meta-learning”. In: *Advances in Neural Information Processing Systems*, pp. 7332–7342.

Yosinski, Jason, Jeff Clune, Yoshua Bengio, and Hod Lipson (2014). “How transferable are features in deep neural networks?” In: *Advances in neural information processing systems* 27.

Yousefi, Safoora, Fatemeh Amrollahi, Mohamed Amgad, Chengliang Dong, Joshua E Lewis, Congzheng Song, David A Gutman, Sameer H Halani, Jose Enrique Velazquez Vega, Daniel J Brat, et al. (2017). “Predicting clinical outcomes from large scale cancer genomic profiles with deep survival models”. In: *Scientific reports* 7.1, pp. 1–11.

Yu, Yang-Hao, Chien-Chang Liao, Wu-Huei Hsu, Hung-Jen Chen, Wei-Chih Liao, Chih-Hsin Muo, Fung-Chang Sung, and Chih-Yi Chen (2011). “Increased lung cancer risk among patients with pulmonary tuberculosis: a population cohort study”. In: *Journal of Thoracic Oncology* 6.1, pp. 32–37.

Zamir, Amir, Alexander Sax, Teresa Yeo, Ouzhan Kar, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas Guibas (2020). *Robust Learning Through Cross-Task Consistency.* arXiv: 2006.04096 `[cs.CV]`.

Zapatka, Marc, Ivan Borozan, Daniel S Brewer, Murat Iskar, Adam Grundhoff, Malik Alawi, Nikita Desai, Holger Sültmann, Holger Moch, Colin S Cooper, et al. (2020). “The landscape of viral associations in human cancers”. In: *Nature genetics* 52.3, pp. 320–330.

Zhang, Marvin, Henrik Marklund, Nikita Dhawan, Abhishek Gupta, Sergey Levine, and Chelsea Finn (2021). “Adaptive risk minimization: Learning to adapt to domain shift”. In: *Advances in Neural Information Processing Systems* 34, pp. 23664–23678.

Zhe, Xuefei, Shifeng Chen, and Hong Yan (2019). “Directional statistics-based deep metric learning for image classification and retrieval”. In: *Pattern Recognition* 93, pp. 113–123.

Zhong, Zhun, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang (2020). “Random Erasing Data Augmentation”. In: *AAAI 2020 : The Thirty-Fourth AAAI Conference on Artificial Intelligence.*

Arnout Devos

Last update: February 28, 2024,
Web: arnoutdevos.github.io,

Experience

Swiss Federal Institute of Technology Lausanne (EPFL) · PhD Candidate Sep 2018 - now
Designed innovative machine (transfer) learning algorithms for settings with small data. *Lausanne, Switzerland*

Sciencepreneurship Summer School · Co-Founder Spring 2023
Led: program (20+ speakers), 7 partnerships ($40k+), marketing (30/110 admitted). *Zurich, Switzerland*

Spotify · PhD Research Intern Summer 2022
Cold-start recommendation of new content to users. *London, UK*

Amazon · PhD Research Intern Summer 2021
Conditional generative modeling on drone sensor data. *Remote*

Stanford University · PhD Research Intern Summer 2018
Developed a meta-learning approach for survival prediction. *Nature Communications 2020.* *Stanford, USA*

USC · Research Intern Fall 2017
Generated imitation learning policies from a reinforcement learned expert in MuJoCo. *Los Angeles, USA*

EPFL · Research Intern Summer 2016
Invented a low-power UWB architecture for brain implants. *IEEE TCAS-II 2018.* *Lausanne, Switzerland*

Academics for Companies · Event Coordinator 2016 - 2017
Organized the 1-day *Student Startup Forum 2017* with 600+ student attendees. *Leuven, Belgium*

Siemens · Software Engineering Intern Summer 2013
Implemented vehicle axle angle measurement papers and performed analysis in Matlab. *Leuven, Belgium*

Education

Swiss Federal Institute of Technology Lausanne (EPFL) · PhD Sep 2018 - now
Ph.D. in Artificial Intelligence. *EPFLinnovators* entrepreneurial track. *Lausanne, Switzerland*

University of Southern California (USC) · MS 2017 - 2018
M.S. in Computer Science, class top 7/1000+ *Los Angeles, USA*

University of Leuven (KU Leuven) · BS & MS^2 2011 - 2017
M.S. in Electrical Engineering, *Summa Cum Laude*, class rank 1/54 *Leuven, Belgium*
M.S. in Management, *Cum Laude*
B.S. in Electrical Engineering & Computer Science, *Cum Laude*

Skills

Computer Skills	Python, Numpy, PyTorch, TensorFlow, Kubernetes, Git, Java, Matlab, SQL, HTML, LaTeX
Languages	English (Fluent, TOEFL 112/120), French (Conversational), German (Basic), Dutch (Native)

Mentorship & Supervision

Alex Chin *('22, Harvard → Jane Street)*, **Yatin Dandi** *('20/'21, IIT Kanpur/EPFL → ELLIS PhD Fellow at EPFL & MPI)*, **Carlos Medina** *('20, EPFL → PhD Bosch/Freiburg)*, **Daniil Dmitriev** *('19, EPFL → ETH AI Center PhD Fellow)*, **Ali Ben Lalah** *('19, EPFL → Apple Zurich)*

Selected Honors

PhD Fellowship, Maria Skłodowska-Curie (Marie-Curie) / EPFLinnovators *(36/1840; $100,000)* 2018 - 2023
Teaching Assistant Award, EPFL Computer and Communication Sciences *(27/287)* 2021
PhD Fellowship, EPFL Computer and Communication Sciences *(50/750; $51,000, declined)* 2018
Phi Kappa Phi PKP Society (top 10% graduate students accross USA) 2018
MS CS Merit Award, USC Computer Science *(7/1000+)* 2018
MS Fellowship, Belgian American Educational Foundation (BAEF) *($75,000)* 2017 - 2018

Teaching Experience

EPFL MGT-413 Entrepreneurship & New Venture Strategy	Guest lecturer ”Investment Memo”, Spring 2023
EPFL COM-308 Internet Analytics	*”proactive & thorough”* - Prof. Grossglauser, Head TA, Spring 2021/2022
EPFL MGT-420 Climate Entrepreneurship	Jury Member & Grader, Fall 2021/2022
Stanford CS106A Code in Place (Python)	TA, Spring 2020; TA Lecturer, Spring 2021
EPFL CS-433 Machine Learning	*”outstanding work”* - Prof. Jaggi, TA, Fall 2020

Selected Peer-reviewed Work

* denotes equal contribution

2021

Devos A.*, Dandi Y.*, *”Model-Agnostic Learning to Meta-Learn”*, NeurIPS 2020 pre-registration workshop & Proceedings of Machine Learning Research (PMLR) 148:155–175, 2021

2020

Qiu L., Zheng H., **Devos A.**, Selby H., Gevaert O. *”A meta-learning approach for genomic survival analysis”*, Nature Communications, Volume 11, 2020.

Devos A., Grossglauser M. *”Regression Networks for Meta-Learning Few-Shot Classification”*, ICML 2020 Workshop on Automated Machine Learning.

Medina C.*, **Devos A.***, Grossglauser M. *”Self-Supervised Prototypical Transfer Learning for Few-Shot Classification”*, ICML 2020 Workshop on Automated Machine Learning.

2019

Devos A.*, Chatel S.*, Grossglauser M. *”[Re] Meta-learning with differentiable closed-form solvers”*, ReScience Journal 2019 (ReScience 2019) & ICLR 2019 Workshop on Reproducibility in Machine Learning.

2018

Devos A., Dhondt J., Stripling E., Baesens B., Broucke S. v., Sukhatme G., *”Profit Maximizing Logistic Regression Modeling for Credit Scoring”*, Proceedings of the IEEE Data Science Workshop (DSW 2018).

Ture K., **Devos A.**, Dehollain C., *”Area and Power Efficient Ultra-Wideband Transmitter Based On Active Inductor”*, in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 65, no. 10, pp. 1325-1329, Oct. 2018.

2016

Devos A., Vigilante M., Reynaert P., *”Multiphase Digitally Controlled Oscillator for Future 5G Phased Arrays in 90 nm CMOS.”* in Proceedings of IEEE NORCAS, pp. 10-14, Copenhagen, 2016.

Service

Reviewer

AISTATS (2022), AutoML (2022), Pre-registration Workshop at NeurIPS (2021), IEEE TPAMI (2021), AutoML Workshop at ICML (2021), Nature Digital Medicine (2021), Swiss Machine Learning day (2019)

Miscellaneous

PhD admissions grader ELLIS (European Lab for Learning and Intelligent Systems) (2021), 1 PhD recommendation letter (2021)