Benchmarking Hierarchical Task Sampling for Few-Shot Multi-Label Domain Adaptation

Joshua Chang[*]
jrchang@stanford.edu

Trenton Chang[*]
tchang97@cs.stanford.edu

Extended Abstract

Land-cover classification has applications in human settlement mapping [Qiu et al., 2020], agricultural yield prediction [Bargiel and Herrmann, 2011], and vegetation monitoring [Tucker et al., 1985]. While previous work focuses on single-label land cover classification, in the real world, land cover is a multi-label problem, as satellite images often contain multiple land types within a single frame. However, multi-label meta-learning is relatively underexplored. Meta-learning is particularly well-suited for multi-label problems with large and sparse label-permutation spaces: while traditional supervised methods require significant amounts of data in each class to learn, meta-learning-based domain adaptation can adeptly handle such sparse label spaces using few-shot learning techniques. Current few-shot meta-learning approaches are not well-defined for few-shot multi-label classification, since for the standard k-shot n-way setup, k and n are no longer trivially well-defined. To combat this, we propose a novel task-definition framework for few-shot multi-label classification based on hierarchical sampling, that first randomly samples a subset of n labels, and subsequently samples training examples to cover those n labels, thus enabling well-defined few-shot multi-label meta-learning. We benchmark on the BigEarthNet [Sumbul et al., 2019] land cover classification task.

We evaluate thoroughly combinations of sampling approaches, multi-label schemes, and meta-learning models in order to evaluate this framework for fitting meta-learning models to the multi-label space. Our task sampling methods are specifically developed to address the multi-label problem and the inherent class imbalances that arise in large multi-label sets. We evaluate two support set sampling methods: *greedy* sampling, which greedily adds training examples to the support set, and *uniform permutation* sampling, which deterministically adds examples with fixed multi-label permutations based on the multi-labels sampled for a particular support set. In addition, we evaluate two multi-label problem transformation techniques: binary relevance transformation [Zhang et al., 2018], which treats each multilabel as a n-way binary classification task, and powerset transformation [Read et al., 2014], which treats each multilabel as a distinct label, yielding a $2^n - 1$ way classification problem. We also perform initial experiments on hyperparameters of our method, namely, the support set size and the label subset size.

Our work demonstrates that this hierarchical sampling-based task framework enables *plug-and-play, effective meta-learning* for multi-label problem. Our paper discusses important considerations for implementing successful meta-learning algorithms tailored to multi-label classification as well as certain challenges that prevent some of the algorithms from performing effectively.

[*]Equal contribution.

CS 330: Deep Multi-Task and Meta Learning, Stanford University, Fall 2020-21. Final course project.

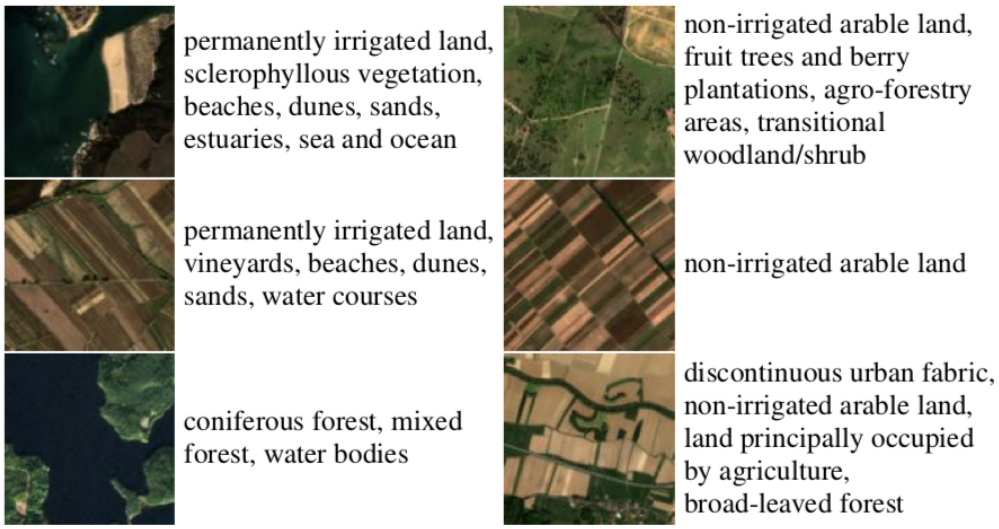


Figure 1: Example satellite images in the BigEarthNet dataset with multi-labels.

To facilitate reproducible research, code for implementing these models and re-running our experiments can be found at `https://github.com/tchainzzz/multilabel_metalearning`.

1 Introduction

Satellite imagery has transformed countless aspects of technology and society from navigation and weather forecasting, to military surveillance and climate modeling. With rapid developments in Artificial Intelligence and Deep Learning in the past couple decades, algorithms have given us the capability to identify, classify and process satellite images at a speed, scale and precision superior to human levels. It should come as no surprise that using AI with satellite images is so heavily utilized, as image recognition is one of the best performing areas of AI, with large neural networks such as Inception [Szegedy et al., 2014] and ResNet-152 [He et al., 2015] approaching near-perfect performance on image classification tasks.

Satellite image recognition presents a challenge for classification algorithms, however, in that for the labels to be specific enough to be useful, there are often multiple classes present in a single image, which makes satellite image classification a multi-label classification problem. For example in the BigEarthNet data set (Figure 1), there are a total of 51 separate land classes, and each image has anywhere between 1 and 12 different class labels, with 95% having 5 or less class labels.

While Sumbal et al. use a traditional CNN classifier adapted to the multi-label space, we aim to transform various meta-learning algorithms instead to the task of multi-label classification. We experiment with two support-set sampling techniques and two types of multi-label problem transformations. We adapted three meta-learning algorithms: memory augmented neural networks (MANN), model-agnostic meta-learning (MAML), and prototypical networks (ProtoNets), to accommodate each of the sampling and multi-label transformation techniques in order to develop a framework for multi-label meta-learning algorithms.

2 Related Work

Land cover classifcation. BigEarthNet [Sumbul et al., 2019], in addition to generating the data set we utilize, ran classification experiments on the satellite land cover imagery using a shallow CNNs

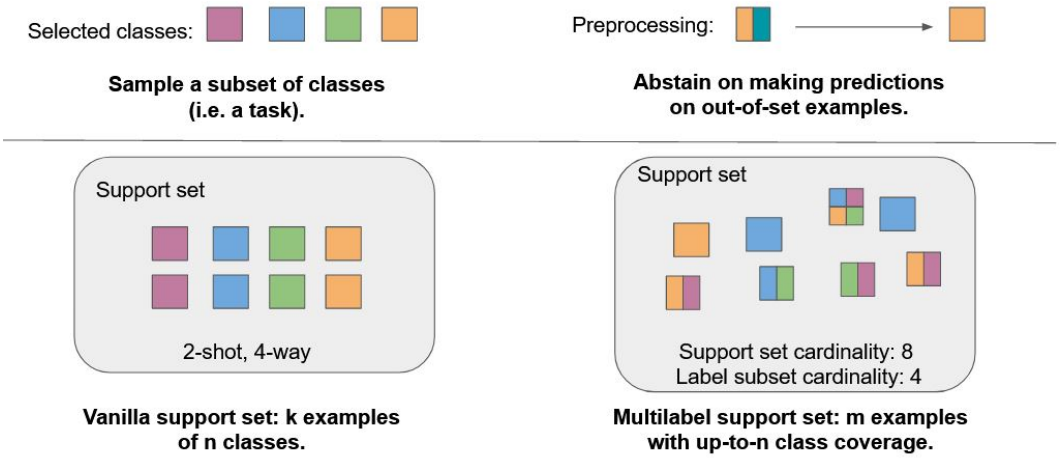


Figure 2: Overview of our multilabel support set sampling method, with a comparison to standard support set sampling.

as well as larger pre-trained models such as Inception [Szegedy et al., 2014]. They found that the shallower models were better suited for the task. Brandt et al. [2020] used deep neural networks to detect over 1.8 billion trees in the West Sahara Desert, which provided breakthrough methods for monitoring trees and climates outside of forrest ecosystems.

Multi-label problem transformation. Binary relevance [Zhang et al., 2018, Montañes et al., 2014] and label powerset [Read et al., 2014, Tsoumakas et al., 2008] methods are the most common multi-label problem transformations. More advanced modifications of these techniques, like classifier chains [Read et al., 2011] or hierarchical decision trees [Vens et al., 2008], extensions of binary relevance transformation, and pruned set ensembles [Read et al., 2008], which ignore rarely occurring multilabels, have also been explored. Another approach is RAKEL [Tsoumakas and Vlahavas, 2007], which samples n-label subsets to train an ensemble of 2^n-way classifiers. The ensemble methods here most resemble meta-learning techniques conceptually, in their usage of an explicit set of learners; however, meta-learning methods decrease memory overhead by dynamically learning to output a single learner.

Multi-label domain adaptation. Bhatia et al. [2015] use a embedding-based approach to learn sparse multilabels, which has been used in recommender/ranking systems Jain et al. [2016] and text classification Liu et al. [2017] applications. Xu et al. [2016] use a label-matrix decomposition into a low-rank and sparse matrix (of arbitrary rank) to model tail labels and common labels separately.

Multi-label meta-learning. Multi-label meta-learning appears relatively underexplored. Wu et al. [2019] apply policy gradient methods to learn per-class thresholds and loss weights for multi-label entity classification. However, this method relies on prior knowledge of a fixed number of pre-determined entity classes, which is too strong of an assumption for this few-shot multi-label classification task.

3 Preliminaries

We provide in this section an overview of the problem setup and our methods. In Section 3.1, we disucss the multilabel few-shot domain adaptation setup. Then, in Section 3.2, we describe our sampling-based framework for generating support/query sets from a multilabel dataset. We then briefly overview the meta-learning methods on which we benchmark our task framework in Section 3.3. Lastly, we discuss some design choices important for this framework in Section 3.4.

3.1 Problem Setup

The standard few-shot domain adaptation setup is as follows. Consider a source dataset $\mathcal{D}_{src} = (\mathcal{X}_{src}, \mathcal{Y}_{src})$ and a target dataset $\mathcal{D}_{tgt} = (\mathcal{X}_{tgt}, \mathcal{Y}_{tgt})$. In the case of BigEarthnet, $\mathcal{X}$ consists of $120 \times 120 \times 3$ tensors representing RGB images, and $\mathcal{Y}$ consists of vectors in $\mathbb{R}^C$, where C represents the cardinality of the set of labels in each domain. In the language of meta-learning, for the single-label case, we sample a task $\mathcal{T}_i$ which comprises n classes from $\mathcal{Y}_{src}$, from which we sample a support set $\mathcal{S}$ with k labeled examples of each class and a query set $\mathcal{Q}$ with k unlabeled examples of each class. This is known as a k-shot, n-way classification problem. We can then optimize some classifier to predict the labels of $\mathcal{Q}$ given supervision from $\mathcal{S}$. At test time, we repeat the same setup, sampling from $\mathcal{Y}_{tgt}$ instead.

3.2 Task Definition Framework

Two key challenges emerge when translating standard few-shot domain adaptation techniques from the single label domain to the multi-label domain:

- The notion of a "shot" is not well-defined: it is unclear what sampling k examples of a particular class means.
- It is not immediately clear how one can sample data from a well-defined set of n classes.

Both questions suggest that a new definition of the support/query set is necessary. Thus, to address this problem, we take some simple relaxations of the definition of a support set as provided in Section 3.1 to create a *hierarchical sampling task generation* framework. The basic idea is to sample random label subsets, from which we sample support sets. Let $\mathcal{L}_{tr}, \mathcal{L}_{ts}$ be disjoint sets of multilabels representing the source and target domains, respectively. At meta-train time, we sample subsets $S \subseteq \mathcal{L}_{tr}$ of cardinality n. Then, we sample a support set with m examples $(\mathbf{x^{(i)}}, \mathbf{y^{(i)}})$, where $x^{(i)}$ is an image and $y^{(i)} \in \{0,1\}^n$, such that the union of labels $\cup_{i=1}^{m} y^{(i)}$ is a subset of S. At meta-test time, we sample from $\mathcal{L}_{ts}$ similarly.

3.3 Datasets and models.

Dataset. We experiment on BigEarthNet, a collection of 590326 satellite images with 51 types of land cover, taken year-round by the Sentinel-2 satellite in 10 countries of Europe from June 2017 to May 2018, covering all seasons. Due to time and memory constraints (the full dataset is 66GB), we train and test on a subset of 10000 images under an approximate 70-30 train-test split derived from partitioning the labels into disjoint train and test sets. As a preprocessing step, we select only the RGB color bands out of the 12 spectral bands in the dataset, subtract the ImageNet [Deng et al., 2009] mean from each channel of the BigEarthnet images.

Meta-learning methods. We experiment with various types of meta-learning methods. First, we examine black-box domain adpatation, testing on MANN [Santoro et al., 2016]. We also examine optimization-based meta-learning, using MAML [Finn et al., 2017] as a baseline. Lastly, we evaluate the utility of ProtoNets [Snell et al., 2017], a metric-based learning setup. We benchmark these techniques on our multi-label task definition framework, evaluating whether models trained on a subset of land cover types can learn to discriminate unseen land cover types. For exact hyperparameter settings and training settings, see the Appendix.

Evaluation metrics. We measure precision, recall, and F1 score for each model, calculated as an unweighted average (macro) over the labels of the query set.

3.4 Design choices

Support/query set sampling method. There are multiple considerations for sampling examples to create a support set that spans a subset of labels $\ell \subset \mathcal{L}$. Given ℓ, the standard single-label setup samples a support set of size $k \times |\ell|$ where k is the number of shots; for the multi-label setup, we can simply set hyperparameter m to represent the total size of the support set. We present two methods of support/query set sampling: *greedy* sampling and *uniform permutation* sampling.

Under a *greedy* sampling technique, we repeatedly sample uniformly at random examples $(x^{(i)}, y^{(i)})$ from the training dataset and accumulate them into our support set until it reaches size m. If $y^{(i)} \cap \ell \neq \emptyset$, we take $y'^{(i)} = y^{(i)} \cap \ell$ and add $(x^{(i)}, y'^{(i)})$ to the support set. Intuitively, we greedily fill our support set with examples that contain land cover classes in ℓ, and abstain on land cover classes appearing in such images outside of ℓ. This method is intuitive and easy to implement, but does not guarantee a balanced distribution of classes in the support.

An alternative method of sampling that addresses the class imbalance problem more directly is to consider all possible permutation of classes in ℓ that appear in the dataset, and then we sample from those permutations in a fixed order, accumulating examples until we reach support set size m. We call this method *uniform permutation* sampling. More formally, we sample some ℓ such that $|\ell| = n$; then, we define a bijection $\tau : \{0,1\}^n \setminus \{0\}^n \to [2^n - 1]$, where $[i]$ is defined as $\{1, 2, \ldots, 2^n - 1\}$ as well as a permutation $\pi : [2^n - 1] \to [2^n - 1]$. In practice, τ is simply the function that converts the the input set of zeros and ones into a binary integer representation, then subtracts one, i.e. $\tau(\{0,1,1\}) = 011_2 - 1 = 2$. Then, for support set $\mathcal{S}$, we add examples such that $\mathcal{S}_i = (x^{(i)}, \pi(\tau(y'^{(i)})))$, where $y'^{(i)}$ is as defined above, and $\mathcal{S}_i$ is the ith element of $\mathcal{S}$ for $i = 1, 2, \ldots, m$. In practice, this can be done be precomputing the indices of examples in the dataset that contain a particular class label for each class.

This sampling technique is more consistent in expectation with the traditional notion of "shots" in few-shot learning: in the standard k-shot setup, the support set contains the same number of examples per class. Under the strong assumption that all permutations are equally likely to appear, uniform permutation sampling trivially results in class-balanced support sets in expectation in the limit of infinite data.

We also present a sketch for an optimal class-balanced sampling approach that dynamically computes permutation sampling probabilities for each support set, which can be found in the Appendix. We leave the evaluation of this technique to future work.

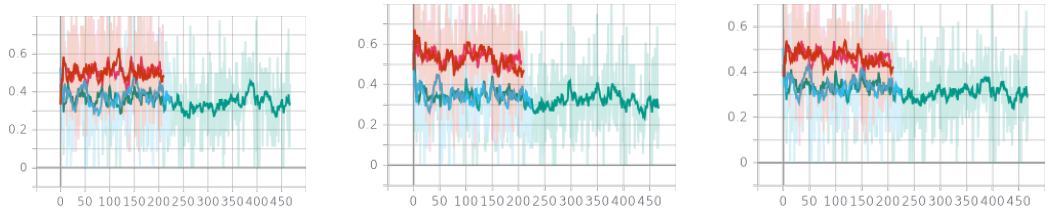
Multi-label transformation technique. Standard few-shot meta-learning frameworks are limited to single-label input, so we evaluate baseline methods for transforming multi-label problems so that we can easily integrate these tasks into any few-shot meta-learning framework. The first method is known as the *powerset* method, which transforms multi-label $y^{(i)}$ into a one-hot label via some bijective map $\tau_{\mathcal{P}} : \{0,1\}^n \setminus \{0\}^n \to 2^n - 1$. This method can be used with no architectural changes to a classifier.

The second method, known as the *binary relevance* method, transforms a multi-label $y^{(i)}$ into an m-hot vector of length n, and performs binary classification on a corresponding entry of the label vector. Thus, this method is only suitable when architectural changes are feasible. Lastly, note that the binary relevance method is equivalent to making an assumption of mutual independence between the labels.

Weight sharing. Since the binary relevance method transforms this problem into a multi-task, single-output problem, an important design choice for the binary relevance method is the extent of weight sharing between the heads. Due to memory constraints, we share weights as much as possible, adopting a simple hard parameter sharing heuristic. For MAML and ProtoNets, we share all weights except for those in the last layer; for MANN, all weights prior to the first LSTM controller layer are shared. We hope to explore this tension more in the future.

4 Results

In this section, we discuss the results of our task-definition framework on a few-shot land cover multilabel classification task under standard few-shot meta-learning architectures. In Section 4.1, we discuss difference between greedy and uniform permutation sampling on the performance of these models. Then, in Section 4.2, we perform the same analysis on the effect of powerset vs. binary relevance multi-label transformation methods. Additionally, we perform a targeted study on MAML, chosen because of its high performance and relatively fast training time, on the effect of support set size (Section 4.3) and label subset size (Section 4.4) on performance of the model.

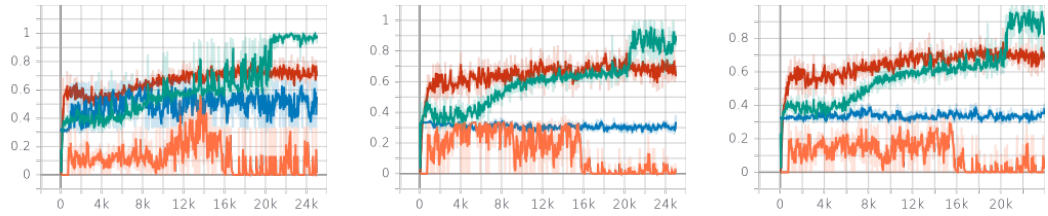


(a) Protonets validation precision with early stopping.

(b) Protonets recall precision with early stopping.

(c) Protonets validation F1 with early stopping.

Figure 3: Protonets training history. From top to bottom (at end of training/right side of graph); pink: uniform + binary relevance (BR), red: greedy + (BR), blue: greedy + powerset, turquoise: uniform + powerset.

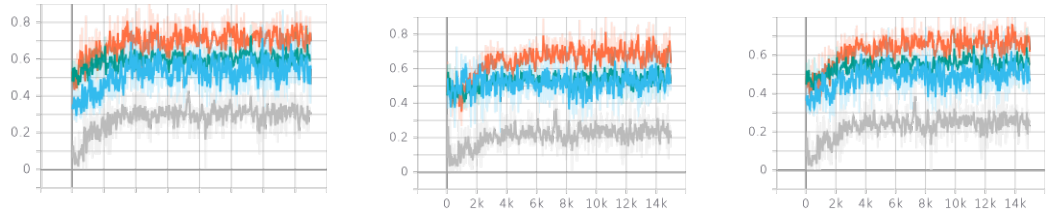


(a) MANN validation precision over 25000 epochs.

(b) MANN validation recall over 25000 epochs.

(c) MANN validation F1 over 25000 epochs.

Figure 4: MANN training history. From top to bottom (at end of training/right side of graph); turquoise: uniform + powerset, red: uniform + binary relevance (BR), blue: greedy + BR, orange: greedy + powerset.

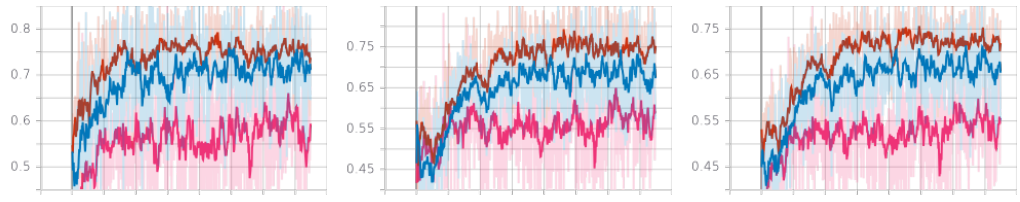


(a) MAML outer-loop validation precision over 15000 epochs.

(b) MAML outer-loop validation recall over 15000 epochs.

(c) MAML outer-loop validation F1 over 15000 epochs.

Figure 5: MAML training history. From top to bottom (at end of training/right side of graph); orange: uniform + binary relevance (BR), turquoise: uniform + powerset, light blue: greedy + BR, gray: greedy + powerset.



(a) Precision

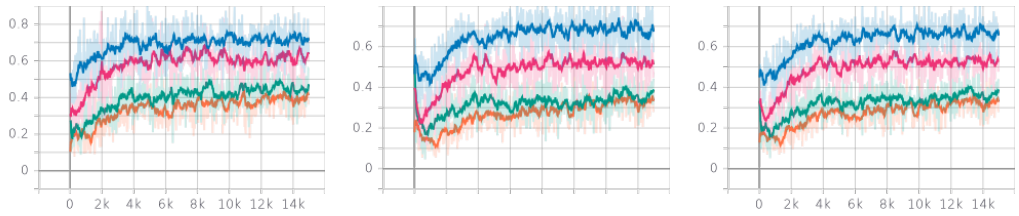
(b) Recall

(c) F1

Figure 6: Outer loop validation performance for MAML, varying support set cardinality. Support set sizes (top to bottom; end-of-training): red=16, blue=8, pink=4.

Table 1: Evaluation time performance of standard few-shot meta-learning algorithms under varying support set sampling schemes and multilabel problem transformation methods.

Model	Sampling Method	Multilabel Transformation	Precision	Recall	F1
ProtoNets	Greedy	Powerset	36.6	33.8	33.4
	Uniform	Powerset	36.0	34.9	33.5
	Greedy	Binary Rel.	**53.7**	**37.4**	**40.1**
	Uniform	Binary Rel.	53.2	36.0	38.8
MANN	Greedy	Powerset	0.0	0.0	0.0
	Uniform	Powerset	**92.9**	**83.3**	**86.3**
	Greedy	Binary Rel.	50.0	31.0	34.9
	Uniform	Binary Rel.	68.8	65.9	65.4
MAML	Greedy	Powerset	23.2	8.2	11.0
	Uniform	Powerset	60.1	56.6	56.0
	Greedy	Binary Rel.	52.1	38.1	40.4
	Uniform	Binary Rel.	**84.5**	**79.4**	**79.5**



(a) Precision (b) Recall (c) F1

Figure 7: Outer loop validation performance for MAML, varying label subset cardinality. Subset sizes (top to bottom; end-of-training): blue=3, pink=4, turquoise=5, orange=6.

4.1 On Support Sampling Methods

In Table 1, we provide results on the effect of sampling methods on the performance of baseline meta-learning methods. In general, it appears that uniform permutation sampling methods outperform greedy sampling methods. For MANN, uniform permutation sampling results in up to a **51.4** point gain in F1 score over the best greedy-sampling method; for MAML, this results in up to a **39.1** point gain in F1 score. This pattern does not hold for ProtoNets; however, performance is consistently low for that particular method. Even in that case, uniform permutation sampling maintains competitive performance, lagging only 1.3 points F1 behind the best greedy-sampling method.

Our results suggest that uniform permutation sampling may be important to achieve high performance in the multi-label setting for domain adaptation. While this is not a particularly surprising result, this demonstrates empirically that greedy sampling paradigms, while faster and easier to implement, are insufficient to provide meta-learning models with sufficient supervision to adapt. Interestingly, regardless of support sampling method, the models trained tend to have relatively higher precision than recall, indicating a low false positive rate.

4.2 Empirical Study of Label Transformation Techniques

In general, binary relevance label transformation works better than powerset label transformation. For ProtoNets, this results in a **6.6** point gain in F1 score over the nearest powerset-based experiment. For MAML in particular, this transformation was essential to maximize the performance of our models: with binary relevant, we achieved gains in F1 score up to **23.5** points over the nearest greedy sampling experiment. Interestingly, compared to the lift from uniform permutation sampling (detailed in Section 4.1), the performance gains from using binary relevance label transformation over powerset label transformation are smaller. This suggests that using uniform permutation sampling is a more important design choice than the label transformation method.

Table 2: Evaluation time performance of MAML with varied support set and label set cardinalities, under binary relevance label transformation and uniform permutation support set sampling. For experiments involving varying support set cardinality, label subset size is 3. For experiments that vary label subset cardinality, support set size is 8.

Support set cardinality	Precision	Recall	F1
4	75.1	75.2	72.8
8	84.5	79.4	79.5
16	86.6	83.0	83.1

Label subset cardinality	Precision	Recall	F1
3	84.5	79.4	79.5
4	69.8	68.9	66.8
5	60.0	52.6	53.2
6	40.4	36.5	35.7

However, we note that MANN's performance under uniform permutation sampling and powerset label transformation was anomalously high; in Figure 4, we have provided validation curves for precision, recall, and F1. Toward the end of training, the experiment uniform permutation sampling and powerset label transformation, represented by the turquoise line, experienced in a spike in all metrics, which we *did not* observe in the training data. Similarly, MANN's performance under greedy sampling and powerset label transformation was anamolously bad, with a precision/recall/F-score of 0 at the end of training. This is possibly attributable to hyperparameter sensitivity with MANN. This largely means that for this particular architecture, more extensive experiments may be necessary in the future to control for variance in the results.

4.3 Empirical Study of Support Set Size

We perform a study of the effect of support set size on model convergence. Again, we focus on MAML under a binary relevance label transformation and uniform permutation sampling. Naturally, increasing support set size improves model performance. We present validation metric curves for precision, recall, and F1 in Figure 6. All lines trend upward and to the right, showing that the model is still able to learn under all support set sizes tested, though performance decreases as support set size decreases.

Notably, doubling support set size only results in sublinear-to-linear gains in accuracy, as seen in Table 2: doubling support set size from 4 to 8 yields a lift of 6.7 F1 points, but doubling the support set size again to 16 yields a subsequent lift of only 3.6 points. Due to memory constraints, we are unable to determine if this pattern continues; however, these initial signals suggest diminishing returns of increasing support set size. In any case, the apparent need for exponential input for linear gains in output is not a particularly promising result. This directs focus for future work to methods for improving multi-label domain adaptation under constrained support set sizes.

4.4 Empirical Study of Label Subset Size

We perform a targeted study on the effects of label subset size on model performance as well. We focus on MAML under a binary relevance label transformation and uniform permutation sampling, because of its good performance and relatively short training time. Unsurprisingly, this framework is not robust to increasing label subset size. We present validation metric curves for precision, recall, and F1 in Figure 7. All lines trend upward and to the right, showing that the model is still able to learn under all label subset sizes tested, though performance decreases as label subset size increases.

However, the way in which increasing label subset size scales with the performance drop is non-trivial. On testing data (Table 2), as label subset size increases, conditioned on constant support set size, F1 decreases linearly: each increment in label subset size is associated with a drop of $12.7 - 17.5$ points F1. Nevertheless, it is promising that model performance is still far above random even accounting for the performance drop, though performance indeed draws closer to random as label subset size increases. This indicates that under our label subset size settings, there is still sufficient supervision for the model to learn. For future work, it would be interesting to explore the tension between

label subset size and support set size in more depth, and find empirical limitations and theoretical guarantees under varying label subset cardinalities.

5 Conclusions and Future Work

Due to the sparse label-permutation space in multi-label classification, traditional algorithms that rely on seeing a specific example numerous times under perform. We attempt a different approach to this problem by using meta-learning to fit the mutli-label space, as meta-learning is able to classify images in a task while only relying on a few examples. Our results show that meta-learning can indeed be successfully adapted to the multi-label space, achieving an F1 score of 86.3 on the best-performing technique (MANN with uniform permutation sampling). We found that binary relevance generally outperforms powerset transformations for multi-label schemes, and that uniform sampling also improves performance over greedy sampling. Furthermore, the lift from using uniform sampling over greedy sampling is greater than that of choosing powerset transformation over binary label transformation.

Next steps for development include varying the choice of inner classification architecture, varying the weight sharing schemes for the meta-learning algorithms with binary relevance implementation, and hyperparameter optimization for the meta-learning models. Other sampling techniques for creating balanced classes or optimizing for rare classes are also significant points of interest for future study.

Contributions

Joshua and Trenton Chang collaborated equally on selecting the problem, and framing our task. Joshua was responsible for implementing the multi-label problem transformation methods, while Trenton wrote data loading and training code for the relevant models. Trenton implemented the different sampling modes. Both partners contributed equally to the final writeup, presentation, and other deliverables.

References

Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your maml, 2019.

Damian Bargiel and Sylvia Herrmann. Multi-temporal land-cover classification of agricultural areas in two european regions with high resolution spotlight terrasar-x data. *Remote Sensing*, 3(5): 859–877, Apr 2011. ISSN 2072-4292. doi: 10.3390/rs3050859. URL `http://dx.doi.org/10.3390/rs3050859`.

Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. Sparse local embeddings for extreme multi-label classification. In *Advances in neural information processing systems*, pages 730–738, 2015.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL `http://arxiv.org/abs/1512.03385`.

G. Hinton and T. Tieleman. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

Himanshu Jain, Yashoteja Prabhu, and Manik Varma. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 935–944, 2016.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124, 2017.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner, 2018.

Elena Montañes, Robin Senge, Jose Barranquero, José Ramón Quevedo, Juan José del Coz, and Eyke Hüllermeier. Dependent binary relevance models for multi-label classification. *Pattern Recognition*, 47(3):1494 – 1508, 2014. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2013.09.029. URL http://www.sciencedirect.com/science/article/pii/S0031320313004019. Handwriting Recognition and other PR Applications.

Chunping Qiu, Michael Schmitt, Christian Geiß, Tzu-Hsin Karen Chen, and Xiao Xiang Zhu. A framework for large-scale mapping of human settlement extent from sentinel-2 images via fully convolutional neural networks. *ISPRS Journal of Photogrammetry and Remote Sensing*, 163: 152 – 170, 2020. ISSN 0924-2716. doi: https://doi.org/10.1016/j.isprsjprs.2020.01.028. URL http://www.sciencedirect.com/science/article/pii/S0924271620300344.

J. Read, A. Puurula, and A. Bifet. Multi-label classification with meta-labels. In *2014 IEEE International Conference on Data Mining*, pages 941–946, 2014.

Jesse Read, Bernhard Pfahringer, and Geoff Holmes. Multi-label classification using ensembles of pruned sets. In *2008 eighth IEEE international conference on data mining*, pages 995–1000. IEEE, 2008.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine learning*, 85(3):333, 2011.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. One-shot learning with memory-augmented neural networks, 2016.

Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning, 2017.

Gencer Sumbul, Marcela Charfuelan, Begum Demir, and Volker Markl. Bigearthnet: A large-scale benchmark archive for remote sensing image understanding. *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, Jul 2019. doi: 10.1109/igarss.2019.8900532. URL http://dx.doi.org/10.1109/IGARSS.2019.8900532.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. URL http://arxiv.org/abs/1409.4842.

Grigorios Tsoumakas and I. Vlahavas. Random k -labelsets: An ensemble method for multilabel classification. In *ECML*, 2007.

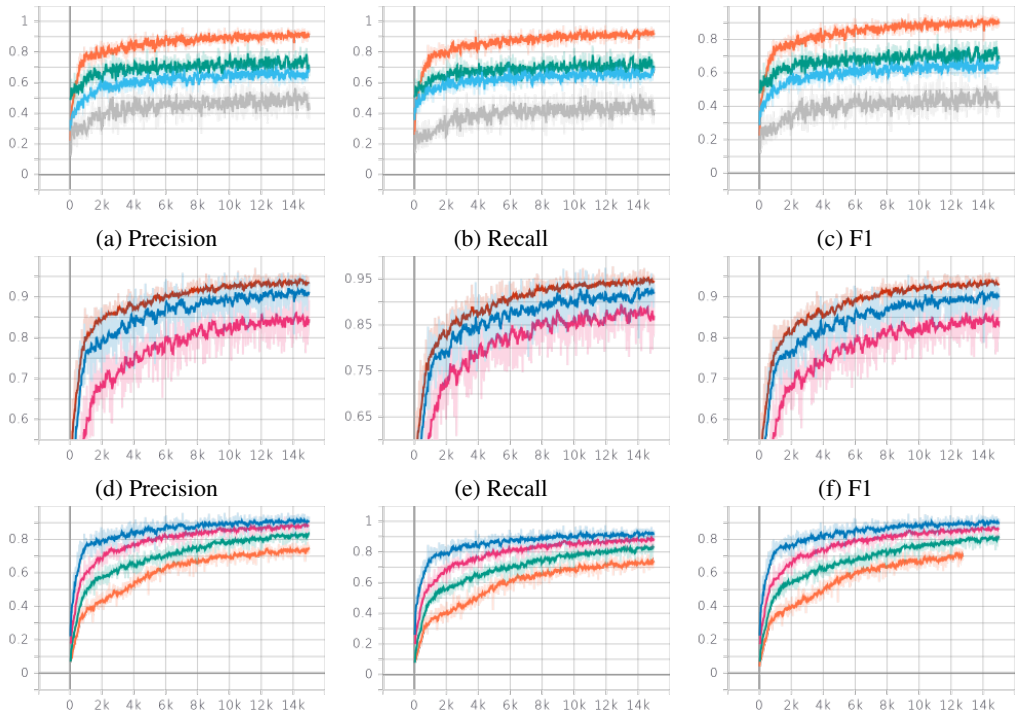
Grigorios Tsoumakas, Ioannis Katakis, and I. Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. 01 2008.

Compton J. Tucker, John R.G. Townshend, and Thomas E. Goff. African land-cover classification using satellite data. *Science*, 227(4685):369–375, 1985. ISSN 0036-8075. doi: 10.1126/science.227.4685.369. URL https://science.sciencemag.org/content/227/4685/369.

Celine Vens, Jan Struyf, Leander Schietgat, Sašo Džeroski, and Hendrik Blockeel. Decision trees for hierarchical multi-label classification. *Machine learning*, 73(2):185, 2008.

Jiawei Wu, Wenhan Xiong, and William Yang Wang. Learning to learn and predict: A meta-learning approach for multi-label classification, 2019.

Chang Xu, Dacheng Tao, and Chao Xu. Robust extreme multi-label learning. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 1275–1284, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939798. URL https://doi.org/10.1145/2939672.2939798.



(a) Precision (b) Recall (c) F1

(d) Precision (e) Recall (f) F1

(g) Precision (h) Recall (i) F1

Figure 8: Outer loop validation performance for MAML, varying sampling mode/label transformation (top), support set cardinality (middle) and label subset cardinality (bottom). Colors are as given in corresponding figures in the body.

Min-Ling Zhang, Yu-Kun Li, Xu-Ying Liu, and Xin Geng. Binary relevance for multi-label learning: an overview. *Frontiers of Computer Science*, 12(2):191–202, 2018.

A MAML Inner Training History

We provide here inner loop training precision/recall/F1 curves for the MAML experiments run.

B Enforcing Class-Balance with Missing Permutations

As a starting point for future work, we provide here a brief sketch of a modification to uniform permutation sampling for enforcing optimal class balance in the case where permutations are missing. As motivation, with increasingly large label subset sizes, permutations are likely to grow increasingly sparse, as the discrete distribution of the cardinality of multi-labels in a natural dataset tends to follow a power-law distribution. Furthermore, for large label subset sizes, the number of possible permutations grows exponentially, which exacerbates any class imbalance issues in the underlying data. Thus, we conjecture that more advanced methods may be necessary to achieve good performance under large label subset cardinalities.

The basic idea is to assign sampling probabilities to each permutation, instead of deterministically sampling each permutation. Suppose we choose label subset size n. Then, we define the probability that a particular class is sampled as $\mathbf{p} = (p_1, p_2, \ldots, p_n)$.

Then, we can define a matrix $A \in \mathbb{R}^{n \times P}$, where $P \geq n$ is the number of permutations of the n classes that exist for a particular label subset, where for an ordering of classes i the entries of A are given by

$$A_{ij} = \begin{cases} 1 & \text{if class } i \text{ appears in permutation } j \\ 0 & \text{otherwise} \end{cases}. \tag{1}$$

The target vector is simply $\frac{1}{n}\mathbf{1}$. Intuitively, the system $A\mathbf{p} = \frac{1}{n}\mathbf{1}$ encodes the constraint that all classes should be sampled equally. For simplicity, we assume that A is full-rank. Then this system is overdetermined; thus, the optimal value of $\mathbf{p}$ can be found by solving the constrained optimization problem

$$\text{minimize} \quad \left\| A\mathbf{p} - \frac{1}{n}\mathbf{1} \right\| \tag{2}$$

$$\text{subject to} \quad \|\mathbf{p}\| = 1; 0 \leq p_i \leq 1, i = 1, 2, \ldots n. \tag{3}$$

This can be solved using computational methods; we leave it to future work to evaluate the efficacy of this sampling method.

C Hyperparameters and Other Settings

Hyperparameters and optimizers. Our hyperparameters and optimizers are as follows: for MANN, we use RMSProp [Hinton and Tieleman, 2012] with a decay of 0.95, momentum of 0.9, and a learning rate of 10^{-4}; for MAML, we use ADAM [Kingma and Ba, 2017] with learning rate 10^{-3} for the outer optimizer and stochastic gradient descent with dynamic learning rate initialized at 0.4 for the inner optimizer following the idea of Antoniou et al. [2019]. For Protonets, we use SGD with learning rate 10^{-3}.

Inner/embedding architecture. For MANN and ProtoNets, we use an embedding architecture before passing the output into the LSTM controller (for MANN) or computing prototypes (for ProtoNets). The architecture is a 4 block convolutional net based on the embedding architecture used in SNAIL [Mishra et al., 2018], another meta-learning architecture for black-box domain adaptation. Each convolutional block consists of 64 filters with a 3×3 kernel, ReLU non-linearity, batch normalization, and 2×2 max-pooling; under binary relevance transformation, we split the architecture at the final dense layer (or last convolutional block for ProtoNets only) to output multiple binary predictions.

Training scheme. We train MANN for 25000 iterations (sampled task batches), MAML for 15000 iterations, and ProtoNets for as many iterations as needed to hit the early stopping threshold with patience 200. For early stopping, we monitor validation loss. Default meta-batch size is 16.

Hardware. Initial experiments with MAML were run on a K80 GPU with 12GB VRAM via Google Cloud Platform; the ablations on support set size and label subset size as well as all MANN experiments were run on a P100 GPU with 16GB VRAM via Google Cloud Platform. Experiments with ProtoNets were done locally using a CPU.

Other. Visualizations were generated using the `TensorboardX` Python package. Due to compatibility issues, we ran MAML and ProtoNets on Tensorflow version 2.3, but ran MANN on Tensorflow version 2.1. BigEarthNet files require the `gdal` package to load; the design of our dataloader was heavily based on those provided at `https://github.com/tensorflow/datasets/blob/master/tensorflow_datasets/image_classification/bigearthnet.py`, the official Tensorflow repository, and `https://gitlab.tubit.tu-berlin.de/rsim/bigearthnet-tools/blob/master/scripts/read_patch.py`, the official BigEarthNet repository.