Modular Meta-learning

Motivation

Previous approaches to meta-learning have focused on finding distributions or initial values of parameters. Our objective is similar, but rather than focusing on transferring information about parameter values, we focus on finding a set of reusable modules that can form components of a solution to a new task, possibly with a small amount of tuning. The authors provide an algorithm, called BounceGrad, which learns a set of modules and then combines them appropriately for a new task.

Diference between MAML and Modular Meta-learning

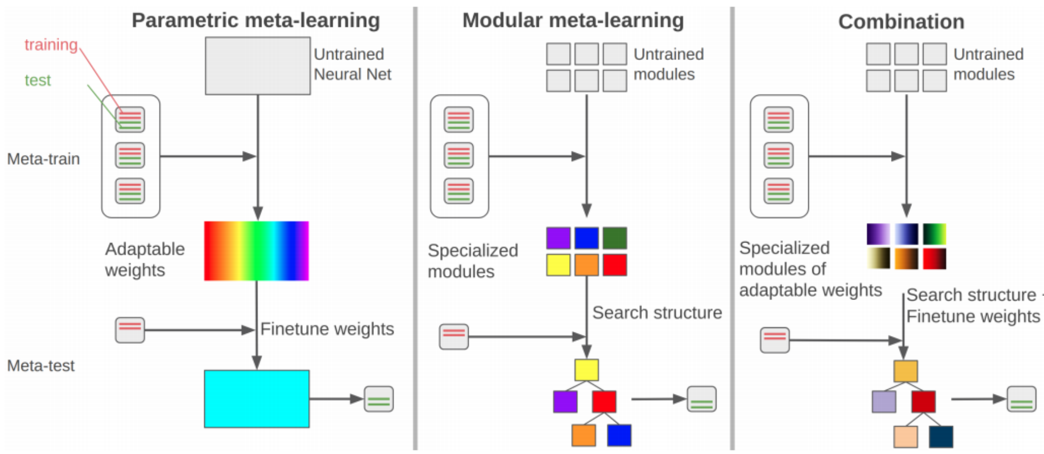
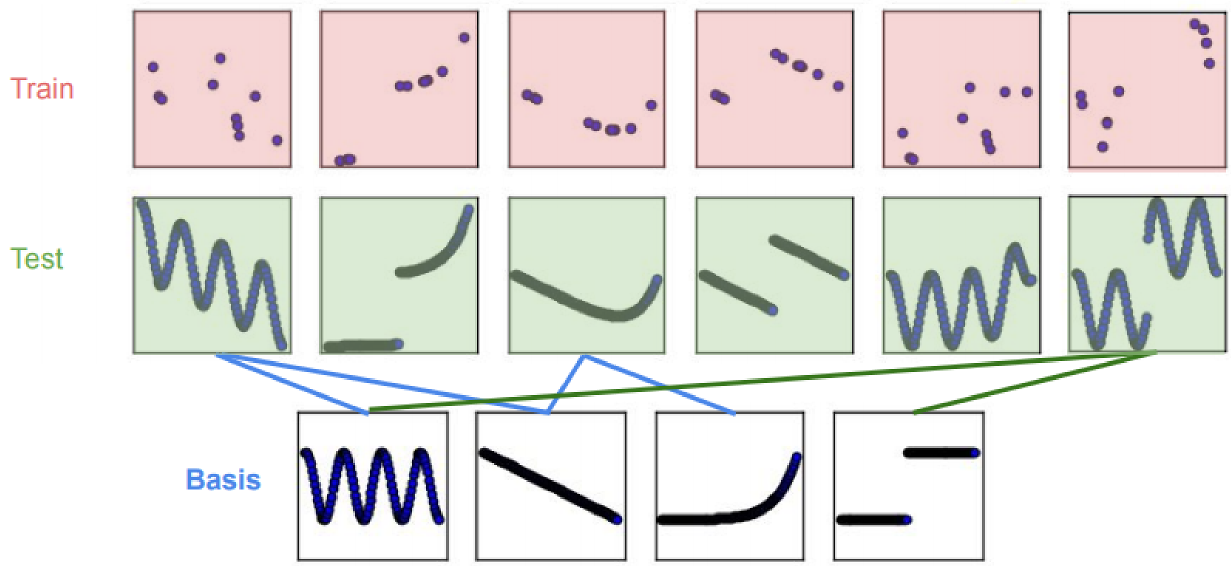


Figure 1 shows that all methods train on a set of related tasks and obtain some flexible intermediate representation. Parametric strategies such as MAML (left) learn a representation that can be quickly adjusted to solve a new task. Our modular meta-learning method (middle) learns a repertoire of modules that can be quickly recombined to solve a new task. A combination of MAML and modular meta-learning (right) learn initial weights for modules that can be combined and adapted for a new task.

Modular Meta-learning

Here, we give a concrete example to explain what is Modular Meta-learning.



Different from MAML to learn a good initialization, **Modular Meta-learning learns a modular decomposition of characteristics shared by similar tasks.**

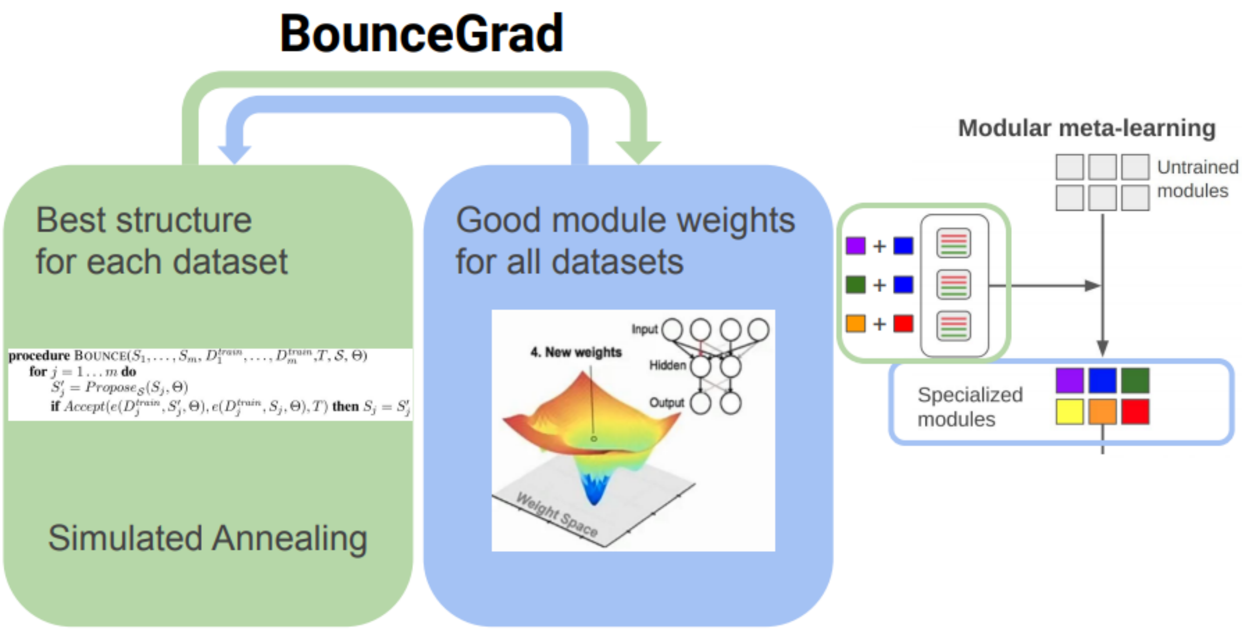
- learn good basis.
- learn which basis to pick and how to compose them together.

How to learn good modules?

Given the specification of a composition rule and a basis set of modules, (C, F, \theta) represents a set of possible functional input-output mappings that will serve as the hypothesis space for the meta-test task. F is a basis set of modules, which are functions f1,…, fk. C is a compositional scheme for forming complex functions from simpler ones, defined by an initial structure and a set of local modification operations on the structure.

Let s be the set of possible structures and s\in S be a particular structure, generated by C. The approach has two phases: an off-line meta-learning phase and an on-line meta-test learning phase.

- **Meta-learning phase**, we take training and validation data sets for tasks 1,…,k as input and generate a parametrization \theta for each module. The objective is to construct modules that will work together as good building blocks for future tasks.
- **Meta-test learning phase**, we take a training data set for the meta-test task as input, as well as S and \Theta ; the output is a compositional form s\in S which includes a selection of modules f1,…,fm to be used in that form. Since \Theta is already specified, the choice of s completely determines a mapping from inputs to outputs.



The optimization problems specified previously are in general quite difficult, requiring a mixed continuous-discrete search in a space with many local optima. The authors proposed BounceGrad algorithm (show in above figure), which performs local searches based on a combination of simulated annealing and gradient descent to find approximately optimal solutions to these problems.

This approach considers optimizaiton S as a inner problem, which has different aim from existing NAS and meta learning papers. This approach is more related to dictionary learning. The aim of this approach is to learn pretrained modules (Theta), such that these modules can be composed for new tasks. In this case, the hyperparameters are actually Theta, but not S.