INSTILLING INDUCTIVE BIASES WITH SUBNETWORKS

Enyan Zhang[1,2],
[1] Department of Computer Science, Brown University
[2] Division of Applied Mathematics, Brown University
enyan_zhang@brown.edu

ABSTRACT

Despite the recent success of artificial neural networks on a variety of tasks, we have little knowledge or control over the exact solutions these models implement. Instilling inductive biases — preferences for some solutions over others — into these models is one promising path toward understanding and controlling their behavior. Much work has been done to study the inherent inductive biases of models and instill different inductive biases through hand-designed architectures or carefully curated training regimens. In this work, we explore a more mechanistic approach: *Subtask Induction*. Our method discovers a functional subnetwork that implements a particular subtask within a trained model and uses it to instill inductive biases towards solutions utilizing that subtask. Subtask Induction is flexible and efficient, and we demonstrate its effectiveness with two experiments. First, we show that Subtask Induction significantly reduces the amount of training data required for a model to adopt a specific, generalizable solution to a modular arithmetic task. Second, we demonstrate that Subtask Induction successfully induces a human-like shape bias while increasing data efficiency for convolutional and transformer-based image classification models. Our code is available at the following *Github repository link*

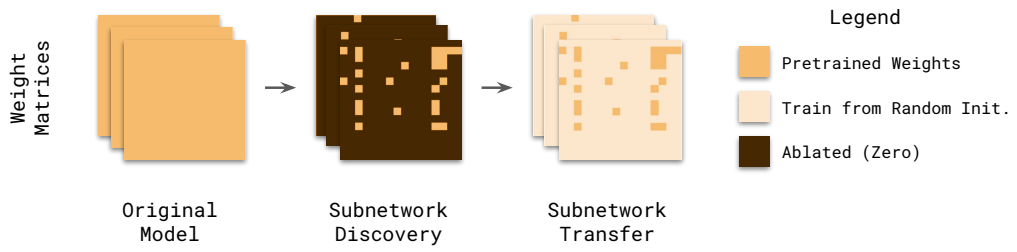

Figure 1: Subtask Induction localizes a subnetwork that implements a certain subtask in a trained neural network and transfers it to a randomly initialized model, thereby instilling an inductive bias towards solutions utilizing the specific subtask. The figure above illustrates the 3 stages of Subtask Induction in our experiments: we first train for a binary weight-level mask representing the subnetwork for a specific subtask through *subnetwork discovery*, then perform *subnetwork transfer* by copying the subnetwork weights to a newly initialized model and keep it frozen while optimizing the re-initialized weights. We demonstrate through two experiments that transferring subnetworks effectively and reliably instills desired inductive biases.