Modular Networks: Learning to Decompose Neural Computation

Louis Kirsch[1,2], Julius Kunze[1], David Barber[1]

[1]Department of Computer Science, University College London
[2]now affiliated with IDSIA, The Swiss AI Lab

@LouisKirschAI
louiskirsch.com/libmodular

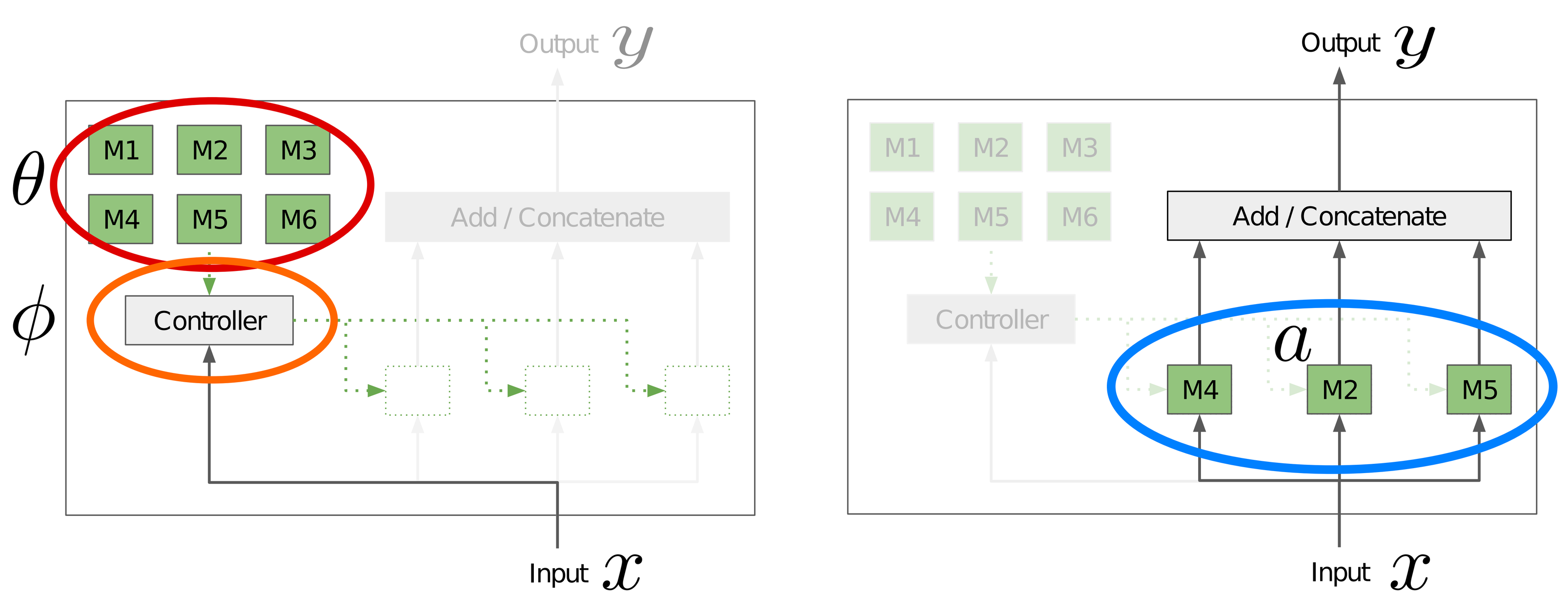
Motivation

- **Scaling up model size** has been vital in the success of deep learning
- Necessary compute resources and training time grow at least linearly with model size
- We solve this by **learning modules** that are conditionally executed
- In contrast to other approaches, we require **no regularization** to avoid module collapse

Introducing Modular Networks

- A 'module' is a sub-network that can be selected by the controller given some input ***x***
- We devise a training algorithm that learns the decomposition of a problem into modules
- A pool of modules is available for execution
- In each modular layer a set of modules ***a*** is chosen by a controller $p(a|x,\phi)$
- Modules have parameters θ ; controllers have ϕ
- Modular layers can be stacked or used as RNNs
- The output ***y*** of the network is thus given by

$$p(y|x,\theta,\phi) = \sum_a p(y|x,a,\theta)p(a|x,\phi)$$

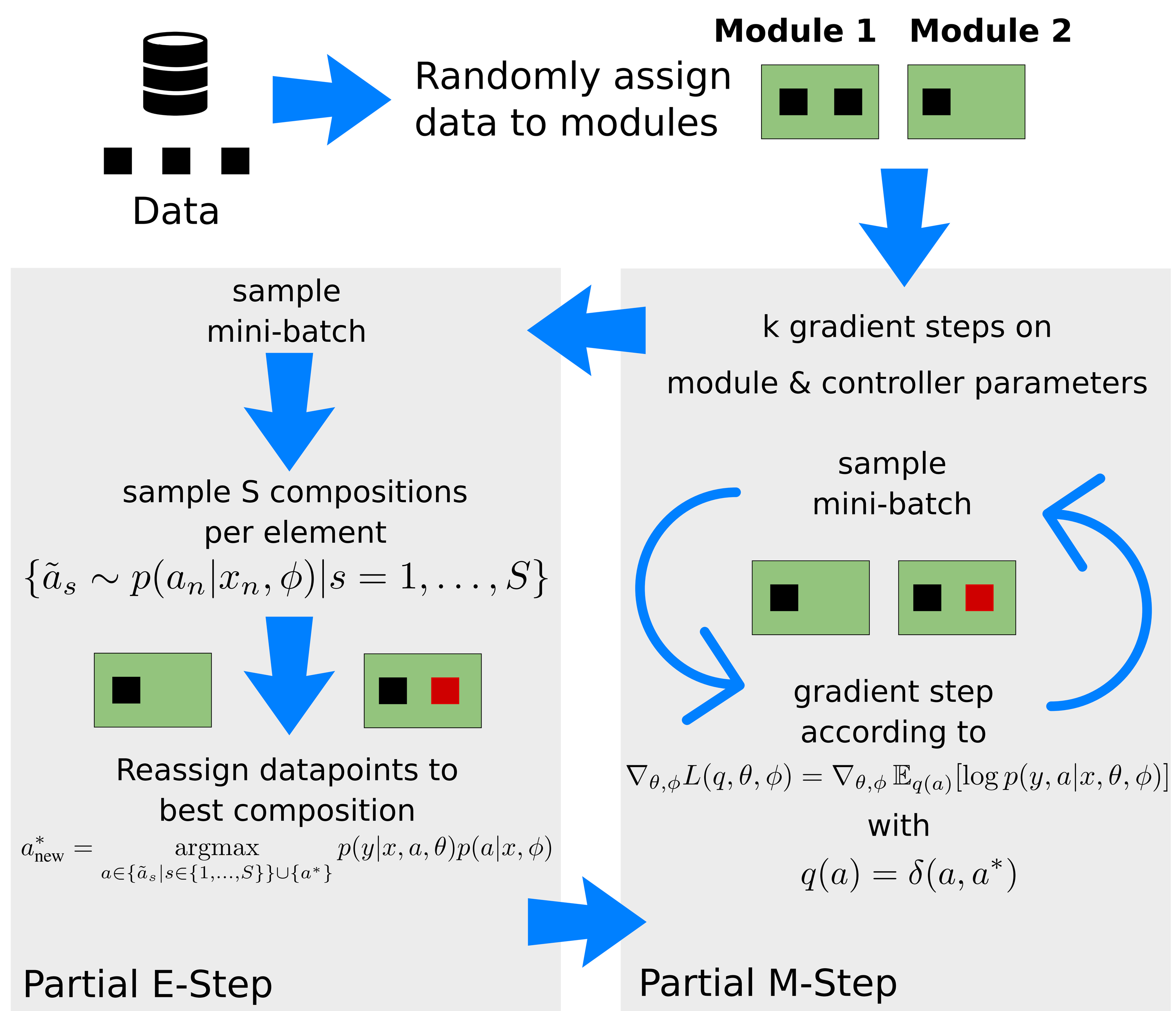


What is module collapse?

- An optimization algorithm designed to pick from a set of modules ignores most of them
- Because the capacity of a few modules is insufficient to learn a good solution, the algorithm gets 'stuck' in a local minimum

Method - Generalized Viterbi EM

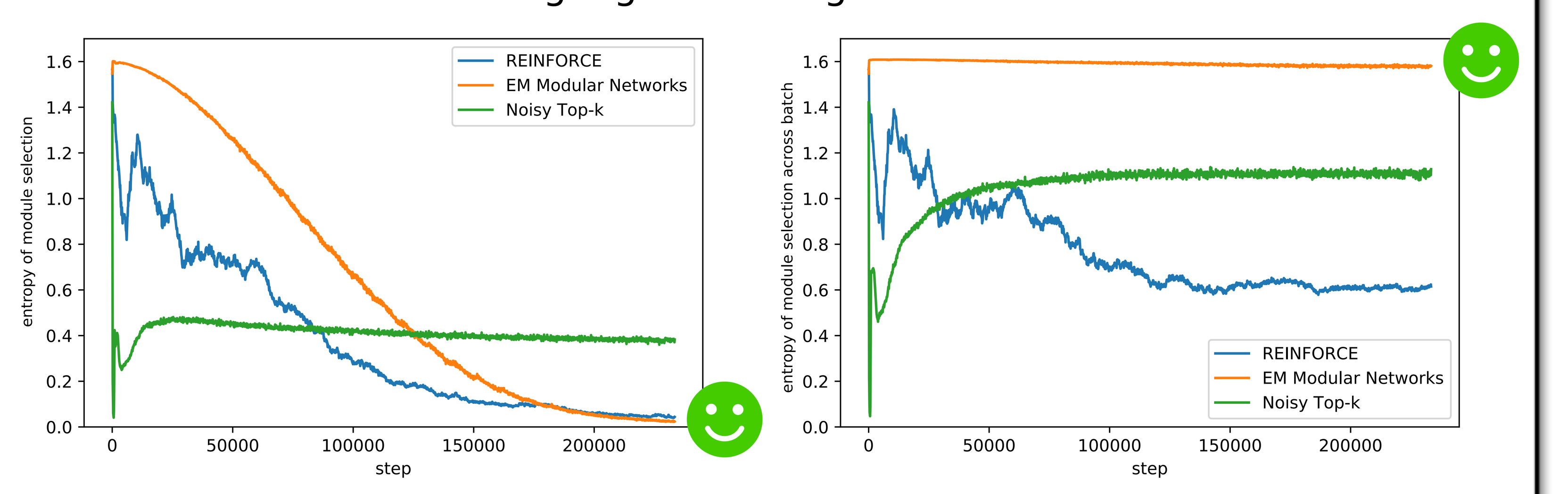
- The ML objective involves an intractible marginalization $p(y|x,\theta,\phi) = \sum_a p(y|x,a,\theta)p(a|x,\phi)$
- We optimize a lower bound instead



Results

- Modular networks completely **avoid module collapse** without regularization

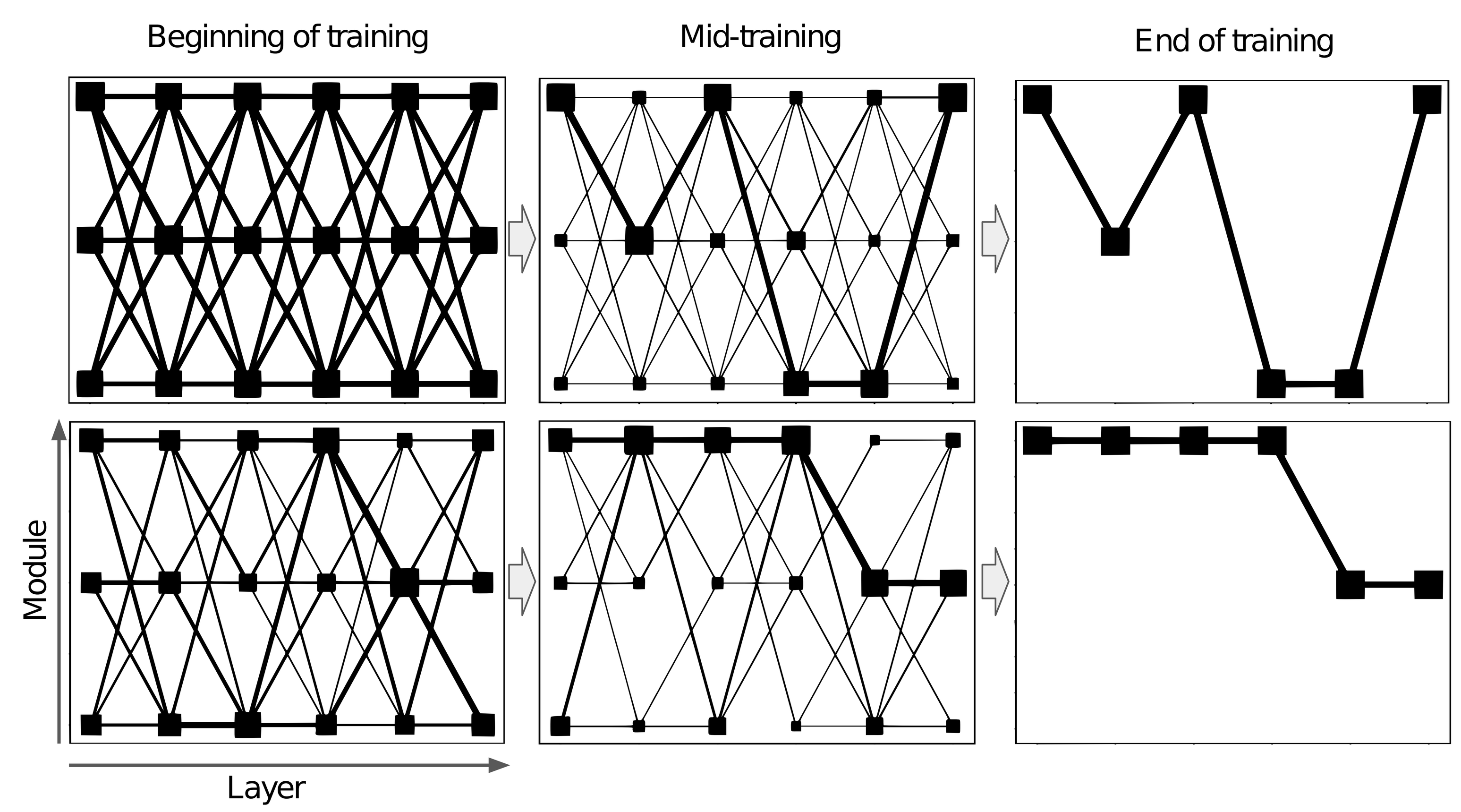
Word-level language modeling on Penn Treebank



The entropy of the controller converges to near zero, thus making the choice of module **deterministic**.

The entropy across the batch remains extremly high, thus **using all of the modules**.

- Modular networks can be interpreted as **clustering** datapoints that require **similar computation together**



- Analyzing modularization on **CIFAR10** image classification
- Two different subsets of datapoints (top and bottom) that use specific modules at the end of training (right)
- Datapoints in each subset start with entirely different modules (left) and **slowly cluster together** (left to right).
- Sizes of nodes (modules) and stroke width of edges (connected modules) describe how many datapoints of the subset use these modules.