NetTailor: Tuning the Architecture, Not Just the Weights

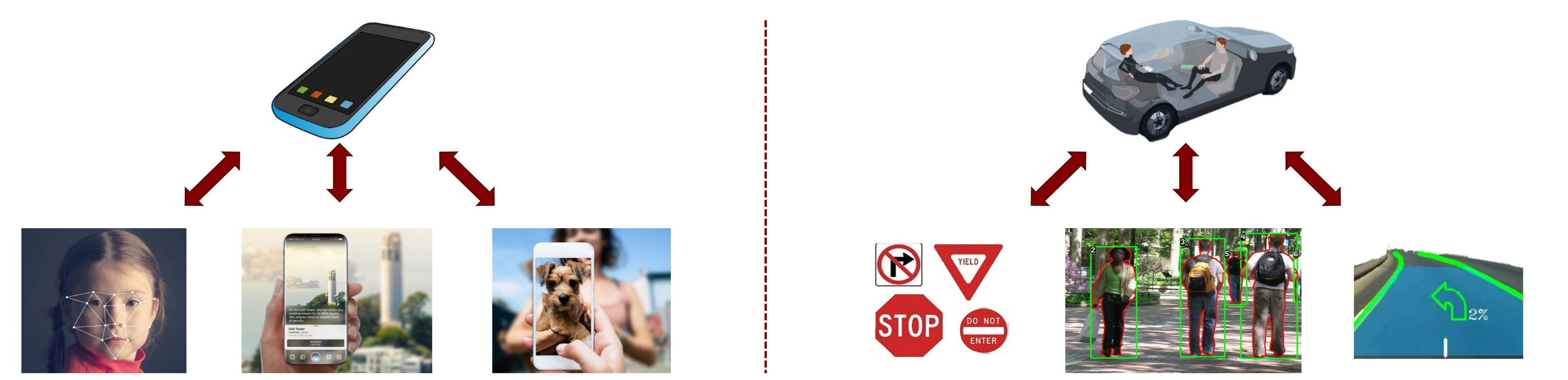
Pedro Morgado, Nuno Vasconcelos
University of California, San Diego

CVPR JUNE 16-21, 2019 LONG BEACH • CALIFORNIA

Abstract

Real-world applications of object recognition often require the ***solution of multiple tasks in a single platform***. We propose a transfer learning procedure, denoted ***NetTailor***, in which ***layers of a pre-trained CNN are used as universal blocks*** that can be ***combined with small task-specific layers to generate new networks***. In this way, NetTailor can adapt the network architecture, not just its weights, to the target task.

Experiments show that ***networks adapted to simpler tasks become significantly smaller than those adapted to complex tasks***. Due to the modular nature of the procedure, this reduction is achieved ***without compromise of either parameter sharing across tasks, or classification accuracy.***

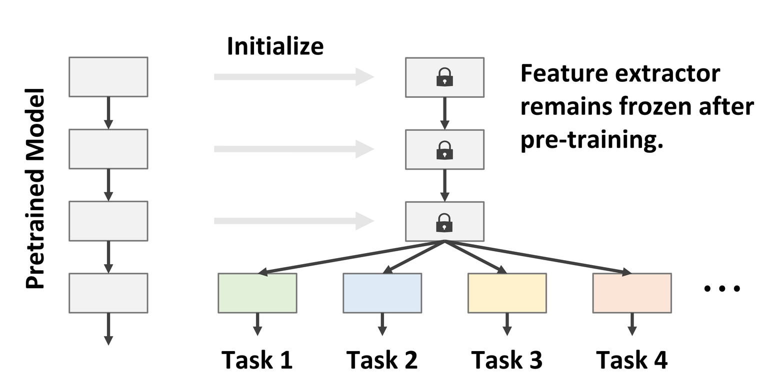


Goals

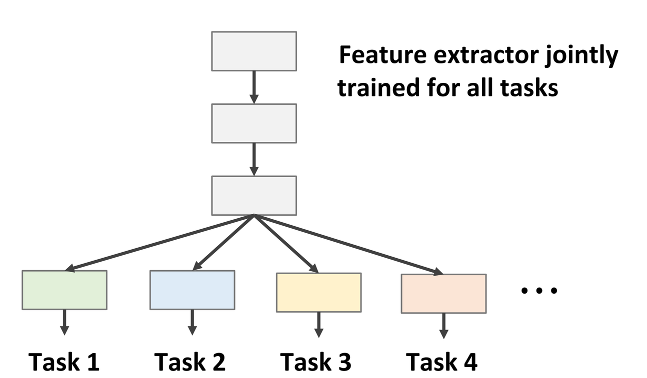
- **HP** ***High performance*** on all tasks
- **SP** Models ***share parameters and computation*** *to be* supported in the same device
- **DD** Models ***trained independently*** of tasks/datasets for **distributed development**.
- **AC** ***Adaptive model complexity*** *(Easier tasks → simple model, Complex → complex)*

Prior Work

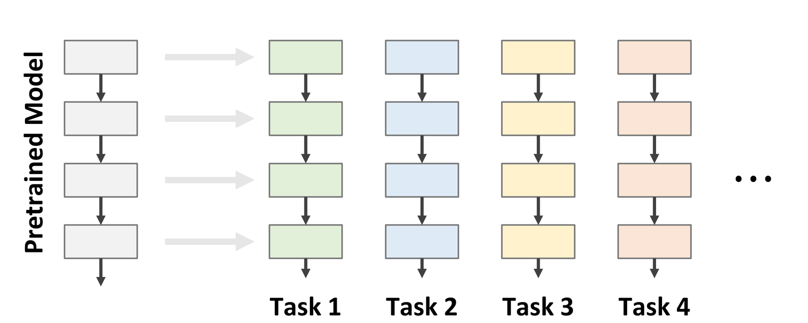
Feature Extraction HP SP DD AC



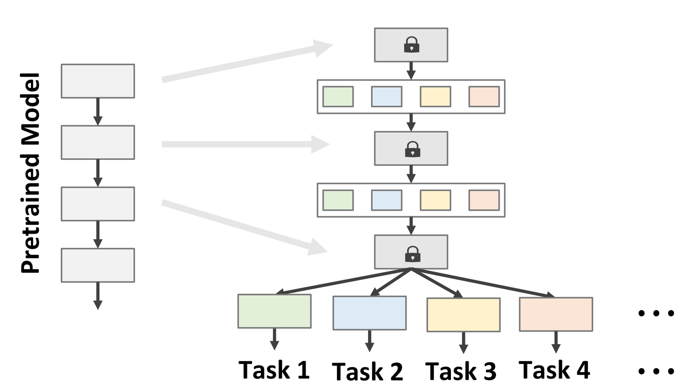
Universal classifier HP SP DD AC



Finetuning HP SP DD AC



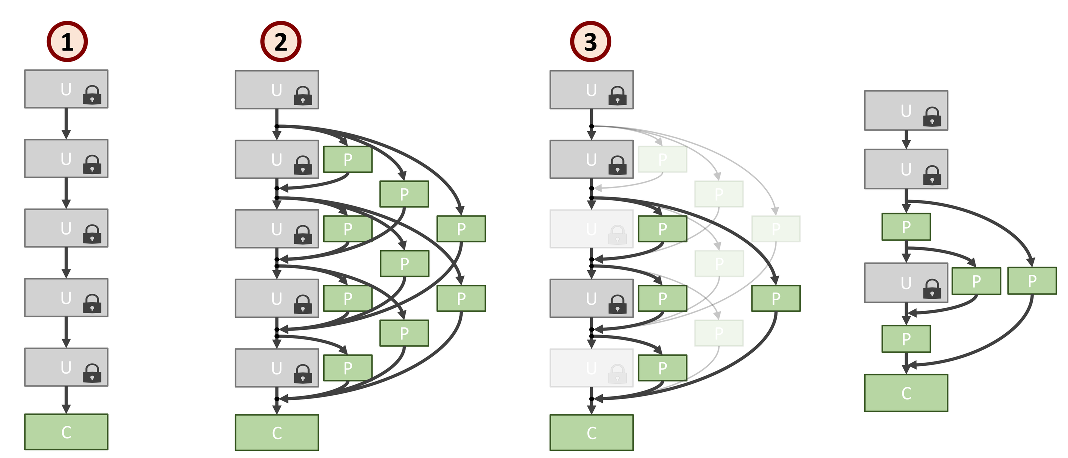
Residual Adapters HP SP DD AC



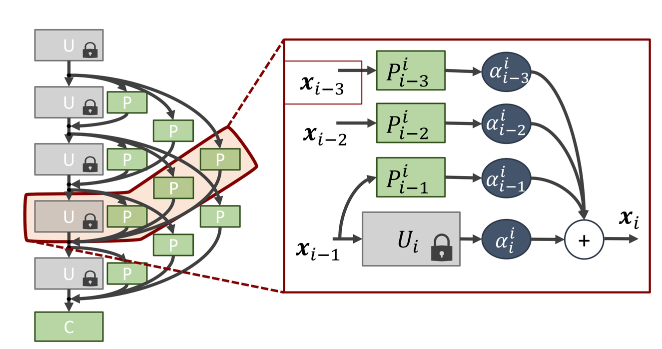
NetTailor

Idea

1. Start with a pre-trained network that remains ***unchanged.***
2. Augment the original network ( ) with a set of small task-specific blocks ( )
3. Find the ***smallest combination of layers and connections*** that solves the task.



Path Selection



An ***attention coefficient*** $\alpha \in [0,1]$ for each block

$$x_i = \alpha_i^i U_i(x_{i-1}) + \sum_k \alpha_k^i P_k^i(x_k)$$

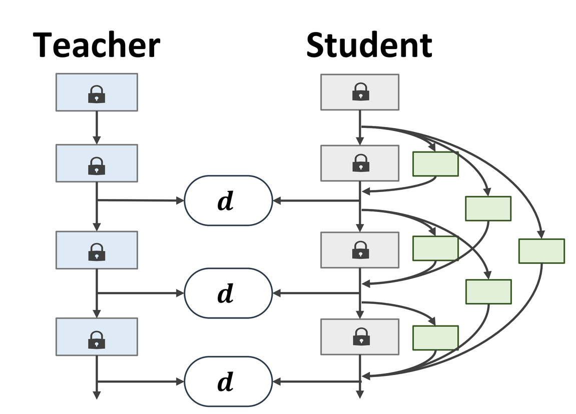
Blocks removed from inference path when α is small.
Coefficients α are parameterized by a ***softmax distribution*** to force network paths to compete

Complexity Constraints

Conditions for removing block B_i^j	Probability
Self exclusion: Small coefficient α associated with the block.	$P\left(R_{self}^{i,j}\right) = 1 - \alpha_i^j$
Input exclusion: When all incoming paths are removed	$P\left(R_{input}^{i,j}\right) = \prod_k (1 - \alpha_{I_k})$
Output exclusion: When all departing paths are removed	$P\left(R_{output}^{i,j}\right) = \prod_k (1 - \alpha_{O_k})$

Expected Complexity $$E[C] = \sum_{i,j} C_i^j \left(1 - P\left(R_{self}^{i,j} \cup R_{input}^{i,j} \cup R_{output}^{i,j}\right)\right)$$

Teacher Network



To preserve performance, a ***teacher network finetuned on the target task*** is used to supervise the student at each stage.

$$L_t(x) = \sum_l \left\|x_l^t - x_l\right\|^2$$

Learning

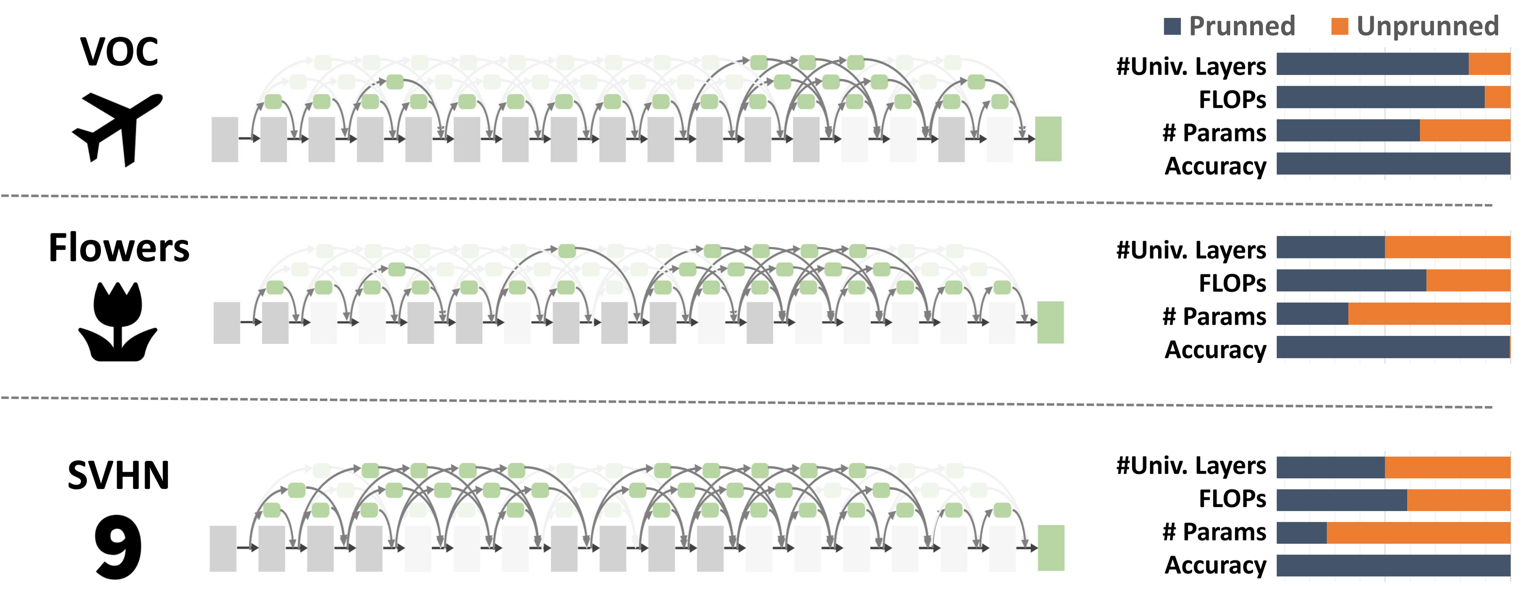
1st **Stage:** Tuning the architecture.
Student model optimized by SGD under the loss $\mathcal{L} = L_c(x,y) + \lambda_1 L_t(x) + \lambda_2 E[C]$.

2nd **Stage:** Pruning and retraining.
Blocks with small α are removed. Remaining parameters retrained with $\lambda_2 = 0$.

Evaluation

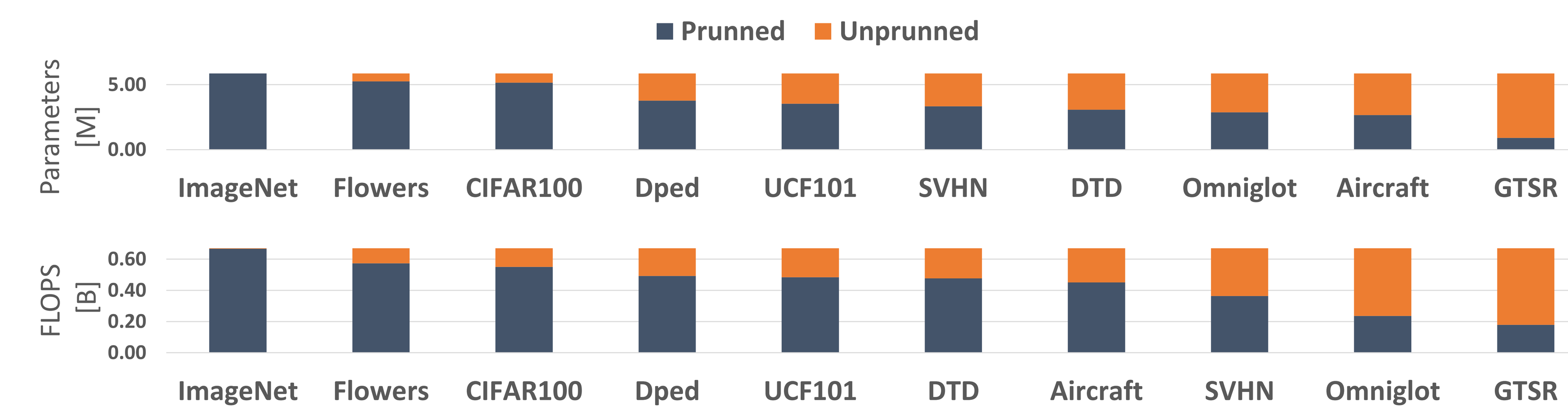
Learned architectures

Complex tasks require larger models, simpler task require smaller models



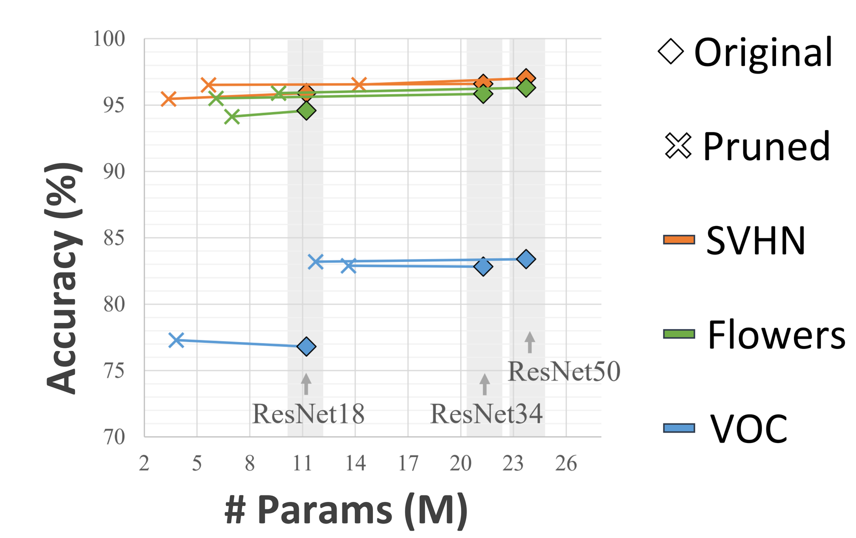
NetTailor on different datasets

Compression rates up to 80% of parameters and 66% of FLOPs



NetTailor on different base networks

ResNet50 pruned to the size of ResNet18 while preserving its performance



Future Work

- Study different block pruning strategies
- Extend NetTailor path selection framework to tasks beyond recognition (e.g. detection, semantic segmentation, multi-task learning)
- Study the benefits of NetTailor to general neural architecture search

Support

FCT Fundação para a Ciência e a Tecnologia NSF

Website

https://pedro-morgado.github.io/nettailor

Repository

https://github.com/pedro-morgado/nettailor