Hardware-Aware Neural Network Analysis

Mark Vero[1], Thorir Mar Ingolfsson[1], Xiaying Wang[1]
Lorenzo Lamberti,[2] Matteo Spallanzani[1], Luca Benini[1,2]

[1]ETH Zürich, [2]Università di Bologna

Why HW-constrained NAS?

Neural Architecture Search (NAS) aims at discovering Deep Neural Network (DNN) topologies that have good task accuracy. NAS algorithms are often time-consuming and computationally expensive. Moreover, they usually ignore the limitations of embedded or edge computing devices. To design better HW for DNNs and better DNNs for constrained HW, we must understand the recurring features of task-accurate architectures. This work investigates how to efficiently explore NAS spaces looking for task-accurate HW-constrained DNNs and how to design better search spaces for HW-constrained DNNs.

We model the **DNN architecture** as a latent variable $\lambda \in \Lambda$. This variable manifests itself through several *observable properties*, mainly 1) the **functional form** f_λ and 2) the **program form** G_λ.

Training-Free Statistics

In its functional form, a DNN is a function

$$f_\lambda : \Theta_\lambda \times X \to Y$$
$$(\theta_\lambda, x) \mapsto f_\lambda(\theta_\lambda, x)$$

that is parametric in $\theta_\lambda \in \Theta_\lambda$. The parameter evolves from a randomly chosen initial condition $\theta_\lambda^{(0)} \sim \mu_{\theta_\lambda^{(0)}}$ over a stochastic trajectory $\{\theta_\lambda^{(0)}, \dots, \theta_\lambda^{(T)}\}$, where $T \in \mathbb{N}$ is the number of iterations of the training algorithm.

Given a DNN f_λ, we define a **statistic** to be a measurable function

$$s : \Theta_\lambda \times (X \times Y) \to S$$
$$(\theta_\lambda, (x, y)) \mapsto s(f_\lambda(\theta_\lambda, \cdot), (x, y))\,.$$

Note that we can measure the statistic for an untrained ($t = 0$), partially trained ($0 < t < T$), or completely trained ($t = T$) network parameter $\theta_\lambda^{(t)}$. Note also that the value of the statistic is stochastic in the point $\theta_\lambda^{(t)}$ of the parameter trajectory and in the test point (x, y). We can describe the latent architecture variable $\lambda \in \Lambda$ through the statistic s after t training iterations by computing

$$s_\lambda \coloneqq \mathbb{E}_{\substack{\theta_\lambda \sim \mu_{\theta_\lambda^{(t)}} \\ (x,y)\sim\mu_{(x,y)}}} [s(f_\lambda(\theta_\lambda, \cdot), (x, y))] \approx \frac{1}{N} \sum_{n=1}^{N} s(f_\lambda(\theta_\lambda^{(t)}(\omega^{(n)}), \cdot), (x(\omega^{(n)}), y(\omega^{(n)})))\,.$$

The most important statistic is **task accuracy**:

$$a(f_\lambda(\theta_\lambda^{(T)}, \cdot), (x, y)) \coloneqq \begin{cases} 1, \text{ if } f_\lambda(\theta_\lambda^{(T)}, x) = y\,, \\ 0, \text{ if } f_\lambda(\theta_\lambda^{(T)}, x) \neq y\,. \end{cases}$$

Task accuracy requires one to run the training algorithm to completion before it can be measured. **Training-free (TF) statistics** are statistics whose distributions, when computed with respect to the distribution $\mu_{\theta_\lambda^{(0)}}$, correlate well with the distribution of task accuracy. *TF statistics provide a much cheaper estimate for the task accuracy of a candidate DNN, since they avoid the need to train it.*

NAS literature on TF statistics has so far proposed four promising candidates:

- the condition number of the Neural Tangent Kernel (NTK);
- the number of regions cut in the input domain X by a ReLU-activated network;
- NAS w/o Training Score (version 1 and version 2).

Clustering Computational Graphs

In its program form, a DNN is a bipartite graph

$$G = (V_M \cup V_K, E_R \cup E_W)$$

called a **computational graph (CG)**, where V_M is a set of memory nodes (e.g., parameters or feature arrays), V_K is a set of kernel nodes (e.g., convolutions or activation operations), $E_R \subset V_M \times V_K$ is a set of read operations, and $E_W \subset V_K \times V_M$ is a set of write operations. We can derive a CG composed only of operations by *projecting* it onto the kernel partition.

We represent a computational graph G_λ using a third-order **adjacency tensor**

$$A_\lambda \in \{0, 1\}^{|V_\lambda| \times |V_\lambda| \times |K|}\,,$$

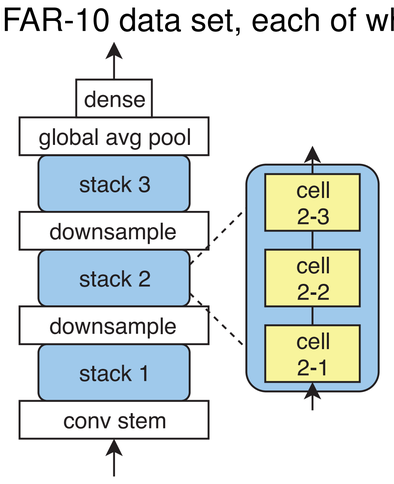
where V_λ is the set of operations building up G_λ, K is the collection of operation types, and $|\cdot|$ denotes set cardinality. We compare two graphs $G_{\lambda_1}, G_{\lambda_2}$ by using two classes of distances:

- probabilistic differences comparing the distributions of operations usage (symmetricised Kullback-Leibler, Jensen-Shannon, Hellinger);
- a transformation distance capturing the cost of transforming one graph into another; the cost model is defined heuristically.

Clustering algorithms (k-means, spectral clustering) create **groups of objects that are similar under the chosen distance**.

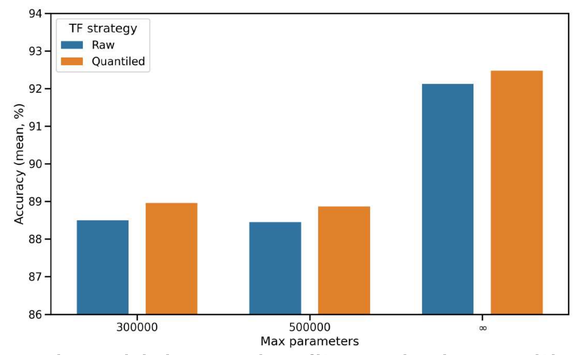
NAS-Bench-101

NAS-Bench-101 has been the first data set of DNN architectures developed to foster NAS research. It contains over $400,000$ unique architectures to solve the CIFAR-10 data set, each of which is annotated with its task accuracy.



NAS-Bench data sets utilise **cell-based parametrisations** to describe different architectures. As shown in the picture to the left, NAS-Bench networks are built concatenating several *cells*; the operations that compose a cell can be chosen and configured independently from those of other cells.

As a first step, we ran a replication experiment with TF statistics. As shown by the right-most column in the barplot, **the task accuracy distribution of the DNN population associated with the best TF statistics quantile is better than the task accuracy distribution for the whole population**.

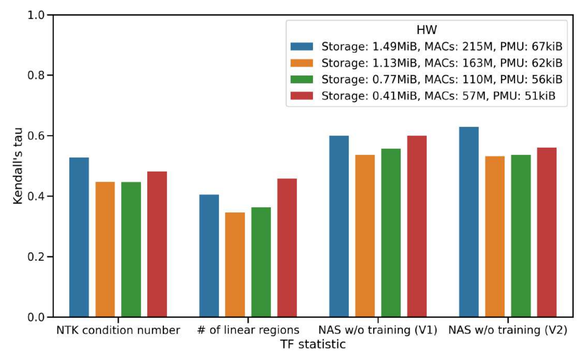


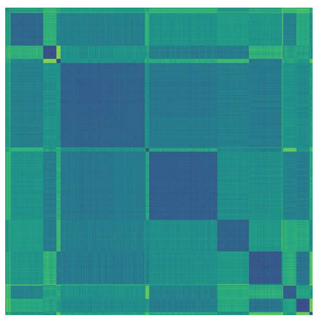
If we constrain the analysis to those networks which can be fit on devices with constrained storage (first and second columns), the predictive power of TF statistics is still preserved.

NAS-Bench-201

The **NAS-Bench-201** contains $5^6 = 15,625$ DNNs, each of which is annotated with its task accuracy over three different image classification data sets (CIFAR-10, CIFAR-100, ImageNet16-120).

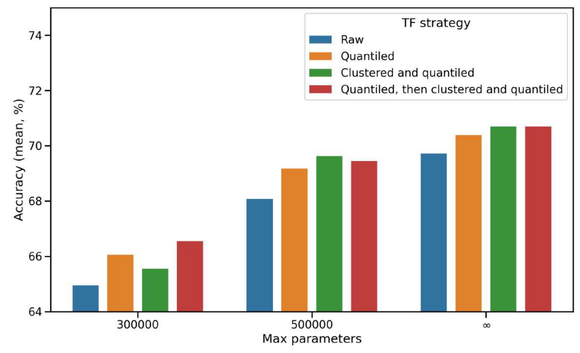
The **predictive power of TF statistics seems independent of HW constraints** even on NAS-Bench-201, as shown by the Kendall's tau coefficients correlating the TF statistics rankings to the task accuracy ranking.



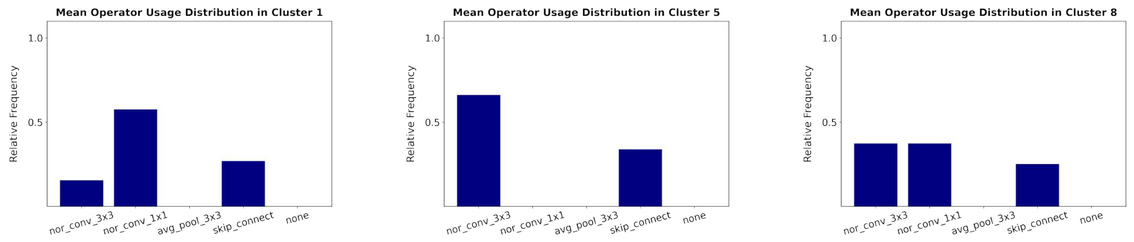


We cluster the CGs of NAS-Bench-201 architectures to **find groups of mutually similar programs**. This similarity between programs also impacts the intra-cluster distributions of TF statistics: indeed, **different clusters exhibit different distributions of TF statistics**.

We can exploit this observation to derive an **unsupervised algorithm to select those program clusters which have the best distributions of TF statistics**. Even a trivial search algorithm such as sampling networks at random from the best cluster can find task-accurate networks.



Looking at the distributions of the operations used by the programs in the best clusters, we see 1) that **good programs do not use disruptive operations** (e.g., pooling) and 2) that **more compact programs mix 3×3 and 1×1 convolutions instead of using only 3×3 convolutions**.



Email: thoriri@iis.ee.ethz.ch