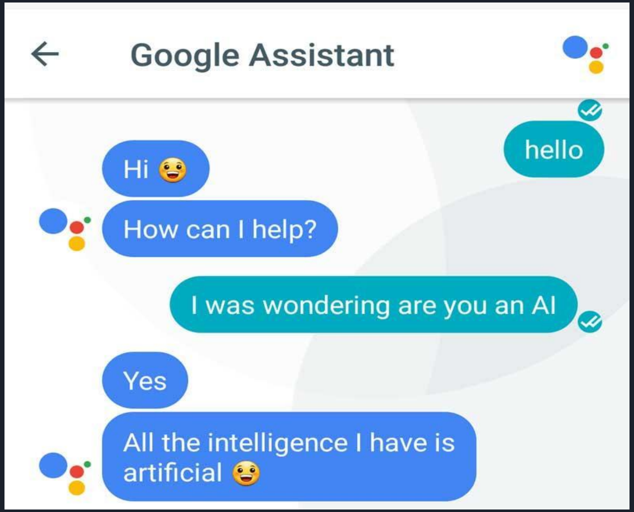
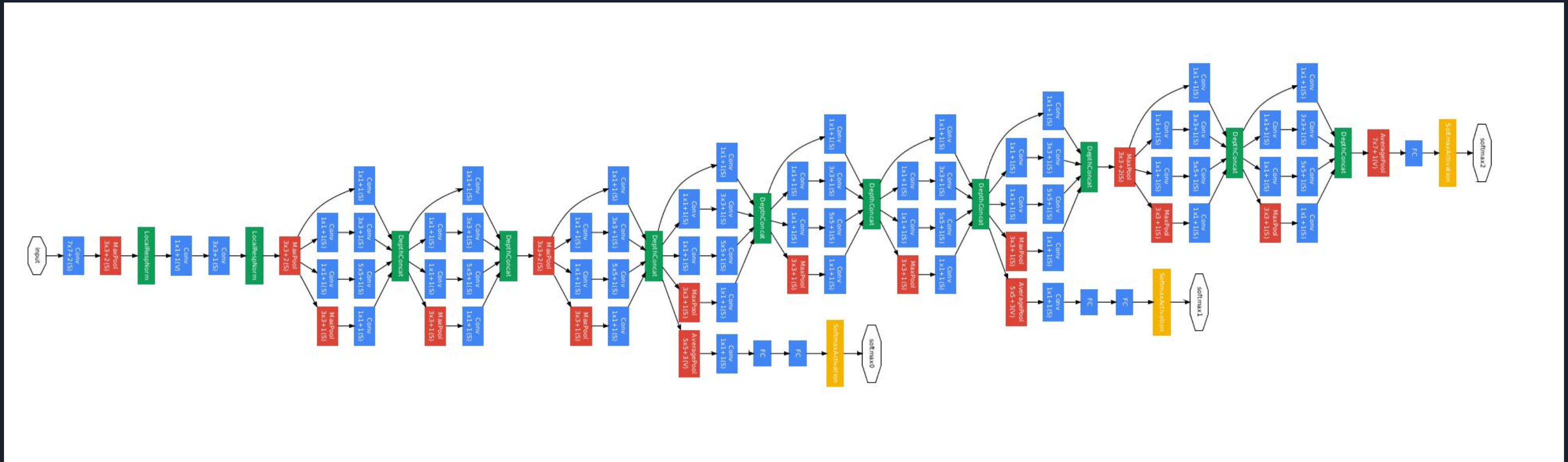
Neural Architecture Search in ARIS (Project DNAD)

George Kyriakides
ge.kyriakides@uom.edu.gr
Konstantinos Margaritis
kmarg@uom.gr

Deep Learning is Great



But Complicated

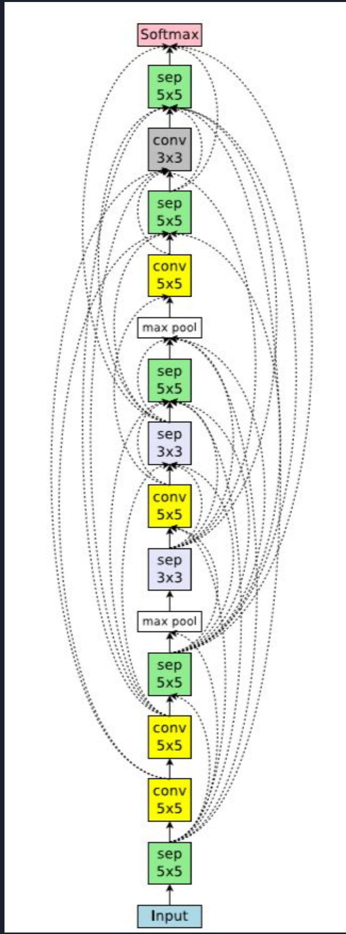


Enter Neural Architecture Search (NAS)

- Utilizing optimization methods in order to find neural network architectures
- Global search space: search for the entire network structure (Exploration)
- Cell search space: search for repeating blocks with a fixed macro-structure (Exploitation)

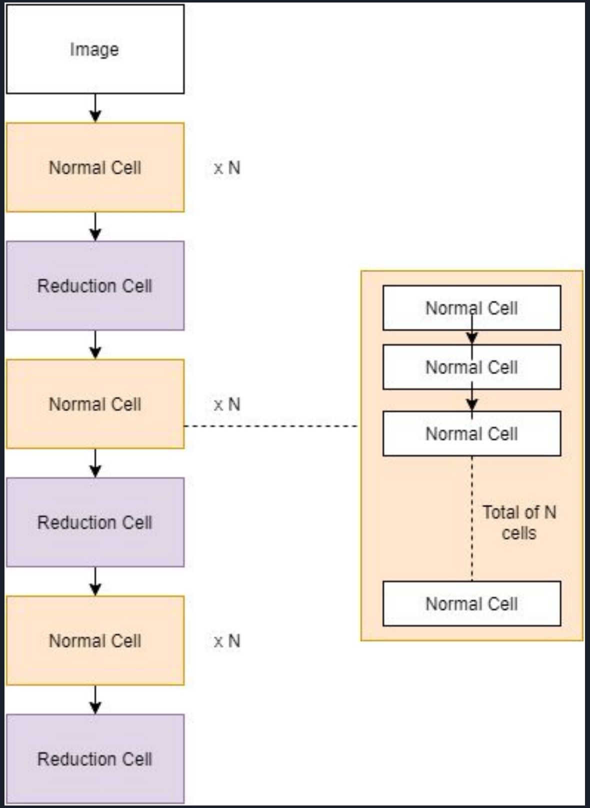
Global Search

- Can "invent" new architectures (in theory)
- Can be applied to new domains
- Less efficient at finding the best architecture



Cell Search

- Leverages previous human experience
- Better at finding state-of-the-art networks
- More difficult to “innovate”
- Requires prior knowledge about well-performing skeletons



Optimization Methods

- Genetic algorithms (CoDeepNEAT)
- Particle swarm optimization (DeepSwarm)
- Reinforcement learning (NAS)
- Sequential model-based optimization (SMBO)
- Gradient Methods (DARTS)
- Many more…

Similarities

- A number of individual networks are evaluated (train/test) ← Very Expensive
- Based on the evaluations, the algorithm selects new networks ← Not so expensive
- Until a stopping criterion is met

Speed-ups

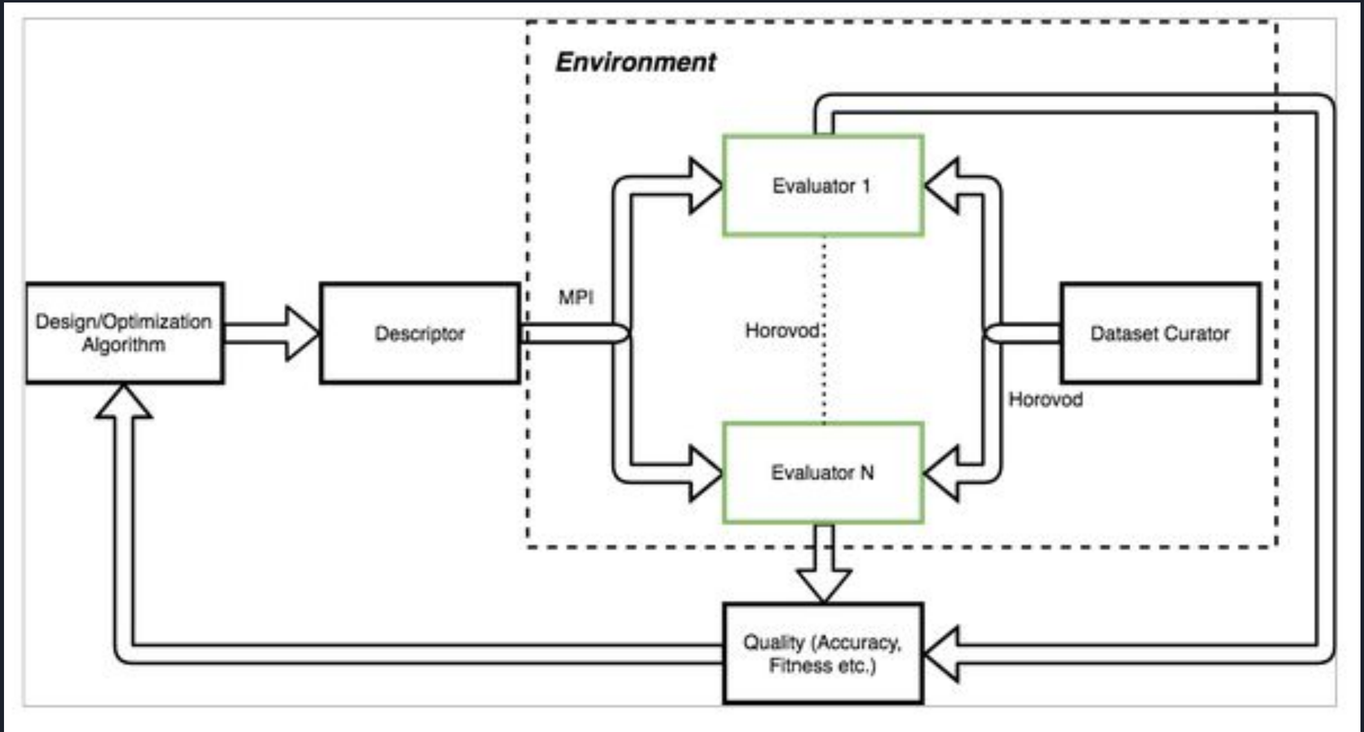
- Algorithmic
  - Evaluate less accurately (Model/Data/Epoch Augmentations)
  - Evaluate less networks (More efficient algorithms)
  - Create predictive models (Bayesian optimization)

- Technical
  - Use faster hardware (GPUs, TPUs)
  - Use more hardware (MPI, Horovood)

Technical Speedups

- MPI (Algorithm Parallelization)
  - Distribute the population amongst N processes (speedup of roughly N) - GA, ES
  - (A)synchronously update a RL controller - A3C, A2C, REINFORCE
  - (A)synchronously update a predictive model - BO
- Horovod (Evaluation Parallelization)
  - Distribute model training amongst N processes (speedup less than N)
  - Depends on networking speed, model size, dataset size
  - Potentially train larger models

Using ARIS



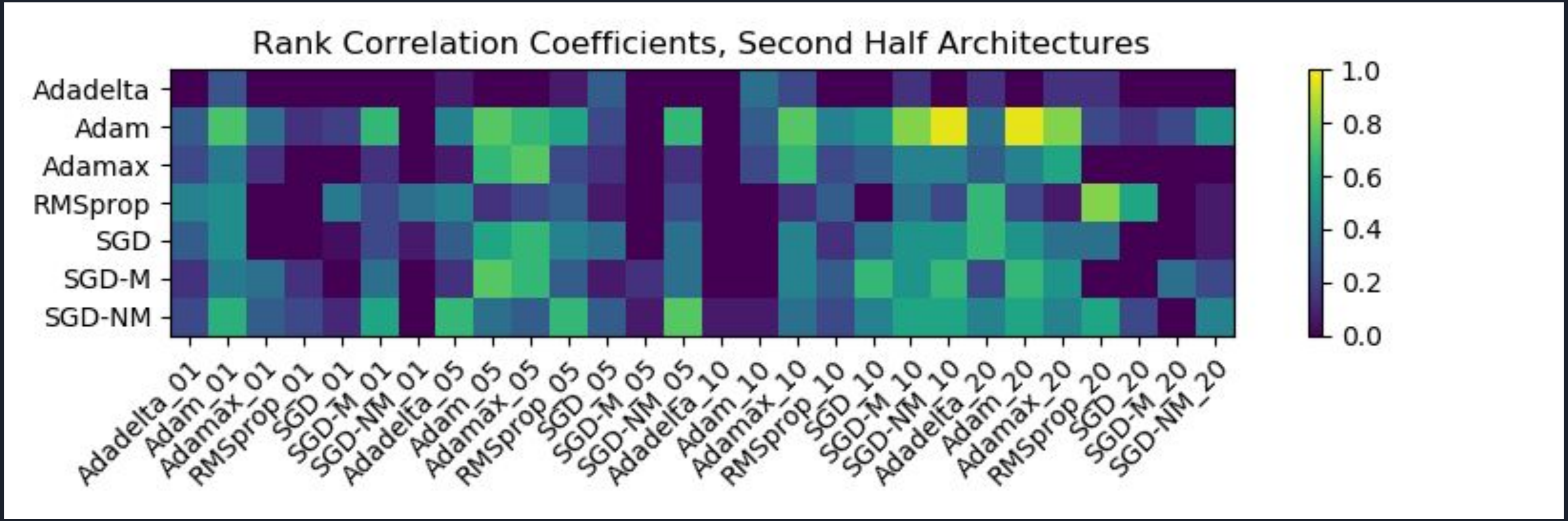
Environment
Evaluator 1
Design/Optimization Algorithm
Descriptor
MPI
Horovod
Dataset Curator
Horovod
Evaluator N
Quality (Accuracy, Fitness etc.)

Some of the Problems Encountered

- Pytorch does not release memory, even if it is not in use (nvidia-smi does not reflect actual usage.
- Using torch.cuda.empty_cache() on a single-node, multi-gpu instance will allocate extra memory on gpu0 and bind subsequent cuda calls to gpu0.
- Load imbalance: Idle workers due to different network sizes and complexities.
- A lot of invalid architectures in global search.

Experiments Conducted (1)

- In-depth study on the effect of utilizing a smaller number of epochs and various optimizers to evaluate candidate architectures



Experiments Conducted (2)

- Regularized evolution of convolutional networks for the Fashion-MNIST dataset on a global search space.

Method	Final accuracy	Search epochs	Training epochs
DENSER [1]	*94.23%* (**94.70%**, test set augment.)	**10**	400
Auto-Keras [7]	93.28%	200	200
DeepSwarm [3]	93.56%	50	100
NASH [5][a]	91.95%	20–105	205
REMNet-256	**94.46%**	**10**	**20**
REMNet-128	*94.26%*	**10**	**20**

[a] As implemented in [7]

Observations

- Selecting and defining search space is the most important aspect of NAS, in terms of end results (Global/Cell, available layer options, regularizations etc.)
- Utilizing augmentation techniques can improve search times, as well as the result quality.
- A high discrepancy between search and final training epochs significantly reduces the correlation between relative architecture performance.

- When arbitrary architectures are generated, the way that layers are merged and how the dimensions are preserved is very important for the final result.
  - Zero-padding or interpolation of layer results and pixel-wise sums works for height/width preservation.
  - 1x1 convolutions work for channels, but fixing the number of channels is better when limited resources are available.

Thank You!