DARTS
Differentiable Architecture Search

PROXYLESSNAS

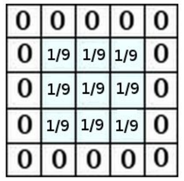
BY AAMIR JAMAL, PAWAN SHETTY, ADAM SPINDLER

INTRODUCTION

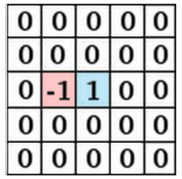
- Neural networks are complicated systems and require a lot of tuning to work properly on certain tasks such as image classification or text classification.
- NAS – Neural Network Architecture Search, are algorithmic solutions to automate the manual process of architecture design.
- Reinforcement learning has been the most popular for the task but is computationally expensive requiring on the order of thousands of GPU hours to find a suitable setup.
- The two papers examined in this work, DARTS and ProxylessNAS, propose techniques to create a continuous loss function that includes a representation of the network architecture so that gradient descent can be used to find an optimal architecture in much less time.
- Since, both the papers are attempting to find the optimal Neural Network Architecture for CNN's (DARTS and Proxyless) and RNN's (DARTS Only). Hence, it would help to briefly overview the functioning of these two varying Neural Networks.

CNN: CONVOLUTIONAL NEURAL NETWORKS

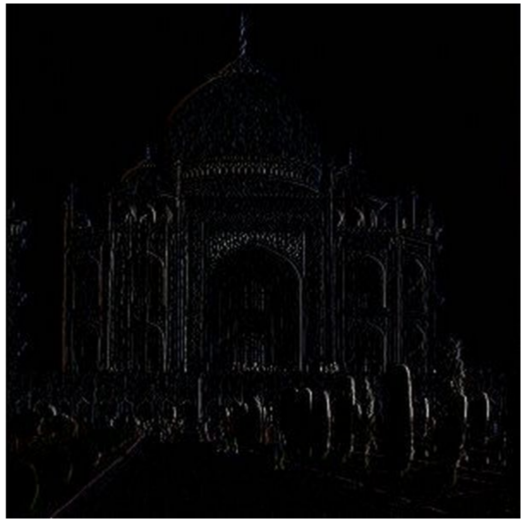
- A Convolutional Neural Network (CNN, or ConvNet) is a class of deep neural networks, most commonly used to analyze visual imagery. They usually consist of multiple hidden layers.
- **Convolutional Layers**
  - Emulates a single neuron. Basically, an N x N Matrix that are used as a feature detector. Output is a feature map which is a representation of the image.
- **RELU Layers**
  - Used to introduce nonlinearity to a system that has been computing linear operations in the previous layers.
- **Pooling Layers**
  - Used to reduce the dimensions of the data by combining output of region in a feature layer together. The combination is done by calculating the average of the previous activations, or taking their averages
- **Fully Connected Layers**
  - Same as the traditional multi-layer perceptron neural network (MLP). The flattened matrix goes through a fully connected layer to classify the images.



0	0	0	0	0
0	1/9	1/9	1/9	0
0	1/9	1/9	1/9	0
0	1/9	1/9	1/9	0
0	0	0	0	0

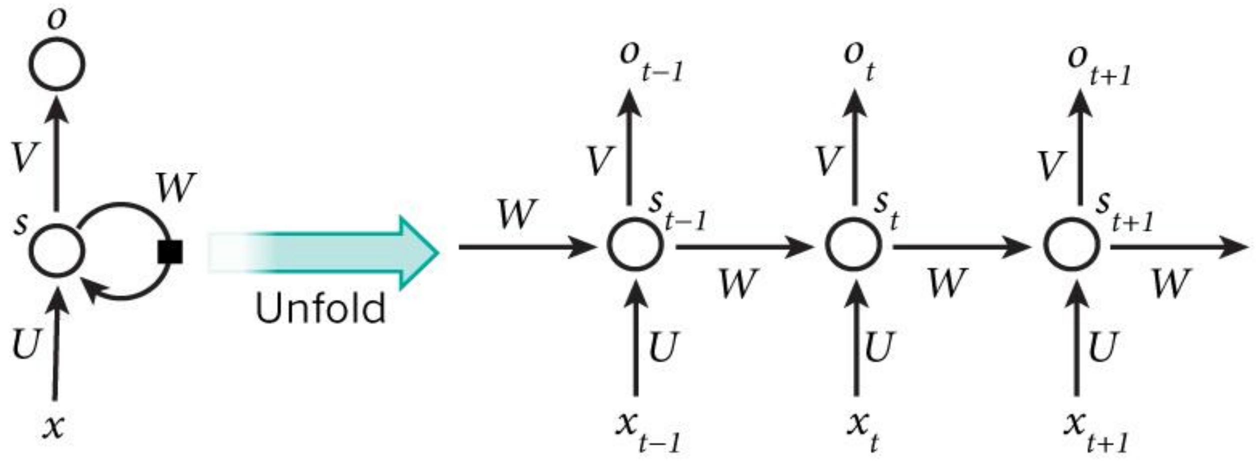


0	0	0	0	0
0	0	0	0	0
0	-1	1	0	0
0	0	0	0	0
0	0	0	0	0



RNN: RECURRENT NEURAL NETWORKS

- Recurrent Neural Network (RNN are widely used for language modeling.
- It captures information about what has been calculated so far.
- We can think of RNNs as multiple copies of the same network, each passing a message to a successor in a sequence.
- Each element performs the same task with the output being dependent on the previous element's computation.
- The input to the hidden node in an RNN is both the input at the current timestep, and the value of its own activation in the previous timestep.



DARTS (DIFFERENTIABLE ARCHITECTURE SEARCH)

Existing state of the art NAS Algorithms

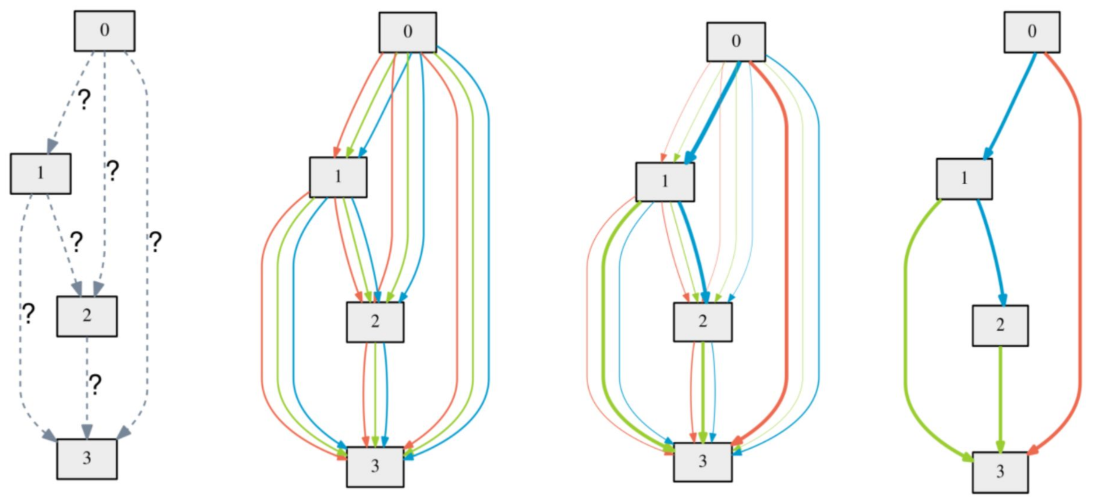
- The existing architecture search algorithms are computationally demanding.
- The search is treated as a black-box optimization problem over a discrete domain.
- A large number of architecture evaluations are required to find the most efficient architecture.

DARTS

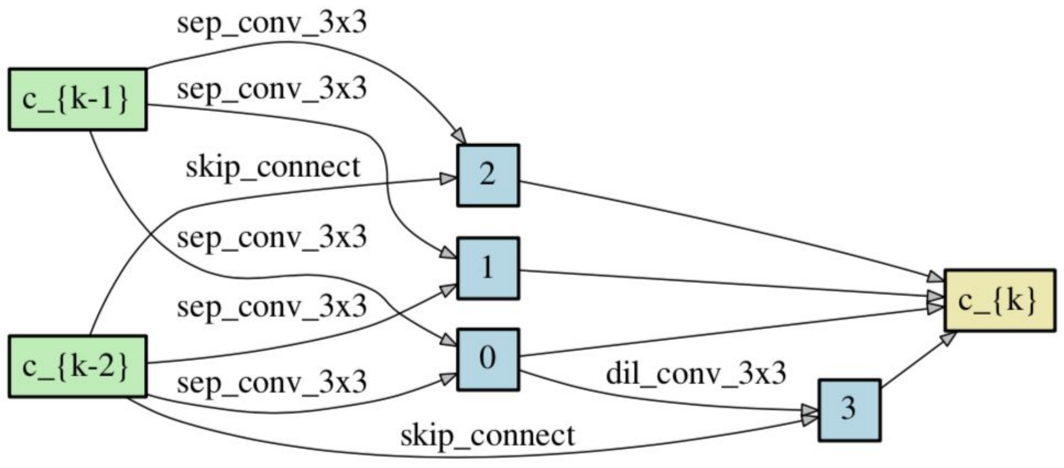
- DARTS uses a continuous search space.
- It performs gradient descent based on the the Validation set performance to optimize the Architecture.
- Does not fine-tune specific aspects of the architecture, but instead learns the entire architecture building block.
- Simpler than many existing NAS approaches and can handle both CNNs and RNNs.

DARTS - Cell

- A cell is a directed acyclic graph with N nodes.
- A node is the output of an operation (like a feature map)
- The edges are the operations (like convolution, max pooling, zero, ReLU) themselves.
- It **optimizes**:
  - The **weights** for the architecture to reduce the training loss on that network.
  - The **architecture**, which is evaluated by taking the validation loss of the network with the current architecture using the weights that it optimized already.



Overview of DARTS (General Cell Structure)



Example of a learned cell on CIFAR-10

DARTS

- The experiments consist of two stages,
  - **Architecture search:** Cell Architectures are searched and the best one are selected
  - **Architecture evaluation:** Best cells are selected and used to construct larger architectures.
- Transferability is also tested on Large Datasets, such as ImageNet and WikiText-2 (WT2).
- Once the architectures are selected, the architectures are initialized with random weights, trained from scratch and results are reported based on the performance on the test set.
- The best cells from the selected architectures are evaluated on ImageNet (mobile setting) and WikiText-2, by stacking them on top of each other.

DARTS

Results

- DARTs was able to provide the lowest error rates on CIFAR-10 and text perplexity on PTB respectively, when compared to other state of the art NAS Algorithms.
- DARTS requires the least number of parameters and the lowest number of GPU days compared to other State of the art NAS Algorithms.
- On CIFAR-10, DARTS required three order of magnitude less computation resources to achieve comparable results to other state of the art architectures (that is 1.5 or 4 GPU days vs 2000 GPU days for NASNet and 3150 GPU days for AmoebaNet) and with a slightly longer search time, DARTS outperformed ENAS by discovering cells with comparable error rates but less parameters.
- Experiments with the cell learned on CIFAR-10 can be transferred to ImageNet. Cells discovered by DARTS also transfers to WikiText2 better than the ones discovered by ENAS.

AmoebaNet, NASNet and ENAS are all state-of-the-art Neural Network Architecture Search Algorithms

PROXYLESSNAS

- DARTS requires a lot of memory
- Similar architecture representation to DARTS
- Trains for specific target hardware
- Over-parameterized network instead of stackable cells
- To reduce training complexity, the paths weights are **binarized** so that only one is used at a time

PROXYLESSNAS

Search Technique

- Freezing weights while training one set of parameters reduces memory

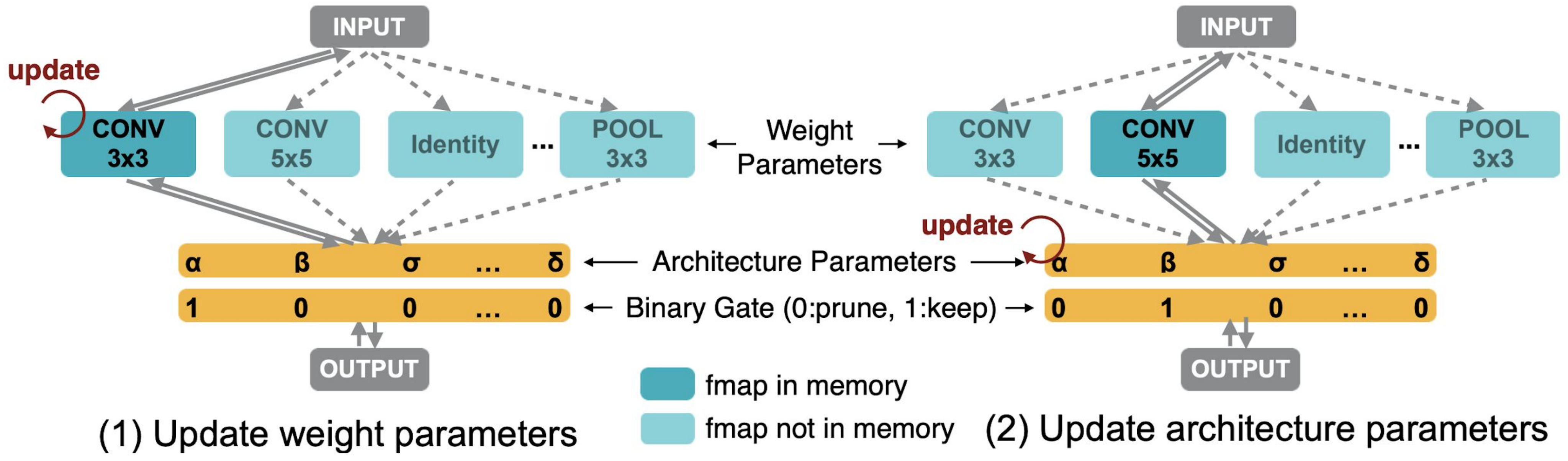
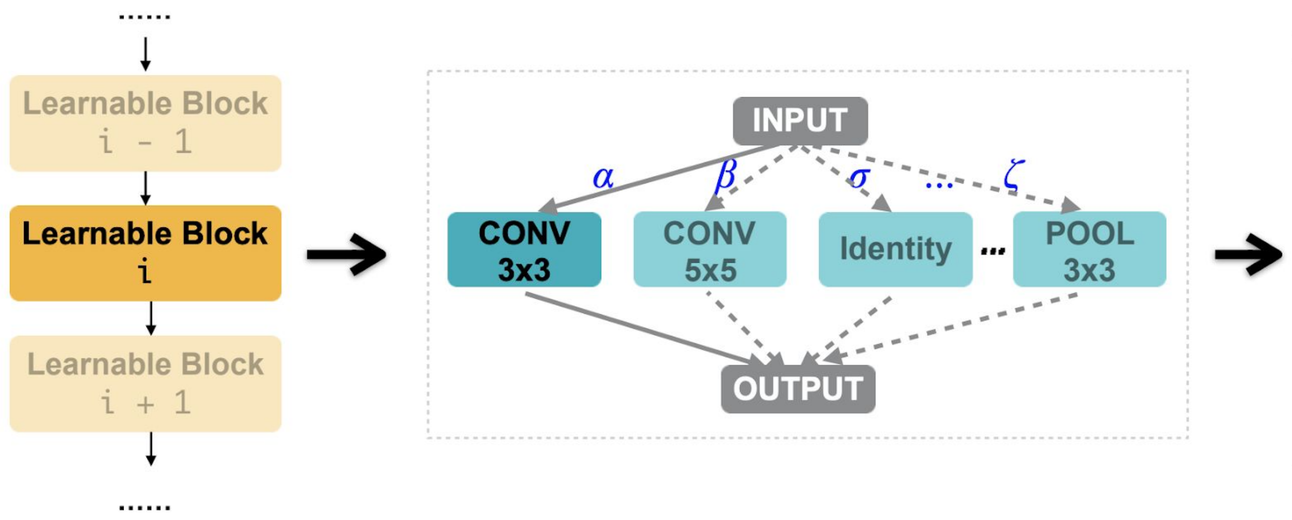


Figure 2: Learning both weight parameters and binarized architecture parameters.

PROXYLESSNAS

Training for specific hardware

- Latency vs. FLOPs
- Regularizing by weights keeps them from getting out of control



$$\begin{aligned}\mathbb{E}[\text{Latency}] = {} & \alpha \times F(\text{conv\_3x3}) + \\ & \beta \times F(\text{conv\_5x5}) + \\ & \sigma \times F(\text{identity}) + \\ & \ldots\ldots \\ & \zeta \times F(\text{pool\_3x3})\end{aligned}$$

$$\mathbb{E}[\text{latency}] = \sum_i \mathbb{E}[\text{latency}_i]$$

$$Loss = Loss_{CE} + \lambda_1 ||w||_2^2 + \lambda_2 \mathbb{E}[\text{latency}]$$

Figure 3: Making latency differentiable by introducing latency regularization loss.

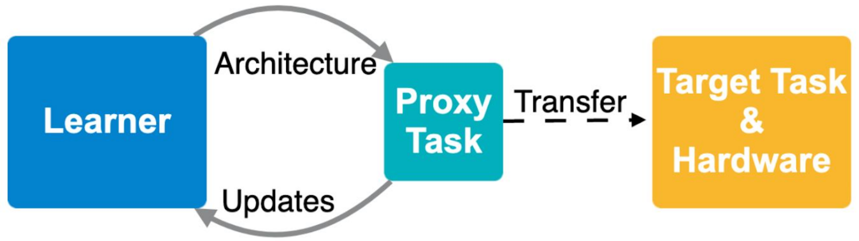
PROXYLESSNAS

Results

- CIFAR-10 2.08% error compared to 2.13% by AmoebaNet-B but uses only 5.7M parameters to 34.9M
- ImageNet GPU 75.1% accuracy compared to 72%
- ImageNet mobile when evaluations must be under 80ms, it ran in 78ms and was 74.6% accurate.
- Without latency regularization, the architecture search finds one with 158ms of latency on the Pixel 1, so it is necessary to explicitly train for it.
- Less computationally expensive, 200 GPU hours while next best is 40,000

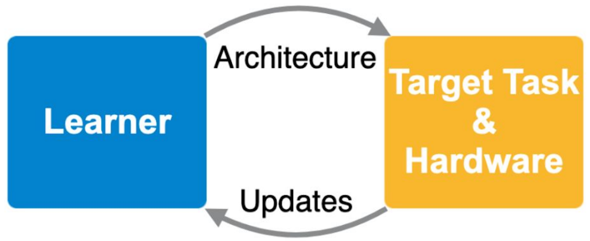
DARTS

- Differentiable architecture search is much faster than previous techniques, but it requires a lot of GPU memory.
- DARTS train the architecture on a smaller, proxy, task like a subset of CIFAR-10.
- The trained architecture is then used on the true target task such as ImageNet.
- Architecture is not tailored to the true task that it was meant to solve so it is unlikely to be the best architecture for the job.



PROXYLESSNAS

- ProxylessNAS learns different architectures for different target hardware and target task.
- ProxylessNAS trains the architecture on the target task and hardware.
- The architecture trained for a GPU can take advantage of the parallelism that the hardware provides and is tested directly on the target task.
- A differentiable function to model the performance of different devices is used, so that the architecture can be learned for a specific device. So the architectures searched will be the best for the task.


Learner
Architecture
Updates
Target Task & Hardware