Neural Architecture Search

Tobias Hörnschemeyer, Nazish Qamar

Introduction

- **Neural Architecture Search** is the task of automatically finding an optimal neural architecture by selecting and combining different basic operations involved in the formation of a deep neural network.

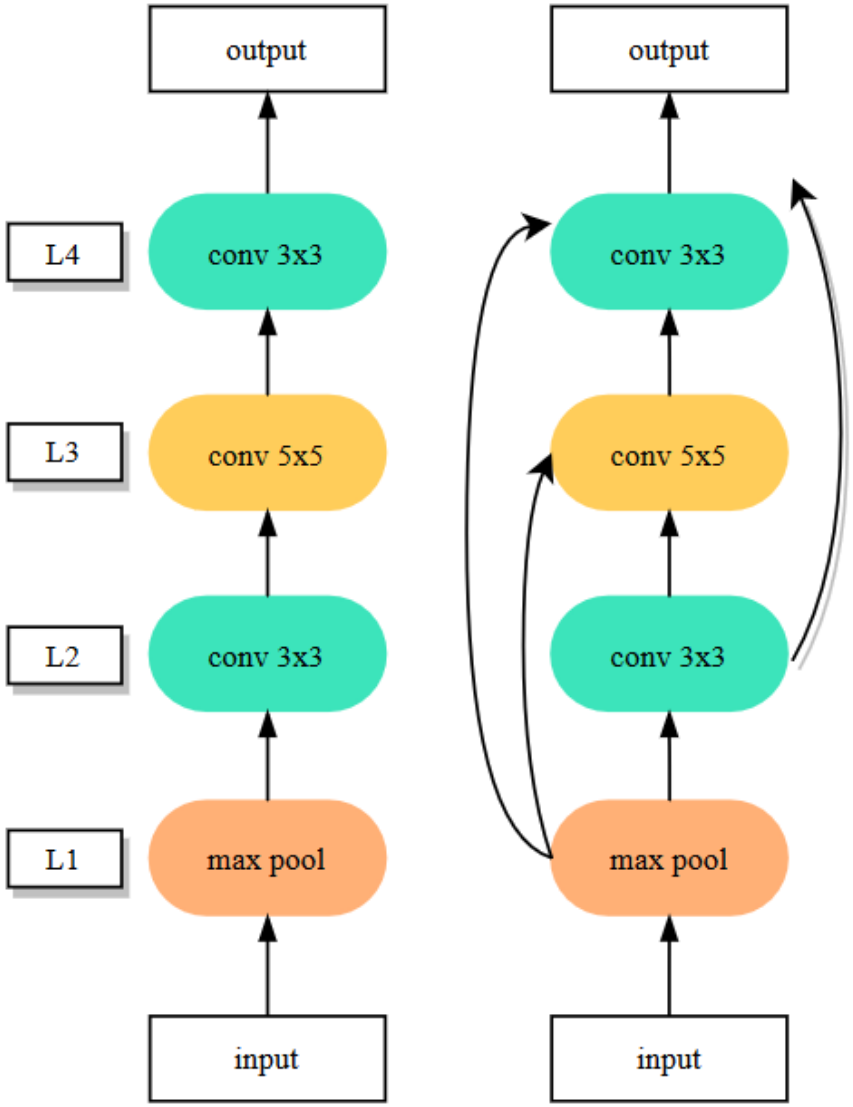
- The selection and combination of these operations vary with the design of a **Search Space** that defines the design principles of neural architectures.
- To improve training times, different **model evaluation methods** can be used

Search spaces

- Entire Structures
- Cell-based
- Hierarchical
- Morphism-based

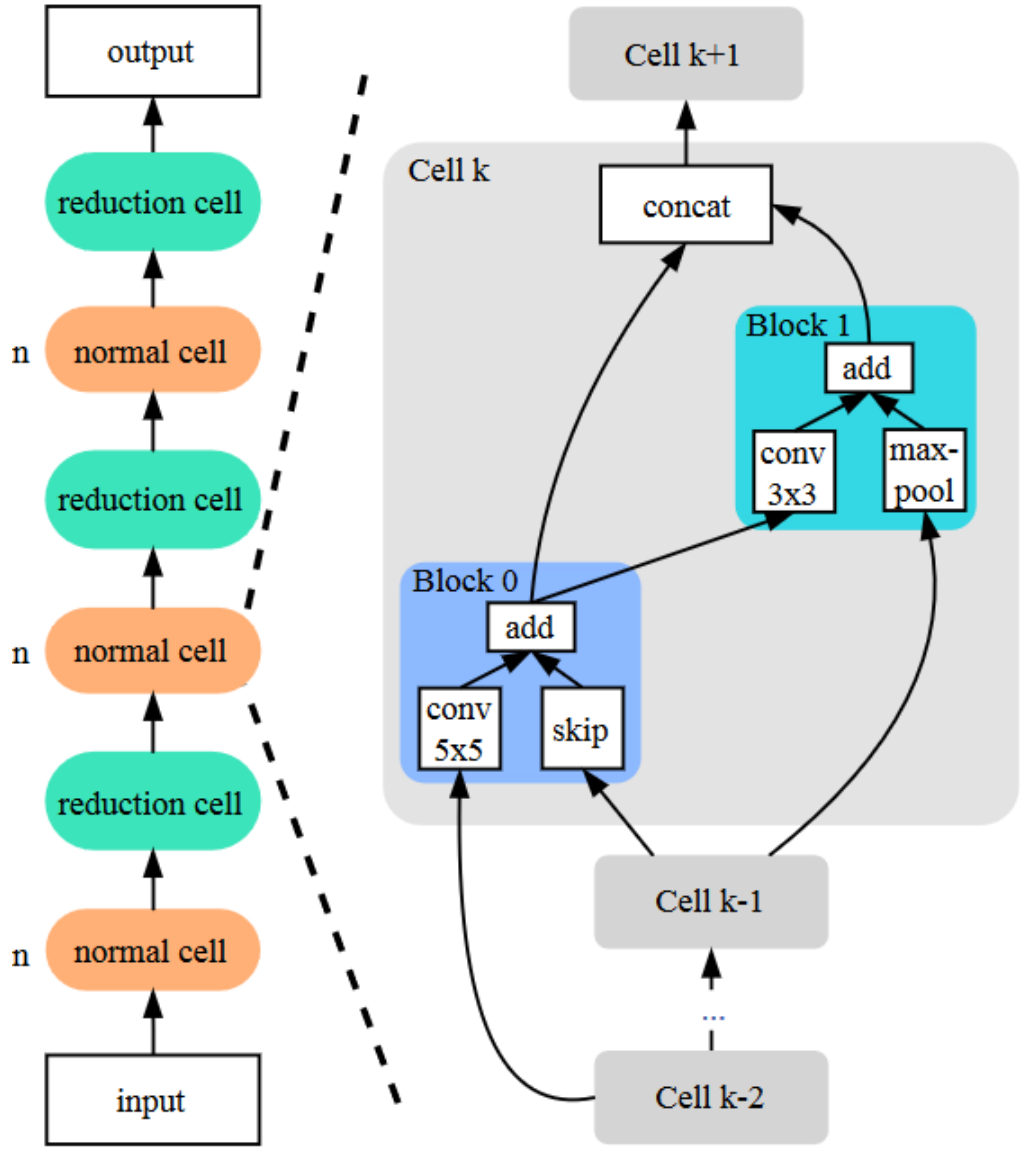
Entire Structures

- All possible architectures in the search space
- Computationally hard to find a good architecture
- Search space might be too large



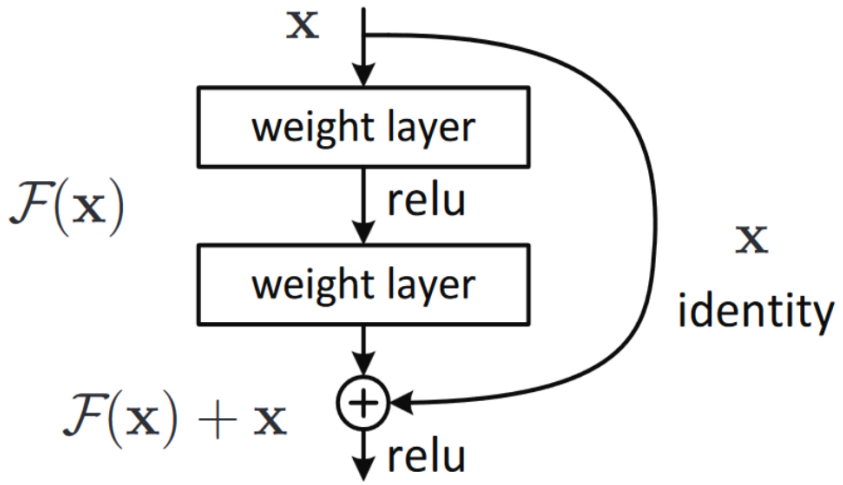
Cell-based

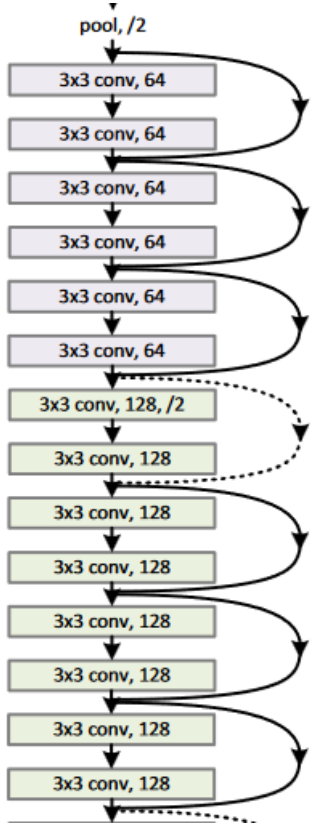
- Many human-made architectures use stacked cells
- Full-architecture search is slow



ResNet

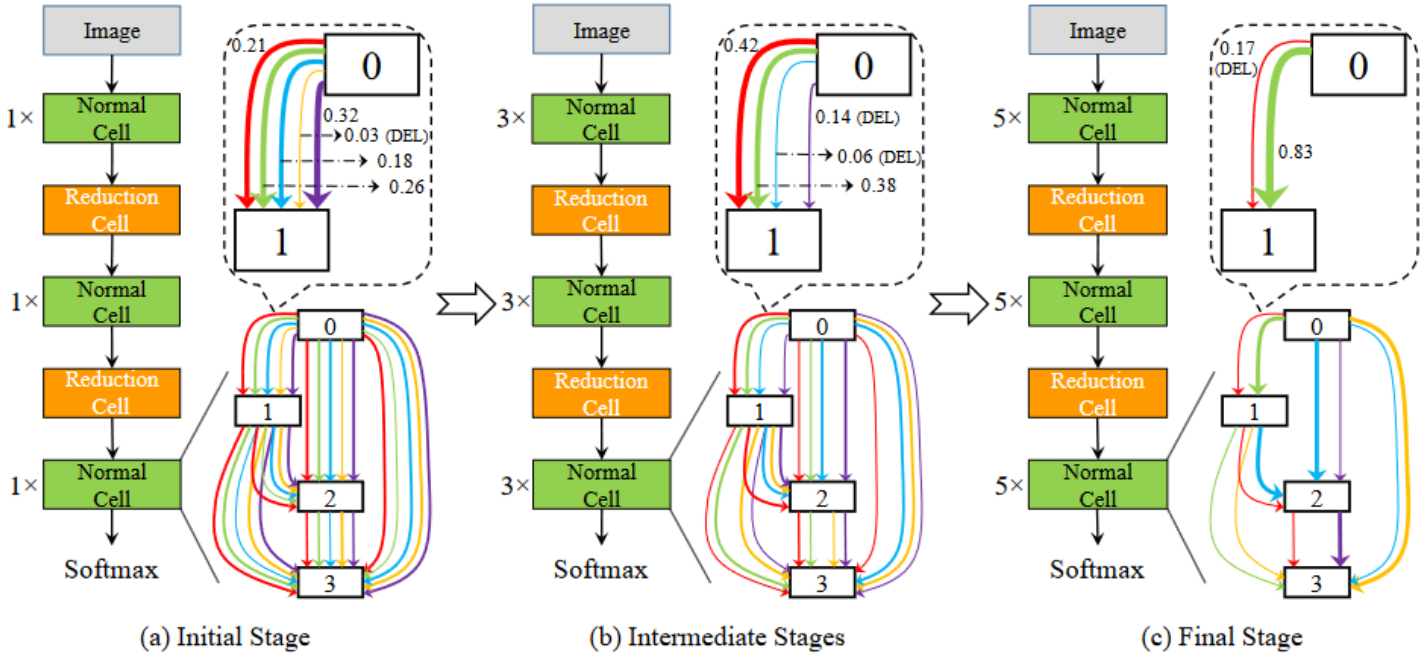
- Stack cells
- Each cell has skip connections to pass gradients
- Cells can consist of multiple convolutional layers



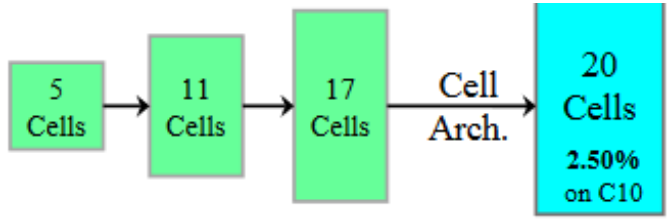


P-DARTS

- Find the architecture of a normal cell using Gradient Descent
- Increase search depth
- Decay dropout
- Regularize skip connections



(a) Initial Stage (b) Intermediate Stages (c) Final Stage



Hierarchical Search Space

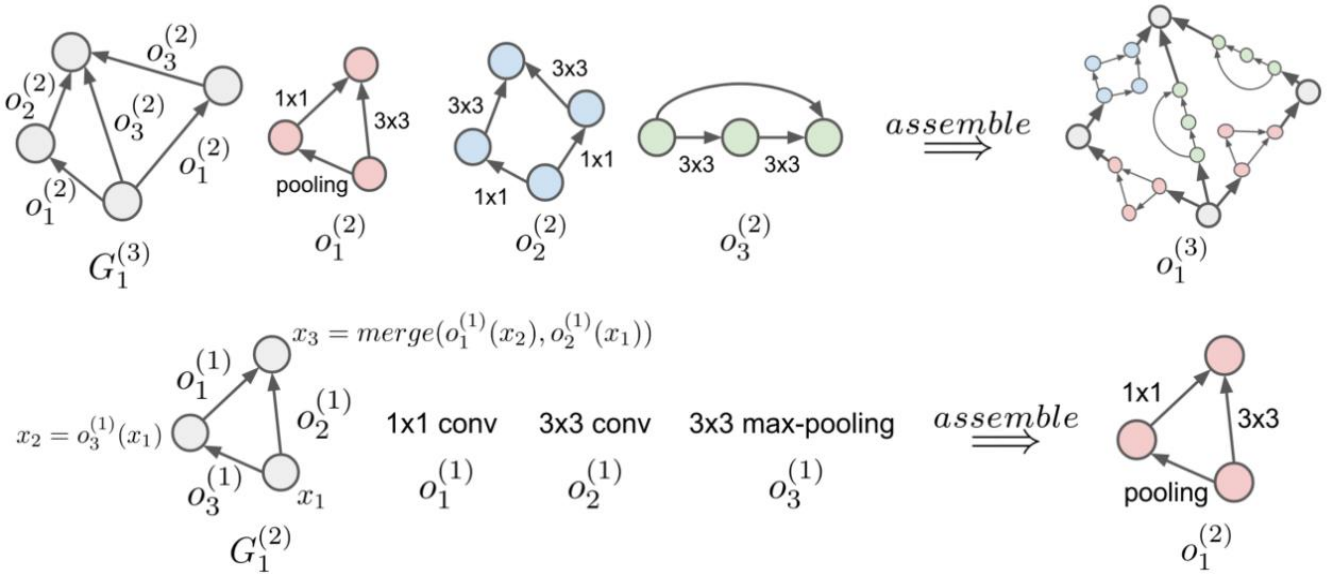
Motivation:

- Cell based approaches follow a two level hierarchy but focus only on the cell level. The network level structure is defined manually.

- Hierarchical based approached jointly learn a combination of repeatable cell and network structures.

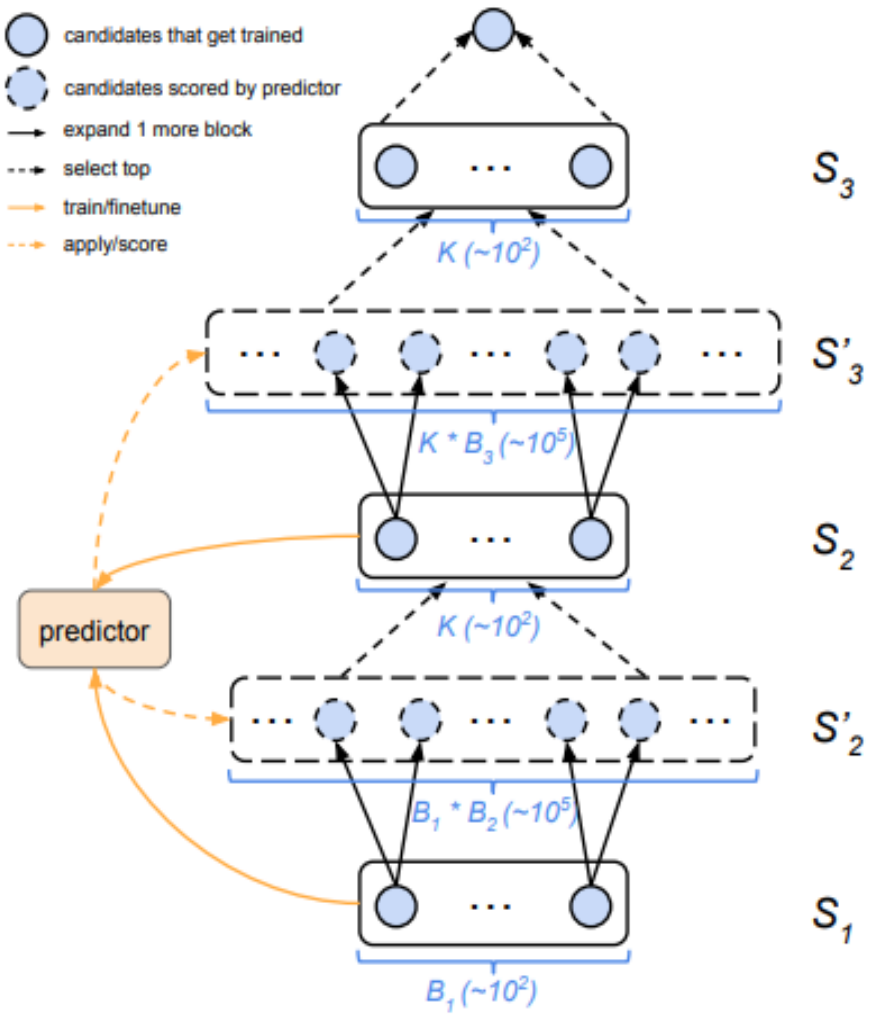
HierNAS (Hierarchical NAS)

- A small set of primitives operations at the bottom level of the hierarchy.
- A hierarchical architechtures is formed by stacking higher level blocks multiple times which are formed using lower level motifs.
- The optimal hierarchical architecture is discovered through evolutionary or random search.

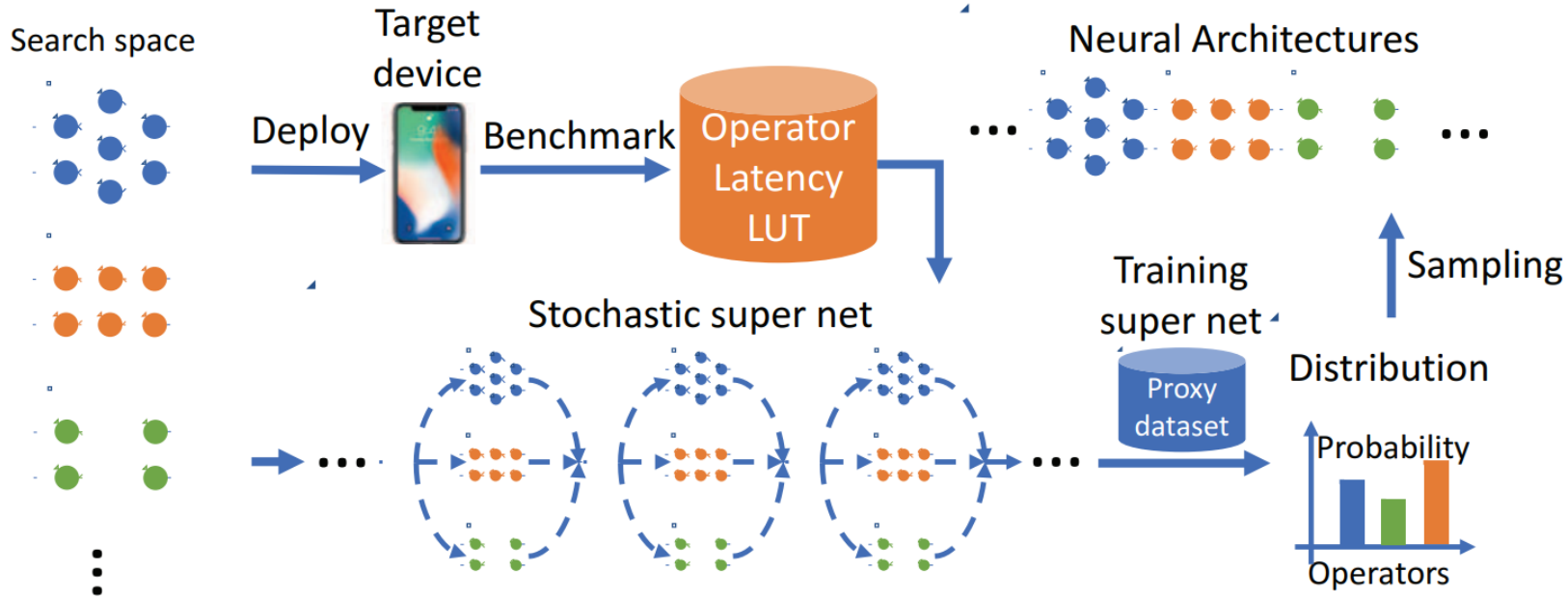


PNAS (Progressive NAS)

- Searches the best space of cell structures, using a progressive search through the space of increasingly complex graphs.
- Learns a cell structure, and then stack this cell multiple times to create the final CNN.
- A learned prediction function identifies efficiently the most promising models instead of exploring all the model at every iteration.

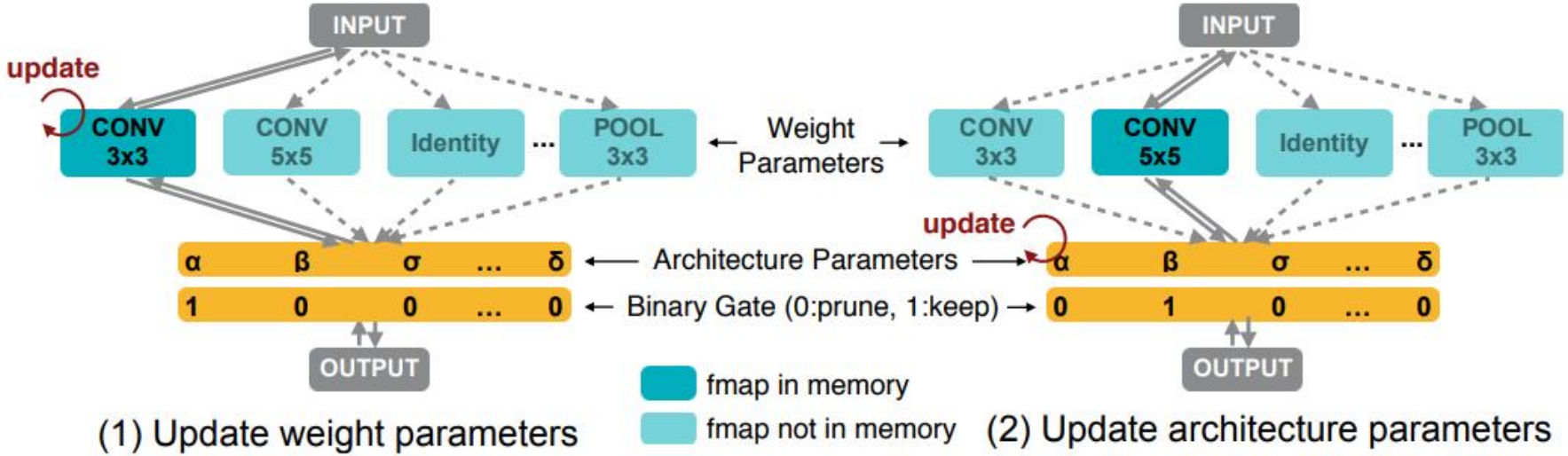


Differentiable NAS (e.g. DARTS)



- DNAS represents the search space by a super net whose operators execute stochastically.
- Instead of solving for the optimal architecture, optimize the probability Pθ of the stochastic super net to achieve the minimum expected loss with a loss function that contains latency term.
- After the training, optimal architectures is obtained by sampling from the architecture distribution Pθ.

ProxylessNAS



- NAS formulated as a path-level pruning process.
- Redundant paths are pruned at the end of training to get a compact optimized architecture.
- Binarized the architecture parameters (1 or 0) and force only one path to be active at run-time.
- A gradient-based approach applied to train these binarized parameters.

Morphism-based

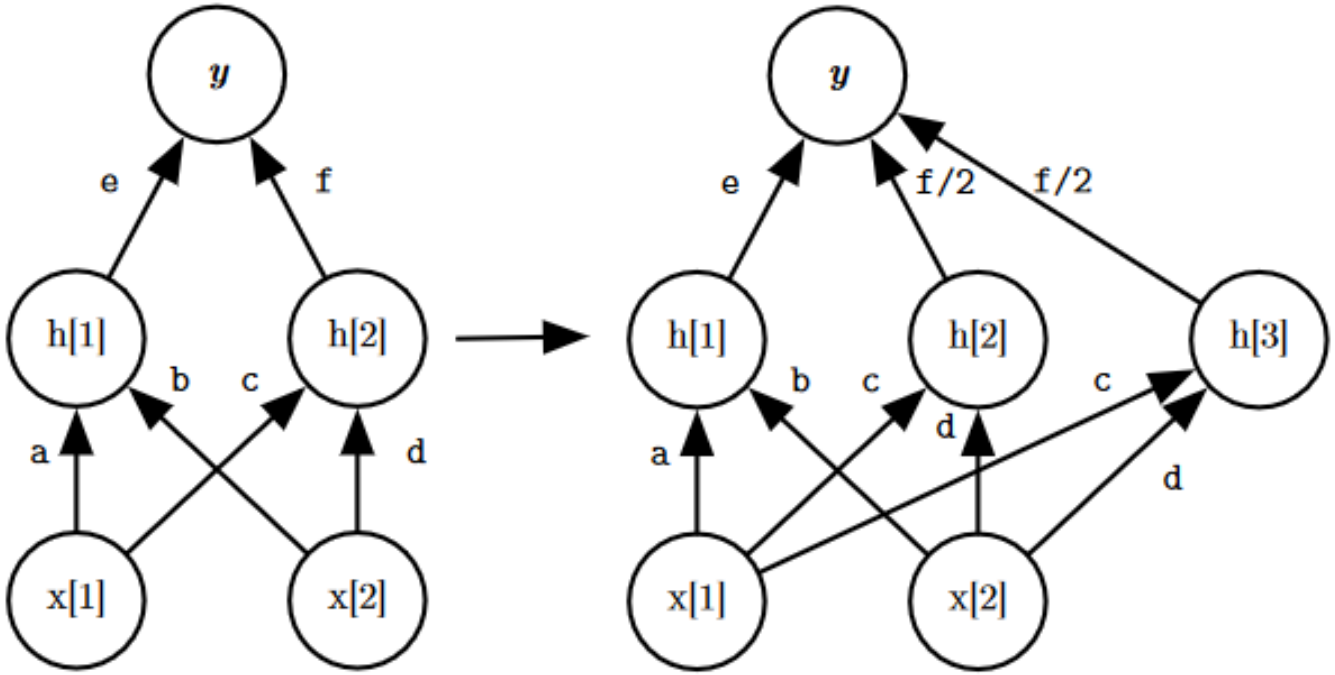
- Change an existing trained architecture
- Keep trained information (weight sharing)
- Only retrain neccessary parts after changes
  - Training of the the new architecture is faster

Net2Net

- Use a small trained network as trainer for a bigger network
- The bigger network training should start with the same loss as the small network
- Might improve training time
- Maybe training from the beginning would end with slightly better results

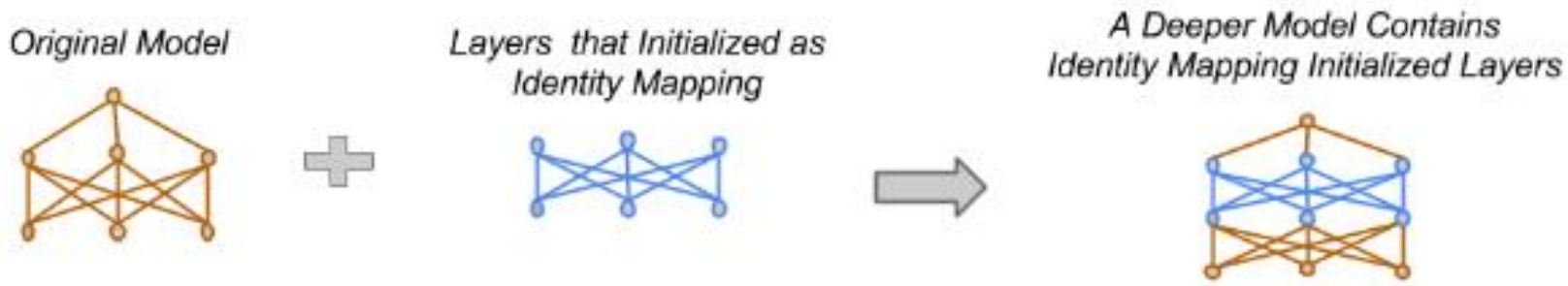
Net2WiderNet

- Increase layer width
- Copy random neurons and normalize output



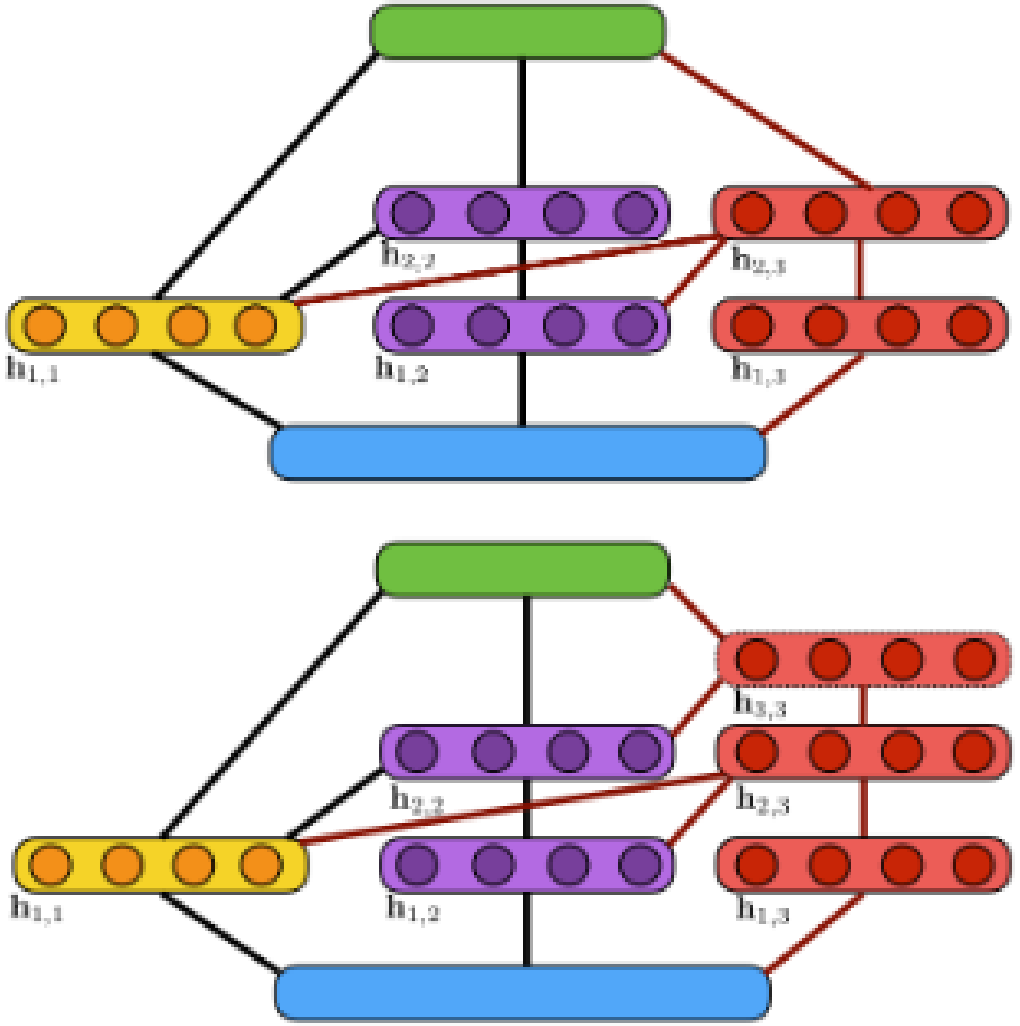
Net2DeeperNet

- Increase the number of layers
- Only works with identity-mapping-able activation functions (ReLu)

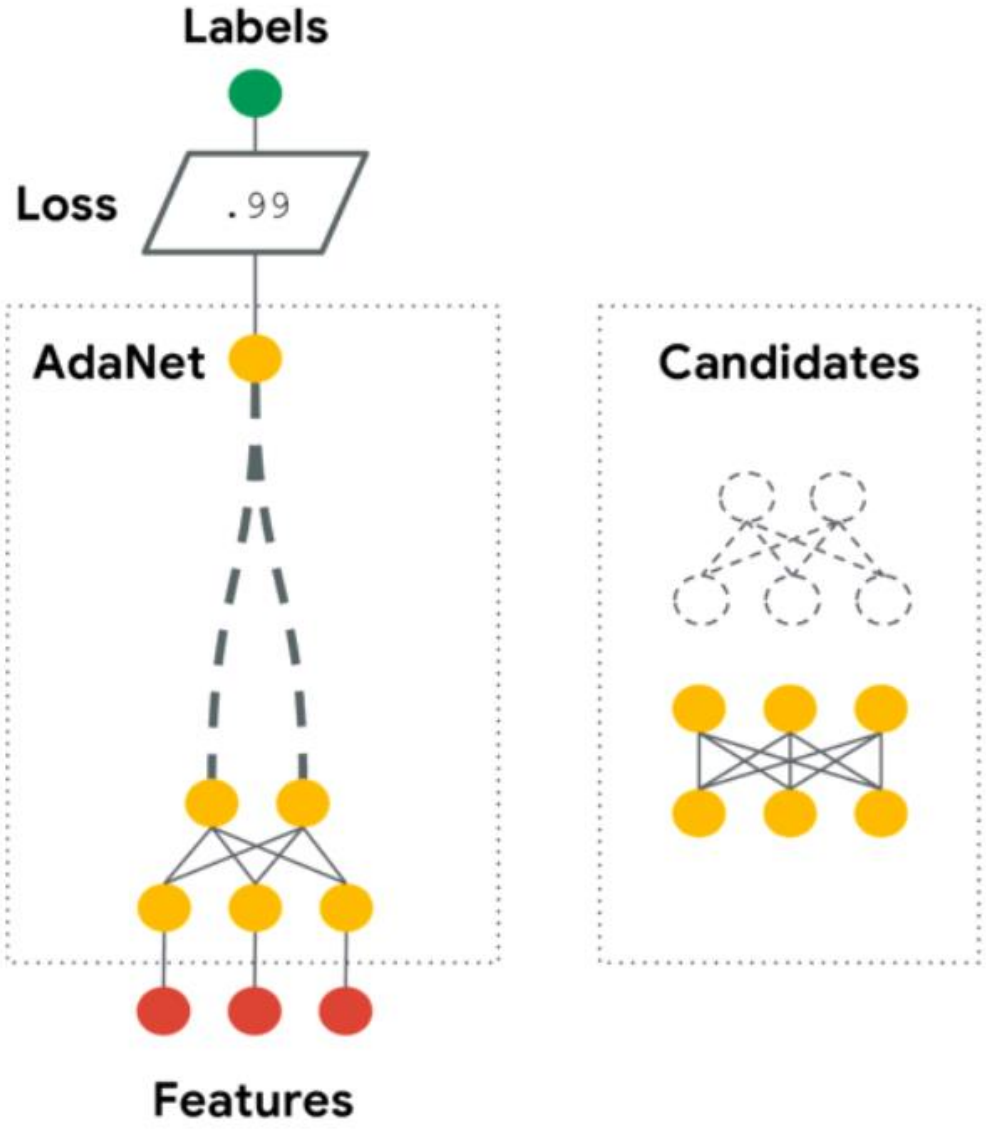


AdaNet

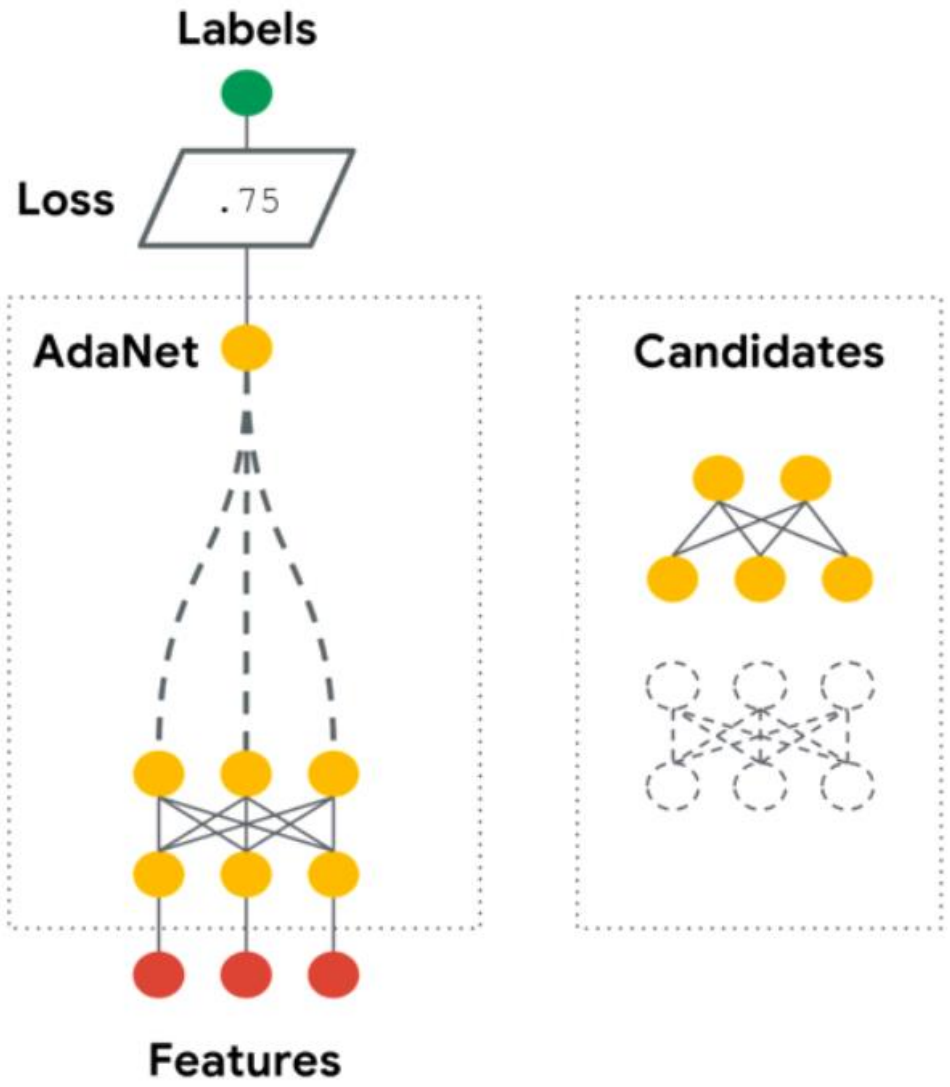
- Multiple parallel sub-networks
- Start with one sub-network
- In each iteration, test a candidate
- Possible candidates:
  - Same number layers
  - One layer more



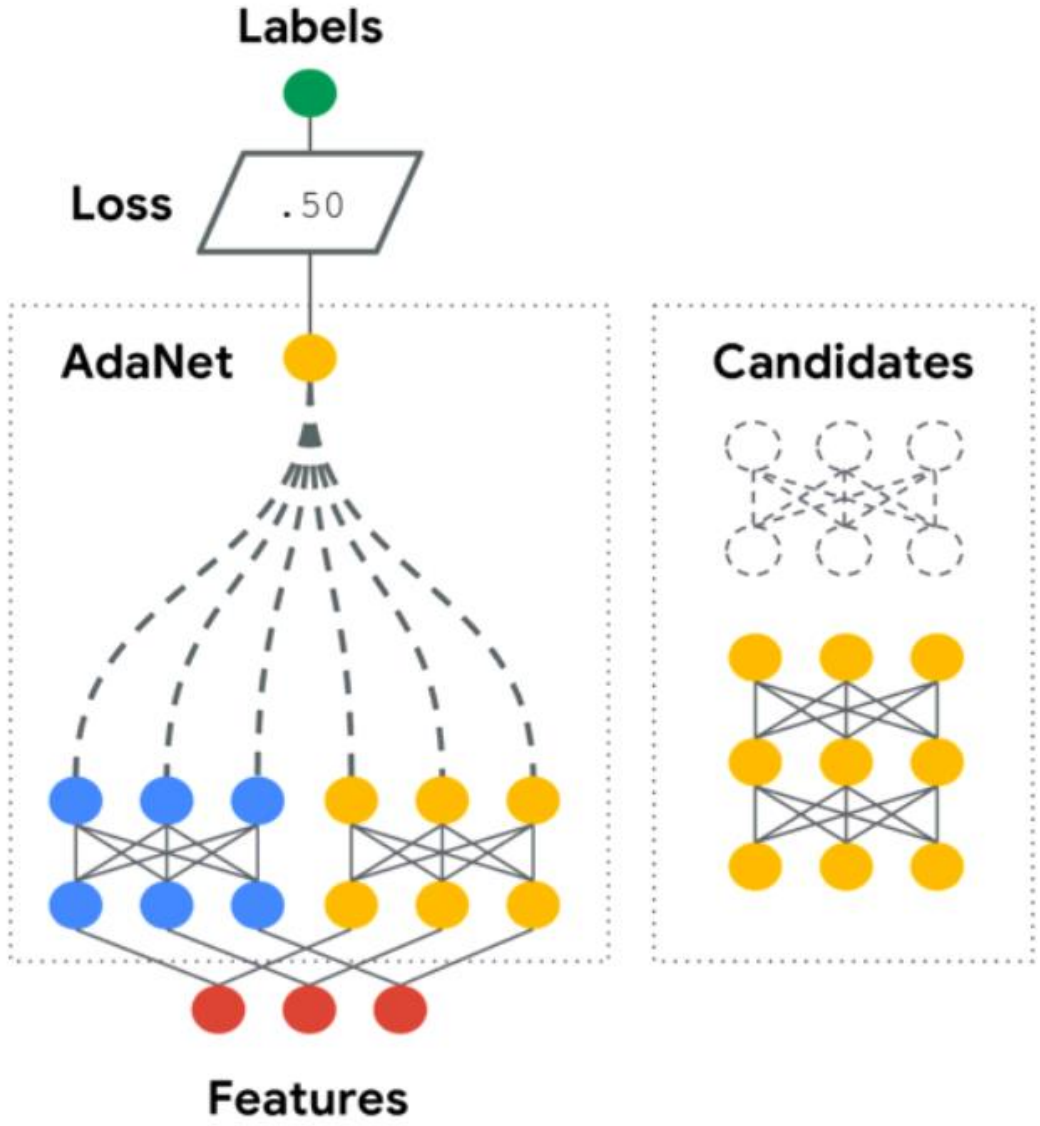
AdaNet



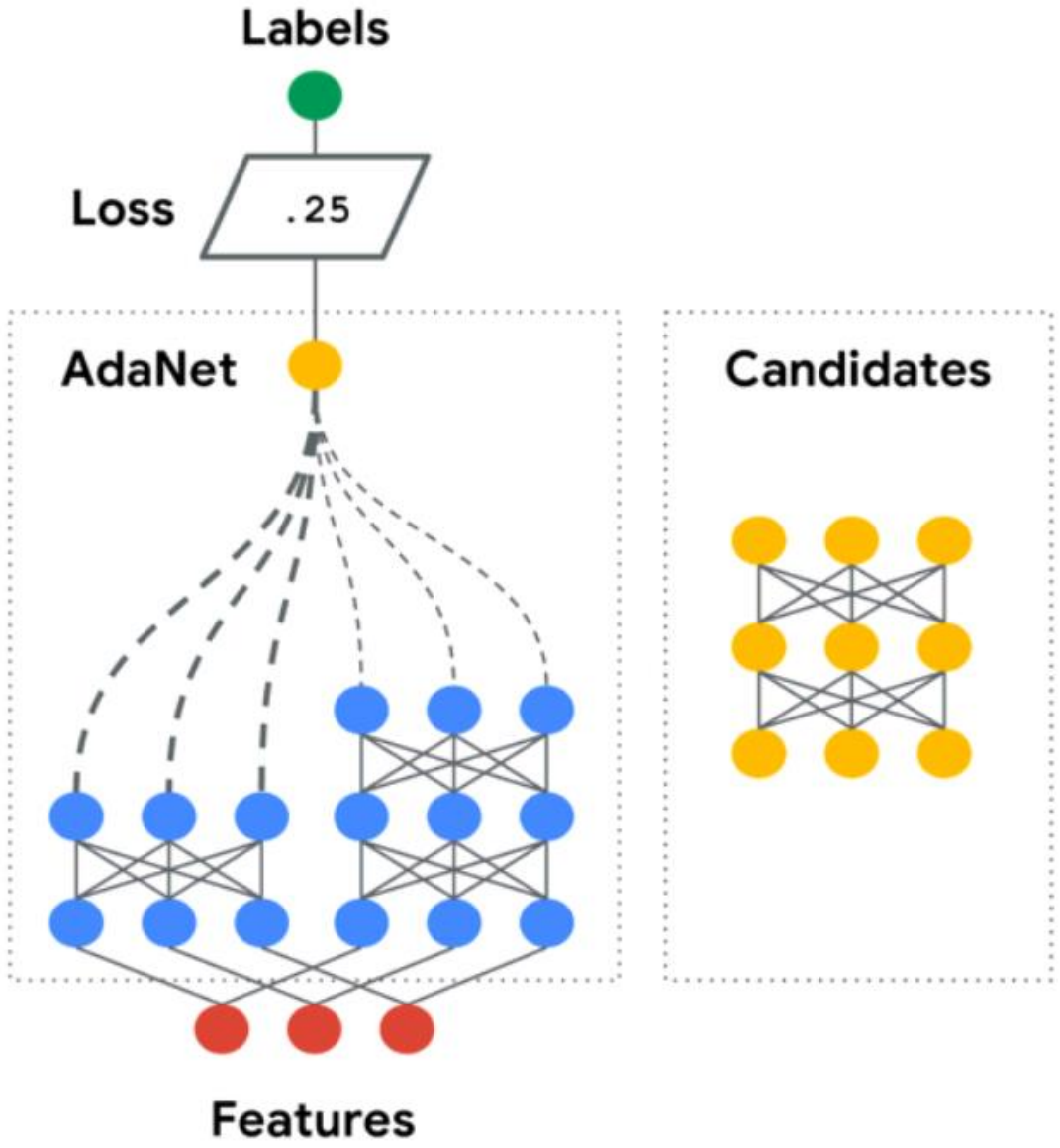
AdaNet



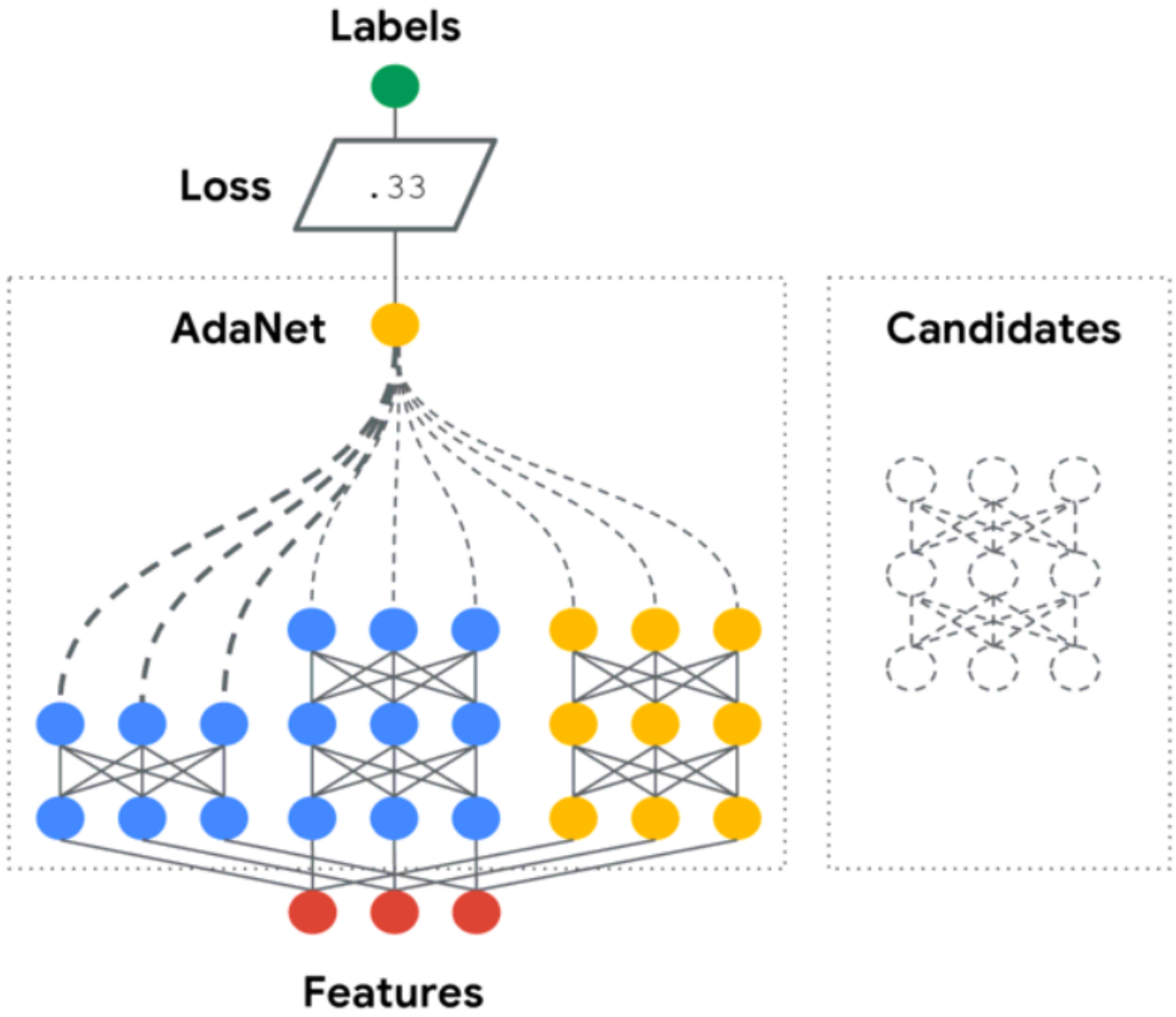
AdaNet



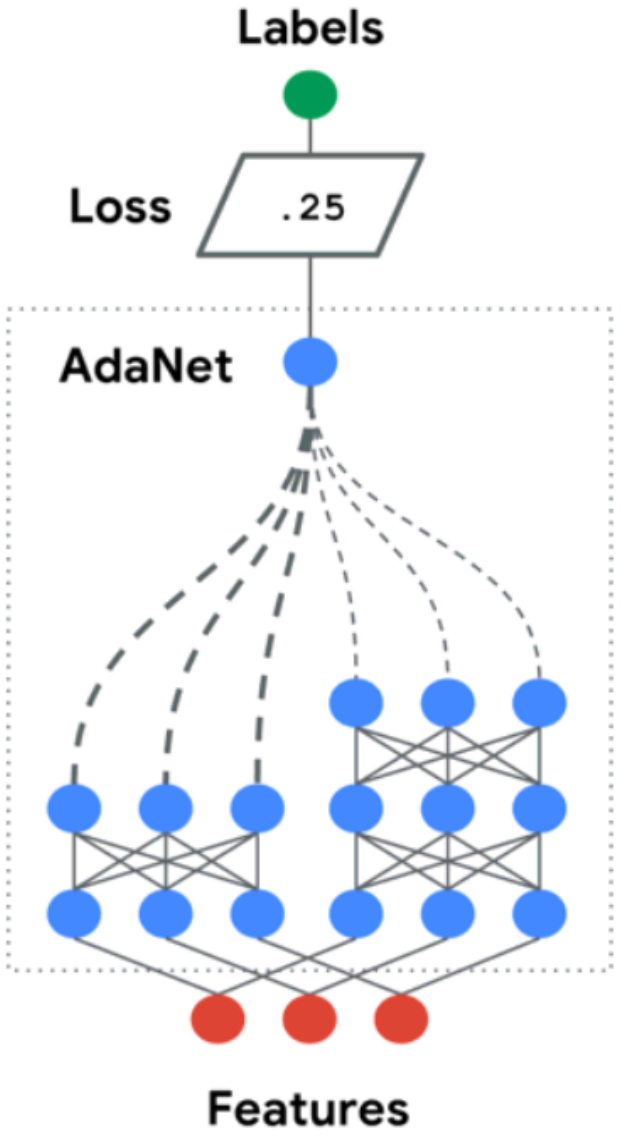
AdaNet



AdaNet

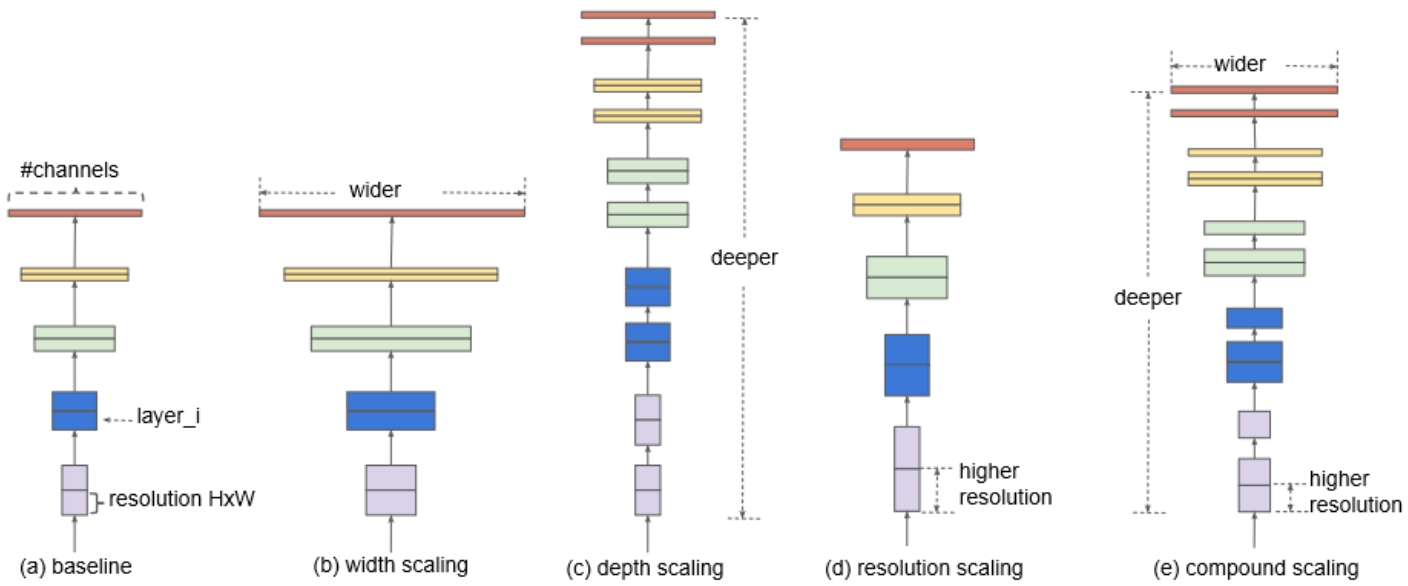


AdaNet



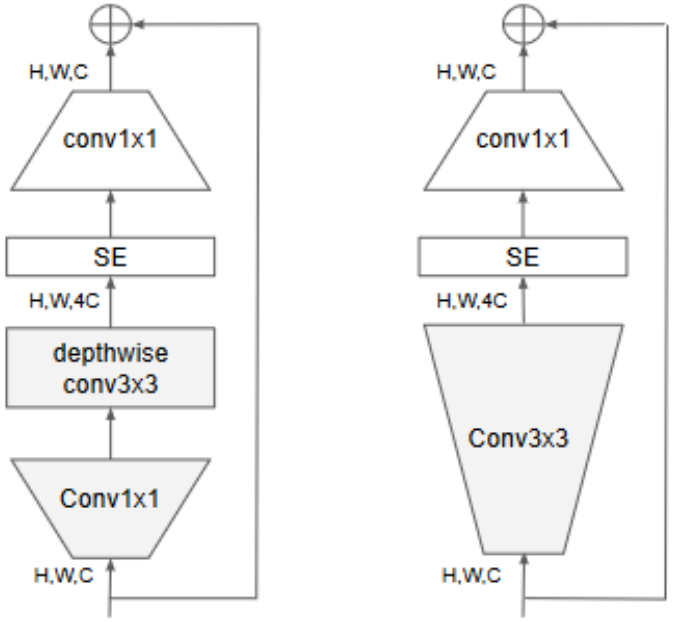
EfficientNet

- Convolutional Network
- High parameter efficiency
- Start with a small network
- Find scaling factors using grid search on baseline network
  - Width
  - Depth
  - Image size
- Scale the network using the scaling factors

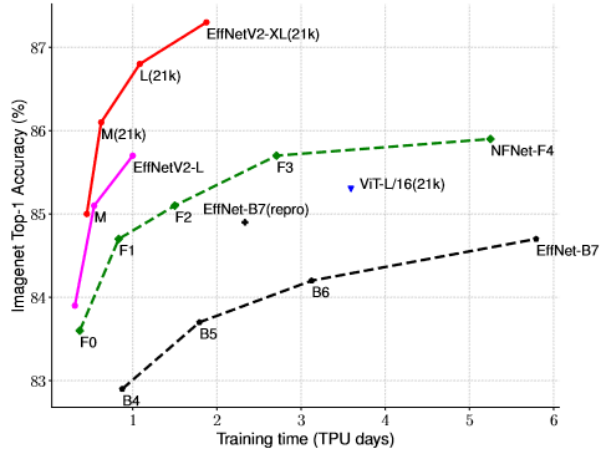


EfficientNetV2

- Further improve training time and parameter efficiency
- Increase image size during training
- Replace DepthWise-Convolution in the first stages to improve training time
- Non-uniform scaling strategy



MBConv(left) and Fused-MBConv(right)



Accuracy Evaluation

NAS	#Params (Millions)	Cifar10 (Top1 Acc)	ImageNet
ResNet-110	1.7	93.57	-
ResNET-152	230	-	70.62
DARTS	3.3 - 4.7	97.23	73.3
P-DARTS	3.4 - 10.5	97.75	75.6
Hierarchical-EAS	15.7	96.25	79.7
PNASNet-5 Mobile	3.2-5.1	96.66	74.2
PNASNet-5 Large	86.1	-	82.9
MnasNet	5.2	-	76.7
AdaNet	-	93.76	-
Efficientnet-B3	12	-	81.5
EfficientNetV2-L	121	99.1	86.8

Model Evaluation Methods

- Used to accelerating the process of model evaluation.
- Methods:
  - Low fidelity
  - Weight sharing
  - Early stopping

Low Fidelity

- **Reducing the training set size**, e.g. FABOLAS- models loss and training time as a function of dataset size to trades off high information with computational cost.

- **Training set with low resolution images**, e.g., training with a downsampled version of ImageNet with same number of classes and images as ImageNet, but with 32x32 pixel images.

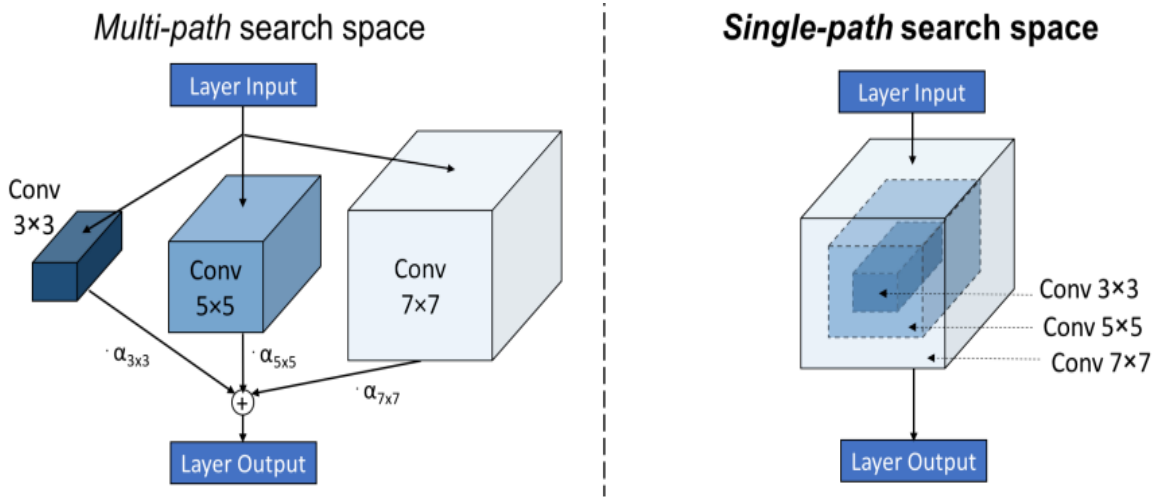
- **Transferable architecture,** e.g., search for the best convolutional architecture on a proxy dataset (e.g. CIFAR-10), and then transfer the learned architecture to ImageNet.

- **Regularize Evolutionary algorithm,** e.g., AmoebaNets, Instead of removing worst model in each evolution cycle, remove the oldest model.

- **Multi-Fidelity Optimization**: e.g. evaluating model configuration on small data subset, but fixing the resulting bias with Transfer Series Expansion (TSE), i.e., learning the low-fidelity correction predictor from linear combination of a set of base predictors.

Weight Sharing

- **Parameter Server Scheme:** shared parameter among minibatches of child architectures trained in parallel and then updated made in all child architecture together.
- **Transfer Learning:** transferring knowledge contained in one neural network to another neural network
- **Single-Path NAS**: Using one single path over parameterized ConvNet to encode all architectural decisions with shared convolutional kernel parameters.

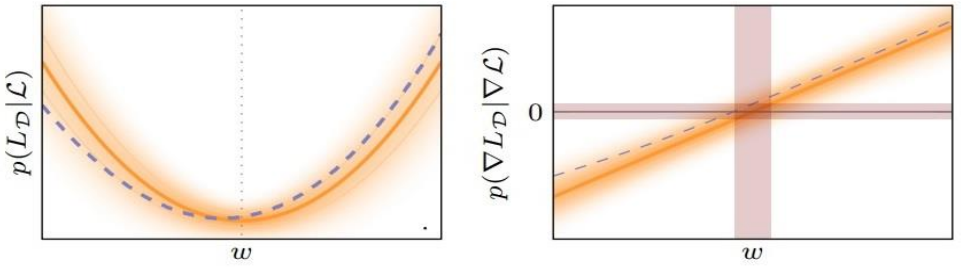


Early Stopping

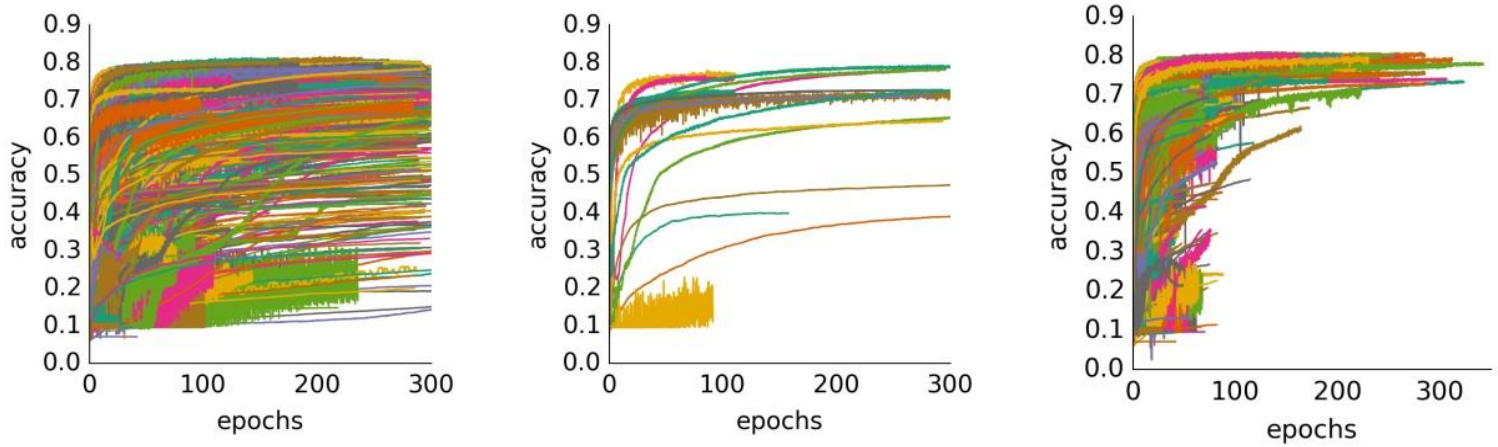
- **Based on Local Statistics** of the computed gradients.

$\mathcal{L}(w)$:= Expectation of Loss

$L_{\mathcal{D}}(w)$:= Empirical Risk



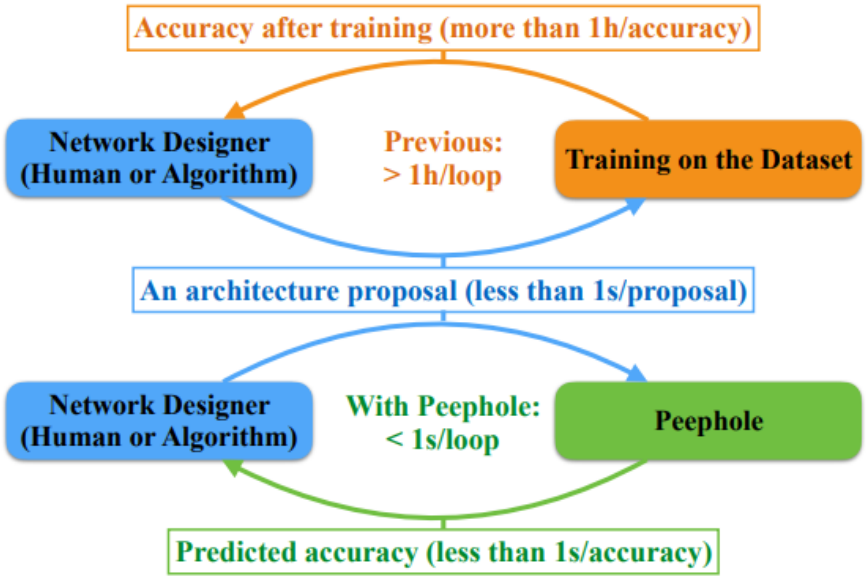
- **Based on Extrapolation of Learning Curve:** formed by combining prediction from mixture of multiple Bayesian probabilistic models.



(a) Without predictive termination (b) Random subset of left figure (c) With predictive termination

Based on a Peephole technique:

- A *Unified Layer Code* and *Layer Embedding* to encode individual layers into vector representation and then integrate different layers into embedding.
- Using *LSTM* to integrate information along sequence of layers.
- A *Block-based Generation scheme to* effectively constrain the sample space (training set)



References

- AdaNet: Adaptive Structural Learning of Artificial Neural Networks (Corinna Cortes et al.)
- AutoML: A Survey of the State-of-the-Art (Xin He et al.)
- DARTS: Differentiable Architecture Search (Hanxiao Liu et al.)
- MnasNet: Platform-Aware Neural Architecture Search for Mobile (Mingxing Tan et al.)
- Net2Net: Accelerating Learning via Knowledge Transfer (Tianqi Chen et al.)
- Progressive DARTS: Bridging the Optimization Gap for NAS in the Wild (Xin Chen et al.)
- Progressive Neural Architecture Search (Chenxi Liu et al.)
- Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets. (Aaron Klein. Et.al.)
- A Downsampled Variant of ImageNet as an Alternative to the CIFAR datasets. (Patryk Chrabaszcz et. Al.)
- Learning Transferable Architectures for Scalable Image Recognition (Barret Zoph et. al.)
- Regularized Evolution for Image Classifier Architecture Search (Esteban Real et. al.)

References

- Multi-Fidelity Automatic Hyper-Parameter Tuning via Transfer Series Expansion (Y. Hu et. al.)
- Neural Architecture Search with Reinforcement Learning (Barret Zoph et. al.)
- Transfer Learning with Neural AutoML (Catherine Wong et. al.)
- Single-Path NAS: Designing Hardware-Efficient ConvNets in less than 4 Hours (Dimitrios Stamoulis et. al.)
- FBNet: Hardware-Aware Efficient ConvNet Design via Differentiable Neural Architecture Search (Bichen Wu et. al.)
- EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks (Mingxing Tan, Quoc V. Le)
- EfficientNetV2: Smaller Models and Faster Training (Mingxing Tan, Quoc V. Le)
- https://paperswithcode.com/sota/image-classification-on-cifar-10
- Early Stopping without a Validation Set (Maren Mahsereci, et. al)
- Speeding up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves (Tobias Domhan, et. al.)
- Peephole: Predicting Network Performance Before Training (B. Dang et. al.)