Augmented Neural Architecture Search with REINFORCE

Olga Xu

Motivation and Related Work

- Hyperparameter search and model architecture design is an important research topic in machine learning field.
- Finding the best hyperparameter setting is crucial in solving machine learning tasks.
- We propose train a model that can automate the process of designing the model architecture
- *Neural Architecture Search with Reinforcement Learning* (NASRL) by Barret Zoph and Quoc V. Le

Hypothesis

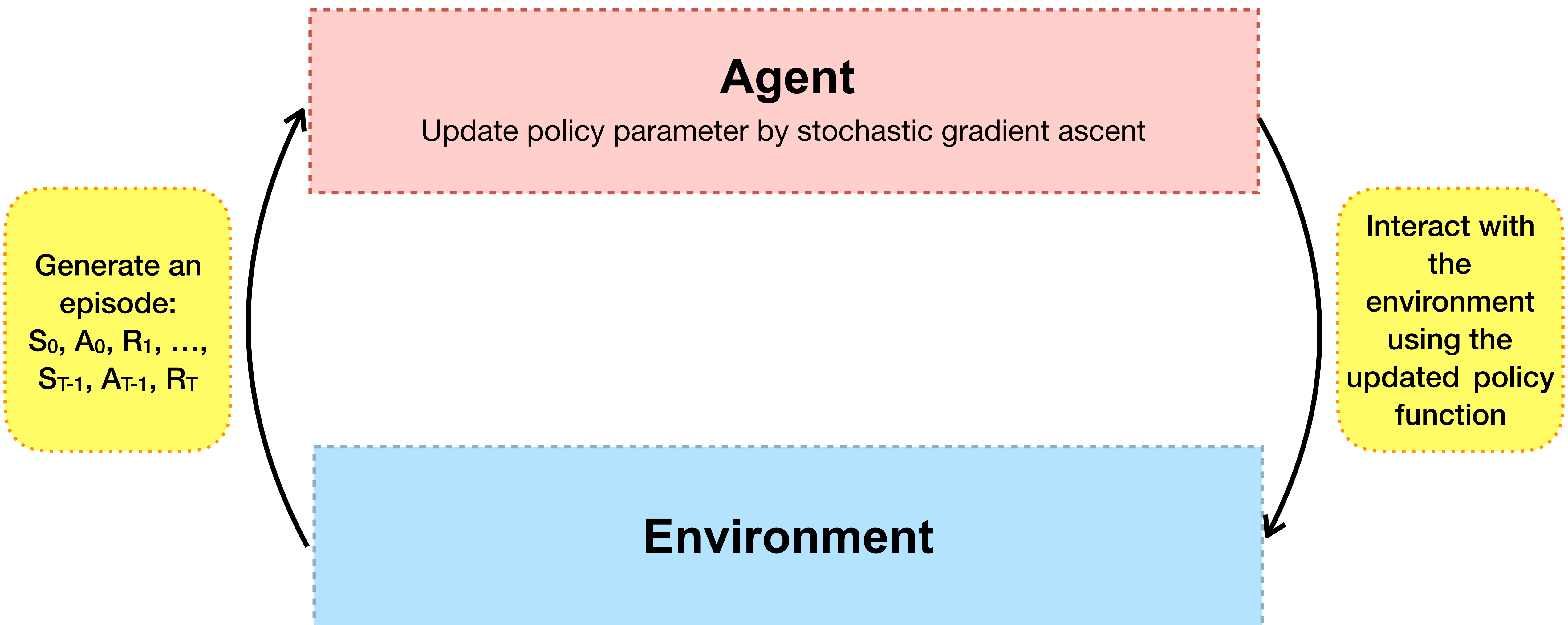
- implement a similar model as NASRL
- However, there are some small alterations between our method and NASRL.
- We define a model architecture by a list of actions the RNN controller samples.
- The crucial assumption we made is that every sub-model of the optimal generated model is a good model (i.e. if the optimal model, $a_{1:T}$, has T layers, $a_{1:t}$ is a good model for every t such that $t<=T$).
- At each episode we use the RNN controller to generate a new model. At each time step, t, of every episode, we samples a layer(action), a_t, with probability calculated by the RNN controller and train architecture a_t to obtain its validation accuracy R_t. The assumption that we made previously helps us to define our return function to be $G_t=R_t+gamma*G_{t-1}$
- Experience replay tree!

Recap: REINFORCE algorithm

REINFORCE, A Monte-Carlo Policy-Gradient Method (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Algorithm parameter: step size $\alpha > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
  Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
  Loop for each step of the episode $t = 0, \ldots, T-1$:
    $G \leftarrow$ return from step t (G_t)
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha\gamma^t G \nabla_{\boldsymbol{\theta}} \ln \pi(A_t|S_t, \boldsymbol{\theta})$

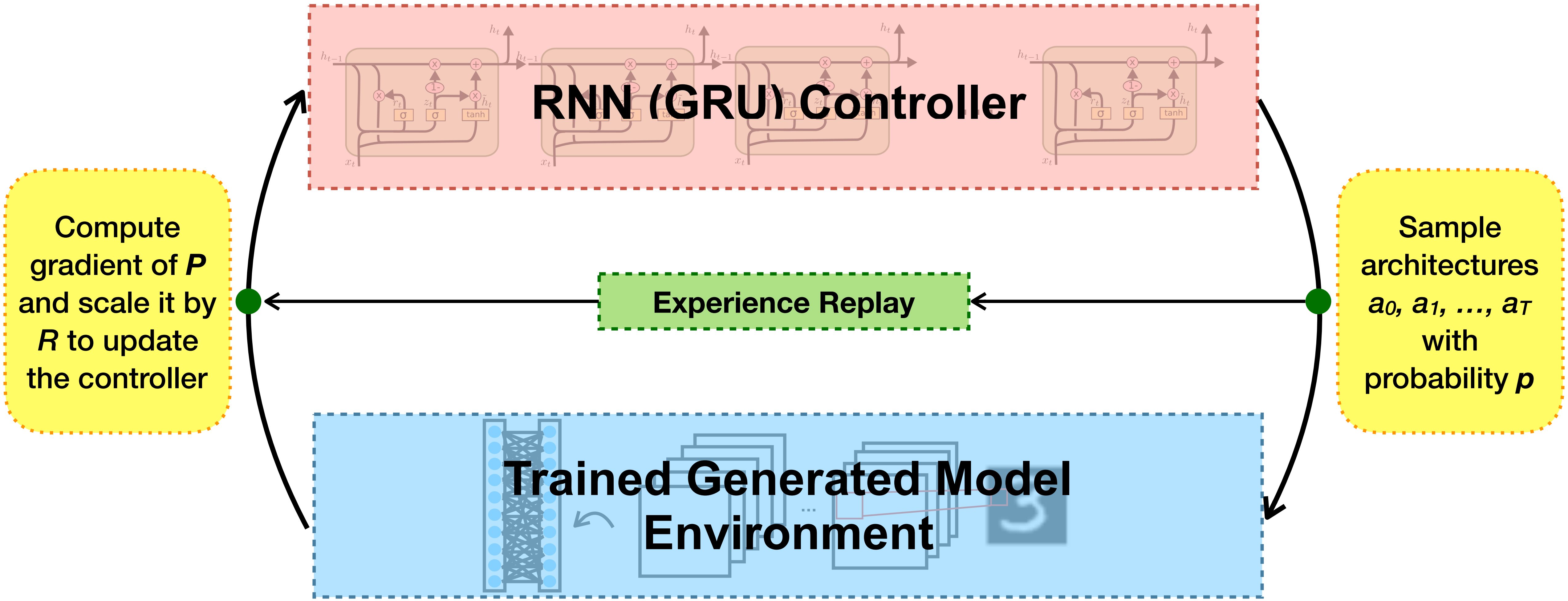


Neural Architecture Search using REINFORCE algorithm

Controller optimizes θ_c to maximize its expected reward, $J(\theta_c) = \mathbb{E}_{P(a_{1:T};\theta_c)}[R]$, where

$$\nabla J(\theta_c) = \frac{1}{m}\sum_{k}^{m}\sum_{t}^{T}\nabla\theta_c \log P(a_t|a_{(t-1):1};\theta_c)(R_k - b)$$

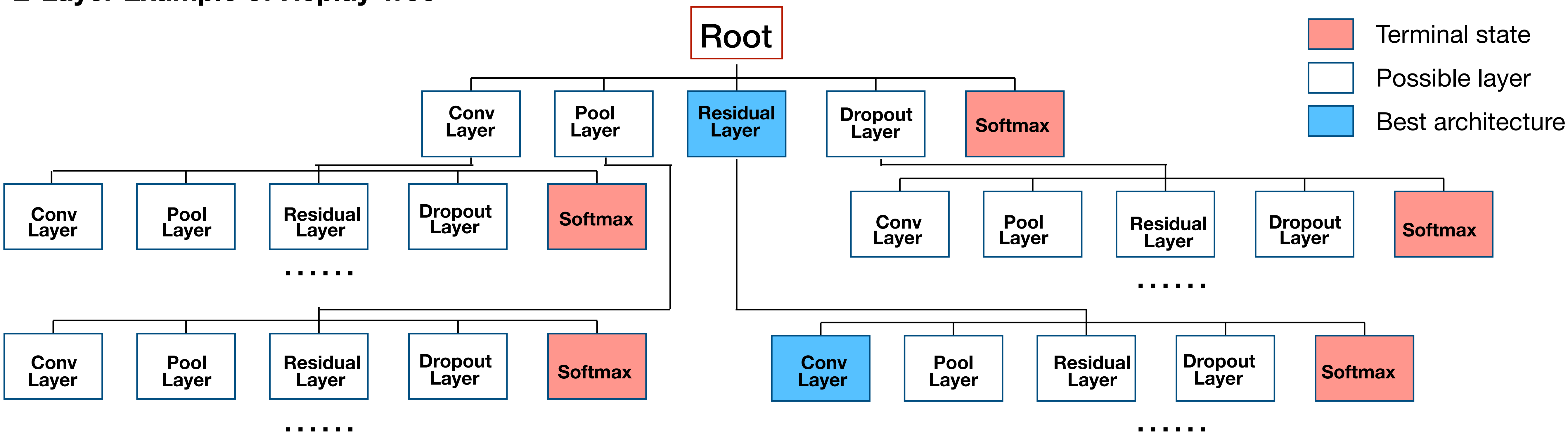
$a_{1:T}$ is a list of action to design the child architecture and *b* is model baseline (in this experiment, *b* is the iterative average of the previous architecture accuracies)



This environment will train the sampled architecture $a_{1:T}$ and will received an accuracy R on the validation set. This R will be the "reward" used to update the controller

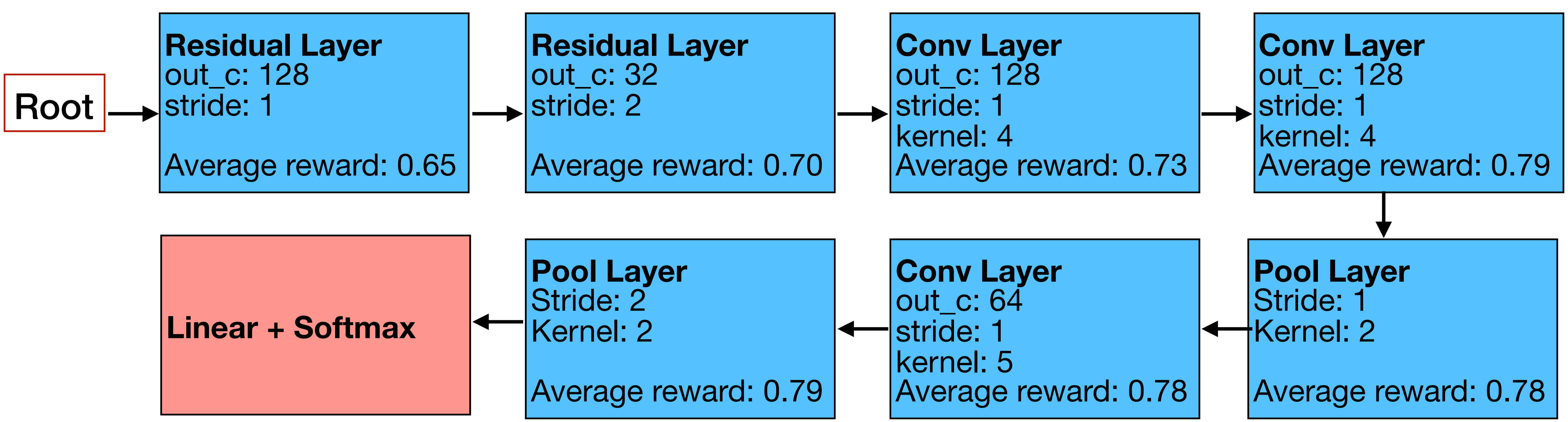
Experience Replay Tree and Model Results

2-Layer Example of Replay Tree



- Every node corresponds to a layer of the classification network
- Every path from root is a possible architecture.
- Every node contains (1) layer's structure, (2) corresponding architecture's average accuracy (R) and (3) count of number of time that architecture has used.

An Architecture Found at episode #40 that Generates 86% Accuracy on CIFAR-10 dataset:



Results and More

Best Architecture found

Layer	Type	Setting
1	Residual	Channel=128; Stride=1
2	Convolution	Channel=32; Kernel_size=3; Stride=1
3	Convolution	Channel=32; Kernel_size=3; Stride=1
4	Residual	Channel=64; Stride=1
5	Pool	Kernel_size=2; Stride=2
6	Residual	Channel=64; Stride=1
7	Residual	Channel=64; Stride=1
8	Convolution	Channel=128; Kernel_size=4; Stride=1
9	Convolution	Channel=64; Kernel_size=3; Stride=1
10	Linear+Softmax	-

Result Table

	MNIST	CIFAR-10
NAS [B. Zoph et., 2016]	-	94.5
CNNAS [M. Phulsuksombati, 2014]	-	77.7
CNN [J. Mairal et., 2014]	99.5	82.18
Our NAS Model	99.2	86.13

Discussion

- Search space:
  - Number of possible layers: 24
  - Max number of layer: 15
  - Possible architecture: $24^{15} \approx 5.04 \times 10^{20}$
- Gradually increased the architectures's max layer bound and number of train iterations
- Total number of architectures trained: 437 (NAS [B. Zoph et., 2016] trained 12,800 architectures on 800 GPUs)
- The model found several interesting features about the architectures:
  - Pool and Max Pool layers doesn't work well as first layer of the architecture
  - 5 layers architecture works the best for MNIST and 10 layers architecture works the best for CIFAR-10