Learnable Embedding Space for Efficient Neural Architecture Compression

Shengcao Cao*, Xiaofang Wang* and Kris M. Kitani

Peking University, Carnegie Mellon University

Compressed Architecture Search

- **Goal**: automatically search for a small and accurate network architecture based on a given large network architecture
- **Bottleneck**: the need to repeatedly evaluate different architectures
- **Proposal**: a **learnable embedding space** over the domain of network architectures that can be used to generate a **priority ordering** of architectures for evaluation

Formulation

Reward Function

$$x^* = \arg\max_{x \in \mathcal{X}} f(x)$$

Domain of Architectures

$$f(x) = C(x)\,(2 - C(x)) \cdot \frac{A(x)}{A(x_{\text{teacher}})} \qquad C(x) = 1 - \frac{\#\text{params}(x)}{\#\text{params}(x_{\text{teacher}})}$$

Validation Accuracy   Compression Ratio

Search with Bayesian Optimization (BO)

- Gaussian Process (GP) Prior: $\mu(\cdot) : \mathcal{X} \mapsto \mathcal{R}$ (Mean Function) $k(\cdot,\cdot) : \mathcal{X} \times \mathcal{X} \mapsto \mathcal{R}$ (Kernel Function)
- At architecture search step t:

  Posterior distribution $p\,(f(x) \mid f(x_{1:t})) \sim \mathcal{N}(\mu_t(x), \sigma_t^2(x))$
- Select the next architecture for evaluation

  Maximize the acquisition function $\mathrm{EI}_t(x) = \mathbb{E}_t[\max(0, f(x) - f_t^*)]$

How to define the kernel function?

Learnable Embedding Space

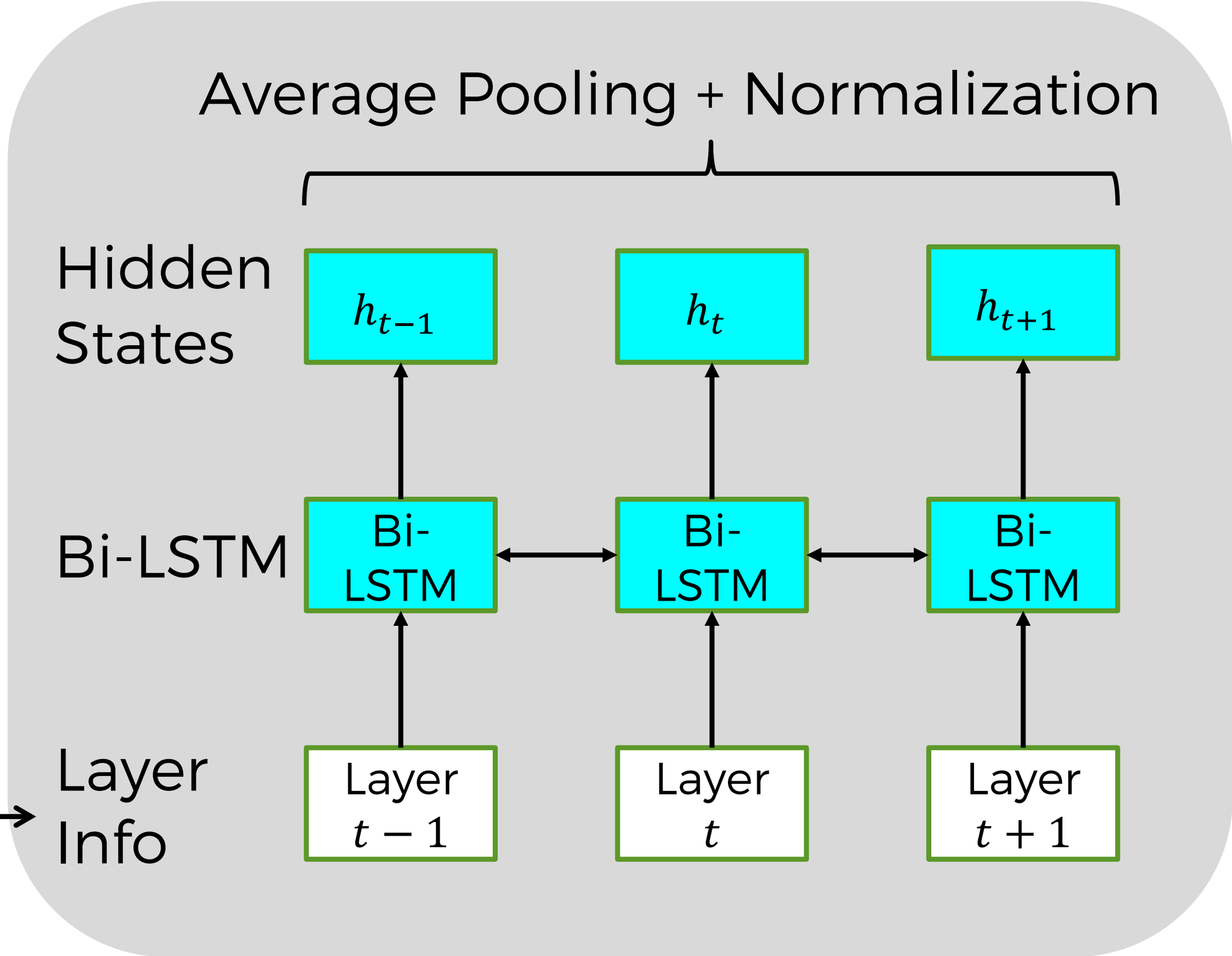
$$h(\cdot;\theta) : \mathcal{X} \mapsto \mathcal{R}^d$$

Architecture Embedding

$$k(x, x'; \theta) = \exp\left(-\frac{\|h(x;\theta) - h(x';\theta)\|^2}{2\sigma^2}\right)$$

RBF Kernel

1. Layer Type: a one-hot vector
2. Layer Attributes: filter size, stride ...
3. Layer Connectivity



Objective Function

$$L(\theta) = -\frac{1}{|D|} \sum_{i : x_i \in D} \log p\,(f(x_i) \mid f(D \setminus x_i); \theta)$$

Posterior Probability

Learn a θ such that the function f is consistent with the GP prior.

$$D = \{x_1, \ldots, x_t\}$$

At architecture step t, we have t evaluated architectures.

Algorithm Sketch

Algorithm 1 Neural Architecture Search with Bayesian Optimization

```
Input: Number of steps T. Number of kernels K. Teacher network x_teacher.
Randomly sample K architectures x_1^1, ..., x_1^K from the search space defined based on x_teacher.
Initialize the set of evaluated architectures D = ∅.
for t = 1, ..., T do
  Evaluate the K architectures x_t^1, ..., x_t^K.
  D = D ∪ {x_t^1, ..., x_t^K}.
  for k = 1, ..., K do
    Randomly initialize the weights of kernel k, denoted as θ^k.
    Randomly sample a subset of D, denoted as D^k.
    Learn θ^k on D^k by minimizing the negative log posterior probability.
    Compute the posterior distribution conditioned on the architectures in D_k with kernel k.
    Maximize the acquisition function and denote the solution as x_{t+1}^k.
  end for
end for
Return the best architecture in D as the solution.
```

Search Space

- Layer Removal
- Layer Shrinkage
- Add Skip Connections

Code

Scan me

Compression Results

CIFAR-100		Accuracy	#Params	Ratio	Times	$f(x)$
VGG-19	Teacher	73.71%	20.09M	-	-	-
	Random Search	68.17% ±1.28%	2.83M ±1.05M	0.8593 ±0.0525	8.04× ±3.78×	0.9046 ±0.0074
	Ours	71.41% ±0.75%	2.61M ±0.61M	0.8699 ±0.0306	7.99× ±1.99×	**0.9518 ±0.0158**
ResNet-18	Teacher	78.68%	11.22M	-	-	-
	Random Search	69.86% ±1.90%	1.26M ±0.54M	0.8878 ±0.0477	10.10× ±4.33×	0.8752 ±0.0137
	N2N	68.01%	2.42M	0.7845	4.64×	0.8242
	Ours	73.83% ±1.11%	1.87M ±0.08M	0.8335 ±0.0073	6.01× ±0.26×	**0.9123 ±0.0151**
ResNet-34	Teacher	78.71%	21.33M	-	-	-
	Random Search	72.33% ±1.53%	3.61M ±0.35M	0.8308 ±0.0162	5.95× ±0.60×	0.8924 ±0.0154
	N2N - removal	70.11%	4.25M	0.8008	5.02×	0.8554
	Ours - removal	74.05% ±0.52%	3.18M ±0.65M	0.8508 ±0.0307	6.88× ±1.31×	0.9192 ±0.0033
	Ours	73.68% ±0.57%	2.36M ±0.15M	0.8895 ±0.0069	9.08× ±0.59×	**0.9246 ±0.0076**
ShuffleNet	Teacher	71.14%	1.06M	-	-	-
	Random Search	64.75% ±2.15%	0.18M ±0.06M	0.8264 ±0.0588	6.37× ±2.68×	0.8803 ±0.0152
	Ours	68.45% ±1.38%	0.23M ±0.04M	0.7855 ±0.0337	4.74× ±0.78×	**0.9171 ±0.0088**

Compression Results on CIFAR-100

	Teacher	Accuracy	#Params	Teacher	Accuracy	#Params
Ours	VGG-19	71.41%	2.61M	ResNet-18	73.83%	1.87M
	ShuffleNet	68.45%	0.23M	ResNet-34	73.68%	2.36M
	Configuration	Accuracy	#Params	Configuration	Accuracy	#Params
ShuffleNet	$0.5 \times (g=1)$	67.71%	0.26M	$1.5 \times (g=1)$	72.43%	2.09M
	$0.5 \times (g=2)$	67.54%	0.27M	$1.5 \times (g=2)$	71.41%	2.07M
	$0.5 \times (g=3)$	67.23%	0.27M	$1.5 \times (g=3)$	71.05%	2.03M
	$0.5 \times (g=4)$	66.83%	0.27M	$1.5 \times (g=4)$	71.86%	1.99M
	$0.5 \times (g=8)$	66.74%	0.31M	$1.5 \times (g=8)$	71.04%	2.08M

Comparison to ShuffleNet [Zhang et al.]

CIFAR-100		Accuracy	#Params	Ratio	Times	$f(x)$
VGG-19	Euclidean	70.95% ±1.07%	2.47M ±1.26M	0.8771 ±0.0627	9.62× ±4.55×	0.9453 ±0.0092
	Marginal	69.90% ±0.69%	1.50M ±0.68M	0.9254 ±0.3382	16.14× ±9.22×	0.9422 ±0.0071
	Posterior	71.41% ±0.75%	2.61M ±0.61M	0.8699 ±0.0306	7.99× ±1.99×	**0.9518 ±0.0158**
ResNet-18	Euclidean	71.67% ±0.67%	1.62M ±0.27M	0.856 ±0.0243	7.07× ±1.09×	0.8917 ±0.0137
	Marginal	72.80% ±1.11%	1.72M ±0.18M	0.8467 ±0.0160	6.57× ±0.67×	0.9033 ±0.0094
	Posterior	73.83% ±1.11%	1.87M ±0.08M	0.8335 ±0.0073	6.01× ±0.26×	**0.9123 ±0.0151**
ResNet-34	Euclidean	72.87% ±1.11%	2.49M ±0.60M	0.8834 ±0.2814	8.90× ±2.04×	0.9127 ±0.0103
	Marginal	73.11% ±0.57%	3.34M ±0.48M	0.8435 ±0.0224	6.47× ±0.89×	0.9059 ±0.0134
	Posterior	73.68% ±0.57%	2.36M ±0.15M	0.8895 ±0.0069	9.08× ±0.59×	**0.9246 ±0.0076**

Choice of the Objective Function

Architecture Visualization