



Meta Architecture Learning

Albert Shaw¹, Wei Wei², Weiyang Liu¹, Le Song^{1,3}, Bo Dai^{1,2}



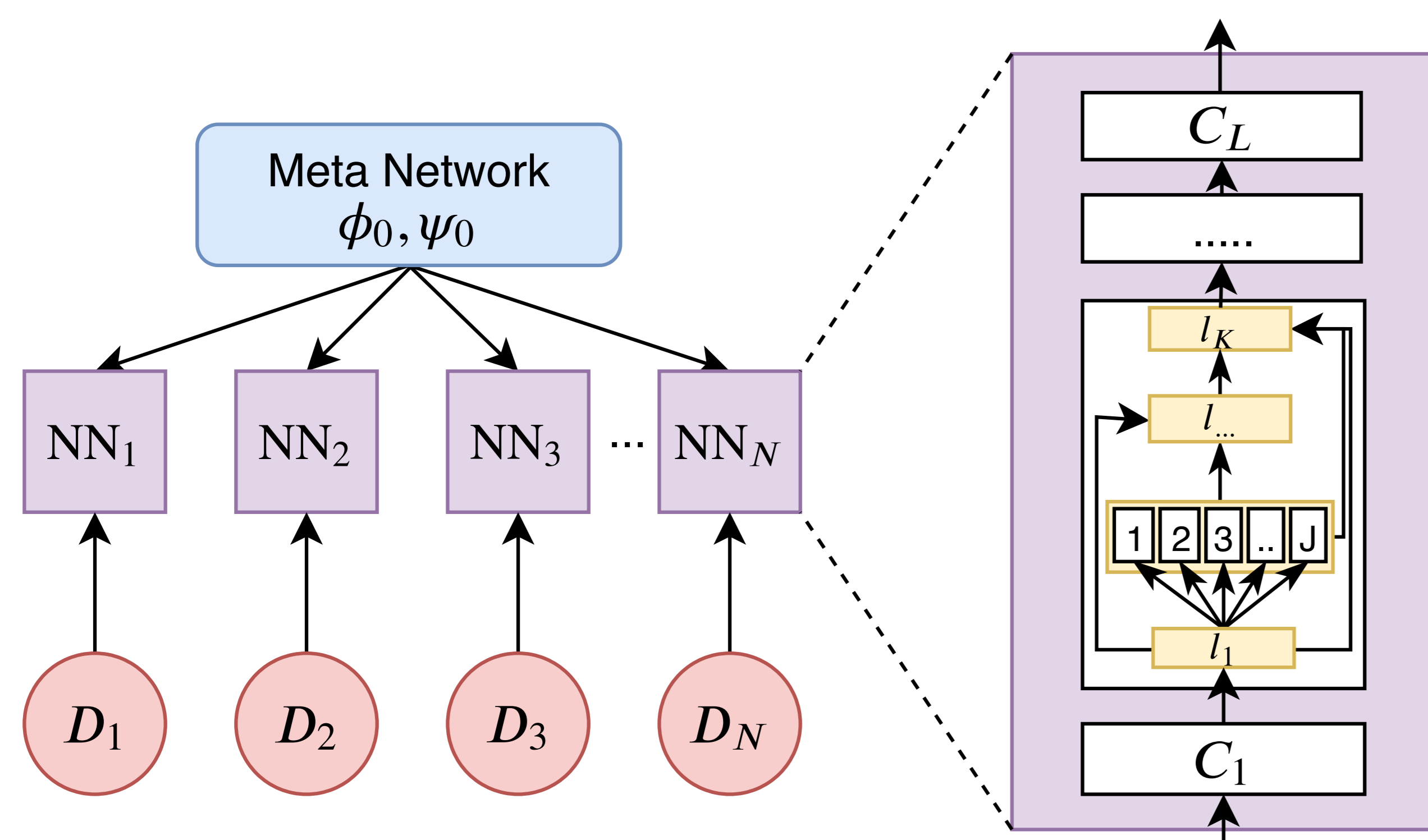
Motivation

While developing techniques to automatically search the space of Neural Network architectures has become a large focus of recent efforts, existing methods can only learn the architecture for a single task at a time. We approach the problem of neural architecture search using Bayesian and Meta-learning techniques to simultaneously learn the architecture and weight distribution for multiple tasks at once.

Main Contribution:

- ▶ We propose a **Bayesian inference** view of architecture learning.
- ▶ We derive a variational inference method to learn the architectures of an entire set of tasks simultaneously using the **optimization embedding** technique to design the parameterization of the posterior.
- ▶ Demonstrates a **concrete algorithm** for Meta Architecture Search that can use a prior trained over multiple tasks to find competitive models for new unseen datasets with just quick adaptation.

Bayesian View of Architecture Search



We consider NAS as an operation selection problem.

$$x_k = \sum_{i=1}^{k-1} (z_{i,k}^T A_i(\theta)) \circ x_i := \sum_{i=1}^{k-1} \sum_{l=1}^L z_{i,k}^l \phi_l^i(x_i; \theta),$$

Assume the probabilistic model as

$$\begin{aligned} \theta &\sim \mathcal{N}(\mu, \sigma^2), \\ z_{i,k} &\sim \text{Categorical}(\alpha_{i,k}), \quad k = 1, \dots, K, \\ y &\sim p(y|x; \theta, z) \propto \exp(-\ell(f(x; \theta, z), y)), \end{aligned}$$

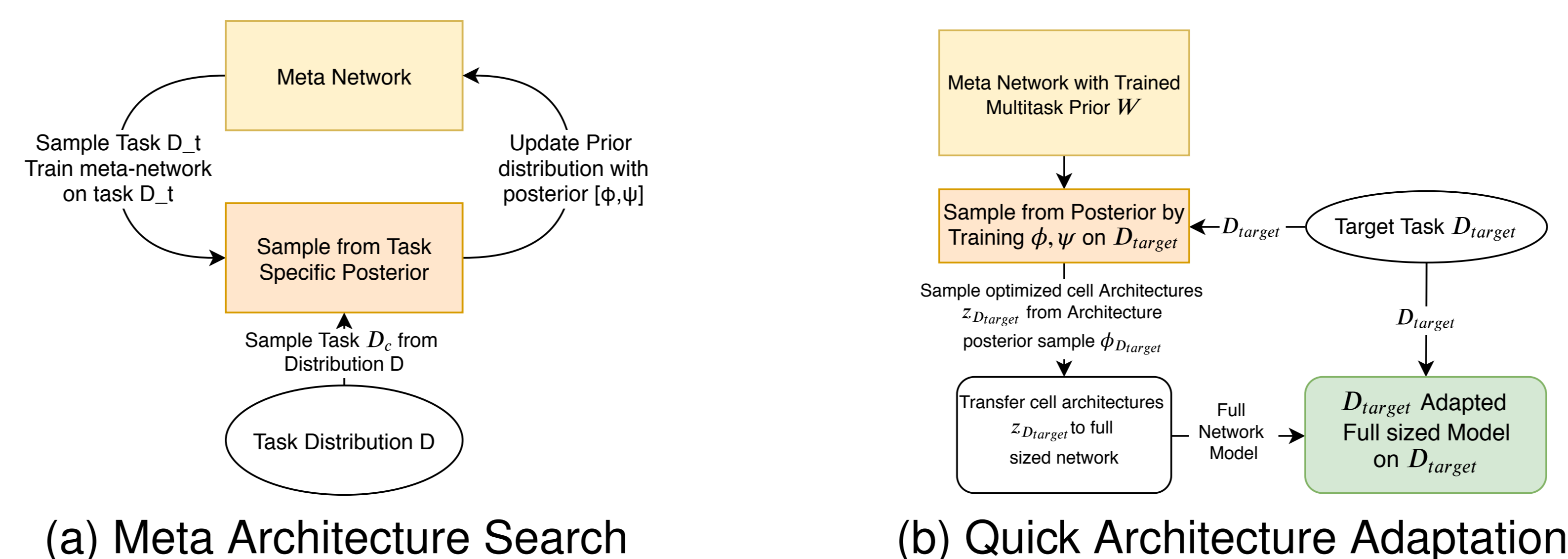
We can estimate the parameters $W(\mu, \sigma, \alpha)$ via MLE.

$$\max_W \hat{\mathbb{E}}_{x,y} \left[\log \int p(y|x; \theta, z) p(z; \alpha) p(\theta; \mu, \sigma) dz d\theta \right].$$

Extended to the meta-learning setting with many tasks \mathcal{D}_i and weight and architecture priors (μ, σ, α) are shared between all the tasks,

$$\max_W \hat{\mathbb{E}}_{\mathcal{D}_i} \hat{\mathbb{E}}_{(x,y) \sim \mathcal{D}_i} \left[\log \int p(y|x; \theta, z) p(z; \alpha) p(\theta; \mu, \sigma) dz d\theta \right]$$

Meta Architecture Search and Adaptation



Variational Inference by Optimization Embedding

We consider the ELBO, since the MLE is intractable due to the integral.

$$\hat{\mathbb{E}}_{\mathcal{D}} \left[\max_{\phi_D, \psi_D} \underbrace{\hat{\mathbb{E}}_{x,y} \mathbb{E}_{\xi, \epsilon} [-\ell(f(x; \theta_D(\epsilon, \psi), z_D(\xi, \phi)), y)] - \log \frac{q_{\phi_D}(z|\mathcal{D})}{p(z; \alpha)} - \log \frac{q_{\psi_D}(\theta|\mathcal{D})}{p(\theta; \mu, \sigma)}}_{L(\phi_D, \psi_D; W)} \right].$$

We approximate $q(z|\mathcal{D})$ with the Gumbel-Softmax distribution.

We follow the parametrized CVB derivation for embedding the optimization procedure for (ϕ, ψ) deriving the explicit form of the parameters ϕ_D and ψ_D . Denoting the $\hat{g}_{\phi_D, \psi_D}(\mathcal{D}, W) = \frac{\partial \hat{L}}{\partial (\phi_D, \psi_D)}$ where \hat{L} is the stochastic approximation for $L(\phi_D, \psi_D; W)$, we have

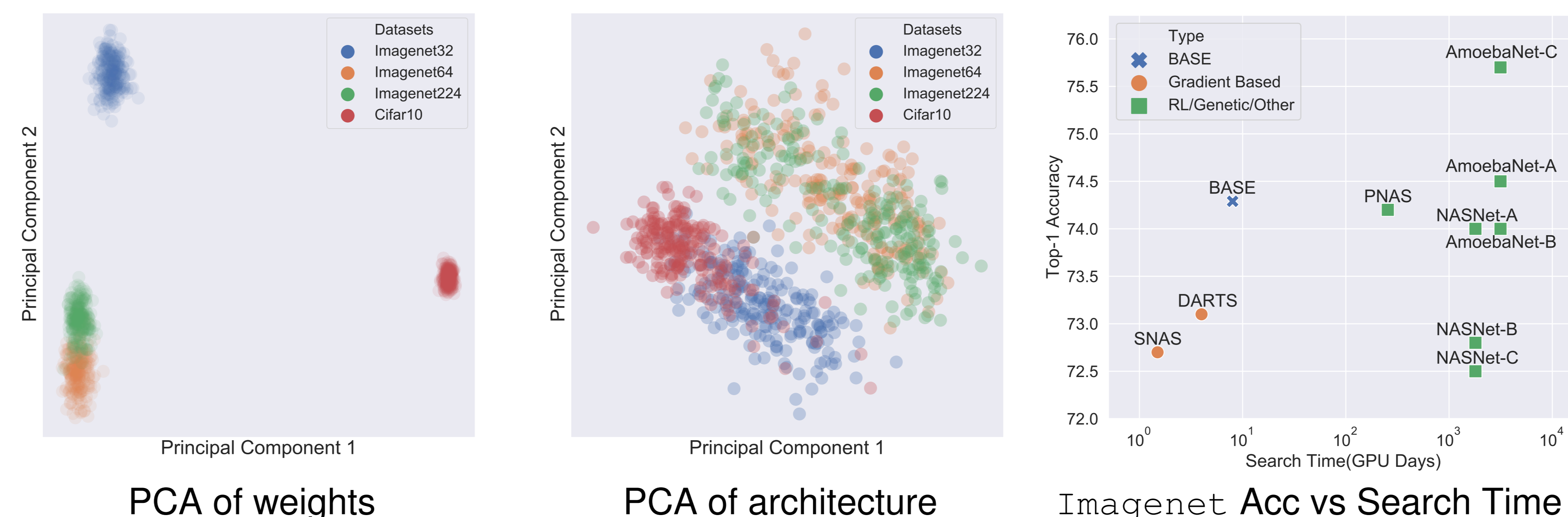
$$[\phi_D^t, \psi_D^t] = \eta_t \hat{g}_{\phi_D, \psi_D}(\mathcal{D}, W) + [\phi_D^{t-1}, \psi_D^{t-1}],$$

We can initialize $(\phi^0, \psi^0) = W$ which is shared across all the tasks. After T optimization steps, we obtain (ϕ_D^T, ψ_D^T) , which leads to $(\theta_D^T(\xi, \psi_D^T), z_D^T(\xi, \phi_D^T))$. In other words, we derive the concrete parameterization of $q(\theta|\mathcal{D})$ and $q(z|\mathcal{D})$ automatically by unfolding the optimization steps.

$$\max_{W, V} \hat{\mathbb{E}}_{\mathcal{D}} \hat{\mathbb{E}}_{x,y} \mathbb{E}_{\xi, \epsilon} \left[-\ell(f(x; \theta_D^T(\epsilon, \psi), z_D^T(\xi, \phi)), y) - \log \frac{q_{\phi_D^T}(z|\mathcal{D})}{p(z; \alpha)} - \log \frac{q_{\psi_D^T}(\theta|\mathcal{D})}{p(\theta; \mu, \sigma)} \right].$$

which can be optimized by stochastic gradient ascent for learning W .

Meta Architecture Search and Adaptation



Bayesian meta-Architecture SEarch (BASE) Algorithm

- 1: Initialize meta-network parameters W_0 .
- 2: **for** $e = 1, \dots, E$ **do**
- 3: Sample C tasks $\{\mathcal{D}_c\}_{c=1}^C \sim \mathcal{D}$.
- 4: **for** \mathcal{D}_c in \mathcal{D} **do**
- 5: Sample $\{x_t, y_t\}_{t=1}^T \sim \mathcal{D}_c$.
- 6: Let $\phi_c^0, \psi_c^0 = W_{e-1}$.
- 7: **for** $t = 1, \dots, T$ **do**
- 8: Sample $\xi \sim \mathcal{G}(0, 1)$.
- 9: Update $[\phi_c^t, \psi_c^t] = [\phi_c^{t-1}, \psi_c^{t-1}] - \eta \nabla_{\phi_c^{t-1}, \psi_c^{t-1}} \hat{L}(f(x_t; \phi_c^{t-1}, \psi_c^{t-1}, \xi), y_t)$.
- 10: Update $W_e = W_{e-1} + \lambda \frac{1}{C} \sum_{c=1}^C ([\phi_c^T, \psi_c^T] - W_{e-1})$.

Classification Accuracy on CIFAR10

Architecture	Top-1 Test Error	Params (M)	Search Time (Gpu Days)
DenseNet-BC	3.46	25.6	-
NASNet-A + cutout	2.65	3.3	1800
AmoebaNet-A + cutout	3.34 ± 0.06	3.2	3150
AmoebaNet-B + cutout	2.55 ± 0.05	2.8	3150
Hierarchical Evo	3.75 ± 0.12	15.7	300
DARTS (1st order)	3.00 ± 0.14	3.3	1.5
DARTS (2nd order)	2.76 ± 0.09	3.3	4
SNAS (single-level)	2.85 ± 0.02	2.8	1.5
ENAS + cutout	2.89	4.6	0.5
PNAS	3.41 ± 0.09	3.2	225
SMASH	4.03	16	1.5
BASE(Multi-task Prior)	3.18	3.22	8
BASE(Imagenet32)	3.00	3.29	0.04 Adap / 8 Meta
BASE(CIFAR10)	2.83	3.07	0.05 Adap / 8 Meta

Classification Accuracy on Imagenet

Architecture	Top-1 Err	Top-5 Err	MACs (M)	Search Time (GPU Days)
NASNet-A	26.0	8.4	564	1800
NASNet-B	27.2	8.7	488	1800
NASNet-C	27.5	9.0	558	1800
AmoebaNet-A	25.5	8.0	555	3150
AmoebaNet-B	26.0	8.5	555	3150
AmoebaNet-C	24.3	7.6	570	3150
PNAS	25.8	8.1	588	225
DARTS	26.9	9.0	595	4
SNAS	27.3	9.2	522	1.5
BASE (Multi-task Prior)	26.12	8.52	544	0 Adap / 8 Meta
BASE (Imagenet)	25.71	8.08	559	0.04 Adap / 8 Meta