# Meta Architecture Learning

Albert Shaw[1], Wei Wei[2], Weiyang Liu[1], Le Song[1], Bo Dai[1,2]

Georgia Institute of Technology

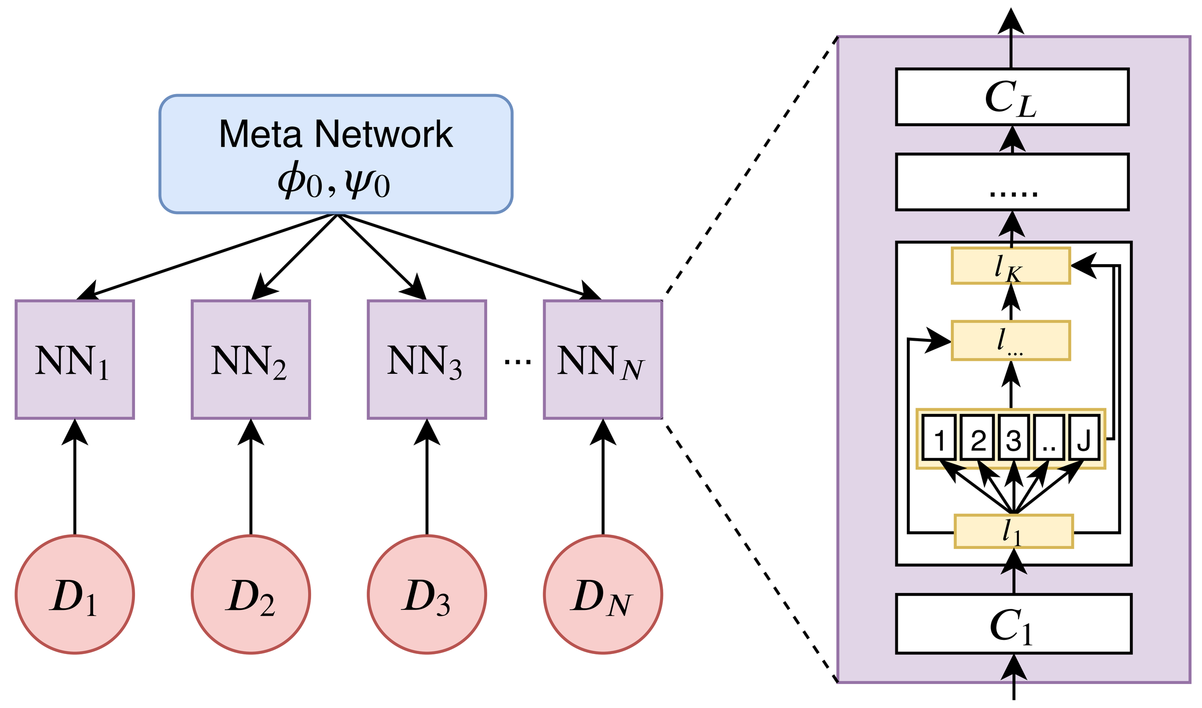
## Motivation

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 use this novel view to derive a variational inference method to learn the architecture of a meta-network, which will be shared across multiple tasks 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 neural network architecture learning as an operation selection problem. Starting with a DenseNet architecture as shown above where the $k$-th layer of the neural network is defined as

$$x_k = \sum_{i=1}^{k-1} \left(z_{i,k}^\top \mathcal{A}_i(\theta)\right) \circ x_i := \sum_{i=1}^{k-1}\sum_{l=1}^{L} z_{i,k}^l \phi_i^l(x_i;\theta),$$

where $\mathcal{A}_i(\theta) = \left[\phi_i^l(\cdot;\theta)\right]_{l=1}^L$ denotes a group of operations from $\mathbb{R}^d \to \mathbb{R}^p$ depending on the parameters $\theta$ and $z_{i,k} \sim \mathcal{C}ategorial(\alpha_{i,k})$ with $\alpha_{i,k}^l \geq 0$, $\sum_{l=1}^L \alpha_{i,k}^l = 1$. Assume the probabilistic model as

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

We can estimate the parameters $W(\mu,\sigma,\alpha)$ via maximum log-likelihood estimation

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

However, the MLE is intractable due to the integral. We consider the ELBO, *i.e.*,

$$\max_W \max_{q(z),q(\theta)} \ -\widehat{\mathbb{E}}_{x,y}\mathbb{E}_{z\sim q(z),\theta\sim q(\theta)}[\ell(f(x;\theta,z),y)] - KL(q||p),$$

whose $p(z) = \prod_{k=1}^K \prod_{i=1}^{k-1} \mathcal{C}ategorial(z_{i,k}) = \prod_{k=1}^K \prod_{i=1}^{k-1} \prod_{l=1}^L \left(\alpha_{i,k}^l\right)^{z_{i,k}^l}$.

## Bayesian Meta-network Architecture Learning

The Bayesian view can be extend to the few-shot meta-learning setting, where we have many tasks, *i.e.*, $\mathcal{D}_t = \left\{x_i^t, y_i^t\right\}_{i=1}^n$.
This can be formulated so that the layers and architecture priors *i.e.*, the hyperparameters $(\mu, \sigma, \alpha)$ are shared between all the tasks.

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

Using the convexity of log-sum-exp we achieve the ELBO

$$\max_W \widehat{\mathbb{E}}_{\mathcal{D}_t}\left[\max_{q(z|\mathcal{D}),q(\theta|\mathcal{D})} \ \widehat{\mathbb{E}}_{(x,y)\sim\mathcal{D}_t}\mathbb{E}_{z\sim q(z|\mathcal{D}),\theta\sim q(\theta|\mathcal{D})}\left[-\ell\left(f(x;\theta,z),y\right)\right] - KL(q||p)\right]$$

With the variational posterior distributions, $q(z|\mathcal{D})$ and $q(\theta|\mathcal{D})$, introduced into the model, we can directly generate the architecture and its corresponding weights based on the posterior. In a sense, the posterior can be understood as the neural network predictive model.

## Variational Inference by Optimization Embedding

The design of the parameterization of the posterior $q(z|\mathcal{D})$ and $q(\theta|\mathcal{D})$ is extremely important, especially in our case where we need to model the dependence w.r.t. the task distributions $\mathcal{D}$.
We assume the $q(\theta|\mathcal{D})$ is Gaussian and the $q(z|\mathcal{D})$ is product of the Categorical distribution which we approximate with Gumbel-Softmax.
Therefore, we have

$$q_\psi(\theta|\mathcal{D}) = \mathcal{N}(\psi_\mu, \psi_\sigma), \quad q_\phi(z_{i,k}|\mathcal{D}) = \Gamma(r)\,\tau^{L-1}\left(\sum_{l=1}^L \frac{\pi_{\mathcal{D},\phi_{i,k}^l,l}}{\left(z_{i,k}^l\right)^\tau}\right)^{-r} \prod_{i=1}^r \left(\frac{\pi_{\mathcal{D},\phi_{i,k}^l,l}}{\left(z_{i,k}^l\right)^{\tau+1}}\right),$$

Then, we can sample $(\theta, z)$ by following,

$$\begin{aligned} \theta_\mathcal{D}(\epsilon,\psi) &= \psi_{\mathcal{D},\mu} + \epsilon\psi_{\mathcal{D},\sigma}, \quad \epsilon\sim\mathcal{N}(0,1), \\ z_{i,k,\mathcal{D}}^l(\xi,\phi) &= \frac{\exp\left(\left(\phi_{\mathcal{D},i,k}^l + \xi^l\right)/\tau\right)}{\sum_{l=1}^L \exp\left(\left(\phi_{i,k}^l+\xi^l\right)/\tau\right)}, \quad \xi^l \sim \mathcal{G}(0,1), \quad l\in\{1,\ldots,L\}, \end{aligned}$$

with $\pi_{x,\phi,i} = \frac{\exp(\phi_{x,i})}{\sum_{i=1}^p \exp(\phi_{x,i})}$ and $\mathcal{G}(0,1)$ denoting the Gumbel distribution.
Plugging the formulation into the ELBO, we arrive at the objective

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

We follow the parametrized CVB derivation for embedding the optimization procedure for $(\phi,\psi)$ deriving the explicit form of the parameters $\phi_\mathcal{D}$ and $\psi_\mathcal{D}$.
Denoting the $\widehat{g}_{\phi_\mathcal{D},\psi_\mathcal{D}}(\mathcal{D},W) = \frac{\partial\widehat{L}}{\partial(\phi_\mathcal{D},\psi_\mathcal{D})}$ where $\widehat{L}$ is the stochastic approximation for $L(\phi_\mathcal{D},\psi_\mathcal{D};W)$, we have

$$\left[\phi_\mathcal{D}^t, \psi_\mathcal{D}^t\right] = \eta_t \widehat{g}_{\phi_\mathcal{D},\psi_\mathcal{D}}(\mathcal{D},W) + \left[\phi_\mathcal{D}^{t-1}, \psi_\mathcal{D}^{t-1}\right],$$

We can initialize $(\phi^0,\psi^0) = W$ which is shared across all the tasks. After $T$ optimization steps, we obtain $(\phi_\mathcal{D}^T, \psi_\mathcal{D}^T)$, which leads to $\left(\theta_\mathcal{D}^T(\xi,\psi_\mathcal{D}^T), z_\mathcal{D}(\xi,\phi_\mathcal{D}^T)\right)$.
In other words, we derive the concrete parameterization of $q(\theta|\mathcal{D})$ and $q(z|\mathcal{D})$ automatically by unfolding the optimization steps. Plugging the ultimate parameterization into $L(\phi_\mathcal{D},\psi_D,W)$, we have

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

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

## Bayesian meta-Architecture SEarch (BASE) Algorithm

```
Initialize meta-network parameters W_0.
for e = 1, ..., E do
    Sample C tasks {D_c}_{c=1}^C ~ D.
    for D_c in D do
        Sample {x_t, y_t}_{t=1}^T ~ D_c.
        Let φ_c^0, ψ_c^0 = W_{e-1}.
        for t = 1, ..., T do
            Sample ξ ~ G(0, 1).
            Update [φ_c^t, ψ_c^t] = [φ_c^{t-1}, ψ_c^{t-1}] − η∇_{φ_c^{t-1}, ψ_c^{t-1}} L̂(f(x_t; φ_c^{t-1}, ψ_c^{t-1}, ξ), y_t).
    Update W_e = W_{e-1} + λ(1/C) Σ_{c=1}^C ([φ_c^T, ψ_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 | $3.34 \pm 0.06$ | 3.2 | 3150 |
| AmoebaNet-B + cutout | $2.55 \pm 0.05$ | **2.8** | 3150 |
| Hierarchical Evo | $3.75 \pm 0.12$ | 15.7 | **300** |
| DARTS (1st order) | $3.00 \pm 0.14$ | 3.3 | 1.5 |
| DARTS (2nd order) | **2.76** $\pm$ **0.09** | 3.3 | 4 |
| SNAS (single-level) | $2.85 \pm 0.02$ | **2.8** | 1.5 |
| ENAS + cutout | 2.89 | 4.6 | **0.5** |
| PNAS | $3.41 \pm 0.09$ | 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 |

[1]Georgia Institute of Technology, [2]Google Brain

Mailto: ashaw596@gatech.edu