



# Differentiable Network Architecture Search

2019-07-8, CERN

Andrey Ustyuzhanin

NRU HSE

YSDA

ICL

# Network Architecture Search

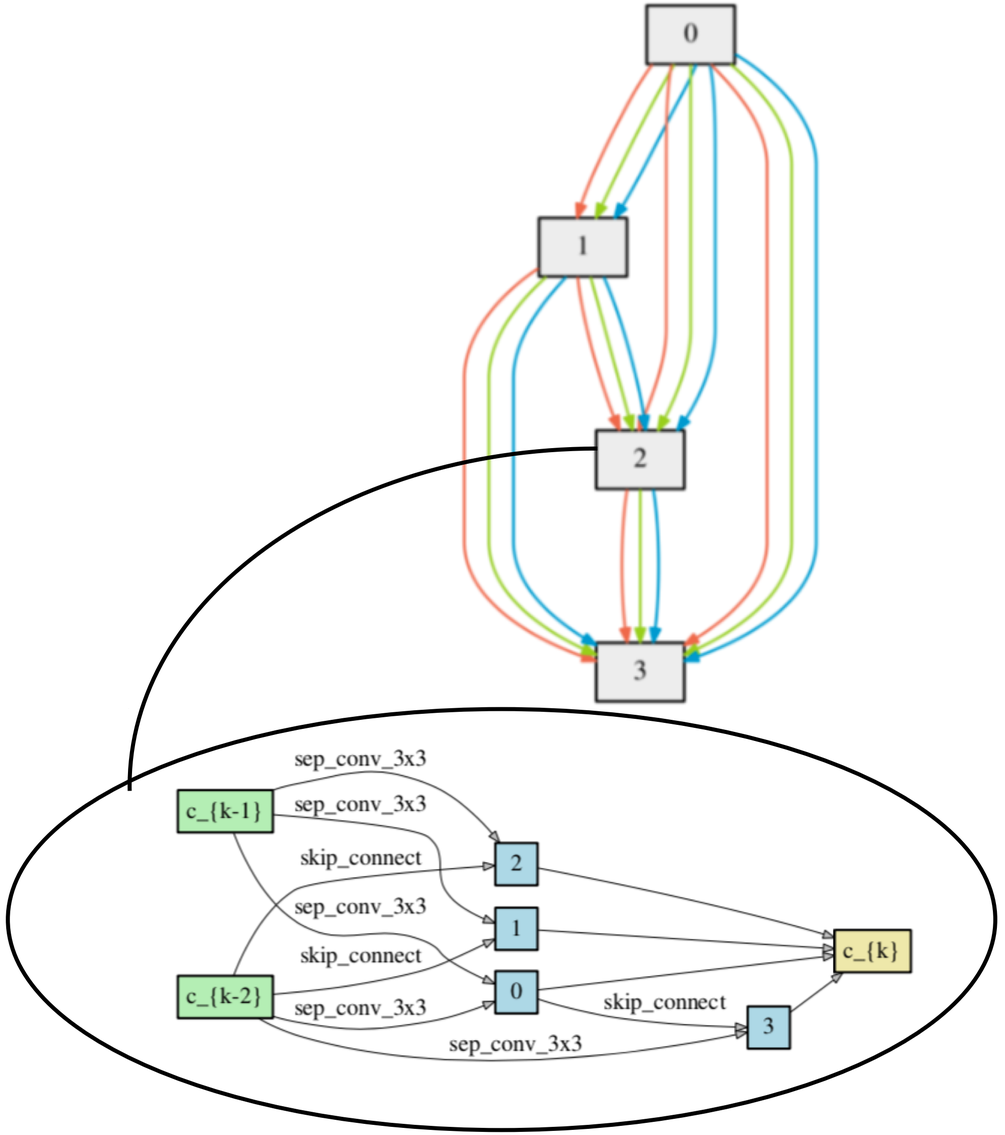


# Basic idea

Search for network topology as a graph of nodes connected by operations.

For simplicity the whole network is split into sub-graphs (cells) connected to each other.

Connection between nodes and cells are parametrized (relaxed) by softmax operation




Liu, Hanxiao and Simonyan, Karen and Yang, Yiming, DARTS: Differentiable Architecture Search, 2018

# Structure of a cell[i]

**0-th** node is the output of cell[i-2], **1-st** node is the output of cell[i-1]

All other nodes are connected to all of its predecessors through operations like convolution, pooling, identity, zero:

$$x^{(i)} = \sum_{j<i} o^{(i,j)}(x^{(j)})$$

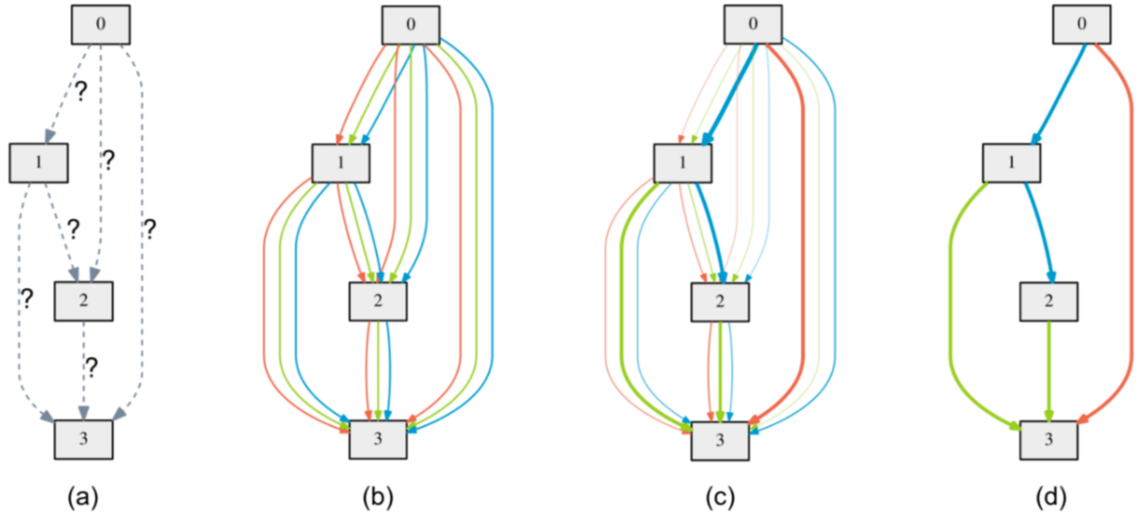
Softmax of all possible operations:

$$\bar{o}^{(i,j)}(x) = \sum_{o \in \mathcal{O}} \frac{\exp(\alpha_o^{(i,j)})}{\sum_{o' \in \mathcal{O}} \exp(\alpha_{o'}^{(i,j)})} o(x)$$

Nodes are parametrized by $\boldsymbol{w}$, connections are parametrized by $\boldsymbol{\alpha}$

# Cell optimisation procedure

$$\min_{\alpha} \quad \mathcal{L}_{val}(w^*(\alpha), \alpha)$$
$$\text{s.t.} \quad w^*(\alpha) = \text{argmin}_w \; \mathcal{L}_{train}(w, \alpha)$$




**Algorithm 1:** DARTS – Differentiable Architecture Search

Create a mixed operation $\bar{o}^{(i,j)}$ parametrized by $\alpha^{(i,j)}$ for each edge $(i, j)$

**while** *not converged* **do**

1. Update weights $w$ by descending $\nabla_w \mathcal{L}_{train}(w, \alpha)$
2. Update architecture $\alpha$ by descending $\nabla_\alpha \mathcal{L}_{val}(w - \xi \nabla_w \mathcal{L}_{train}(w, \alpha), \alpha)$

Replace $\bar{o}^{(i,j)}$ with $o^{(i,j)} = \text{argmax}_{o \in \mathcal{O}} \; \alpha_o^{(i,j)}$ for each edge $(i, j)$

# Network structure

Initialise array of cells according to ‘genotype’ (specific cell structure trained before)

Connect each cell to two previous neighbors

Output of the final cell is the output of the network

Learn through backprop

# Discussion

Trains reasonable amount of time ~ 1 day on GTX 1080
Works both for CNNs and RNNs
No rigorous guarantees, but:

Table 1: Comparison with state-of-the-art image classifiers on CIFAR-10. Results marked with † were obtained by training the corresponding architectures using our setup.

| Architecture | Test Error (%) | Params (M) | Search Cost (GPU days) | Search Method |
|---|---|---|---|---|
| DenseNet-BC (Huang et al., 2017) | 3.46 | 25.6 | – | manual |
| NASNet-A + cutout (Zoph et al., 2017) | 2.65 | 3.3 | 1800 | RL |
| NASNet-A + cutout (Zoph et al., 2017)† | 2.83 | 3.1 | 3150 | RL |
| AmoebaNet-A + cutout (Real et al., 2018) | $3.34 \pm 0.06$ | 3.2 | 3150 | evolution |
| AmoebaNet-A + cutout (Real et al., 2018)† | 3.12 | 3.1 | 3150 | evolution |
| AmoebaNet-B + cutout (Real et al., 2018) | $2.55 \pm 0.05$ | 2.8 | 3150 | evolution |
| Hierarchical Evo (Liu et al., 2017b) | $3.75 \pm 0.12$ | 15.7 | 300 | evolution |
| PNAS (Liu et al., 2017a) | $3.41 \pm 0.09$ | 3.2 | 225 | SMBO |
| ENAS + cutout (Pham et al., 2018b) | 2.89 | 4.6 | 0.5 | RL |
| Random + cutout | 3.49 | 3.1 | – | – |
| DARTS (first order) + cutout | 2.94 | 2.9 | 1.5 | gradient-based |
| DARTS (second order) + cutout | $2.83 \pm 0.06$ | 3.4 | 4 | gradient-based |

# Play on your own

Original repository: https://github.com/quark0/darts (supports pytorch 0.4)

Updated repository: https://github.com/dragen1860/DARTS-PyTorch

```
git clone <> ; cd DARTS-PyTorch
nvidia-smi # check if you have GPU onboard
mkdir exp
 python train_search.py
<wait for genotype to evolve, add it to genotype.py>
python visualize.py NAME_OF_YOUR_GENOTYPE
python train.py —genotype NAME_OF_YOUR_GENOTYPE
```

# Other approaches

**Reinforcement learning**
Neural Architecture Search with Reinforcement Learning (Zoph and Le, 2016)
NASNet (Zoph *et al.*, 2017)
ENAS (Pham *et al.*, 2018)

**Evolutionary algorithm**
Hierarchical Evo (Liu *et al.*, 2017)
AmoebaNet (Real *et al.*, 2018)

**Sequential model-based optimisation (SMBO)**
PNAS (Liu *et al.*, 2017)

**Bayesian optimisation**
Auto-Keras (Jin *et al.*, 2018)
NASBOT (Kandasamy *et al.* 2018)

**Gradient-based optimisation**
SNAS (Xie *et al.*, 2018)
DARTS (Liu *et al.*, 2018)