Understanding and Robustifying Differentiable Architecture Search

Arber Zela[1], Thomas Elsken[2,1], Tonmoy Saikia[1], Yassine Marrakchi[1], Thomas Brox[1] & Frank Hutter[1,2]

[1]Department of Computer Science, University of Freiburg
{zelaa, saikiat, marrakch, brox, fh}@cs.uni-freiburg.de
[2]Bosch Center for Artificial Intelligence
Thomas.Elsken@de.bosch.com

February 19, 2020

Accepted as Oral at ICLR 2020

The Choice of Architecture Matters

- Performance improvements on various tasks mostly due to novel architectural design choices

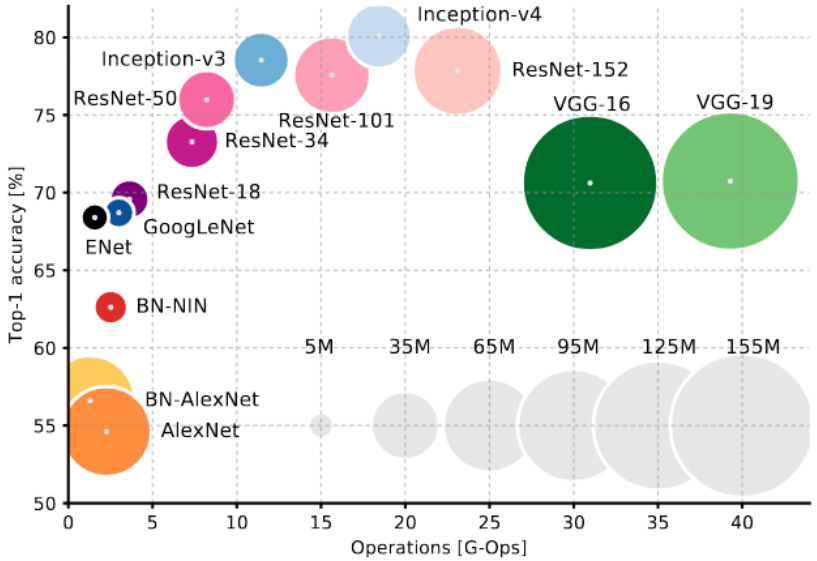


Figure: Larger circles, more network parameters [Canziani et al. 2017]

The Choice of Architecture Matters

- Performance improvements on various tasks mostly due to novel architectural design choices

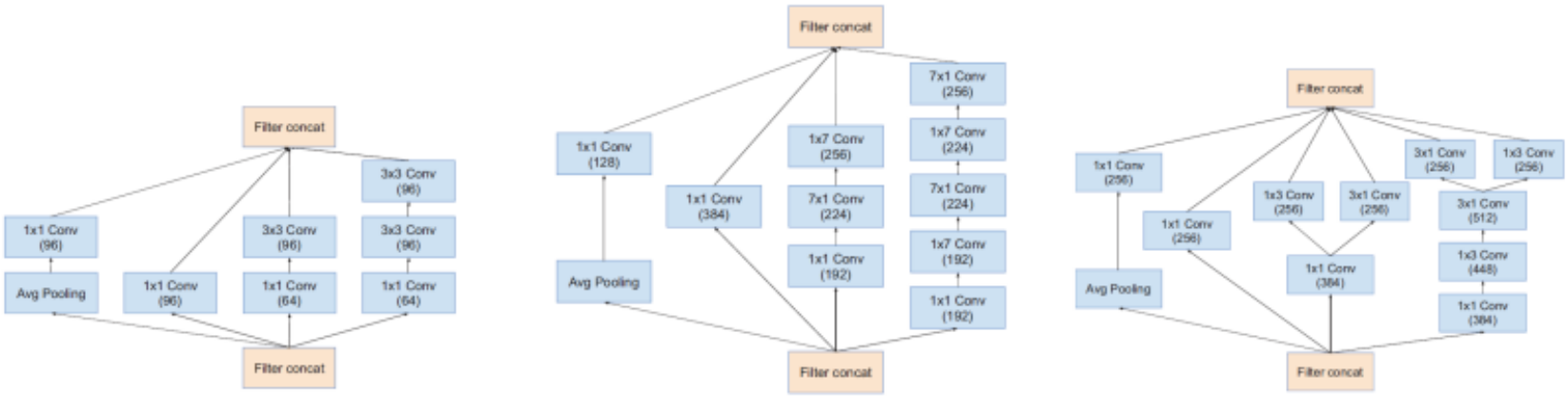


Figure: Inception-v4 modules [Szegedy et al. '17]

- Designing network architectures is hard, requiring lots of human efforts
  - Can we automate this design process?

Towards efficient Neural Architecture Search (NAS)

- RL & Evolution for NAS by Google Brain [Quoc Le's group, '16-'18]
  - New state-of-the-art results for CIFAR-10, ImageNet, Penn Treebank
  - Large computational demands
    - 800 GPUs for 2 weeks; 12800 architectures evaluated
  - Code not public

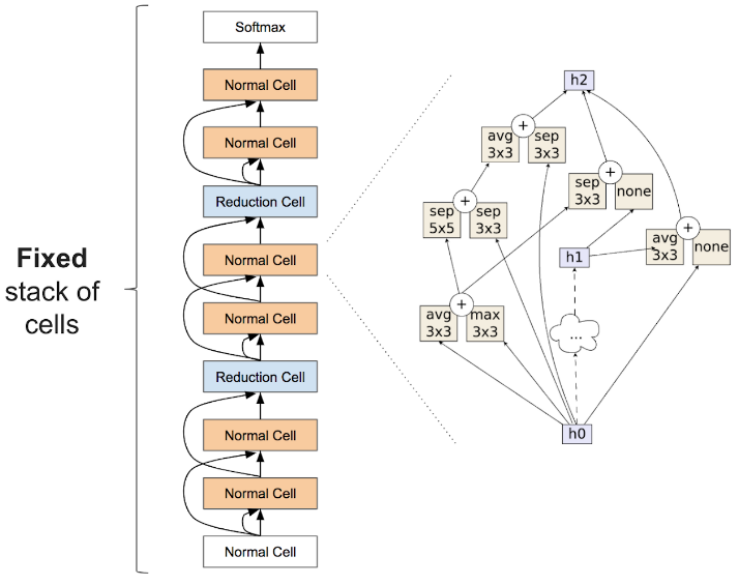


Figure taken from FastAI

Towards efficient Neural Architecture Search (NAS)

- RL & Evolution for NAS by Google Brain [Quoc Le's group, '16-'18]
  - New state-of-the-art results for CIFAR-10, ImageNet, Penn Treebank
  - Large computational demands
    - 800 GPUs for 2 weeks; 12800 architectures evaluated
  - Code not public

- Weight sharing/One-shot NAS [Pham et al,'18; Bender et al, '18; Liu et al, '19; Xie et al. '19; Cai et al. '19, Zhang et al. '19]
  - All possible architectures are subgraphs of a large supergraph (the one-shot model)
  - Weights are shared between different architectures with common edges/nodes in the supergraph
  - Search costs reduced to $<$ 1 GPU day.

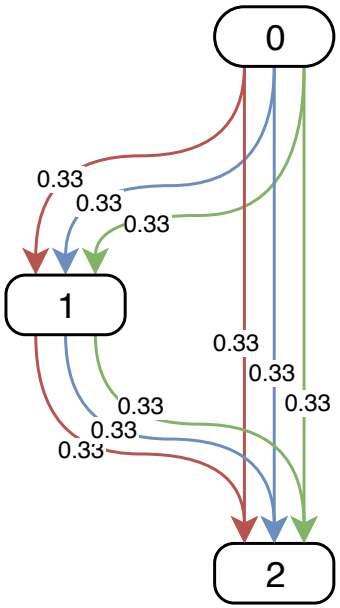
Differentiable NAS (DARTS) [Liu et al. '19]

- Neural Network as Directed Acyclic Graph
  - Nodes: fixed operators (element-wise addition, concatenation) on feature maps
  - Edges: operations ($sep\_conv\_3\times3$, $sep\_conv\_5\times5$, $dil\_conv\_3\times3$, $dil\_conv\_5\times5$, $max\_pool\_3\times3$, $avg\_pool\_3\times3$, *identity* and *zero*)

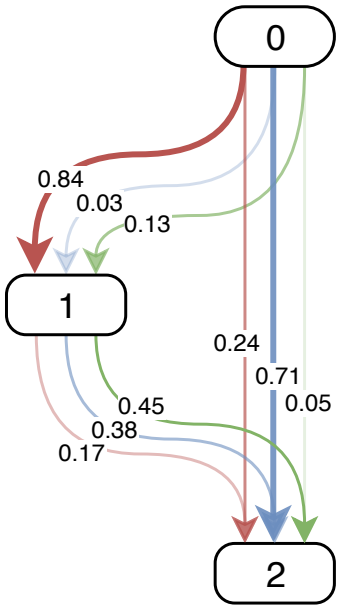
- Between 2 nodes: Categorical choice for which operation to use
  - Relax this discrete space to a continuous representation using a convex combination of these choices (MixedOps) $\longrightarrow$ one-shot model
  - Use SGD to search in the space of architectures.

Differentiable Architecture Search (DARTS) [Liu et al. '19]

- $x^{(j)} = \sum_{i<j} \tilde{o}^{(i,j)}(x^{(i)}) = \sum_{i<j} \sum_{o \in \mathcal{O}} \frac{e^{\alpha_o^{(i,j)}}}{\sum_{o' \in \mathcal{O}} e^{\alpha_{o'}^{(i,j)}}} o(x^{(i)})$



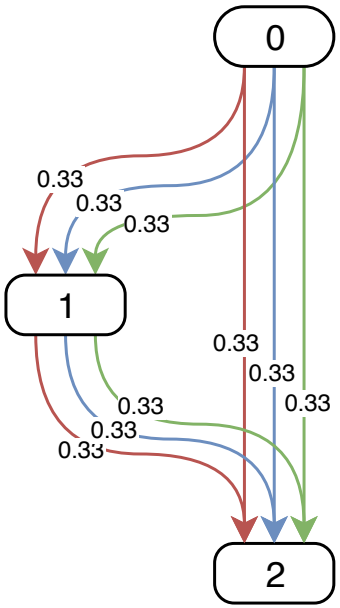
(a) Search start



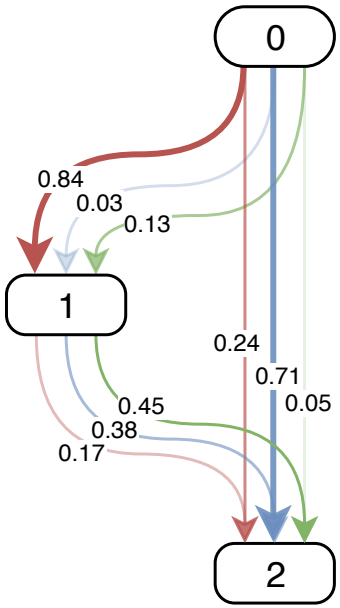
(b) Search end

Differentiable Architecture Search (DARTS) [Liu et al. '19]

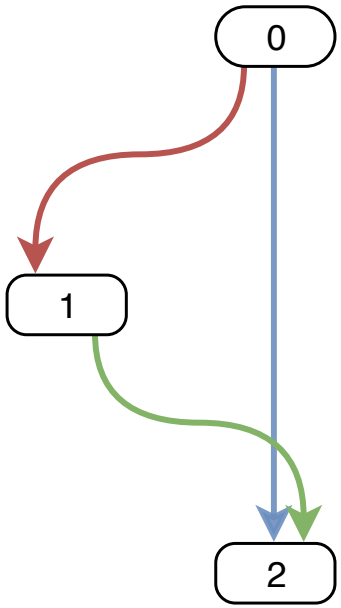
- $x^{(j)} = \sum_{i<j} \tilde{o}^{(i,j)}(x^{(i)}) = \sum_{i<j} \sum_{o \in \mathcal{O}} \frac{e^{\alpha_o^{(i,j)}}}{\sum_{o' \in \mathcal{O}} e^{\alpha_{o'}^{(i,j)}}} o(x^{(i)})$
- $o^{(i,j)} \in \arg\max_{o \in \mathcal{O}} \alpha_o^{(i,j)}$



(d) Search start



(e) Search end



(f) Final cell

DARTS: Architecture Optimization

- Optimizing both $\mathcal{L}_{train}$ and $\mathcal{L}_{valid}$ corresponds to a bilevel optimization problem:

$$\min_{\alpha}\{f(\alpha) \triangleq \mathcal{L}_{valid}(w^*(\alpha), \alpha)\}$$
$$s.t. \quad w^*(\alpha) = \arg\min_{w} \mathcal{L}_{train}(w, \alpha),$$

  where

  - $\alpha \longrightarrow$ architectural weights
  - $w \longrightarrow$ operation weights

DARTS: Architecture Optimization

- Optimizing both $\mathcal{L}_{train}$ and $\mathcal{L}_{valid}$ corresponds to a bilevel optimization problem:

$$\min_{\alpha}\{f(\alpha) \triangleq \mathcal{L}_{valid}(w^*(\alpha), \alpha)\}$$
$$s.t. \quad w^*(\alpha) = \arg\min_{w} \mathcal{L}_{train}(w, \alpha),$$

  where

  - $\alpha \longrightarrow$ architectural weights
  - $w \longrightarrow$ operation weights

- Approximate $w^*(\alpha) \approx w - \xi \nabla_w \mathcal{L}_{train}(w, \alpha)$
- The optimization alternates between:
  1. Update w by $\nabla_w \mathcal{L}_{train}(w, \alpha)$
  2. Update α by $\nabla_\alpha \mathcal{L}_{valid}(\mathbf{w} - \xi \nabla_w \mathcal{L}_{train}(w, \alpha), \alpha)$

Works quite well on many benchmarks

- Original CNN space: 8 operations on each MixedOp
- 28 MixedOPs in total
- $> 10^{23}$ possible architectures
- $< 3\%$ on CIFAR-10 in less than 1 GPU day of search

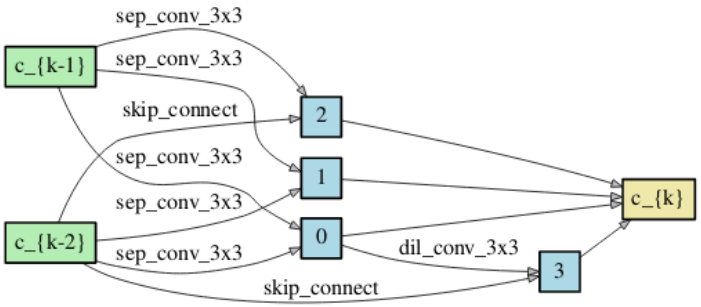


Figure 4: Normal cell learned on CIFAR-10.

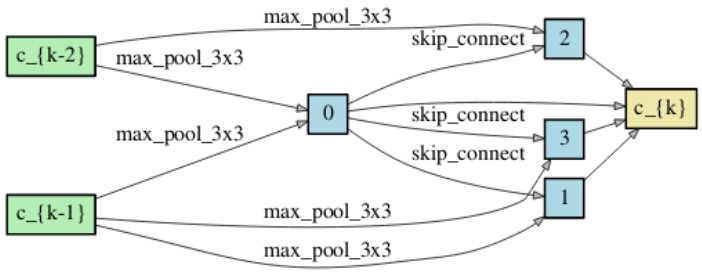


Figure 5: Reduction cell learned on CIFAR-10.

But not always...

S1: This search space uses a different set of two operators per edge, derived by iteratively running DARTs and pruning unimportant operations.

S2: $\{3 \times 3$ *SepConv*, *SkipConnect*$\}$.

S3: $\{3 \times 3$ *SepConv*, *SkipConnect*, *Zero*$\}$,

S4: $\{3 \times 3$ *SepConv*, *Noise*$\}$.

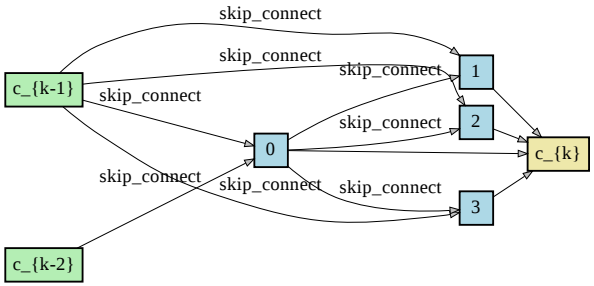
But not always...

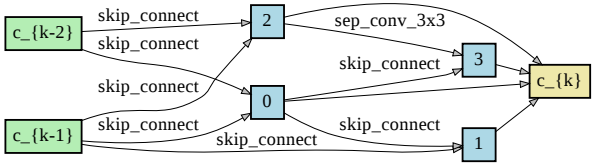
S1: This search space uses a different set of two operators per edge, derived by iteratively running DARTs and pruning unimportant operations.

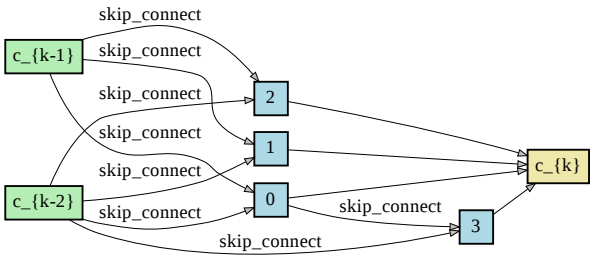
S2: $\{3 \times 3$ *SepConv*, *SkipConnect*$\}$.

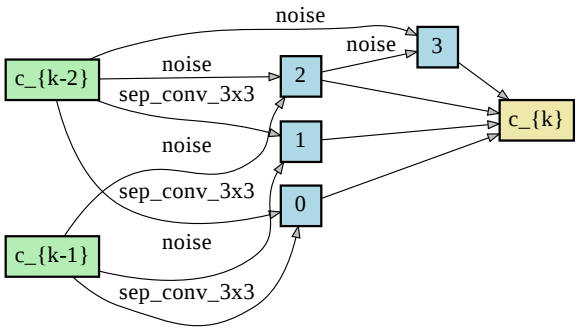
S3: $\{3 \times 3$ *SepConv*, *SkipConnect*, *Zero*$\}$,

S4: $\{3 \times 3$ *SepConv*, *Noise*$\}$.









Architecture overfitting

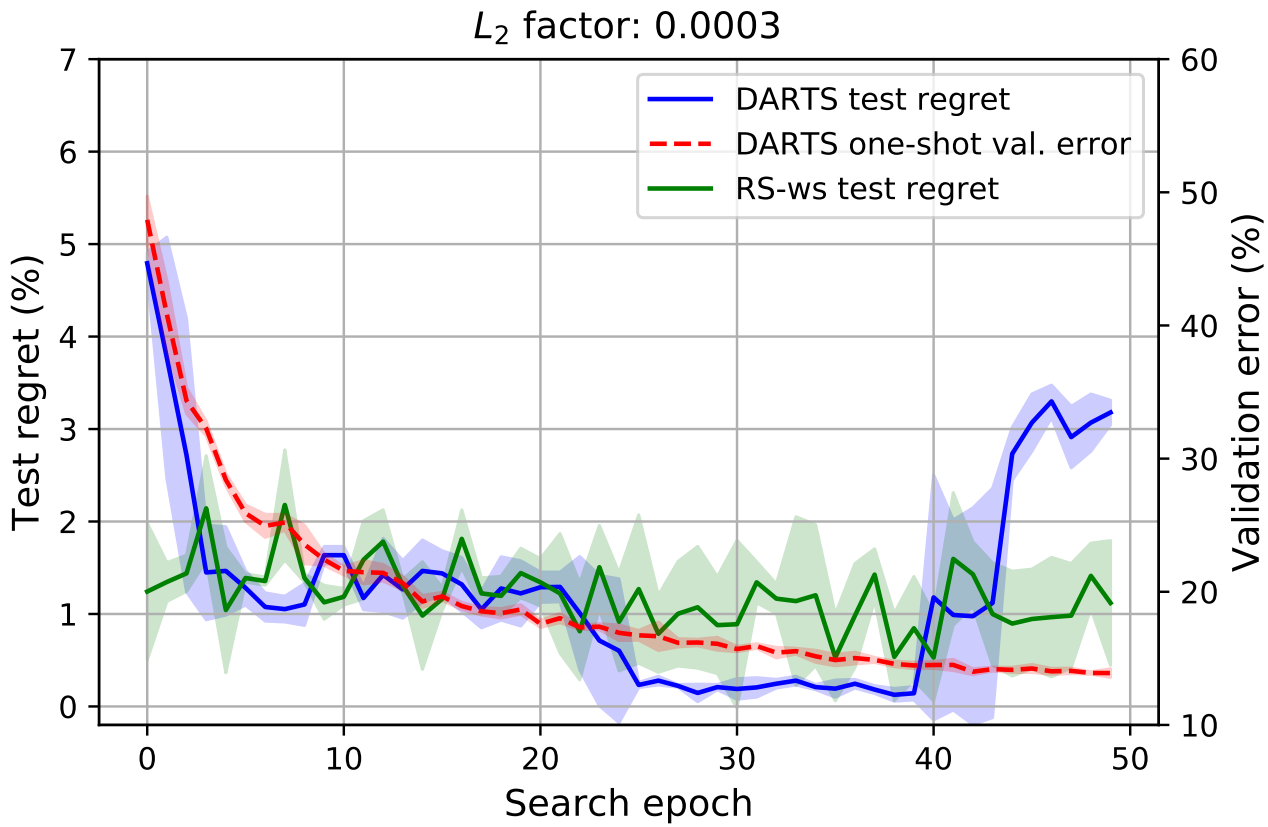
S5: Very small search space with known global optimum.

- 81 possible architectures trained 3 independent times using the default DARTS settings.

Architecture overfitting

S5: Very small search space with known global optimum.

- 81 possible architectures trained 3 independent times using the default DARTS settings.
- Architectural parameters start overfitting to the validation set.



Architecture overfitting

- What would be a good feature that would detect overfitting without training and evaluating the architectures from scratch (too expensive!)?

Architecture overfitting

- What would be a good feature that would detect overfitting without training and evaluating the architectures from scratch (too expensive!)?
- **HINT:** flatness/sharpness of minimas, e.g. in large vs. small batch size training of NN is a good indicator of generalization.

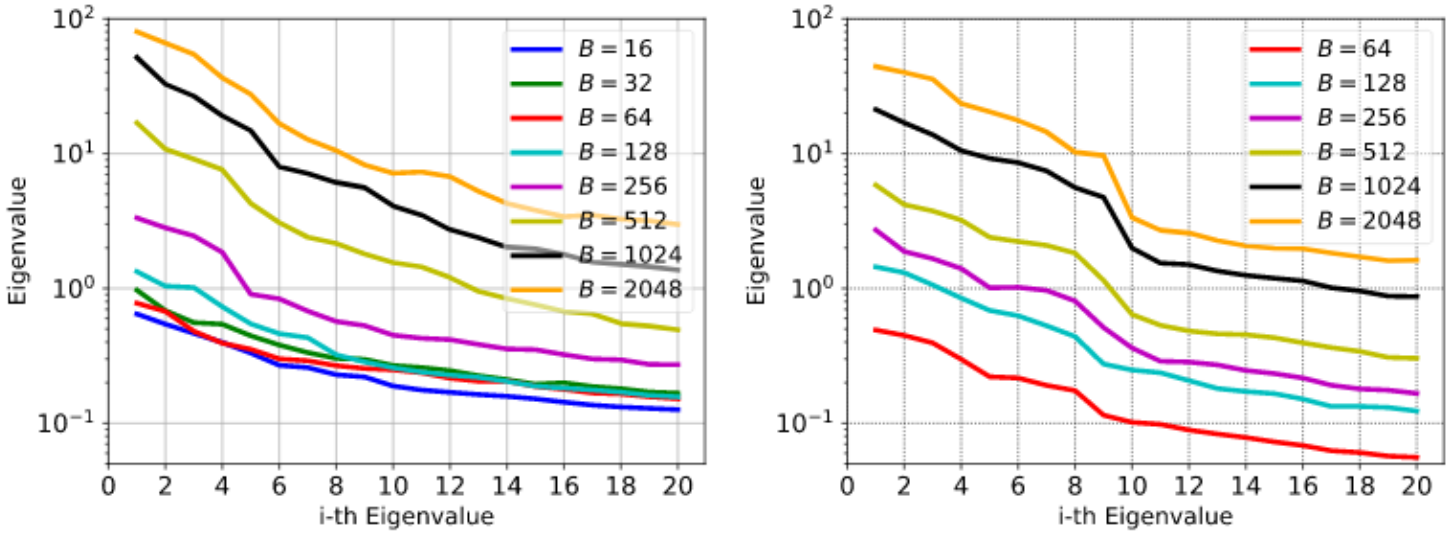


Figure 1: *Top 20 eigenvalues of the Hessian is shown for C1 on CIFAR-10 (left) and M1 on MNIST (right) datasets. The spectrum is computed using power iteration with relative error of* 1E-4.

[2]

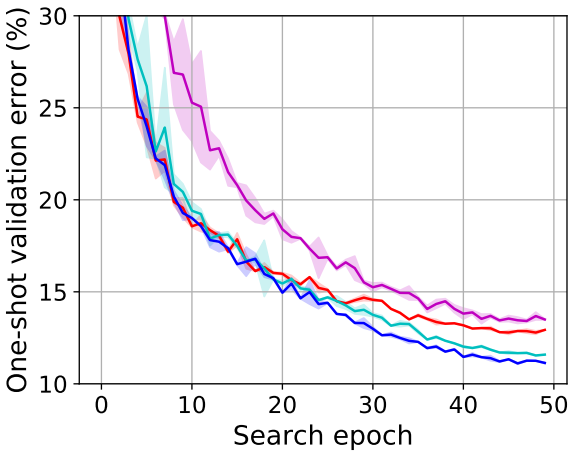
[2] Hessian-based Analysis of Large Batch Training and Robustness to Adversaries. Yao et al. NeurIPS '18

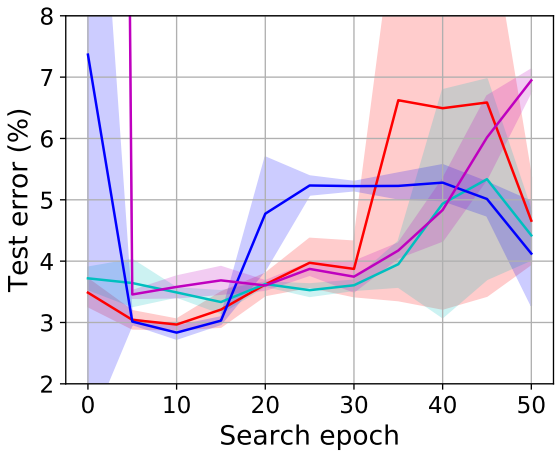
Generalization of architectures and sharpness of minimas

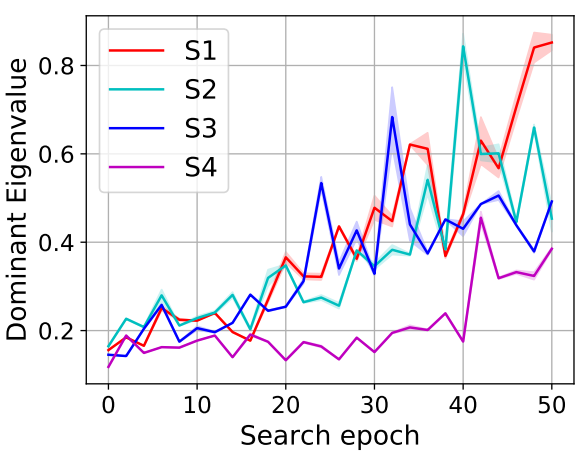
- Compute the full Hessian $\nabla^2_\alpha \mathcal{L}_{val}$ on a randomly sampled mini-batch from the validation set.

Generalization of architectures and sharpness of minimas

- Compute the full Hessian $\nabla^2_\alpha \mathcal{L}_{val}$ on a randomly sampled mini-batch from the validation set.
- The dominant EV starts increasing at the point where the architecture generalization error starts increasing.

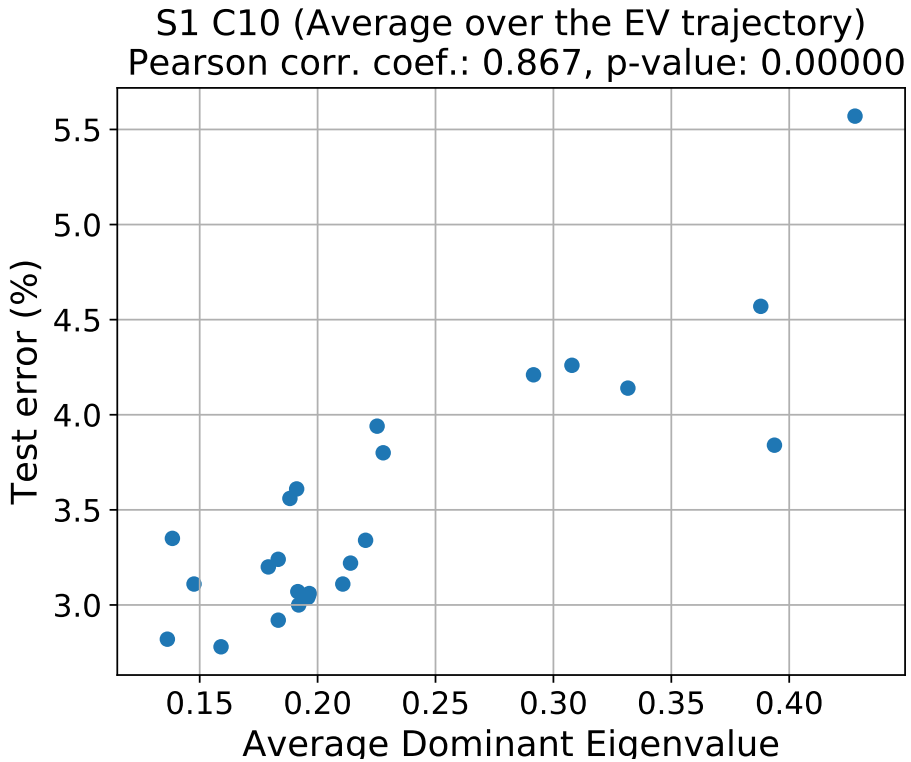






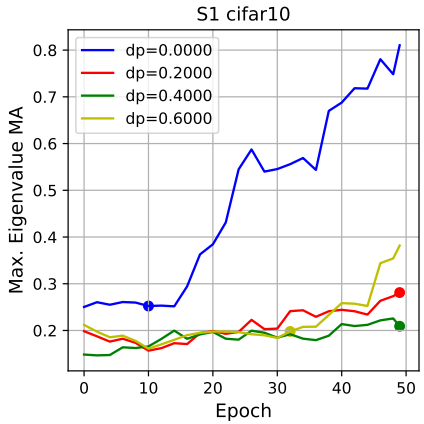
Generalization of architectures and sharpness of minimas

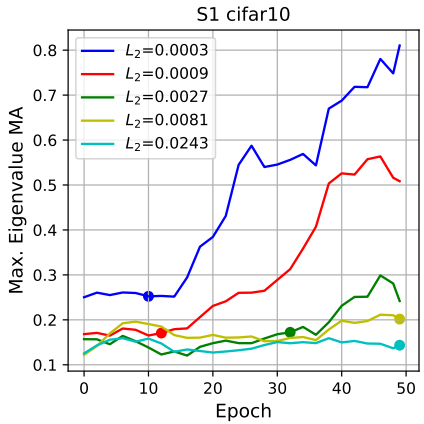
- Compute the full Hessian $\nabla^2_\alpha \mathcal{L}_{val}$ on a randomly sampled mini-batch from the validation set.
- The dominant EV starts increasing at the point where the architecture generalization error starts increasing.
- High correlation between generalization and the dominant eigenvalue (EV)



Early Stopping and Meta-regularization

- **Goal**: Keep the dominant eigenvalue to a low value
  - Early stop whenever the EV increases rapidly
  - Regularize the inner problem

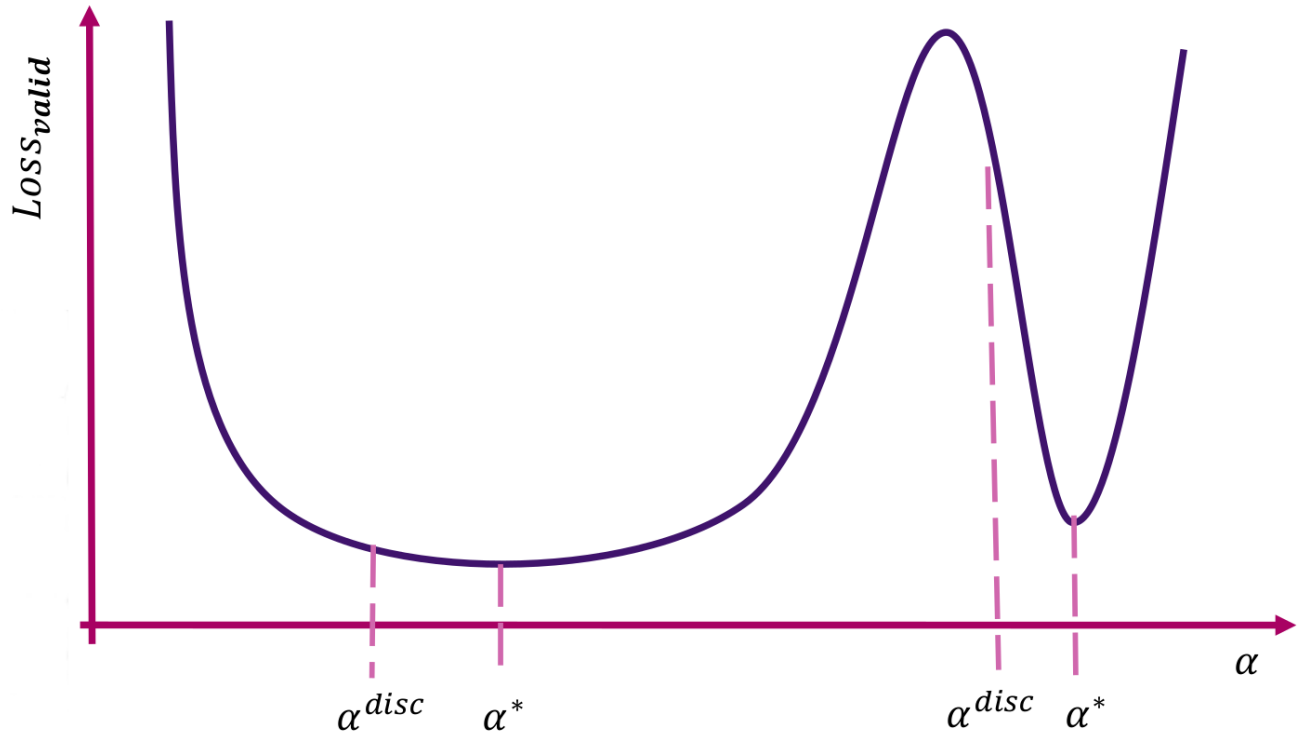




Benchmark		DARTS	DARTS-ES
C10	S1	4.66 ± 0.71	**3.05** $\pm$ 0.07
	S2	4.42 ± 0.40	**3.41** $\pm$ 0.14
	S3	4.12 ± 0.85	**3.71** $\pm$ 1.14
	S4	6.95 ± 0.18	**4.17** $\pm$ 0.21
C100	S1	29.93 ± 0.41	**28.90** $\pm$ 0.81
	S2	28.75 ± 0.92	**24.68** $\pm$ 1.43
	S3	29.01 ± 0.24	**26.99** $\pm$ 1.79
	S4	24.77 ± 1.51	**23.90** $\pm$ 2.01
SVHN	S1	9.88 ± 5.50	**2.80** $\pm$ 0.09
	S2	3.69 ± 0.12	**2.68** $\pm$ 0.18
	S3	4.00 ± 1.01	**2.78** $\pm$ 0.29
	S4	2.90 ± 0.02	**2.55** $\pm$ 0.15

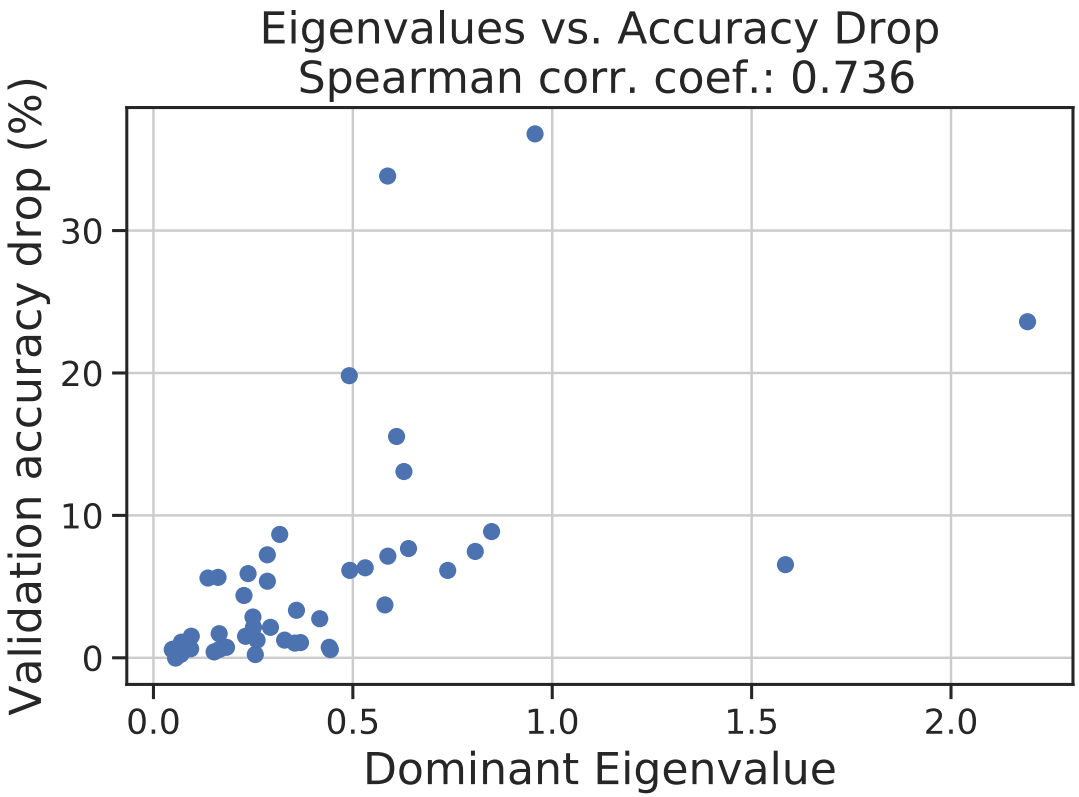
How the curvature relates with generalization?

- Sharp minimas much more sensitive to variations in the input space.
- **DARTS discretizes (i.e. takes argmax over α) to get the final architecture.**



How the curvature relates with generalization?

- Sharp minimas much more sensitive to variations in the input space.
- **DARTS discretizes (i.e. takes argmax over α) to get the final architecture.**



- Evaluate the found architectures with the search model weights. Report the accuracy drop relative to the search model performance.

How the curvature relates with generalization?

- Sharp minimas much more sensitive to variations in the input space.
- **DARTS discretizes (i.e. takes argmax over α) to get the final architecture.**

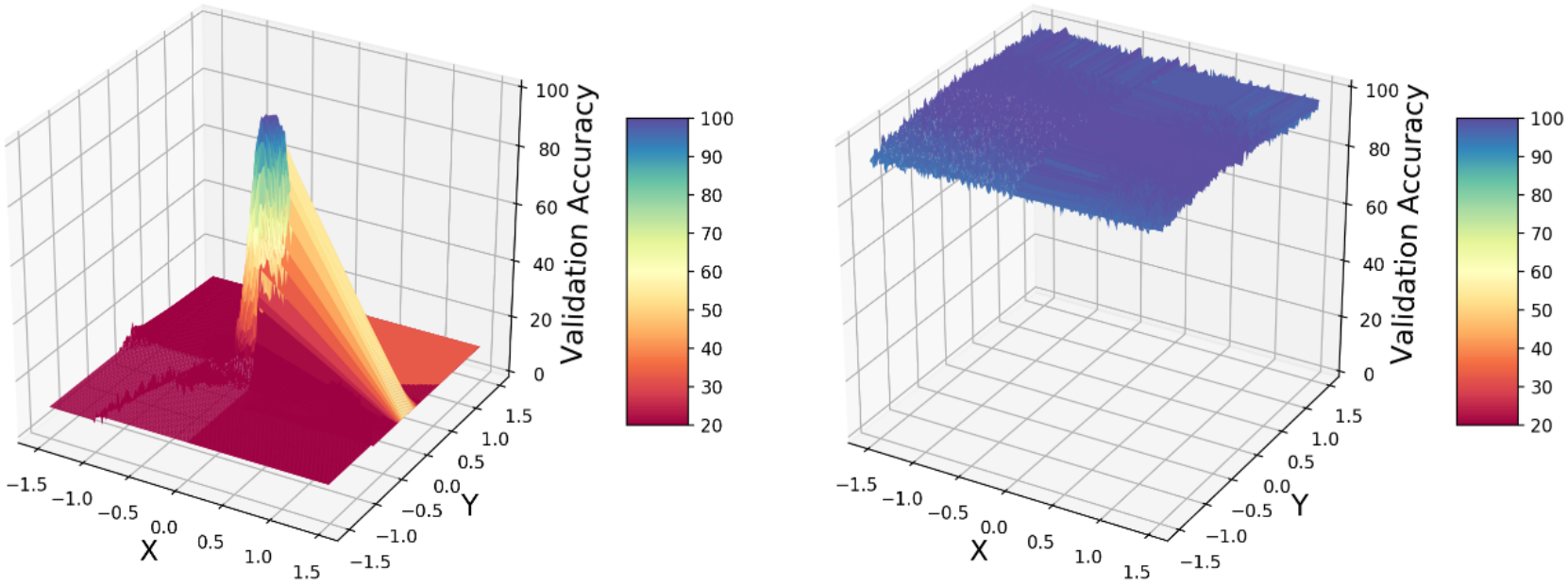


Figure: Taken from SDARTS-RS [Chen & Hsieh, 2020]

Benchmark Results

Empirical evaluation of practical robustified versions of DARTS. Each entry is the test error after retraining the selected architecture as usual. The best method for each setting is boldface and underlined, the second best boldface.

Benchmark		RS-ws	DARTS	R-DARTS(DP)	R-DARTS(L2)	DARTS-ES	DARTS-ADA
C10	S1	3.23	3.84	3.11	**2.78**	**3.01**	3.10
	S2	3.66	4.85	3.48	**3.31**	**3.26**	3.35
	S3	2.95	3.34	2.93	**2.51**	2.74	**2.59**
	S4	8.07	7.20	**3.58**	**3.56**	3.71	4.84
C100	S1	**23.30**	29.46	25.93	24.25	28.37	**24.03**
	S2	**21.21**	26.05	22.30	**22.24**	23.25	23.52
	S3	23.75	28.90	**22.36**	23.99	23.73	**23.37**
	S4	28.19	22.85	22.18	**21.94**	**21.26**	23.20
SVHN	S1	2.59	4.58	**2.55**	4.79	2.72	**2.53**
	S2	2.72	3.53	**2.52**	**2.51**	2.60	2.54
	S3	2.87	3.41	**2.49**	**2.48**	2.50	2.50
	S4	3.46	3.05	2.61	**2.50**	2.51	**2.46**

More results

Effect of regularization for disparity estimation. Search was conducted on FlyingThings3D (FT) and then evaluated on both FT and Sintel.

Aug. Scale	**One-shot valid EPE**	**FT test EPE**	**Sintel test EPE**	**Params (M)**
0.0	4.49	3.83	5.69	9.65
0.1	3.53	3.75	5.97	9.65
0.5	3.28	3.37	5.22	9.43
1.0	4.61	3.12	5.47	12.46
1.5	5.23	2.60	4.15	12.57
2.0	7.45	**2.33**	**3.76**	12.25
L_2 reg. factor	**One-shot valid EPE**	**FT test EPE**	**Sintel test EPE**	**Params (M)**
3×10^{-4}	3.95	3.25	6.13	11.00
9×10^{-4}	5.97	**2.30**	4.12	13.92
27×10^{-4}	4.25	2.72	4.83	10.29
81×10^{-4}	4.61	**2.34**	**3.85**	12.16

DARTS vs. RobustDARTS on the original DARTS search spaces. We show mean $\pm$ stddev for 5 repetitions.

Benchmark	**DARTS**	**R-DARTS(L2)**
C10	**2.91 $\pm$ 0.25**	**2.95 $\pm$ 0.21**
C100	20.58 $\pm$ 0.44	**18.01 $\pm$ 0.26**
SVHN	2.46 $\pm$ 0.09	**2.17 $\pm$ 0.09**
PTB	58.64	**57.59**

Conclusions

1. We identify 12 NAS benchmarks in which standard DARTS yields degenerate architectures with poor test performance.
2. We show that there is a strong correlation between the sharpness of minimas and the architecture's generalization error.
3. Based on these observations we propose regularizers in the architectural level, such as:
   - EV-based early stopping
   - (Adaptive) regularization in the inner objective of DARTS