Efficient Academic Talk

NeurIPS 2099

Jane Musterfrau

Department of Computer Science
University of Example

Joint work with: John Doe, Alice Johnson

February 13, 2026

Motivation

Why neural architecture search matters

Manual architecture design is:

- Time-consuming (weeks to months) (Smith & Doe, 2023)
- Requires deep expertise
- Often suboptimal

Motivation

Why neural architecture search matters

The Promise of NAS

Automatically discover optimal architectures for specific tasks and constraints.

But current methods face challenges:

- Computational cost (thousands of GPU hours)
- Search space limitations
- Poor generalization

Related Work

Building on prior foundations

Reinforcement Learning

- NASNet (Zoph et al.)
- High performance
- Extremely costly

Evolutionary Methods

- AmoebaNet
- Good diversity
- Slow convergence

Gradient-Based

- DARTS (Liu et al.)
- Fast optimization
- Memory intensive

One-Shot Methods

- Weight sharing
- Efficient
- Ranking correlation issues

Zoph et al. 2018; Liu et al. 2019

Our Contribution

Research Question

Can we achieve SOTA NAS performance with 10x less compute while maintaining architecture quality?

Key Contributions

1. Novel gradient approximation reducing memory by 60%
2. Progressive search space pruning strategy
3. Theoretical analysis of convergence guarantees
4. SOTA results on CIFAR-10, ImageNet, and NAS-Bench-201

Methodology

Problem Formulation

Formalizing the search objective

Let $\mathcal{A}$ be the architecture space and $\mathcal{D}$ the dataset.

$$\alpha^* = \arg\min_{\alpha \in \mathcal{A}} \mathcal{L}_{\text{val}}(w^*(\alpha), \alpha)$$

where $w^*(\alpha) = \arg\min_w \mathcal{L}_{\text{train}}(w, \alpha)$

Bi-Level Optimization

The nested optimization makes NAS computationally expensive.

Liu et al. 2019

Gradient Approximation

First-order approximation eliminates expensive Hessian computation

Based on DARTS

$$\nabla_\alpha \mathcal{L}_{\text{val}} \approx \nabla_\alpha \mathcal{L}_{\text{val}} - \xi \nabla^2_{\alpha,w} \mathcal{L}_{\text{train}} \cdot \nabla_w \mathcal{L}_{\text{val}}$$

ξ	Learning rate for weight update
$\nabla^2_{\alpha,w}$	Mixed partial derivatives (approximated)
$\mathcal{L}_{\text{val}}, \mathcal{L}_{\text{train}}$	Validation and training losses

Implementation

```python
def darts_step(model, arch_params, train_data, val_data, xi=0.01):
    """One step of DARTS: bilevel optimization."""
    # 1. Update weights on training data
    w_loss = model.loss(train_data)
    w_loss.backward()
    optimizer_w.step()

    # 2. Update architecture on validation data
    a_loss = model.loss(val_data)
    a_loss.backward()
    optimizer_a.step()

    return a_loss.item()
```

Progressive Search Space Pruning

Reducing complexity during search

Epoch 0: Full search space

All $|\mathcal{O}|^E$ possible architectures

Progressive Search Space Pruning

Reducing complexity during search

Epoch 50: After first pruning

Top 50% operations retained

Operations with $\alpha_i < \tau$ are pruned.

Progressive Search Space Pruning

Reducing complexity during search

Epoch 100: Final architecture

Discrete architecture

Result

60% memory reduction with improved stability.

Results

Main Results: CIFAR-10

Comparison with state-of-the-art methods

Method	Error (%)	Params (M)	GPU Days
NASNet-A	2.65	3.3	1800
AmoebaNet-A	2.55	3.2	3150
DARTS (2nd)	2.76	3.3	4.0
PC-DARTS	2.57	3.6	0.1
Ours	**2.48**	3.4	**0.3**

Key Finding

Our method achieves lowest error with only 0.3 GPU days of search.

Ablation Study

Understanding component contributions

Gradient Approx.

Without: 2.71% With: 2.56%

−0.15%

Progressive Pruning

Without: 2.63% With: 2.56%

−0.07%

Combined

Baseline: 2.76% Full: 2.48%

−0.28%

> Insight
>
> Both components contribute, with gradient approximation having larger impact.

Transferability to ImageNet

Architectures found on CIFAR-10, evaluated on ImageNet

Top-1 Error	23.8%
Top-5 Error	7.1%
Parameters	5.3M
FLOPs	574M

Comparison

- MobileNetV2: 28.0% (3.4M)
- EfficientNet-B0: 23.7% (5.3M)
- **Ours: 23.8% (5.3M)**

Competitive with hand-designed efficient architectures.

Discussion

Key Takeaways

Main Finding

Efficient gradient-based NAS can match or exceed expensive methods at a fraction of the cost.

Practical Implications

1. Democratizes NAS for resource-constrained researchers
2. Enables rapid architecture iteration
3. Opens new research directions in efficient search

Limitations

Current Limitations

- Search space still manually designed
- Limited to differentiable operations
- Proxy task gap (CIFAR $\longrightarrow$ ImageNet)
- Reproducibility challenges with random seeds

Future work should address these through:

- Learned search spaces
- Non-differentiable operation handling
- Direct large-scale search

Future Directions

Short-term

- Multi-objective NAS
- Hardware-aware search
- Robustness constraints

Medium-term

- Zero-cost proxies
- Transfer learning
- Foundation model adaptation

Long-term

- Fully automated ML
- Self-improving systems
- Theoretical foundations

Summary

Conclusion

What We Showed

1. Novel gradient approximation for efficient NAS
2. Progressive pruning reduces memory 60%
3. SOTA results: 2.48% error on CIFAR-10
4. Only 0.3 GPU days (10x faster than DARTS)

Open Questions

How can we further close the gap between proxy and target tasks?

Acknowledgements

Logo University Computing Center for GPU resources

Logo National Science Foundation Grant 12345

Logo Industry Partner for dataset access

Special thanks to the anonymous reviewers for constructive feedback.

References

Liu, H., Simonyan, K., & Yang, Y. (2019,). DARTS: Differentiable Architecture Search. *International Conference on Learning Representations*.

Smith, J., & Doe, J. (2023). Advances in Modern Presentation Design. *Journal of Visual Communication*, *15*, 1–20.

Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2018). Learning Transferable Architectures for Scalable Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8697–8710.

Thank You

Questions?

jane.smith@university.edu

github.com/jsmith/efficient-nas

arxiv.org/abs/2024.xxxxx