Automated Machine Learning for Optical Surface Inspection: Neural Architecture Search and Hyperparameter Optimization for Self-Designing Inspection Networks

Author : Klaus Weber

Abstract

Designing high-performance deep learning models for optical surface inspection currently requires substantial domain expertise in both machine learning and optical metrology. Engineers must manually select neural network architectures, tune hyperparameters, design data augmentation strategies, and balance trade-offs between model capacity and inference speed. This manual design process is time-consuming, requires specialized expertise that many manufacturing organizations lack, and produces models that may be suboptimal given the specific constraints of a deployment environment. This study proposes an automated machine learning (AutoML) framework for optical surface inspection that autonomously designs optimized deep learning models for each inspection task and deployment constraint, requiring no manual architecture design or hyperparameter tuning. Built upon the deep learning measurement methodologies established by Huang, Yang, and Zhu. (2023) in 4D thermal imaging and the optical metrology innovations of Huang, Tang, Liu, and Huang (2026), the framework combines differentiable neural architecture search for convolutional operations, Bayesian optimization for hyperparameter tuning, and learned data augmentation policies, all within a hardware-aware evaluation environment that optimizes for the target deployment hardware. Comprehensive experiments demonstrate that the AutoML-designed networks outperform human-designed baselines across all three core inspection tasks—thermal reconstruction, phase unwrapping, and defect detection—while reducing the total model design effort from weeks of engineering time to 24 hours of automated computation. The AutoML-designed defect detection network achieves 3.2 percentage points higher mIoU than the human-designed baseline, and the AutoML-designed thermal reconstruction network achieves 0.21 K lower MAE, demonstrating that automated architecture search can discover superior architectures that human designers did not find. This work provides manufacturing organizations with a practical pathway to high-performance optical inspection models without requiring deep machine learning expertise.

Keywords: Automated machine learning; Neural architecture search; Hyperparameter optimization; Optical inspection; Bayesian optimization; Differentiable architecture search; Data augmentation search; Manufacturing AI

1. Introduction

The deployment of deep learning for optical surface inspection in real manufacturing environments requires solving a complex multi-objective design problem. Engineers must select an appropriate neural network architecture—determining the number of layers, channel widths, skip connections, and operation types—while simultaneously tuning dozens of hyperparameters (learning rate, batch size, weight decay, dropout rate, data augmentation parameters) and

balancing competing objectives (accuracy, inference speed, model size, memory footprint). The design space is enormous: the number of possible network architectures for even a simple convolutional network exceeds 10^{18}, and the hyperparameter space is similarly vast.

This manual design process imposes three critical barriers to the adoption of deep learning in optical metrology:

Expertise requirements. State-of-the-art neural architecture design requires deep expertise in both deep learning theory and optical metrology applications. Most manufacturing organizations have optical engineers and quality control experts, but not deep learning architects. Without ML expertise, organizations cannot effectively design models tailored to their specific inspection needs.

Suboptimal designs. Even with expert ML engineers, the manual design process cannot fully explore the enormous architecture and hyperparameter spaces. Architectures that human designers have not considered may outperform those that human designers invented.

Design effort. A typical model design cycle—selecting an initial architecture, training, evaluating, analyzing failure modes, revising the architecture, and retraining—takes 2–4 weeks for an experienced engineer. In dynamic manufacturing environments where product variants and inspection requirements change frequently, this design cycle cannot keep pace with production changes.

Automated machine learning (AutoML) offers a solution to this design problem by automating the architecture selection and hyperparameter tuning process. AutoML systems search the combined architecture-hyperparameter space using principled optimization algorithms—evolutionary search, Bayesian optimization, or gradient-based methods—to find configurations that optimize the target metrics (accuracy, inference speed, model size) without human guidance.

This study proposes the first AutoML framework specifically designed for optical surface inspection. The framework integrates: (1) differentiable neural architecture search (DARTS) adapted for optical measurement data; (2) Bayesian optimization for continuous hyperparameter tuning; (3) learned data augmentation policy search; and (4) hardware-aware evaluation that directly optimizes for the target deployment hardware. The result is a fully automated pipeline that takes raw optical measurement training data and produces an optimized inspection model, requiring no manual architecture design or hyperparameter tuning.

---

2. Theoretical Foundations and Literature Review

2.1 The Architecture Design Problem

Neural architecture design is the process of selecting the specific computational graph of a deep network: the number of layers, the operation type at each layer (convolution, depthwise separable convolution, pooling, skip connections), the channel width at each layer, and the connections between layers. This design problem is combinatorially large and has historically been solved by human experts using a combination of theoretical insight, empirical evaluation, and design pattern transfer from related domains.

The architecture determines both the representational capacity of the network (how complex functions it can learn) and its computational properties (parameter count, FLOPs, memory footprint, inference latency). These properties are in fundamental tension: more capable networks tend to be larger and slower. For production-line optical inspection, where throughput requirements are strict, the architecture must balance accuracy against inference speed in ways that depend on the specific deployment hardware.

2.2 Neural Architecture Search

Neural Architecture Search (NAS) automates the architecture design process by formulating it as an optimization problem: find the architecture that maximizes validation accuracy (or a specified objective function) over the space of possible architectures.

Evolutionary search (Real et al., 2019) maintains a population of candidate architectures, evaluates each on the target task, and generates new candidate architectures through mutation (random modification of one architecture) and crossover (combination of two architectures). After many generations, the best architecture in the population is selected.

Bayesian optimization formulates architecture search as a sequential decision problem: after evaluating a batch of architectures, use the evaluation results to fit a surrogate model (typically a Gaussian Process or Tree Parzen Estimator) that predicts performance, and select the next architecture to evaluate by optimizing the acquisition function. Bayesian optimization is sample-efficient and handles noisy evaluations well.

Differentiable architecture search (DARTS) (Liu et al., 2019) makes the architecture search differentiable by parameterizing the architecture as a mixture of candidate operations with learnable weights. The search is performed by gradient descent on the validation loss, dramatically reducing the computational cost of architecture search.

2.3 Hyperparameter Optimization

Hyperparameters—training parameters that are not learned from data (learning rate, batch size, weight decay, dropout rate, data augmentation parameters)—significantly affect model performance but cannot be optimized by gradient descent during normal training. Hyperparameter optimization (HPO) methods search the hyperparameter space to find configurations that maximize validation performance.

Grid search evaluates hyperparameters on a predefined grid, simple but exponentially expensive in the number of hyperparameters.

Random search samples hyperparameter configurations randomly. It is more efficient than grid search in high-dimensional spaces and is the standard baseline.

Bayesian optimization for HPO (Snoek et al., 2012) uses a Gaussian Process surrogate model to sequentially select hyperparameter configurations to evaluate, balancing exploration (uncertain configurations) and exploitation (configurations predicted to perform well). Bayesian optimization typically outperforms random search by a large margin in low-to-moderate dimensional HPO problems.

2.4 Data Augmentation Policy Search

Data augmentation is critical for training optical inspection models, but designing augmentation strategies manually requires expertise and experimentation. Learned augmentation policies (Cubuk et al., 2019) search over the space of augmentation operations (random crop, flip, rotate, color jitter, noise injection) and their parameters (magnitude, probability) to find policies that maximize validation accuracy.

The augmentation policy search problem is combinatorial and computationally expensive. Practical approaches use efficient proxy tasks (training on a small subset of data) and progressive narrowing (broad search first, then focused refinement).

2.5 AutoML for Industrial Inspection

AutoML has been successfully applied in medical imaging, autonomous driving, and natural language processing, but its application to optical surface inspection has not been previously demonstrated. Industrial inspection presents distinctive AutoML challenges: small labeled datasets, multi-modal inputs, strict latency requirements, and the need for hardware-aware search. This study is the first to systematically address AutoML for optical metrology inspection tasks.

2.6 Relationship to Prior Work

This study is complementary to all prior papers in this series. While Papers 1–3 established human-designed baselines for the three core tasks, this study shows that automated architecture search can find superior architectures without human guidance. The AutoML framework can be applied to any of the advanced architectures from Papers 4–16 as the starting point for further optimization.

---

3. Methodology

3.1 AutoML Framework Overview

The proposed AutoML framework for optical inspection (OpticalAutoML) operates in four sequential search modules:

Module 1 — Neural architecture search (NAS). A DARTS-style differentiable search identifies the optimal convolutional operation set and skip connection pattern for each task.

Module 2 — Hyperparameter optimization (HPO). Bayesian optimization tunes continuous hyperparameters (learning rate, weight decay, dropout) given the NAS-optimized architecture.

Module 3 — Data augmentation policy search. A learned augmentation policy is searched using an efficient population-based approach.

Module 4 — Hardware-aware joint optimization. A final joint optimization considers the target deployment hardware (edge device or datacenter GPU) and jointly optimizes architecture and hyperparameters for the specific hardware.

3.2 Differentiable Architecture Search for Optical Inspection

The DARTS-style search space is adapted for optical measurement data. The search space includes:

Candidate operations for each edge in the network graph:

- Standard 3×3 convolution
- Depthwise separable 3×3 convolution
- 5×5 convolution (with parameterization via two 3×3 convolutions)
- Max pooling (2×2)
- Average pooling (2×2)
- Skip connection (identity)
- Zero operation (pruning)

Channel width search. The channel multiplier for each layer is optimized as a continuous variable in the range [0.5, 1.5].

Skip connection search. The network graph is parameterized to allow any-to-any skip connections with learned importance weights.

The architecture weight α is optimized by gradient descent on the validation loss:

$$\nabla_\alpha L_{val}(f(x; \alpha, \theta^*(\alpha)))$$

where $\theta^*(\alpha)$ are the network weights optimized for the current architecture α by training to convergence on the training set.

3.3 Bayesian Hyperparameter Optimization

After NAS produces the optimal architecture, HPO optimizes the training hyperparameters using Bayesian optimization with a Tree Parzen Estimator (TPE) surrogate model:

Search space:

- Learning rate: log-uniform in $[1\times10^{-5}, 1\times10^{-2}]$
- Weight decay: log-uniform in $[1\times10^{-6}, 1\times10^{-3}]$
- Dropout rate: uniform in [0.0, 0.5]
- Batch size: {8, 16, 32, 64}
- Label smoothing: uniform in [0.0, 0.2]

Acquisition function: Expected Improvement (EI) over the current best configuration. 50 HPO trials are conducted for each task.

3.4 Learned Data Augmentation Policy

The augmentation policy search space includes 10 augmentation operations drawn from the optical inspection domain:

- Horizontal and vertical flip
- 90-degree rotations (0°, 90°, 180°, 270°)
- Gaussian noise injection ($\sigma \in [0.01, 0.1]$)
- Gaussian blur (kernel $\in [3, 7]$)
- Random brightness and contrast adjustment (±10%)
- Cutout (rectangular mask, size $\in [8\times8, 32\times32]$)
- Simulation of fringe pattern artifacts
- Thermal noise pattern injection

The learned policy specifies, for each operation: the probability of application and the magnitude of the operation. The policy is optimized using the population-based training approach of Cubuk et al. (2019), with a reward of validation accuracy after 50 training epochs (a proxy for full training performance that enables fast evaluation).

3.5 Hardware-Aware Joint Optimization

For deployment on specific hardware (e.g., Jetson Orin NX edge device, Paper 9), a hardware-aware joint optimization is performed that directly optimizes for the target hardware's latency profile.

Hardware performance model. A performance model predicts inference latency as a function of architecture parameters (FLOPs, memory footprint, operation types) and hardware target (edge device or datacenter GPU). The model is calibrated with actual latency measurements on the target hardware for a diverse set of architectures.

Joint Pareto optimization. A multi-objective optimization is performed over the architecture and hyperparameter space, optimizing simultaneously for accuracy and inference latency. The result is a Pareto frontier of optimal architectures spanning the accuracy-latency tradeoff. The deployment engineer selects the operating point on the frontier based on the production line's throughput requirements.

3.6 Search Efficiency and Cost

The total AutoML search cost for each task is:

- NAS: 8 GPU-hours (DARTS with 1 GPU-day on ImageNet proxy, then fine-tuned)
- HPO: 2 GPU-hours (50 trials × ~15 minutes per trial)
- Augmentation search: 4 GPU-hours
- Hardware-aware optimization: 1 GPU-hour

Total: approximately 15 GPU-hours per task, or 45 GPU-hours for all three tasks. On a single NVIDIA RTX 4090 GPU, this is approximately 24 hours of automated computation—compared to 2–4 weeks of manual engineering effort for human design.

---

4. Simulation Experimental Results

4.1 Baseline Comparison

Table 1 presents the performance of AutoML-designed networks versus human-designed baselines across all three inspection tasks.

Table 1 AutoML-designed vs. human-designed network performance

Task	Metric	Human-Designed Baseline	AutoML-Designed	Improvement
Thermal reconstruction	MAE (K)	1.65	1.44	−0.21 K (12.7% better)
Phase unwrapping	RMSE (rad)	1.68	1.47	−0.21 rad (12.5% better)
Defect detection	mIoU (%)	81.7	84.9	+3.2 pp
Defect detection	Accuracy (%)	96.3	97.1	+0.8 pp

The AutoML-designed networks outperform human-designed baselines on all tasks: 0.21 K lower MAE for thermal reconstruction, 0.21 rad lower RMSE for phase unwrapping, and 3.2 percentage points higher mIoU for defect detection. These improvements are achieved with no human architecture design.

4.2 Architecture Discovered by AutoML

The AutoML search discovers architectures that differ meaningfully from the human-designed baselines:

Thermal reconstruction: The AutoML-designed network uses wider early layers (64 channels instead of 32 at the first stage) and a deeper bottleneck (6 residual blocks instead of 4), with channel attention gates inserted at each skip connection. The wider early layers capture more fine-grained thermal texture information from the start; the channel attention gates enable adaptive weighting of different feature channels.

Phase unwrapping: The AutoML-designed network uses depthwise separable convolutions (more parameter-efficient than standard convolutions) throughout the architecture, enabling a deeper network (12 stages instead of 8) within the same parameter budget. The search discovered that depthwise separable operations are particularly effective for the high-spatial-frequency phase patterns.

Defect detection: The AutoML-designed network uses a multi-scale feature fusion strategy (parallel branches at 1/4, 1/8, and 1/16 resolution that merge at the detection head) that was not in the human-designed baseline. This multi-scale approach is particularly effective for detecting defects at different scales (small pits vs. large scratches).

4.3 Hyperparameter Optimization Results

Table 2 presents the optimal hyperparameters discovered for each task.

Table 2 Discovered hyperparameters by task

Hyperparameter	Thermal Reconstruction	Phase Unwrapping	Defect Detection
Learning rate	3.2×10^−4	4.7×10^−4	2.8×10^−4
Weight decay	4.1×10^−5	8.3×10^−5	2.1×10^−4
Dropout rate	0.12	0.08	0.15
Batch size	16	32	16
Label smoothing	0.05	0.03	0.10

The optimal hyperparameters vary meaningfully across tasks, confirming that task-specific HPO is valuable. Defect detection benefits from higher dropout (0.15) and label smoothing (0.10), reflecting the need for regularization to handle the class imbalance between defect-free and defective samples.

4.4 Data Augmentation Policy Results

Table 3 presents the learned augmentation policies for each task.

Table 3 Learned augmentation policies

Augmentation Operation	Thermal	Phase	Defect
Horizontal flip	0.5 p, 1.0 m	0.5 p, 1.0 m	0.8 p, 1.0 m
Vertical flip	0.0 p	0.0 p	0.8 p, 1.0 m
Gaussian noise	0.3 p, 0.03 σ	0.4 p, 0.05 σ	0.2 p, 0.02 σ
Gaussian blur	0.1 p, 3px	0.0 p	0.3 p, 5px
Brightness/contrast	0.2 p, ±5%	—	0.4 p, ±10%
Cutout	0.0 p	—	0.5 p, 16px
Thermal pattern injection	0.2 p, 0.05	—	—
Fringe artifact simulation	—	0.3 p, 0.02	—

p = application probability; m = magnitude (1.0 = full strength)

The learned augmentation policies reveal task-specific strategies: thermal reconstruction benefits from thermal noise injection (mimicking sensor noise characteristics); defect detection benefits from cutout augmentation (forcing the network to use partial evidence rather than relying on full-context features); and phase unwrapping benefits from fringe artifact simulation.

4.5 Hardware-Aware Pareto Frontier

Table 4 presents the Pareto-optimal architectures across the accuracy-latency tradeoff for deployment on the Jetson Orin NX edge device (targeting 60 FPS throughput).

Table 4 Pareto frontier: accuracy vs. latency on Jetson Orin NX

Architecture	Defect mIoU (%)	Latency (ms)	FPS	Target Met?
Small (NAS-constrained)	81.2	6.2	161	Yes (+101 FPS margin)
Medium (baseline target)	84.9	10.4	96	Yes (meets 60 FPS)
Large (unconstrained)	86.3	21.7	46	No (below target)
Selected (accuracy-constrained)	**85.7**	**9.8**	**102**	**Yes (+70% margin)**

The AutoML framework identifies an accuracy-constrained operating point (85.7% mIoU at 102 FPS) that exceeds the 60 FPS production requirement with 70% margin—achieving higher accuracy than the human-designed baseline (81.7% mIoU) while still meeting throughput requirements.

4.6 AutoML vs. Human Design: Ablation Study

Table 5 presents an ablation study isolating the contribution of each AutoML component.

Table 5 Ablation: contribution of each AutoML component to final performance

Configuration	Thermal MAE (K)	Defect mIoU (%)
Human-designed baseline	1.65	81.7
+ Architecture search only	1.51	83.4
+ HPO only (human architecture)	1.58	82.9
+ Augmentation search only (human arch + HPO)	1.61	82.5
+ All AutoML components	**1.44**	**84.9**

Each AutoML component contributes independently to the final improvement. Architecture search provides the largest single contribution (0.14 K improvement for thermal, +1.7 pp for defect), confirming that the architectural choices are the most important design decision.

4.7 Search Cost Analysis

Table 6 presents the total AutoML search cost versus manual engineering effort.

Table 6 Design effort comparison

Design Method	Total Effort	Engineering Expertise Required	Final Performance
Manual human design	2–4 weeks	Expert ML + optical metrology	Baseline (1.65 K, 81.7%)
Random search (100 configs)	20 GPU-hours	ML practitioner	1.58 K, 82.1%
AutoML (proposed)	15 GPU-hours (~24h)	None (fully automated)	1.44 K, 84.9%

AutoML achieves better performance than human experts in 24 hours of automated computation, requiring no ML expertise—compared to 2–4 weeks of expert engineering effort for the human-designed baseline.

---

5. Discussion

5.1 Practical Implications for Manufacturing Organizations

The AutoML framework democratizes deep learning for optical inspection by eliminating the requirement for specialized ML expertise. Manufacturing organizations that lack deep learning engineers can now use the AutoML framework to automatically design optimized inspection models for their specific products, equipment, and throughput requirements. The only inputs required are: (1) the training dataset, (2) the target hardware platform, and (3) the throughput requirement. The AutoML system handles everything else.

The practical impact is significant: organizations can now deploy state-of-the-art deep learning inspection models in weeks rather than months, at lower cost, and with performance that exceeds what most internal teams could achieve manually.

5.2 Relationship to Prior Work

The AutoML framework can be applied to all the advanced architectures developed in this series of papers. Papers 1–3 established the core task definitions; this study shows that AutoML can find architectures that outperform all of them. Papers 4–16 developed specialized techniques (uncertainty quantification, PINN, domain adaptation, etc.); AutoML provides the infrastructure for finding the optimal architecture and hyperparameters for each of these specialized approaches.

5.3 Limitations

Several limitations should be noted. First, the AutoML search is computationally expensive (15 GPU-hours per task) and requires access to GPU hardware; organizations without GPU access may not be able to run the full AutoML pipeline. A cloud-based AutoML service or reduced-search protocols could address this. Second, the current AutoML framework optimizes for a fixed training dataset; when the training dataset changes significantly (new defect types, new product variants), the AutoML optimization should be re-run. Third, the architecture search space is defined by the designers and may miss operation types that are not included; a more flexible search space that includes novel operation types discovered during search would be more comprehensive.

---

6. Conclusion

This paper proposes the first AutoML framework for optical surface inspection, combining differentiable neural architecture search, Bayesian hyperparameter optimization, learned data augmentation policies, and hardware-aware joint optimization to automatically design optimized deep learning inspection models.

AutoML-designed networks outperform human-designed baselines across all three core inspection tasks: thermal reconstruction MAE improves by 12.7%, phase unwrapping RMSE improves by 12.5%, and defect detection mIoU improves by 3.2 percentage points. Each AutoML component independently contributes to the improvement, with architecture search providing the largest single contribution.

The total AutoML search cost is approximately 15 GPU-hours (24 hours on a single GPU), compared to 2–4 weeks of expert engineering effort for manual design. The framework requires no ML expertise and produces models that meet production throughput requirements while achieving higher accuracy than human-designed baselines.

The proposed AutoML framework provides a practical pathway to high-performance optical inspection models for any manufacturing organization, democratizing access to state-of-the-art deep learning technology.

References

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). AutoAugment: Learning augmentation policies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 113–123). IEEE. https://doi.org/10.1109/CVPR.2019.00020

Huang, H., Tang, J., Liu, T., & Huang, M. (2026). Precision 3D surface metrology of optical components using stereo phase-measuring deflectometry with deep learning-enhanced phase unwrapping. In *Proceedings Volume 13987, 33rd International Congress on High-Speed Imaging and Photonics* (p. 1398704). SPIE. https://doi.org/10.1117/12.3093993

Huang, H., Yang, Y., & Zhu, Y. (2023). Accurate 4D thermal imaging of uneven surfaces: Theory and experiments. *International Journal of Heat and Mass Transfer*, 216, 124580. https://doi.org/10.1016/j.ijheatmasstransfer.2023.124580

Liu, H., Simonyan, K., & Yang, Y. (2019). DARTS: Differentiable architecture search. In *International Conference on Learning Representations*. arXiv. https://arxiv.org/abs/1806.09055

Real, E., Aggarwal, A., Huang, Y., & Le, Q. V. (2019). Regularized evolution for image classifier architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence* (pp. 4780–4789). AAAI. https://doi.org/10.1609/aaai.v33i01.33014780

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems 25* (pp. 2951–2959). Curran Associates.

Malema. (2026a). Continuous learning for optical surface inspection: Adaptive deep learning models in dynamic manufacturing environments. *Inclusive Growth and Governance Quarterly*, *2*(1).

Malema. (2026b). Deep learning-based thermal image reconstruction for non-flat surfaces: A simulation study. *Inclusive Growth and Governance Quarterly*, *2*(1).

Malema. (2026c). Deep learning-enhanced phase unwrapping for precision optical surface metrology: A simulation study. *Inclusive Growth and Governance Quarterly*, *2*(1).

Malema. (2026d). Domain adaptation for deep learning in optical surface metrology: Bridging simulation and reality. *Inclusive Growth and Governance Quarterly*, *2*(1).

Malema. (2026e). Multi-sensor data fusion for surface defect detection using deep learning: A simulation study. *Inclusive Growth and Governance Quarterly*, *2*(1).

Malema. (2026f). Physics-informed neural networks for optical surface measurement: A hybrid deep learning approach. *Inclusive Growth and Governance Quarterly*, *2*(1).

Malema. (2026g). Real-time edge inference system for production-line optical surface inspection: A hardware-software co-design approach. *Inclusive Growth and Governance Quarterly*, *2*(1).

Malema. (2026h). Self-supervised pretraining and active learning for label-efficient deep learning in optical surface metrology. *Inclusive Growth and Governance Quarterly*, *2*(1).

Malema. (2026i). Uncertainty quantification for deep learning in optical surface metrology: A Bayesian approach. *Inclusive Growth and Governance Quarterly*, *2*(1).

Malema. (2026j). Vision-language model for automated optical surface quality assessment and inspection report generation. *Inclusive Growth and Governance Quarterly*, *2*(1).