



Vrije Universiteit Amsterdam
Faculty of Science
Master of Science in Business Analytics

Proxy-constrained Bayesian Hyperparameter and Data Composition Optimisation for Automated Vehicle Damage Detection

Bodine Stubbé
2661872

Supervisors:
Prof. Dr. Sandjai Bhulai
Dr. Robin van Ruitenbeek

Master's Thesis

Amsterdam, July, 2026

Abstract

Hyperparameter optimisation is central to the performance of machine learning models, yet in production settings it is often skipped: full training runs are so slow that searching over configurations becomes infeasible. In parallel, the field of AI has shifted from model-centric towards data-centric AI, where performance is improved by curating the right data in the right proportions. This thesis addresses both developments in practice, at a company performing automated vehicle damage detection with YOLO models. It develops a compute-constrained proxy model, trained in roughly an eighth of the full model’s time by reducing the amount of data, architecture size, and image resolution, and uses it to optimise training hyperparameters and training-data composition. Configurations found on the proxy are transferred to the full-scale model, and the fidelity of that transfer is quantified through a purpose-built metric, the proxy calibration ratio.

Within a controlled AdamW regime, proxy-constrained hyperparameter optimisation improved full-scale test performance over untuned defaults in all three seeds. Optimising training-data composition did not improve over either baseline; this negative result is fully diagnosed rather than left opaque, while the comparison of transfer fidelity between the two parameter spaces was inconclusive at three seeds. Transfer was class-asymmetric throughout: the majority class (scratch) transferred faithfully in both studies, while the minority class (dent) transferred weakly or, in the composition study, reversed sign at full scale. The calibration ratio proved informative per class but degenerate at the aggregate level, where a near-zero denominator makes it explode. A recurring finding is that the framework’s reliability depends on the alignment of the evaluation sets: where validation and test diverged, as here, firm conclusions were limited. The practical value of the approach therefore remains to be confirmed against the company’s production model, which uses a different optimiser.

The thesis makes three contributions: empirical evidence that proxy-constrained hyperparameter optimisation can improve full-scale test performance over untuned defaults, with class-asymmetric transfer; a fully diagnosed negative result for data-composition optimisation, decomposing the outcome into the costs of the resampling mechanism, an aggregate objective blind to per-class trades, and an insufficient search budget; and the introduction of the proxy calibration ratio as a transfer-fidelity diagnostic, including its instability near the noise floor.

Contents

Contents	iii
1 Introduction	1
1.1 Background	1
1.2 Research setting	2
1.3 Objective and research questions	2
1.4 Contributions	3
1.5 Thesis structure	3
2 Literature	4
2.1 Object Detection	4
2.1.1 Background	4
2.2 Hyperparameter Optimisation	6
2.2.1 Background	6
2.2.2 Problem Formulation	7
2.3 Methods for Hyperparameter Optimisation (HPO)	7
2.3.1 Bayesian Optimisation	8
2.3.2 Multi-Fidelity Optimisation and Bayesian Optimisation and Hyperband (BOHB)	9
2.4 Hyperparameter Optimisation (HPO) for Object Detection Models	10
2.5 Data Composition Optimisation	10
2.6 Constrained HPO	11
2.7 Research Gap	12
3 Data	13
3.1 Dataset Overview and Capture Setup	13
3.2 Label Architecture and Parsing	15
3.3 Class Distribution	16
3.3.1 Sub-type composition of the training set	16
3.3.2 Cross-Split Comparison	17
3.4 Pre-Experimental Data Preparation	18
4 Methodology	20
4.1 Experimental Framework	20
4.2 Evaluation Metrics	21
4.2.1 mAP@50	21
4.2.2 mAP@50:95	21
4.2.3 Per-Class Average Precision (AP)	21
4.2.4 Proxy Calibration Ratio	21
4.3 Full Scale Retrain Model	22
4.4 Proxy Regime Design	24
4.4.1 Proxy Configuration	24

4.4.2	Remaining Configuration Settings	25
4.4.3	Proxy Transfer	27
4.5	Summary: Proxy and Retrain Comparison	28
4.6	Experimental Study 1 (E1): Proxy-Guided HPO	28
4.6.1	Experimental Design	28
4.6.2	Search Space	31
4.7	Experimental Study 2 (E2): Data Composition Optimisation	32
4.7.1	Motivation	32
4.7.2	Stratification Design	33
4.7.3	Sampling Mechanism	35
4.7.4	Search Space	36
4.7.5	Experimental Design	37
5	Results	40
5.1	Experimental Study 1 (E1)	40
5.1.1	Effect of proxy-guided HPO on full-scale performance (SQ1A)	40
5.1.2	Effect of HPO algorithm choice on proxy-to-full transfer fidelity (SQ1B)	45
5.2	Experimental Study 2 (E2)	48
5.2.1	Performance Improvement with Data Composition Optimisation (SQ2A)	48
5.2.2	Proxy-to-Full Transfer Fidelity for Data Optimisation and Training Hyper-parameters (SQ2B)	55
6	Discussion	57
6.1	Interpretation of the findings and limitations	57
6.2	Practical Implications and Recommendations	59
6.3	Limitations and Future Research	61
7	Conclusion	63
	References	65
	Appendices	70
A	Appendix	70
A.1	Methodology	70
A.1.1	Proxy configuration selection experiments	70
A.1.2	Preliminary Experiments: Training Plot Proxy Model	71
A.1.3	Frame audit	72
A.1.4	Strata counts on the full training set	72
A.2	Results	73
A.2.1	Experimental Study 1 performance metrics across both evaluation sets	73
A.2.2	Experimental Study 1 validation $\Delta\text{mAP@50}$	74
A.2.3	Experimental Study 1 full ρ results per seed	75
A.2.4	Experimental Study 2 performance metrics across both evaluation sets	76
A.2.5	Experimental Study 1 validation $\Delta\text{mAP@50}$	78
A.2.6	Experimental Study 2 full ρ results per seed	79
A.2.7	ρ difference between Experimental Study 1 (E1) and Experimental Study 2 (E2)	80
A.2.8	Decomposition of costs between different conditions in E2	80

Acronyms

- ABOHB** Adjusted Bayesian Optimisation with Hyperband. 20, 21, 24, 28–30, 32, 40, 42, 44–47, 58, 59, 63
- AdamW** Adam with decoupled Weight decay. 23, 26, 28, 31
- AP** Average Precision. iii, 6, 21, 40, 42, 44, 57
- AutoML** Automated Machine Learning. 7
- BO** Bayesian Optimisation. 1, 7–10, 12, 20, 21, 24, 28–34, 36–40, 42, 44–48, 53–56, 58, 59
- BOHB** Bayesian Optimisation and Hyperband. iii, 1, 9, 10, 12, 20, 30
- CI** Confidence Interval. 53, 55, 56
- CNN** Convolutional Neural Network. 4, 8, 12, 32, 33, 59
- COCO** Common Objects in Context. 6, 21
- CUDA** Compute Unified Device Architecture. 31
- E1** Experimental Study 1. iv, 3, 20, 21, 26, 28, 32, 33, 35, 37, 39, 40, 48, 55, 57–61, 73–75, 78, 80
- E2** Experimental Study 2. iv, 3, 20, 21, 32, 33, 35, 37, 38, 48, 55, 57–61, 72, 76, 77, 79, 80
- EI** Expected Improvement. 8
- FN** False Negative. 5
- FP** False Positive. 5
- GP** Gaussian Process. 8, 9
- GPU** Graphics Processing Unit. 24, 25, 28, 30
- HPO** Hyperparameter Optimisation. iii, 1, 2, 6, 7, 9–12, 20, 22, 23, 25, 27–31, 40–46, 57
- IoU** Intersection over Union. 5, 6, 21, 40
- mAP** mean Average Precision. 6, 21
- ML** Machine Learning. 6, 7
- MLOps** Machine Learning Operations. 30
- MuSGD** Muon Stochastic Gradient Descent. 22, 23

NMS Non-Maximum Supression. 5

RFS Repeat Factor Sampling. 11, 32, 35, 36

ROI Region of Interest. 4, 13–19, 24, 25, 32–36, 38, 44, 58, 72

SD Standard Deviation. 48

SGD Stochastic Gradient Descent. 22

SH Successive Halving. 9, 10, 30

SQ1a Sub question 1a. 3

SQ1b Sub question 1b. 3

SQ1b Sub question 1b. 3, 33, 35, 59

SQ2a Sub question 2a. 3

TP True Positive. 5

TPE Tree-structured Parzen Estimator. 9, 10, 29, 30, 32, 33, 37, 57

VRAM Video Random Access Memory. 24, 30

YOLO You Only Look Once. 4–6, 9, 10, 23, 29, 31, 35

Introduction

1.1 Background

Hyperparameter Optimisation (HPO) is an important part of the production of machine learning models, often facilitating a significant improvement in performance over default configurations (A Ilembayo et al., 2024; Bergstra, Bardenet et al., 2011). However, the cost of performance improvement is twofold: knowing how to find optimal configurations and the time spent finding them.

One may tune a model manually, but this procedure requires domain knowledge and offers no guarantees of optimality (Bergstra, Bardenet et al., 2011). More popular automated methods are Grid Search and Random Search, which involve sampling across a grid or randomly from a range of values across various parameters (Snoek, Larochelle and R. Adams, 2012). Although these methods automate the search, they disregard previous evaluations when selecting new configurations.

In an attempt to increase efficiency and converge toward optimality, more sophisticated methods use the evaluation results of previous configurations to inform the selection of new ones. The most common among these is Bayesian Optimisation (BO), which focuses search on those regions of the parameter space believed to yield the most promising validation scores (Bergstra, Bardenet et al., 2011). BO, albeit more efficient than Grid or Random Search, can still be slow to converge and computationally expensive. In large-scale production systems, where a single training run can already consume significant compute and time, the cost of optimisation is therefore often foregone, as it does not justify the marginal performance benefit.

This illustrates a disparity that recurs across machine learning research: methods that demonstrate clear value in settings with abundant budget are rarely adopted in production environments where compute budgets are fixed. Continued research has responded to this along two complementary lines. The first makes the search itself cheaper by allocating budget adaptively across configurations, terminating unpromising trials before they consume a full training run; bandit-based multi-fidelity methods such as Hyperband, and its combination with BO in Bayesian Optimisation and Hyperband (BOHB), are the canonical instances (L. Li et al., 2018; Falkner, Klein and Hutter, 2018). The second makes each evaluation cheaper by conducting the search on a reduced subset of the training data using a smaller model (Swersky, Snoek and R. P. Adams, 2013). This proxy configuration reduces search cost substantially, on the assumption that configurations identified as promising under constrained conditions will transfer when the model is retrained at full scale. These two mechanisms are not mutually exclusive, and whether the more aggressive search machinery of the first line preserves that transfer once it is layered onto the proxy of the second is not established.

In parallel, a broader shift from model-centric to data-centric AI has emerged in recent years (Jakubik et al., 2024). This view emphasises that the composition of training data, not only the model architecture and training procedure, is a primary determinant of final performance. One aspect of this is data refinement: optimising which examples are included in training and in what proportions. It is known that class imbalances can affect model performance and that not all data points contribute equally to what a model learns.

Despite this, tuning data composition and training hyperparameters is typically treated independently in the literature (Zha et al., 2025). Furthermore, most HPO studies are conducted without accounting for the compute constraints typical of production environments, limiting their practical applicability (Bergstra, Bardenet et al., 2011; Bergstra, Yamins and Cox, 2013). This thesis addresses both dimensions within a single unified framework.

1.2 Research setting

The research is conducted at Lensor, a company specialising in automated vehicle inspection and car damage detection. Lensor’s main products are AI-powered inspection gates through which vehicles drive while cameras capture images that are processed by YOLO-based deep learning models to detect surface damage. A full training run of Lensor’s production model takes approximately twenty-eight hours, which has made hyperparameter optimisation prohibitively expensive and left training configurations at their defaults.

Lensor’s setting is a precise instantiation of the two sources of suboptimality described above. First, training hyperparameters are untuned due to model training times infeasible for a full hyperparameter search. Second, the training dataset exhibits a hidden sub-type imbalance that is invisible to the training objective. Therefore, this research seeks to investigate the possibility of optimising first the training hyperparameters and second the data composition using a multi-factor proxy model. The proxy model is defined by a reduced dataset, a smaller architecture, and a lower image resolution, to respect the time and compute constraints. The goal of the investigation is to determine whether such optimum configurations under the proxy model extrapolate to improved performance on a full-scale retrain model, compared to performance at default configurations. Additionally, the effect of algorithm choice on transfer for the training hyperparameters is investigated to acquire a better understanding of the proxy model’s sensitivity and to inform subsequent methodological choices for the optimisation of the data composition. The novelty of this research lies in the quantification of this transfer through a newly defined proxy calibration ratio, which compares the performance gain observed at full scale to the performance gain observed at proxy scale, where gain is defined as the performance difference between the tuned and untuned configuration.

1.3 Objective and research questions

To achieve the given objective, this thesis investigates the following research question:

To what extent do configurations optimised via a compute-constrained proxy model transfer to full-scale car damage detection models, and does optimising training data composition under the same proxy constraints provide more reliable transfer than optimising training hyperparameters?

To answer this question, the following sub-research questions (SQ) are addressed:

- 1a) Given a fixed compute budget, does proxy-guided HPO of training parameters improve full-scale model performance over untuned defaults?
- 1b) Does the choice of HPO algorithm affect how faithfully proxy-observed improvements transfer to full-scale training?
- 2a) Does optimising training data composition with respect to damage sub-types under the proxy framework improve detection performance over the baseline?
- 2b) Is the proxy calibration ratio for data-space parameters higher than for training hyperparameters?

These sub-questions are investigated through two experimental studies. E1 optimises the training hyperparameters, addressing Sub question 1a (SQ1a) and Sub question 1b (SQ1b); E2 optimises the training data composition, addressing Sub question 2a (SQ2a) and Sub question 1b (SQ1b).

1.4 Contributions

Through these two experimental studies, this thesis makes the following contributions:

First, it provides empirical evidence that proxy-constrained optimisation of training hyperparameters yields configurations that improve full-scale test performance over untuned defaults, with a class-asymmetric transfer profile in which the majority class transfers more faithfully than the minority class.

Second, it provides a fully diagnosed negative result for proxy-constrained optimisation of training data composition. Rather than a bare failure, the outcome is decomposed into a quantified resampling-mechanism cost, a search budget dominated by random startup, and a corner solution that concentrated on the already-dominant stratum.

Third, it introduces the *proxy calibration ratio*, a diagnostic for proxy-to-full transfer fidelity, and characterises its behaviour. In particular, it shows that the ratio can best be interpreted per class, because an aggregate ratio conceals class-asymmetric transfer, and that its aggregate form becomes unstable when the proxy gain approaches the run-to-run noise floor.

1.5 Thesis structure

The remainder of this thesis is structured as follows: Chapter 2 outlines the theoretical background on object detection and the target models, alongside a literature review that contextualises the research gaps. Chapter 3 examines the dataset and motivates the data composition optimisation task. Chapter 4 details the methodology and experimental setup for both studies, followed by the presentation of the results in Chapter 5. Finally, Chapter 6 interprets these findings in a broader discussion, and Chapter 7 concludes the thesis.

Literature

2.1 Object Detection

2.1.1 Background

Object detection is the task of simultaneously localising and classifying objects within images or video frames and is a core problem in computer vision. You Only Look Once (YOLO) models are state-of-the-art object detection models, first introduced by Redmon et al. (2016), and currently in use by Lensor for automated vehicle damage detection. YOLO models are Convolutional Neural Networks (CNNs) that simultaneously predict bounding boxes and their respective class probabilities in a single forward pass (Redmon et al., 2016). Ultralytics has developed a versatile ecosystem of YOLO variants, differing in speed, accuracy, and parameter count (Glenn and Qiu, 2026). Lensor specifically uses a YOLO26 model, released in early 2026 (Glenn and Qiu, 2026).

YOLO models have evolved considerably over successive iterations, with each addressing shortcomings of its predecessor. One development is particularly noteworthy is the shift to anchor-free models. Early YOLO models were anchor-based, relying on anchor boxes, reference rectangles of fixed size and aspect ratio tiled across a feature map, to perform detection. In effect, anchor boxes provide predefined templates against which objects are recognised, at the cost of a set of hyperparameters governing their configuration, which often need tuning for the right performance. YOLOX removed the need for anchor boxes, and thus for these hyperparameters (Ge et al., 2021). Of particular relevance to this work, anchor-free detection carries no size prior, and so generalises more readily to object-size distributions unlike the one the anchors would have been fitted to. YOLO26 is likewise anchor-free meaning it predicts box geometry directly from feature-map locations.

Fundamentals

A bounding box, sometimes called a Region of Interest (ROI), is a rectangular region that defines the spatial location and boundaries of an object within an image. Object detection models predict bounding boxes over regions of an image together with a *confidence score*, which reflects the model's joint estimate of whether an object is present and how likely it belongs to a particular class (Redmon et al., 2016). A predicted bounding box B_p is compared against the ground truth bounding box B_{gt} , which marks the actual object of interest. Figure 2.1 shows an example detection from Lensor's pipeline.

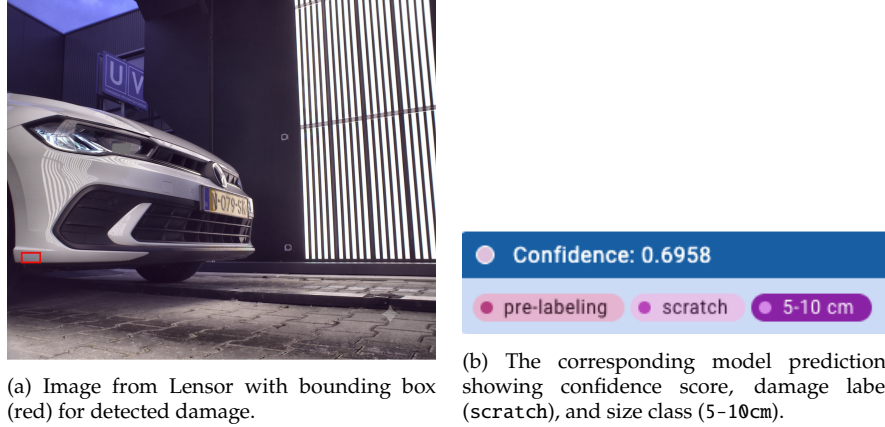


Figure 2.1: An example object detection output from Lensor's pipeline. The model predicts a bounding box localising the damage region, a class label, and a confidence score.

To determine whether a predicted bounding box is correct, the Intersection over Union (IoU) is used. IoU measures the spatial overlap between B_p and B_{gt} :

$$\text{IoU} = \frac{\text{area}(B_p \cap B_{gt})}{\text{area}(B_p \cup B_{gt})} \quad (2.1)$$

yielding a value between 0 (no overlap) and 1 (perfect alignment). Given an IoU threshold τ , predictions are then classified into three categories (Padilla, Netto and Da Silva, 2020):

- **True Positive (TP):** A detection whose IoU with a ground truth box meets or exceeds τ .
- **False Positive (FP):** A detection with no sufficiently overlapping ground truth box, either a spurious detection or a misplaced one.
- **False Negative (FN):** A ground truth box for which no detection was made.

Object detection models typically emit many candidate boxes, several of which localise the same object. In the conventional pipeline these duplicates are removed by Non-Maximum Suppression (NMS), which retains, per class, the highest-confidence box and suppresses any lower-confidence box whose IoU with it exceeds a fixed suppression threshold. This threshold is distinct from the evaluation threshold τ (Neubeck and Van Gool, 2006). YOLO26 instead adopts an end-to-end pipeline in which NMS is foregone, the model produces a single detection per object directly (Jocher et al., 2026). This end-to-end head is the default configuration and is the one used throughout this work.

Evaluation accordingly proceeds in two stages, or three where NMS is retained. First, predictions below a confidence threshold δ are discarded; the default value in recent YOLO models is 0.25 (Glenn and Qiu, 2026). Where NMS is used, the surviving predictions are then deduplicated. Finally, the remaining boxes are matched against the IoU threshold τ to assign TP, FP, or FN labels.

From these, Precision (P) and Recall (R) are defined as (Padilla, Netto and Da Silva, 2020):

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\text{all detections}} \quad (2.2)$$

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\text{all ground truths}} \quad (2.3)$$

Precision measures the fraction of detections that are correct, while Recall measures the fraction of ground truth objects that are found (R. Kaur and Singh, 2023). A good object detection system

maximises both: finding all objects while producing few spurious detections. However, Precision and Recall are both sensitive to the choice of confidence threshold δ ; raising δ increases Precision but reduces Recall, and vice versa. Reporting performance at a single threshold is therefore insufficient.

AP resolves this by integrating over all confidence thresholds. By varying δ across all possible values, one traces a precision-recall curve, and AP is defined as the area under this curve (Everingham et al., 2010):

$$AP = \int_0^1 p(r) dr \quad (2.4)$$

where $p(r)$ is precision at recall level r . AP is computed independently per class. Averaging over all N classes gives the mean Average Precision (mAP):

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_{C_i} \quad (2.5)$$

where AP_{C_i} is the average precision of class C_i . mAP is the standard summary metric for object detection (Everingham et al., 2010). In practice, computing the exact continuous integral is nearly infeasible. Therefore, modern benchmarks approximate this area. Following the introduction of the Common Objects in Context (COCO) dataset by Lin et al. (2015), AP is typically calculated as the mean of interpolated precision values, sampled at 101 evenly spaced recall levels. The mAP reported by Ultralytics for YOLO models relies on the COCO evaluation API, which strictly implements this 101-point interpolation.

The IoU threshold τ remains a second axis of evaluation. mAP@50 fixes $\tau = 0.50$, accepting any detection with at least 50% spatial overlap as correct (Lin et al., 2015). mAP@50:95 averages mAP over $\tau \in \{0.50, 0.55, \dots, 0.95\}$, penalising imprecise localisation more heavily. mAP@50:95 is therefore a stricter and more informative measure of production-level detection quality (Lin et al., 2015).

2.2 Hyperparameter Optimisation

2.2.1 Background

Developing effective Machine Learning (ML) models is a complex task. One aspect involves selecting the appropriate algorithm, while another concerns determining the optimal configuration for that algorithm.

This latter task is known as hyperparameter tuning. In contrast to parameter optimisation, which occurs as the model learns from its training data, hyperparameter tuning is performed before the final training process begins (Yang and Shami, 2020). Hyperparameters can dictate elements of the model's architecture, such as the number of nodes and layers in a neural network, or govern the training process itself. Examples of the latter include the learning rate, batch size, and number of training epochs.

Manual tuning refers to selecting these configurations by hand (Yang and Shami, 2020). While it remains one of the simplest and most traditional approaches, manual tuning quickly becomes ineffective as models grow in complexity, the number of parameters increases, and researchers must navigate non-linear hyperparameter interactions and time-consuming model evaluations (Yang and Shami, 2020).

The importance of identifying optimal hyperparameters cannot be overstated; several studies demonstrate that rigorous tuning significantly improves final model performance (Wojciuk et al., 2024; L. Liao et al., 2022).

Consequently, automating this search to systematically find the best-performing model configuration is defined as Hyperparameter Optimisation (HPO). This constitutes a central subdomain

of Automated Machine Learning (AutoML), which aims to automate the end-to-end development and implementation of ML models (Hutter, Kotthoff and Vanschoren, 2019).

2.2.2 Problem Formulation

The process of finding the optimal hyperparameters can be defined mathematically as follows:

Let $\mathcal{A}$ denote a machine learning algorithm with N hyperparameters, where the domain of the n -th hyperparameter is denoted Λ_n . The joint hyperparameter configuration space is then defined as

$$\Lambda = \Lambda_1 \times \Lambda_2 \times \cdots \times \Lambda_N$$

and a specific configuration is written as $\lambda \in \Lambda$.

Hyperparameter domains are heterogeneous by nature: a hyperparameter may be real-valued, integer-valued, binary, or categorical (Hutter, Kotthoff and Vanschoren, 2019). In practice, continuous and discrete domains are typically bounded, and the configuration space may exhibit conditionality, where a hyperparameter is only active given the value of another (Hutter, Kotthoff and Vanschoren, 2019).

Given a dataset $\mathcal{D}$ drawn from an underlying distribution $\mathcal{M}$, split into training and validation portions $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{val}}$, the objective of HPO is to find the configuration λ^* that minimises a validation loss $\mathcal{L}$:

$$\lambda^* = \arg \min_{\lambda \in \Lambda} \mathcal{L}(\mathcal{A}_\lambda, \mathcal{D}_{\text{train}}, \mathcal{D}_{\text{val}}) \quad (2.6)$$

where $\mathcal{A}_\lambda$ denotes algorithm $\mathcal{A}$ instantiated with configuration λ , trained on $\mathcal{D}_{\text{train}}$ and evaluated on $\mathcal{D}_{\text{val}}$. Several properties make this a particularly challenging optimisation problem. The objective function $\mathcal{L}$ is a black-box: it yields no gradient information and its analytical form is unknown (Shahriari et al., 2016). It is expensive to evaluate, as each query requires a full model training run. It is noisy, since results vary across random seeds. And the configuration space Λ is typically non-convex and mixed-type, precluding the use of standard continuous optimisation methods (Yang and Shami, 2020). Given these constraints, a large number of typical optimisation algorithms, such as Gradient Descent or Newton’s method, are not applicable to the task of HPO.

2.3 Methods for HPO

Several hyperparameter optimisation algorithms exist, varying in complexity and subsequent performance.

The simplest model-free methods are Grid Search and Random Search. In Grid Search, a user specifies a finite set of values for each hyperparameter, and the algorithm evaluates the Cartesian product of these sets (Hutter, Kotthoff and Vanschoren, 2019). However, the curse of dimensionality manifests quickly, as the number of function evaluations grows exponentially with the dimensionality of the hyperparameter space. Consequently, significant compute and time are spent evaluating unimportant configurations, limiting the method’s effectiveness in large hyperparameter spaces.

In Random Search, random configurations are sampled from the hyperparameter configuration space until a specified search budget is exhausted (Hutter, Kotthoff and Vanschoren, 2019). Random Search often outperforms Grid Search when certain hyperparameters are more important than others, a property that holds true in most practical scenarios (Bergstra and Bengio, 2012). Furthermore, Random Search is typically used as a baseline in comparative hyperparameter optimisation studies because it can achieve performance close to the optimum if given enough resources (Bergstra, Bardenet et al., 2011; Bergstra, Yamins and Cox, 2013; Hutter, Kotthoff and Vanschoren, 2019).

In contrast to Grid Search and Random Search, guided search methods offer a more intelligent approach by utilising the results of previous function evaluations to inform the selection of the next configuration. Among the most popular of these methods is Bayesian Optimisation.

2.3.1 Bayesian Optimisation

Bayesian Optimisation (BO) is a state-of-the-art framework for evaluating expensive black-box functions and has thus emerged as an effective tool for efficient hyperparameter optimisation (Bergstra, Bardenet et al., 2011; Snoek, Larochelle and R. Adams, 2012; Swersky, Snoek and R. P. Adams, 2013).

The theory and underlying algorithms were first introduced by Kushner (1964), Zhilinskas (1975), and Mockus (1975) and Mockus (1989), but gained significant traction after the work of Jones and Schonlau (1998) popularised them. However, BO was firmly established as a tool for hyperparameter tuning through the work of Snoek, Larochelle and R. Adams (2012). They demonstrated that BO could find better hyperparameters faster than a human expert (via manual tuning) for a CNN on the CIFAR-10 dataset, beating the state-of-the-art by over 3%.

Many papers have since emerged that utilise BO for hyperparameter tuning. For example, some explore multi-task optimisation (Swersky, Snoek and R. P. Adams, 2013), while others compare its performance against alternative methods like Random Search across different settings (Bergstra, Bardenet et al., 2011; Bergstra, Yamins and Cox, 2013). Notably, Bergstra, Bardenet et al. (2011) found that BO outperforms Random Search in deep belief networks, which are a specific class of deep neural networks. Furthermore, Bergstra, Yamins and Cox (2013) indicate that BO identified the best-performing hyperparameter configurations on the CIFAR-10 dataset compared to Random Search, achieving this in a fraction of the time for a single-layer network.

Algorithm BO is an iterative algorithm consisting of two primary components: a probabilistic surrogate model to approximate the black-box function to be optimised ($\mathcal{L}$) and an acquisition function. The acquisition function is responsible for determining the next sampling point (Shahriari et al., 2016).

At each iteration, the surrogate model is fitted to all observations of the target function collected so far. The acquisition function then uses the predictive distribution of this probabilistic model to determine the utility of candidate configurations, balancing the exploration of unvisited regions against the exploitation of promising ones. Since evaluating the acquisition function is computationally cheap relative to querying the black-box objective, it can be optimised thoroughly.

Acquisition Functions Among the acquisition functions proposed in the literature, Expected Improvement (EI) (Jones and Schonlau, 1998) is a common choice:

$$E[I(\lambda)] = E[\max(f_{\min} - y, 0)] \quad (2.7)$$

This function admits a closed-form expression when the model prediction y at configuration λ follows a normal distribution:

$$E[I(\lambda)] = (f_{\min} - \mu(\lambda)) \Phi\left(\frac{f_{\min} - \mu(\lambda)}{\sigma}\right) + \sigma \phi\left(\frac{f_{\min} - \mu(\lambda)}{\sigma}\right) \quad (2.8)$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ denote the standard normal density and cumulative distribution functions respectively, and $f_{\min}$ is the best function value observed so far.

Surrogate models Gaussian Processes (GPs) are the most commonly employed surrogate models for the black-box optimisation function (Kushner, 1964; Mockus, 1975; Snoek, Larochelle and R. Adams, 2012). Formally, a GP is fully specified by a mean function $m(\lambda)$ and a covariance function $k(\lambda, \lambda')$:

$$f(\lambda) \sim \mathcal{GP}(m(\lambda), k(\lambda, \lambda')) \quad (2.9)$$

Given a set of observations $\mathcal{D}_{1:t} = \{(\lambda_i, y_i)\}_{i=1}^t$, the GP posterior yields a predictive distribution over the objective value at any unseen configuration λ , with predictive mean $\mu(\lambda)$ and variance

$\sigma^2(\lambda)$ available in closed form. It is precisely this predictive distribution that the acquisition function consumes to determine the next configuration to evaluate, as described above.

The appeal of the GP as a surrogate lies in its ability to quantify uncertainty: regions of the configuration space that have been densely sampled carry low predictive variance, while unvisited regions carry high variance. This uncertainty estimate is what enables the principled exploration and exploitation trade-off of BO.

An alternative surrogate model is the Tree-structured Parzen Estimator (TPE), introduced by Bergstra, Bardenet et al. (2011). Rather than placing a GP prior directly over the objective function $\mathcal{L}$, TPE models the relationship between configurations and performance indirectly by fitting two density estimators. Concretely, the observed configurations are partitioned into a set of well-performing configurations $\ell(\lambda)$ and a set of poorly-performing configurations $g(\lambda)$, according to a quantile threshold γ on the observed objective values:

$$p(\lambda | y) = \begin{cases} \ell(\lambda) & \text{if } y < \gamma \\ g(\lambda) & \text{if } y \geq \gamma \end{cases} \quad (2.10)$$

The acquisition function is then approximated by the ratio $\ell(\lambda)/g(\lambda)$, such that configurations are proposed from regions where good performance is dense and poor performance is sparse. Each density is estimated using a tree-structured kernel, which naturally handles the mixed-type and conditional hyperparameter spaces common in practice.

A key advantage of TPE over GP-based surrogates is its scalability to high-dimensional configuration spaces, provided there is sufficient search budget. GP inference scales cubically with the number of observations and degrades in high dimensions, whereas TPE operates on each hyperparameter dimension independently, making it considerably more tractable as the search space grows (Hutter, Kotthoff and Vanschoren, 2019). Bergstra, Yamins and Cox (2013) exploit this property explicitly, demonstrating that TPE scales effectively to hundreds of hyperparameter dimensions in vision architectures. This makes TPE a particularly suitable surrogate for computer vision models such as YOLO, where the configuration space is high-dimensional and model evaluations are expensive.

2.3.2 Multi-Fidelity Optimisation and BOHB

A fundamental limitation of standard BO in the HPO setting is that each function evaluation requires a full model training run, which quickly becomes prohibitive as model complexity and dataset size grow (Krizhevsky, Sutskever and Hinton, 2017). Multi-fidelity optimisation addresses this by evaluating configurations on cheap approximations of the true objective function first. The concept of multi-fidelity stems from the work by Petrak (2000). In “Fast Subsampling Performance Estimates for Classification Algorithm Selection” he noticed that using a subsample of the data is an effective way to choose an algorithm across different configurations. In recent research, examples of multi-fidelity methods include training on a data subset, with a reduced model, at lower image resolution, or for fewer epochs. In some approaches, the process then only promotes the most promising configurations to higher-fidelity evaluations (L. Li et al., 2018).

Successive Halving (SH) and Hyperband SH is one of the simplest multi-fidelity algorithm selection methods, developed by Jamieson and Talwalkar (2016). Given a budget B and n initial configurations, each configuration is evaluated for a minimum resource allocation r . The top $1/\eta$ fraction of configurations are then promoted to the next round with η times the resource, and the remainder are discarded. This continues until a single configuration remains. The procedure is efficient when many configurations can be ruled out cheaply, but introduces a sensitivity to the choice of n and η : a large n explores broadly but allocates little resource to each configuration initially, while a small n evaluates each configuration more thoroughly but at the cost of exploration.

Hyperband (L. Li et al., 2018) resolves this sensitivity by running multiple SH brackets in parallel, each with a different n/η trade-off spanning the spectrum from aggressive early elimina-

tion to conservative full evaluation. Configurations within each bracket are sampled uniformly at random. By treating HPO as a pure resource allocation problem rather than a modelling problem, Hyperband achieves strong empirical performance with minimal assumptions, and outperforms standard BO in wall-clock time on several benchmarks (L. Li et al., 2018). However, because new configurations are always sampled randomly, Hyperband does not incorporate any information about which regions of the configuration space have performed well, forgoing the surrogate-guided proposal that makes BO sample-efficient. As such, SH and Hyperband are more related to Random Search than BO with respect to sampling.

Bayesian Optimisation and Hyperband Bayesian Optimisation and Hyperband (BOHB), introduced by Falkner, Klein and Hutter (2018), combines the resource allocation strategy of Hyperband with the surrogate-guided proposal of BO. Specifically, BOHB replaces the random sampling of new configurations in Hyperband with a TPE-based density ratio model. At each iteration, the TPE model is fitted to all observations collected at the highest fidelity seen so far, and proposes the next configuration by maximising the $\ell(\lambda)/g(\lambda)$ ratio. The Hyperband bracketing and elimination schedule is retained unchanged. This yields an algorithm that simultaneously benefits from the broad exploration and early stopping of Hyperband and the informed proposal distribution of BO. Falkner, Klein and Hutter (2018) demonstrate that BOHB matches or outperforms both Hyperband and standard BO across a range of benchmarks, achieving strong performance at substantially reduced wall-clock cost.

2.4 Hyperparameter Optimisation (HPO) for Object Detection Models

Various works investigate the effect of hyperparameter optimisation of YOLO models, albeit in the context of different use cases. For example, in “Effectiveness of YOLO Architectures in Tree Detection”, Moraes et al. (2025) investigate the impact of hyperparameter optimisation, while comparing three different YOLO models, in the context of tree detection. Hoang, Lee and Park (2025) use BO to optimise the hyperparameters of various YOLO models for fire detection, while Pascual-González et al. (2025) optimise the hyperparameters of a YOLOv8 model in the setting of detection and assessment of invasive coronary angiography lesions. The hyperparameters to be optimised are commonly related to training and include: learning rate, momentum, batch size, weight decay, and (warm-up) epochs (Moraes et al., 2025; Hoang, Lee and Park, 2025; Pascual-González et al., 2025). In all cases, optimisation of hyperparameters beats default configuration performance, thereby highlighting the need for careful hyperparameter selection for YOLO models.

2.5 Data Composition Optimisation

As highlighted by the research on hyperparameter optimisation (for YOLO models) described above, most of the literature focuses on the optimisation of the training procedure for a given dataset. This is in line with the belief that model optimisation is most important to producing better AI systems; the cornerstone of model-centric AI. However, recent developments highlight the importance of the selection and curation of data, birthing a new paradigm called data-centric AI (Jakubik et al., 2024). The focus is not on accumulating more data, but rather on more appropriate data (Jakubik et al., 2024).

In “A Survey of Dataset Refinement for Problems in Computer Vision Datasets”, Wan et al. (2024) highlight class imbalance as one of the most common problems in the computer vision domain, also known as the *imbalanced class learning* problem. Class imbalance refers to an uneven distribution of training samples across classes, which can cause models to favour the over-represented class and degrade performance on minority classes (H. Kaur, Pannu and Malhi, 2020). Class imbalance may occur at the class prediction level or within classes. For the latter,

most of the literature investigates the imbalance between easy and hard samples within a class (Xu et al., 2025).

Addressing class imbalance is frequently done through sampling-based or loss-based approaches (Wan et al., 2024). Sampling-based approaches modify the training data directly by changing which examples the model trains on, thereby altering the data distribution. This includes oversampling the minority class, undersampling the majority class, or a combination of both (Wan et al., 2024; Buda, Maki and Mazurowski, 2018). However, in object detection, there is a many-to-many relationship between images and classes, with one image possibly containing multiple classes. Therefore, naive oversampling per class causes other classes to be oversampled as well. As a result, Repeat Factor Sampling (RFS), introduced by Gupta, Dollar and Girshick (2019), is a specific type of oversampling that tries to remediate this issue. RFS is an image-level oversampling method that aims to increase the frequency of underrepresented classes by specifically repeating the images that contain them during training (Gupta, Dollar and Girshick, 2019). The method is governed by two structural design choices: a category-level repeat factor r_c , defined as:

$$r_c = \max(1, \sqrt{t/f_c}) \quad (2.11)$$

and an image-level repeat factor:

$$r_i = \max_{c \in i} r_c \quad (2.12)$$

where f_c is the fraction of training images that contain at least one instance of class c and t is a hyperparameter that controls the threshold at which oversampling begins. Practically, if $f_c \geq t$, then there is no oversampling for class c . Then, during training, each image is repeated according to its repeat factor r_i . The use of a maximisation operator in Equation 2.12 entails that when an image contains multiple classes, it inherits the most aggressive single class target.

Loss-based approaches, sometimes termed cost-sensitive learning (Wan et al., 2024), leave the training data unchanged but modify the loss function to change how the model learns from the data it trains on. This can be done through simple heuristics such as inverse class frequency weighting (Huang et al., 2016) or more principled meta-learning formulations. Among the latter, Ren et al. (2018) propose a meta-learning algorithm that casts data weighting as a bilevel optimisation problem: the outer loop assigns a continuous weight to each training sample and updates those weights by minimising loss on a small, clean validation set; the inner loop trains the model under the current weights. This establishes a key principle: that a held-out validation signal can guide the assignment of importance to training data.

Two limitations persist across these approaches. First, both sampling and loss-based approaches are typically operationalised at the level of the class label, or at the level of the individual sample (Ren et al., 2018). Neither granularity targets imbalance defined along structural axis definitions of the data, such as visually distinct sub-types (within a class). Second, when a validation signal is used to direct data composition, as in Ren et al. (2018), weights are updated at every gradient step through nested optimisation, binding the search inseparably to the training loop. No transferable composition is produced that exists independently of the training run; the optimised weights are a by-product of a completed trajectory, and cannot be evaluated at a reduced scale, deployed, or re-used independently of the run that produced them.

A related but distinct question concerns whether data composition can be optimised prior to training, in a manner that is driven by a validation signal, operates at a granularity defined by structural sub-types of the data, and remains compatible with the cost constraints that make HPO feasible in production settings. This question sits at the intersection of data-centric AI and constrained HPO, and is taken up in the following section.

2.6 Constrained HPO

Lastly, we have already highlighted that the development of multi-fidelity methods attempts to address the time- and compute-intensive nature of standard hyperparameter optimisation methods. Essentially, in such a setting, a search is conducted on a cheaper approximation of the

full training configuration to compare the performance of various hyperparameter configurations before committing to expensive full-scale training.

In some studies, the transfer of best-found configurations on a subset of the data to the final model is naturally assumed. For example, Wahyudi, Soesanti and Nugroho (2023) perform BO of hyperparameters on just 25% of the training data, while testing the final results of the best-found configurations on the test data. Other research specifically investigates the effect of searches conducted under such proxy constraints. For example, Wojciuk et al. (2024) conduct a sub-study investigating the performance difference when BO HPO is performed on a subset of the training set as opposed to the full training set in classification CNNs. Interestingly, this study reports higher accuracy on the test set when optimised on a data subset, and higher still when that subset was also reweighted (Wojciuk et al., 2024), suggesting that proxy-constrained HPO can be not only more efficient but potentially beneficial. However, this result is reported as an empirical observation rather than a formally characterised finding: the degree to which proxy-observed improvement predicts full-scale improvement is not quantified, and it remains unclear whether this transfer holds consistently across different hyperparameter types, model scales, or dataset sizes. Furthermore, all existing proxy HPO work, including Wojciuk et al. (2024), restricts the search space to training procedure hyperparameters. Whether data composition parameters may be similarly optimised under proxy constraints, and whether their effects extrapolate more or less faithfully than training procedure parameters, has not been studied.

2.7 Research Gap

This chapter has reviewed the literature across four interconnected areas: hyperparameter optimisation methods, HPO for object detection models, data-centric AI and dataset refinement, and constrained proxy HPO. Across these areas, four interconnected gaps emerge that jointly motivate this research.

First, the transfer fidelity of proxy HPO has not been formally quantified. Existing proxy HPO studies also typically employ one approximation technique like reducing data, whereas a production-realistic proxy combines reduced data, a smaller architecture, and lower resolution, the combined transfer behaviour of which remains uncharacterised.

Second, comparisons between HPO algorithms under proxy constraints have focused on raw performance and wall-clock efficiency, not transfer fidelity. Whether BOHB’s broader exploration produces configurations that transfer more faithfully than standard BO, a question its multi-fidelity construction makes particularly natural in the proxy setting, has not been studied.

Third, data composition optimisation has not been framed at the sub-type level within a proxy HPO pipeline. Existing approaches operate either at the individual sample level or at the class level. Methods using a bilevel optimisation loop with a validation signal are coupled to full-scale training and therefore infeasible under compute constraints. No work optimises sampling weights over sub-types defined by structural axes of the data distribution, such as the damage type and size class sub-types documented in Chapter 3.

Fourth, the comparative transfer fidelity of data composition parameters and training hyperparameters has not been studied. All existing proxy HPO work restricts the search space to training procedure parameters.

These gaps motivate the four sub-questions of this thesis. The first two addresses Gaps 1 and 2 by examining whether proxy-guided HPO over a multi-factor proxy improves on an untuned production baseline in automated vehicle damage detection, and whether the choice of HPO algorithm affects transfer fidelity, focusing on BO and an adjusted version of BOHB. The last two address Gaps 3 and 4 by examining whether sub-type sampling weights improve detection performance and whether they transfer more faithfully than training hyperparameters.

Data

This section provides an overview of the data used in the research and highlights noticeable areas of interest that motivate the subsequent methodology.

3.1 Dataset Overview and Capture Setup

Lensor utilises two gates to capture pictures across the entirety of cars driving through them. The most recent gate, Umbra 3A, is covered and features eight cameras positioned at various angles. Images are captured at a 2448×2048 pixel (px) resolution, and the total number of images taken depends on the size of the car.

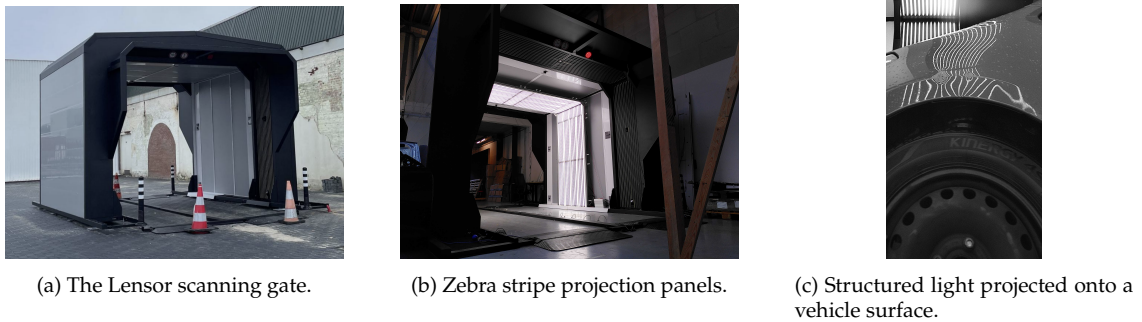


Figure 3.1: The Lensor structured light capture setup.

The gate features a mixture of diffused light and thin panels (see Figure 3.1b) that project a structured light pattern onto the car. Deviated kinks in the pattern clearly highlight dents (see Figure 3.1c), thereby assisting in damage detection.

The pictures are taken from different angles and stored on an Amazon cloud platform. All images collected for damage detection are saved to iteratively bulk up the current datasets. Subsequently, the AI models are constantly trained and updated.

All collected data, being a product of the production setting and thus receiving predictions, is annotated (*prelabelled*). A team of moderators reviews the annotated data to add, confirm, or remove the predicted labels for the detected ROIs. If bounding boxes are placed at an incorrect position, the moderators adjust them appropriately. As a result, two main datasets emerge: a moderated dataset, where all images have been reviewed, and a dataset containing the complete scans. In effect, the moderated dataset is a subset of the complete scans dataset.

From these two datasets, a curated split was created to produce the training, validation, and test sets. The data is currently partitioned into a 90/6/4 split, respectively, which effectively serves as a 90/10 split between training and evaluation data. The test set is used to benchmark production performance and strictly contains data from the moderated dataset. Conversely, both the training and validation sets contain a mix of moderated and un-moderated full-scan data. Over time, the

training and (to a lesser extent) validation sets have grown as more data is collected through the portal, while the test set has remained frozen to ensure fair and consistent model comparison. This heavy allocation toward the training set is a deliberate choice, as complex tasks such as object detection require substantial volumes of data to train faithfully (Bishop, 2006).

The only constraint Lensor maintains when adding further data to the train, validation, and test sets is that a single licence plate can only appear in one split to prevent data leakage.

As such, the split between the train, validation, and test sets is random, bounded only by the licence plate constraint and the requirement that the test set exclusively contains frames from the moderated dataset.

Each image in the dataset is accompanied by extensive metadata, of which the most relevant fields used are shown in Table 3.1

Table 3.1: Hierarchical structure of the Lensor dataset metadata.

Level	Field	Description	Values / range
Scan	scan_id	Unique identifier grouping all frames from a single vehicle capture	UUID string
	dataset_split	Train, validation, or test assignment	{train, validation, test}
	is_negative	Whether the scan contains no damage instances	{True, False}
Frame	frame_id	Unique identifier for a single captured image	UUID string
	image_uri	File path to the image	string
	camera_id	Camera mounted on the pole	{1, 2}
	iteration_id	Spatial position around the vehicle	integer $\in \{0, 1, \dots, 35\}$
ROI	damage_type	Annotated damage category	{dent, dent_zone, scratch, scratch_zone}
	size_class	Annotated damage size bin	{0-2cm, 2-5cm, 5-10cm, 10-30cm, 30+cm, small, medium, large}
	x_c, y_c	Bounding box centre coordinates in pixels	positive integer (px)
	w, h	Bounding box width and height in pixels	positive integer (px)

3.2 Label Architecture and Parsing



(a) Example of a single dent: the depression bends the structured light pattern, creating a visible circular distortion.



(b) Example of dent zone: presence of multiple dent circles in close proximity.



(c) Example of a single scratch.



(d) Example of scratch zone.

Figure 3.2: Visual examples of the four types of damage.

Label Architecture Four main labels are used in the annotation of the ROIs in the data: scratch zone, scratch, dent zone, and dent (see Figure 3.2 for examples). However, before training, scratch zone is mapped to scratch, and dent zone to dent. Final predictions are scoped to either dents or scratches. For most ROIs, a size class is available, of which the core categories are: 0-2 cm, 2-5 cm, 5-10 cm, 10-30 cm, and 30+ cm. Some ROIs have a qualitative size label (small, medium, large) as a result of an annotators' experiment. However, that labelling has now been discontinued.

Parsing Upon parsing the images, all ROIs were extracted and collected in a table. Most importantly, for each subset (train, validation, and test), duplicate frames and ROIs were removed. These duplicates are present due to the fact that some damages are visible across multiple images, and some frames that are in the complete scans dataset are also present in the moderated dataset.

3.3 Class Distribution

3.3.1 Sub-type composition of the training set

Table 3.2 shows the most important findings regarding the class distribution of damage types and size classes for the ROIs in the training dataset.

Table 3.2: Cross-tabulation of the number of ROIs per damage sub-type and size class in the training set.

Size Class	Dent	Dent Zone	Scratch	Scratch Zone	Total
0–2 cm	1,438	1	691	59	2,189
2–5 cm	4,916	0	2,258	640	7,814
5–10 cm	1,664	62	2,288	1,156	5,170
10–30 cm	601	91	2,194	1,859	4,745
30+ cm	203	155	610	1,140	2,108
Small	264	6	435	87	792
Medium	55	16	148	126	345
Large	20	5	19	233	277
Total	9,161	336	8,643	5,300	23,440

The training set comprises 18,841 frames with 23,440 annotated ROIs across the four damage sub-type labels, as shown in Table 3.2. A total of 1,414 ROIs carry qualitative size labels (small, medium, large), amounting to 6.0% of all training ROIs. These represent an experimental labelling convention that is currently discontinued. Of the 18,841 training frames, 4,739 (25.2%) are negative frames containing no damage annotations.

At the top training label level, the ratio between the scratch and dent classes is mild at 1.47:1 (Krawczyk, 2016). However, when considering class imbalance across damage type and size class jointly, a considerably greater imbalance emerges. The most common cell, dent 2-5 cm, contains 4,916 ROIs; the least common non-trivial cell, scratch zone 0-2 cm, contains 59 ROIs, yielding an 83:1 ratio. The dent zone 0–2 cm cell ($n = 1$) is excluded from this ratio as a singleton artefact reflecting the near-absence of dent zone annotations at the smallest size class rather than a meaningful audit cell. This sub-type imbalance is entirely invisible to the model, which is trained on binary scratch and dent labels.

3.3.2 Cross-Split Comparison

Table 3.3: Damage sub-type proportions across train, validation, and test splits.

Damage Sub-type	Train (%)	Val (%)	Test (%)
Dent	39.1	40.0	23.9
Dent Zone	1.4	0.0	0.0
Scratch	36.9	36.9	54.9
Scratch Zone	22.6	23.1	21.2
Total ROIs	23,440	2,778	1,752

Table 3.4: Cross-tabulation of damage sub-type and size class ROIs for the validation and test sets.

(a) Validation set					
Size Class	Dent	Dent Zone	Scratch	Scratch Zone	Total
0–2 cm	73	0	72	2	147
2–5 cm	377	0	433	11	821
5–10 cm	302	0	134	164	600
10–30 cm	227	0	370	242	839
30+ cm	132	0	17	222	371
Total	1,111	0	1,026	641	2,778

(b) Test set					
Size Class	Dent	Dent Zone	Scratch	Scratch Zone	Total
0–2 cm	30	0	78	0	108
2–5 cm	191	0	393	41	625
5–10 cm	132	0	278	61	471
10–30 cm	47	0	172	149	368
30+ cm	19	0	40	121	180
Total	419	0	961	372	1,752

The validation set comprises 1,253 frames with 2,778 ROIs and 67 negative frames (5.3%). The test set comprises 949 frames, 1,752 ROIs, and 179 negative frames (18.9%).

Comparing the train, validation, and test sets, several findings emerge. As shown in Table 3.3, dent zone is entirely absent from both validation and test sets. Therefore, per-class evaluation cannot distinguish it from dent regardless of the training-label treatment; combined with its low overall count (336 ROIs in train), this supports merging dent zone into the dent training label. Furthermore, qualitative size labels (small, medium, large) appear exclusively in train.

At the training label level, Table 3.5 shows that the validation set (1.50:1) is closely aligned with the train set (1.47:1), whereas the test set diverges substantially at 3.18:1. The test set is considerably more scratch-heavy than either the train or validation sets.

However, delving one level deeper, Table 3.4 displays the cross-tabulation of damage type and size for the validation and test sets. Comparing this to Table 3.2, one observes that the modal cell in the training set is dent at 2–5 cm, but for the validation and test sets, this is scratch at 2–5 cm. Furthermore, from Table 3.4, one observes that the internal size distribution of each damage class differs between the validation and test sets. For dent, we see that 33.9% of validation dent ROIs

Table 3.5: Class ratio at training label level across splits.

Training Label	Train		Validation		Test	
	Count	%	Count	%	Count	%
Dent (incl. dent zone)	9,497	40.5	1,111	40.0	419	23.9
Scratch (incl. scratch zone)	13,943	59.5	1,667	60.0	1,333	76.1
Scratch:dent ratio	1.47:1		1.50:1		3.18:1	

Table 3.6: Negative frame rate across splits.

Split	Total Frames	Negative Frames	Negative Rate (%)
Train	18,841	4,739	25.2
Validation	1,253	67	5.3
Test	949	179	18.9

fall in the 2–5 cm modal size and 20.4% in the 10–30 cm tail, against 45.6% and 11.2% respectively in the test set. Test dent is more concentrated in modal sizes, whereas validation dent is more size-diverse. Scratch and scratch zone exhibit smaller within-class compositional differences between the validation and test sets.

The negative frame rate comparison in Table 3.6 reveals a further discrepancy. The train (25.2%) and test (18.9%) sets are reasonably aligned, while the validation set (5.3%) is the outlier.

Taken together, the validation set deviates from the test set in negative frame rate, training-label class proportions, and the within-class size distribution of the dent class.

3.4 Pre-Experimental Data Preparation

Label Collapse Given the findings from the data audit, ROIs annotated with qualitative size labels (small, medium, large) were excluded from the training set. These account for 1,414 ROIs (6.0% of training ROIs), are absent from both the validation and test sets, and cannot be reliably mapped to the centimetre-based size classes used elsewhere in the dataset. Retaining them would introduce an unresolvable asymmetry across splits. Removing them leaves 528 frames without any remaining annotation. These frames are dropped rather than reclassified as negatives, since their emptiness is an artefact of the label exclusion rather than a genuine absence of damage. The training set is thereby reduced from 18,841 to 18,313 frames, with 22,026 ROIs remaining.

Validation Set Remediation Standard machine learning practice assumes that training, validation, and test sets are drawn from the same underlying distribution, as this is a prerequisite for the validation set to serve as a reliable proxy for generalisation performance (Bishop, 2006; James et al., 2013). The data audit revealed systematic divergence between the validation and test sets on two independent dimensions: negative frame rate and class proportions.

In Lensor’s context, the test set has been curated by hand to reflect the production environment and is therefore treated as the distributional reference point. In the context of hyperparameter optimisation, the validation set is used to guide the configuration search: candidate configurations are evaluated against validation performance, and the best configuration is selected on that basis. If the validation set does not reflect the production distribution, the search is guided toward configurations that perform well on an unrepresentative signal, undermining the validity of the optimisation process. It is therefore necessary to restructure the validation set to better reflect the defining properties of the test set.

Although the scratch:dent ratio differs between the validation set (1.50:1) and the test set (3.18:1), neither constitutes a severe class imbalance (Krawczyk, 2016). Resolving this would

require holding out a disproportionate share of scratch-dominant scans from the training set, depleting the sub-type diversity available for the data selection optimisation. The residual class ratio gap is acknowledged as a known limitation.

Furthermore, the negative frame rate discrepancy is a significant concern. The validation set at 5.3% negatives versus the test set at 18.9% means that false positive behaviour on empty frames is substantially underpenalised during optimisation. Nogueira-Rodríguez et al. (2023) specifically found that an increased inclusion of negatives improved precision at an equal recall rate in a medical object detection task, thereby emphasising the need for a similar negative sample rate between optimisation and evaluation conditions.

Therefore, the validation set remediation prioritises matching the test negative frame rate over strictly matching the class ratio.

Validation set remediation involved randomly sampling negative frames from the train set at the scan level to avoid leakage, adding those to the validation set, and removing them from the training set. In total, 194 frames were transferred. In the new validation set, there are 1,447 total frames, of which 261 are negative, making the negative frame percentage 18.0%, which is much closer to the test set.

Following the label collapse and the transfer of 194 negative frames to validation, the training set comprises 18,119 frames, of which 4,545 (25.1%) are negative, and 22,026 ROIs.

The training set retains a higher negative rate (25.1%) than either evaluation set. This is not remediated: the alignment that matters for the optimisation is between the validation signal that guides the search and the test set that defines production performance.

Methodology

This chapter outlines the overarching experimental framework and justifies subsequent choices in defining the research design.

4.1 Experimental Framework

The main purpose of this research is to investigate to what extent the results of an HPO process on a proxy model transfer to a full model with respect to performance improvement. A proxy model is defined as a model that emulates the full model with respect to configuration rankings, but is of smaller size, trains on a reduced data set, and uses a smaller image size, thereby reducing compute demands and improving the efficiency of the HPO process.

Two parameter spaces are investigated, each through its own experimental study:

- **Experimental Study 1 (E1):** proxy HPO transfer for *training hyperparameters*, comparing Bayesian Optimisation (BO) and an adjusted version of Bayesian Optimisation and Hyperband (ABOHB) (see paragraph 4.6.1).
- **Experimental Study 2 (E2):** proxy HPO transfer for *training data composition*.

Both studies follow the same two-phase structure. In the search phase, a hyperparameter search is conducted on the proxy model, a cheaper approximation of the full training configuration. The search phase produces a candidate configuration that is then carried to the retrain phase, in which the best-found configuration is used to train both the proxy and the full-scale model.

Connecting the two studies is the proxy calibration ratio ρ , the diagnostic introduced in this thesis to quantify the transfer effect. It is computed identically for both studies, which is what enables the central comparison of whether data composition parameters extrapolate more or less faithfully than training procedure parameters under identical proxy conditions. The ratio is defined formally, together with its interpretation and the conditions under which it is interpretable, in Section 4.2.4.

4.2 Evaluation Metrics

All metrics used in this thesis are defined centrally in this section, as they are shared across E1 and E2.

4.2.1 mAP@50

mAP@50 is used as the optimisation signal for the BO and Adjusted Bayesian Optimisation with Hyperband (ABOHB) search during the proxy phase. That is, in Equation 2.6 the validation loss ($\mathcal{L}$) is substituted with mAP@50, making it a maximisation rather than a minimisation problem. At reduced data scale and image resolution, mAP@50:95 compresses configurations into a band where differences between configurations are indistinguishable from noise, as the stricter IoU thresholds penalise localisation imprecision that is a consequence of the proxy regime rather than a meaningful signal about configuration quality. mAP@50, with its single IoU threshold of 0.50, provides a coarser but more stable ranking signal under these conditions. mAP@50 is applied consistently across all proxy phase evaluations so that rankings remain comparable.

4.2.2 mAP@50:95

mAP@50:95 is adopted as the primary reported metric for all full retrain conditions on the test set, consistent with the COCO evaluation standard introduced by Lin et al. (2015). By averaging over IoU thresholds from 0.50 to 0.95 in steps of 0.05, it penalises localisation imprecision more harshly than mAP@50 and is therefore a more informative measure of fine-grained detection ability. All full retrain results are reported as mean $\pm$ standard deviation across three independent seeds. mAP@50 is additionally reported for all full retrain conditions to enable the computation of the calibration ratio in a consistent metric space, as described below.

4.2.3 Per-Class AP

Per-class AP for scratch and dent is reported separately for all full retrain conditions across both experimental studies. Aggregate mAP can mask class-specific effects: a sampling weight that improves dent AP at the expense of scratch AP would be invisible in the aggregate number, which is precisely the failure mode that the second experimental study investigates.

4.2.4 Proxy Calibration Ratio

The proxy calibration ratio ρ is the primary novel metric of this thesis. It quantifies the degree to which improvements found at proxy scale transfer to full production scale. It is defined as:

$$\rho = \frac{\Delta_{\text{full}}}{\Delta_{\text{proxy}}} \quad (4.1)$$

where Δ_{proxy} is the mAP@50 improvement of the best-found configuration over its baseline at proxy scale, and Δ_{full} is the mAP@50 improvement of that same configuration over the same baseline after full retrain. Both terms are computed in mAP@50 space so that the ratio is interpretable on a consistent scale.

Interpretation In effect, ρ expresses the full-scale effect as a proportion of the proxy effect. Its magnitude is the size of the full-scale effect per unit of proxy effect, and its sign records whether the full-scale effect moved with the proxy effect or against it. The metric is read on a continuum anchored at zero, with the regimes set out in Table 4.1. A positive ρ admits a proportional reading: the two scales agree on direction, and the magnitude measures how faithfully the proxy effect reproduced. A ρ near zero records no transfer, in the sense that the effect did not materialise at full scale. A negative ρ records a directional disagreement between the scales. Hence, its magnitude is not a fidelity measure, since there is no directional agreement whose proportionality could be

assessed; it establishes only that the proxy did not predict the direction of the full-scale effect. Negative ρ can arise from either term. A negative numerator is the substantive case, indicating a best-found proxy configuration that is no better than the baseline at full scale. A negative denominator is the converse and, for a searched condition, the less expected case: a best-found configuration that is worse than the baseline at proxy scale but nonetheless improves at full scale.

Table 4.1: Regimes of ρ . The magnitude is always the size of the full-scale effect per unit of proxy effect; the sign records directional agreement between the two scales.

Regime	Reading
$\rho \geq 1$	Faithful or understated transfer: the full-scale effect is at least as large as the proxy predicted.
$0 < \rho < 1$	Attenuated transfer: the effect is real and in the predicted direction, but the proxy overstated it.
$\rho \approx 0$	No transfer: the full-scale effect did not materialise. The sign of ρ in this regime is not interpretable.
$\rho < 0$	Reversal: the full-scale effect moved against the proxy.

Fidelity is not desirability ρ measures transfer fidelity, not whether an effect is wanted. A configuration whose proxy-scale *degradation* reproduces proportionally at full scale transfers faithfully, and ρ is correspondingly positive, even though the outcome is harmful: the proxy correctly quantified the damage the configuration would do. Whether an effect is beneficial is read from the sign of Δ_{full} .

Conditions for interpretability ρ is a ratio, and its behaviour is governed by its denominator. As Δ_{proxy} approaches zero, the ratio diverges: perturbations of the denominator well within the run-to-run variation of a single training run produce arbitrarily large changes in ρ . Interpretability degrades continuously as the proxy effect shrinks toward the noise floor. Where the proxy effect is small enough that the resulting ratio cannot be read as a measurement of transfer, this is stated explicitly.

4.3 Full Scale Retrain Model

The exploration of proxy models for HPO is driven by Lensor’s current computational bottleneck, wherein excessive training times render direct HPO unfeasible.

Resolving Lensor’s issue directly would require a proxy model that remains as close to the production setting as possible so that configurations can transfer directly. Optimiser choice, in particular, must match at proxy and full scale, given its dependent relationship with the learning rate, which is one of the parameters under investigation (see Section 4.6.2) (Johnson et al., 2024). To illustrate this, Yuan, Hu and Lu (2022) compare variants of Stochastic Gradient Descent (SGD) optimisers to Adam-based optimisers across various image classification and natural language processing tasks. They found that the optimal learning rate for their SGD optimisers is of a magnitude 10^2 to 10^3 larger than for the Adam-based optimisers.

However, Lensor’s production model uses the auto optimiser setting, which in YOLO26 defaults to Muon Stochastic Gradient Descent (MuSGD) (Glenn and Qiu, 2026). MuSGD applies orthogonalised gradient updates designed to stabilise optimisation at a large scale. This makes it well-suited to the production setting but inappropriate for the proxy regime, where reduced data volume and model size (see Section 4.4) produce noisier gradients that benefit from adaptive learning rates rather than orthogonalised SGD updates.

Adam-family optimisers are well-suited to noisy gradient regimes (Montavon, Orr and Müller, 2012), and Adam with decoupled Weight decay (AdamW) has performed particularly well for YOLO-family detection models (Syamsul and Wibowo, 2024; Bhosale and Angal, 2025). AdamW improves on vanilla Adam by correctly decoupling weight decay from the gradient update (Loshchilov and Hutter, 2019). Preliminary experiments confirmed that AdamW produced stable convergence at proxy scale (Appendix A.1.1, run 7), whereas MuSGD at the same batch size produced a near-zero mAP@50, which is insufficient for configuration ranking. Therefore, AdamW is significantly better suited for the proxy scale.

As a result, this research is conducted with a full-scale retrain model operating under an AdamW regime, which is strictly mirrored in the proxy model to ensure consistency for a fair calibration ratio.

Consequently, the full-scale retrain configuration serves as a controlled anchor against which proxy-derived HPO improvements can be measured. Optimiser aside, it is heavily inspired by Lensor’s production training pipeline in terms of dataset scale, image resolution, and batch size.

The full-scale retrain baseline does not reproduce the absolute performance of Lensor’s deployed production model (Appendix A.1.1, run 16). This is a deliberate scope decision: the research investigates proxy-to-full transfer under a consistent AdamW regime, evaluated on Lensor’s production test set. No claim is made in subsequent chapters that absolute performance values transfer directly to Lensor’s deployment. Rather, the investigation focuses on whether HPO under the proxy regime identifies configurations that improve full-scale performance under the same controlled optimiser. The full retrain configuration is detailed in Table 4.3.

4.4 Proxy Regime Design

4.4.1 Proxy Configuration

The goal of the proxy model is to rank hyperparameter configurations in the same relative order as the full model would, while requiring substantially less compute per evaluation. Three design requirements govern the proxy: it must produce a sufficiently stable mAP@50 signal to distinguish meaningfully between configurations; its training dynamics must be close enough to the full retrain regime for sensitivity rankings to transfer; and the combined proxy pipeline must be fast enough to make numerous evaluations in the BO and ABOHB algorithms tractable within the available compute budget.

Coleman et al. (2020) demonstrate that proxy models, produced through smaller architectures, reduced epoch counts, or layer removal, preserve configuration rankings that correlate strongly with those obtained under full training conditions. Absolute proxy performance is necessarily lower than full retrain performance, but this is inconsequential when the goal is relative ordering rather than absolute matching.

Three important levers govern the computational cost of a training run: model architecture, training data volume, and image resolution (Swersky, Snoek and R. P. Adams, 2013; Wahyudi, Soesanti and Nugroho, 2023; Coleman et al., 2020). The proxy configuration is therefore defined by reducing all three relative to the full retrain, with each reduction justified on both practical and methodological grounds below.

Model size The full-scale retrain model is yolo26m, with 20.4 million parameters (Glenn and Qiu, 2026). Two smaller variants are available: yolo26s at 9.5 million parameters and yolo26n at 2.4 million parameters.

The proxy architecture must share broadly similar training dynamics with the full-scale retrain model for hyperparameter sensitivity rankings to transfer reliably. yolo26n is excluded on this basis: its 2.4 million parameters represent an 8.5× reduction from yolo26m, spanning two architectural size classes with meaningfully different depth multipliers, width multipliers, and feature extraction stages. Parameters that appear insensitive on nano may matter substantially on medium, not because the dataset scale differs, but because the architectures process gradients differently. yolo26s, at a 2.2× reduction, remains closer to the medium model architecturally, thereby more closely sharing comparable feature extraction behaviour and making sensitivity transfer substantially more plausible. Furthermore, preliminary experiments (see Appendix A.1.1) confirm that yolo26s at the selected proxy configuration already trains in approximately two hours per run, providing more than sufficient compute reduction without the additional transfer risk of nano.

Image Resolution The actual images are captured at a 2448×2048 px resolution, resulting in an aspect ratio of approximately 1.2. The full retrain configuration trains at 1600×1600 px, following Lensor’s training pipeline. Higher resolutions are infeasible given current Graphics Processing Unit (GPU) Video Random Access Memory (VRAM) (see Section 4.6.1). Ultralytics resizes images following the aspect ratio such that the longest side equals 1600 px, while padding the shorter side to produce a 1600×1600 px input. Training on rectangular images was omitted since Ultralytics reports that it may affect detection performance (Glenn and Qiu, 2026).

Selecting an appropriate proxy resolution required resolving a domain-specific conflict between training efficiency and signal quality. Mosaic augmentation combines four images into a single training example, rendering each sub-image at approximately half the input resolution.

Damage ROIs in the Lensor dataset are small relative to the total frame area. At sufficiently low input resolutions, mosaic augmentation reduces ROIs to sizes so small that it degrades the training signal. Preliminary experiments confirmed this effect: enabling mosaic at 1024 px reduced the validation mAP@50 by 25% compared to training without mosaic at the same resolution (Appendix A.1.1, runs 3 and 6).

An image resolution of 1240 px was selected as the minimum at which mosaic augmentation remains viable for this domain. At this resolution, each mosaic sub-image is approximately 620 px effective resolution, providing meaningfully more spatial context per ROI than at 1024 px. Preliminary experiments at this configuration produced a mean validation mAP@50 of 0.123 across two independent runs (Appendix A.1.1, runs 7 and 8).

Proxy Data Subset Construction The proxy configuration uses a 25% subset of the training data to make HPO-scale experimentation computationally feasible. Swersky, Snoek and R. P. Adams (2013) demonstrate that a 25% data subset reduces optimisation time by 57% while preserving configuration rankings, providing empirical justification for this fraction.

Two constraints governed the subset construction. First, sampling must operate at the scan level rather than the frame level: frames within a scan capture the same physical car from correlated positions and are not independent observations. Sampling individual frames would create partial scans and inflate the apparent diversity of the proxy set. Second, the subset must preserve the relative distribution of damage sub-types present in the full training set, so that the optimisation signal observed at proxy scale reflects the same latent structure.

The second constraint is met by stratified sampling over the audit cells defined in Chapter 3, the cross-product of damage type and size class. Purely negative scans carry no damage annotation and are therefore treated as an additional sampling group, so that the negative frame rate is preserved. We sampled 25% of scans within each group, applying a floor of one scan per group to prevent rare sub-types from being absent from the proxy entirely.

The final proxy training set comprises 251 scans out of 979 (25.6%), with 4,991 frames, 6,490 ROIs, and 1,247 negative frames (25.0%), closely matching the train negative frame rate of 25.1%. Table 4.2 confirms that the proxy scan distribution closely mirrors the full training distribution, with all cell-level deviations remaining below 0.7 percentage points.

Table 4.2: Scan-level audit-cell distribution: train vs. proxy subset

Damage Type	Size (cm)	Train Scans	Train %	Proxy Scans	Proxy %	Diff
Dent	2–5	254	30.38	64	29.77	−0.62
Scratch	5–10	105	12.56	27	12.56	0.00
Scratch	2–5	103	12.32	26	12.09	−0.23
Scratch	10–30	102	12.20	26	12.09	−0.11
Dent	5–10	52	6.22	13	6.05	−0.17
Scratch Zone	10–30	47	5.62	12	5.58	−0.04
Scratch Zone	30+	42	5.02	11	5.12	+0.09
Dent	0–2	27	3.23	7	3.26	+0.03
Scratch	0–2	27	3.23	7	3.26	+0.03
Scratch Zone	5–10	27	3.23	7	3.26	+0.03
Dent	10–30	21	2.51	6	2.79	+0.28
Scratch	30+	16	1.91	4	1.86	−0.05
Scratch Zone	2–5	9	1.08	3	1.40	+0.32
Dent	30+	2	0.24	1	0.47	+0.23
Dent Zone	30+	2	0.24	1	0.47	+0.23
Negative	–	143	14.61	36	14.34	−0.27
Total		979	100.00	251	100.00	

4.4.2 Remaining Configuration Settings

Batch Size GPU memory constraints at 1600 px production resolution limit the hardware batch size to 4 for both yolo26m and yolo26s. Although larger batch sizes are feasible at proxy resolu-

ution (1240 px), the proxy configuration uses batch size 4 to match the full retrain setting. This consistency is methodologically necessary for two reasons.

First, batch size interacts with learning rate through the linear scaling rule: larger batches require proportionally larger learning rates to maintain equivalent gradient dynamics (S. Li et al., 2024). Since learning rate is the primary dimension under investigation in E1 (see Section 4.6.2), a batch size mismatch between the proxy and full retrain would introduce a systematic confound into the calibration ratio.

Augmentation decisions Mosaic augmentation, first introduced by Bochkovskiy, Wang and H.-Y. M. Liao (2020), combines four training images into a single composite example, increasing contextual diversity per gradient update and improving small object detection by exposing the model to a wider range of spatial arrangements. Multi-scale training complements this by randomly varying the input resolution within each batch, encouraging scale-invariant feature representations across the full range of damage sizes in the dataset. The number of degrees dictates the range to which images may be randomly rotated to improve recognition at various orientations. All settings are enabled consistently across proxy and full retrain phases, matching the original Lensor production configuration and ensuring that augmentation does not contribute as a confound to the proxy calibration ratio. The resolution constraint that mosaic imposes at proxy scale is addressed by the 1240 px resolution selection described above.

Learning rate, Weight Decay, Warmup Epochs The learning rate chosen for the proxy and full retrain configuration at baseline is 0.001, consistent with the most commonly chosen value for the AdamW optimiser (Bishop, 2006; Syamsul and Wibowo, 2024). For Weight Decay and Warmup Epochs, the default parameters present in the full-scale retrain model have been chosen, which align with the defaults of the YOLO26 models in the Ultralytics library (Glenn and Qiu, 2026).

4.4.3 Proxy Transfer

To ascertain the transfer validity potential of the proxy model’s best-found configurations, a preliminary experiment was conducted on the mAP@50 behaviour of the proxy and full retrain models on the validation set at different learning rates.

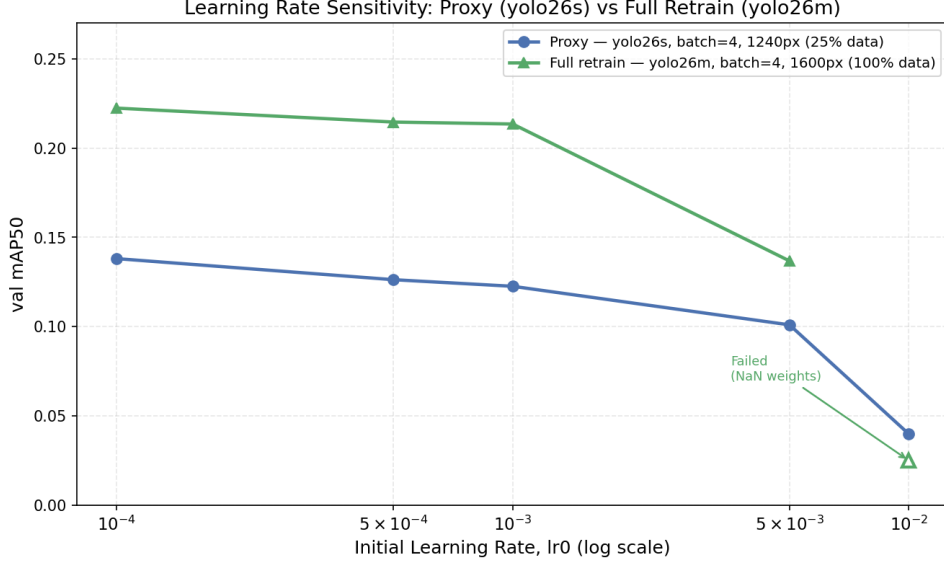


Figure 4.1: Learning rate sensitivity across proxy and full retrain configurations.

Figure 4.1 presents the validation mAP@50 response to learning rate across five values spanning the HPO search range, evaluated on the yolo26s proxy (batch=4) and the yolo26m full retrain. Three conclusions can be drawn. First, all configurations agree at the extremes: at $lr0 \geq 5 \times 10^{-3}$ performance degrades clearly across all curves, while performance peaks at the lowest tested value of $lr0 = 10^{-4}$, confirming the proxy correctly identifies which regions of the search space to favour and avoid. Second, the curves exhibit a broadly similar shape: a monotonic decline from $lr0 = 10^{-4}$ onwards, and are approximately parallel with a consistent vertical offset reflecting the absolute performance gap between proxy and full retrain scale. Third, the failure at $lr0 = 10^{-2}$ on the full retrain and poor performance at both $lr0 = 5 \times 10^{-3}$ and $lr0 = 10^{-2}$ on the proxy validate the upper bound of the search space at 5×10^{-3} . Notably, both proxy and full retrain peak at the leftmost tested value without sign of plateauing, suggesting the optimal learning rate may lie below 10^{-4} ; this motivates extending the lower bound of the search space to 10^{-5} . Overall, this qualitative agreement supports the use of the yolo26s proxy for identifying viable learning rate regions, and further supports the hypothesis that a proxy model may be useful for configuration rankings in HPO for a more complex production model.

4.5 Summary: Proxy and Retrain Comparison

Table 4.3 displays the final proxy model and compares it with the full retrain configuration designed to emulate the Lensor production model.

Table 4.3: Proxy and full retrain configurations compared.

Setting	Proxy	Full Retrain
Model variant	yolo26s	yolo26m
Parameters	9.5M	20.4M
Training data	25%	100%
Image size	1240px	1600px
Batch size	4	4
Optimiser	AdamW	AdamW
lr0	0.001	0.001
Momentum	0.937	0.937
Weight decay	0.0005	0.0005
Warmup epochs	3	3
Mosaic	1.0	1.0
Multi-scale	True	True
Degrees	5.0	5.0
Epochs	100	100
mAP@50 baseline	0.123 ^a	0.214 ^b
GPU hours	~2.17 hrs	~15.73 hrs

^a Mean of two independent runs (0.1226, 0.1229); run-to-run variation ± 0.0003 confirms configuration stability.

^b Single indicative run. Formal baseline mAP@50 is established across three seeds in the Experimental Study 1 design.

^{a, b} Performance is measured on the validation set.

4.6 Experimental Study 1 (E1): Proxy-Guided HPO

Experimental Study 1 (E1) evaluates whether proxy-guided Hyperparameter Optimisation (HPO) can improve over untuned defaults when transferred to the full retrain configuration, and analyses the difference between two HPO algorithms: Bayesian Optimisation (BO) and Adjusted Bayesian Optimisation with Hyperband (ABOHB). The central hypothesis is that ABOHB’s multi-fidelity mechanism allows it to explore more of the search space within the same cumulative compute budget as BO. Whether this broader exploration translates into a configuration that improves full-scale performance over the BO-best configuration, and whether the resulting calibration ratios differ between methods, are treated as open empirical questions. The direction of any calibration ratio difference is not assumed a priori: broader exploration could plausibly produce configurations that transfer more faithfully to full scale, or, conversely, configurations that exploit the proxy regime at the expense of full-scale generalisation.

4.6.1 Experimental Design

Table 4.4 outlines the experimental conditions for E1. For each algorithm used in the search phase, all trials across the three runs will be pooled, from which the best-of-three configuration will be selected for the retrain phase.

Table 4.4: Experimental Study 1 experimental conditions. All proxy runs use the proxy configuration defined in Section 4.4; all full retrains use the configuration in Section 4.3

Phase	Model	Condition	Runs	Notes
Search	Proxy	BO search	3	20 evaluations per run
Search	Proxy	ABOHB search	3	compute budget matched to BO
Train	Proxy	Baseline (untuned defaults)	3	3 independent seeds
Train	Proxy	BO-best of 3 configs	3	3 independent seeds
Train	Proxy	ABOHB-best of 3 configs	3	3 independent seeds
Train	Full	Baseline (untuned defaults)	3	3 independent seeds
Train	Full	BO-best of 3 configs	3	3 independent seeds
Train	Full	ABOHB-best of 3 configs	3	3 independent seeds

Variance Decomposition Within each search, the YOLO training seed is fixed at 0 so that TPE observes a deterministic objective surface; across the three independent searches, the TPE sampler seeds are varied (42, 43, 44) to capture search-trajectory variance. At the retrain phase, for both the proxy and full model, three seeds (42, 43, 44) drive Ultralytics’ single-seed mechanism, which jointly controls weight initialisation, data order, augmentation sampling, and gradient noise. This approach randomises multiple sources of variance, as recommended by Bouthillier et al. (2021). Seeds are paired across conditions (baseline, BO-best, ABOHB-best), enabling matched-seed analyses including the per-seed calibration ratio rather than a ratio-of-means.

For the calibration ratio ρ , specifically, it must be noted that the seed pairing does not remove the instability of the ratio noted in Section 4.2.4; it relocates it. Where a single seed’s Δ_{proxy} falls near zero, that seed’s ρ diverges, and because the across-seed summary is an unweighted mean, one such seed dominates the reported value. The per-seed ratios are therefore reported in full, and the across-seed mean is treated as a summary that is meaningful only where every contributing denominator does not converge to zero.

All retrain results are reported per seed and as mean $\pm$ sample standard deviation across the three runs. Given the small sample size, the calibration ratio is additionally reported with 95% bootstrap percentile confidence intervals (DiCiccio and Efron, 1996), and findings across seeds are framed in terms of effect size and directional consistency rather than null-hypothesis significance testing. The intervals will serve as a descriptive statistic as opposed to an inferential measure because of the limited inference they offer due to the small sample size.

Algorithm Comparison To enable a fair comparison between BO and ABOHB, both are matched on cumulative training compute via post-hoc analysis. BO is allocated 20 evaluations per search run; at 100 epochs per evaluation, this corresponds to a maximum theoretical budget of 2,000 training epochs. BO is run to completion across three seeds, and the average cumulative training time across child training tasks (excluding queue wait and pipeline overhead) defines the matched budget $T_{\text{train,BO}}$. Post-hoc matching rather than runtime enforcement via ClearML’s time limit parameter is used because that parameter is non-functional in the version of ClearML deployed at Lensor: a units mismatch in its budget-tracking logic prevents the auto-stop condition from ever firing within practical compute ranges. Post-hoc analysis is similarly employed by Falkner, Klein and Hutter (2018) in HPO benchmarking and yields a more rigorous comparison than runtime enforcement would, since it isolates algorithmic differences from orchestration variability such as queue contention, container setup, and concurrency effects. ABOHB is run with an optimisation time limit set to $1.1 \times T_{\text{train,BO}}$ in wall-clock minutes, providing a 10% margin to absorb empirically observed orchestration overhead, and the number of configuration evaluations is set high enough not to bind. Post-search, ABOHB trials are ordered by completion timestamp, and cumulative training time is computed sequentially; the largest k satisfying cumulative training time $\leq T_{\text{train,BO}}$ defines the matched-budget set, and the best-performing configuration among trials 1 to k is

selected as the ABOHB-best configuration.

A budget of 20 evaluations for BO is a deliberate design choice that reflects the compute-constrained production setting this research targets. At approximately 2.16 hours per proxy evaluation (Section 4.4), 20 evaluations amounts to roughly 43 GPU hours per search run, or approximately two days of continuous compute. This budget may not guarantee full convergence to the global optimum of the search space, nor does it aim to: the research question is explicitly about whether a practically affordable HPO search produces a configuration that improves over untuned defaults when transferred to full scale.

Implementation Both BO and ABOHB are implemented using the Optuna optimisation framework (Akiba et al., 2019), integrated with the ClearML Machine Learning Operations (MLOps) platform for experiment tracking and hyperparameter optimisation scheduling (ClearML 2024). The original BOHB algorithm of Falkner, Klein and Hutter (2018) is implemented in hpBandster, which presents installation challenges within Lensor’s containerised production environment due to dependency conflicts in its Pyro4 and ConfigSpace requirements. We therefore implement an adapted version of the algorithm in Optuna, following the practical implementation strategy of Nguyen and Liu (2025). This implementation captures BOHB’s Hyperband-style multi-fidelity pruning mechanism through Optuna’s HyperbandPruner, combined with a multivariate TPE sampler for model-driven configuration sampling. It does not include BOHB’s budget-conditional Kernel Density Estimation, in which separate density estimators are maintained per fidelity level rather than pooling observations across budgets; this is acknowledged as a limitation. Therefore, we refer to this specific method as Adjusted Bayesian Optimisation with Hyperband (ABOHB).

For BO, likewise, the multivariate TPE is used as a surrogate model to match the underlying sampler used in the ABOHB implementation. This ensures that the two methods differ only in their budget *allocation* strategy, thereby isolating the effect of multi-fidelity exploration from the effect of surrogate model choice.

Both BO and ABOHB are run sequentially, without concurrent trial execution. Parallelism would degrade the TPE sampler’s quality, since the sampler conditions only on completed trials, and concurrent trials would be sampled from a model that has not yet observed the in-flight trials’ outcomes.

For the ABOHB implementation, the HyperbandPruner requires a parameter specification. The minimum budget ($b_{\min}$), maximum budget ($b_{\max}$), and reduction factor (η) must be set. The maximum budget corresponds to a full training round (100 epochs). The minimum budget defines the lowest fidelity threshold at which configurations can be reliably evaluated and ranked.

According to L. Li et al. (2018), η should be tuned to practical user constraints. Larger values of η correspond to more aggressive elimination at each rung, thereby reducing the number of SH brackets in the Hyperband schedule. The default choice is $\eta = 3$, which for our $b_{\max} = 100$ and a theoretical $b_{\min} = 1$ would produce five SH brackets, with the most aggressive bracket evaluating configurations after just 1 epoch (Falkner, Klein and Hutter, 2018). However, because the warmup epochs in our search space vary between 0 and 5, configuration evaluation must take place after at least 5 epochs. Furthermore, the empirical training curves of the proxy model (defined in Section 4.4; see Appendix A.1.2) demonstrate that stability of the mAP@50 signal only emerges around 10 epochs.

Consequently, we set $b_{\min} = 10$ to prevent the premature termination of promising configurations due to initialisation noise. Given $\eta = 3$, the maximum number of brackets is $s_{\max} = \lfloor \log_3(100/10) \rfloor = 2$, producing three SH brackets per cycle (L. Li et al., 2018). Theoretically, these brackets begin with 9, 5, and 3 configurations, respectively, with the final exploitation bracket ($s = 0$) training its configurations for the full 100 epochs immediately. The actual number of cycles completed within the matched cumulative training compute budget is determined empirically and matched to $T_{\text{train},\text{BO}}$ post-hoc, as described above. The exact bracket-internal scheduling depends on Optuna’s implementation, which may diverge slightly from the original specification of Falkner, Klein and Hutter (2018).

All experiments are run on a single NVIDIA GeForce RTX 4090 GPU (24 GB VRAM). The

environment utilises Python 3.10 and the Compute Unified Device Architecture (CUDA) 12.2 toolkit for hardware acceleration.

4.6.2 Search Space

Table 4.5 defines the HPO search space. The four parameters are selected from the pool of most commonly tuned hyperparameters in YOLO-based detection models (Hoang, Lee and Park, 2025; Wahyudi, Soesanti and Nugroho, 2023; Pascual-González et al., 2025).

Table 4.5: HPO search space for Experimental Study 1.

Parameter	Sampling	Range	Default
lr0	log-uniform	$[10^{-5}, 5 \cdot 10^{-3}]$	10^{-3}
momentum	uniform	$[0.900, 0.999]$	0.937
weight_decay	log-uniform	$[10^{-5}, 10^{-2}]$	5×10^{-4}
warmup_epochs	uniform	$[0, 5]$	3

Learning Rate and Momentum Learning rate is among the most influential hyperparameters in neural network training, directly governing the magnitude of parameter updates at each step (Bishop, 2006; Bengio, 2012; Smith, 2018). For AdamW, the parameter update rule is:

$$\theta_{t+1} = \theta_t - \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} - \alpha \lambda \theta_t \quad (4.2)$$

where θ_t is the vector of network weights at step t , α is the learning rate, $\hat{m}_t$ and $\hat{v}_t$ are the bias-corrected first and second moment estimates of the gradient, ϵ is a small constant for numerical stability, and λ is the weight decay coefficient. The first moment estimate $\hat{m}_t$ is controlled by β_1 , the momentum parameter. Learning rate and momentum are therefore structurally coupled in the AdamW update: the effective step size at any given iteration is a joint function of both. This interdependence means that independent screening of each parameter in isolation would be uninformative; the optimal learning rate depends on which momentum value is active, and vice versa. Joint search via BO is therefore not just preferable but methodologically necessary for these two parameters given the fixed choice of AdamW.

Weight Decay The final term in Equation 4.2, $\alpha \lambda \theta_t$, is the decoupled weight decay introduced by Loshchilov and Hutter (2019). Unlike standard Adam, AdamW applies weight decay directly to the weights independently of the gradient scaling, giving it a clean and consistent regularisation interpretation across all search configurations. However, at the proxy scale, the model is trained on 25% of the full dataset, which reduces the implicit regularisation that arises from data volume and sampling stochasticity. The optimal weight decay under these reduced-data conditions may therefore differ from what is optimal at full scale. Whether that difference materialises remains a question, making weight decay a parameter of particular interest in the cross-scale transfer analysis.

Warmup Epochs Warmup epochs govern the initial ramp-up of the learning rate from near-zero to its target value α , preventing destabilising gradient updates before the optimiser has accumulated reliable moment estimates. The optimal warmup length is directly coupled to the target learning rate: a higher learning rate requires a longer ramp, while an unnecessarily long warmup delays convergence. Since the learning rate is under search, a joint search that includes warmup epochs is of direct interest.

4.7 Experimental Study 2 (E2): Data Composition Optimisation

Experimental Study 2 (E2) investigates whether optimising the training data composition with respect to damage sub-types within the proxy framework improves detection performance, and whether transfer under the proxy framework is more faithful for data-space parameters than for training hyperparameters.

4.7.1 Motivation

Section 3 established that the Lensor training data exhibits a mild class imbalance at the prediction-label level (1.47:1, scratch:dent) but a substantial sub-type imbalance at the damage type $\times$ size class level, reaching 83:1 after excluding cells with fewer than ten ROIs. This sub-type imbalance is invisible to the training objective, since the loss is computed against the collapsed training labels, under which scratch and scratch zone are treated as a single category, and dent and dent zone are merged similarly. The most operationally relevant compositional issue in the data therefore operates below the granularity at which the model is optimised.

E2 targets this hidden imbalance by treating the composition of the training set at sub-type level as the object of optimisation. This level is operationalised as nine strata, defined by pooling the audit cells along the damage type and size class axes (Section 4.7.2); the granularity used is coarser than the full cross-product of these axes but finer than the prediction-level label.

To address the imbalance, a weighted resampling scheme is used, in which each weight is attached to a stratum and governs its representation in the training set. Each stratum is assigned a continuous weight, and the optimisation searches over these weights against a held-out validation signal while the inner loop performs standard training of the proxy model under the resampled data. This is the standard bilevel formulation of hyperparameter optimisation, with stratum sampling weights as the search variables rather than training-procedure parameters. It inherits from Ren et al. (2018) the principle that a validation signal can guide the assignment of importance to training data, but differs from the per-step meta-optimisation of Ren et al. (2018) in where the outer loop operates: Ren et al. (2018) recompute example weights at every gradient step against a validation signal, so the optimised weights exist only relative to that specific training trajectory. Here the outer loop runs offline, with BO searching over compositional configurations across complete training runs. The optimised weight vector therefore exists as a standalone artefact that can be re-used independently of the training run that discovered it.

The choice of resampling over loss reweighting is dictated by the granularity. Although loss reweighting at sub-type granularity is mechanically conceivable, since per-ROI loss contributions could be scaled by the weight of each ROI's stratum, Ultralytics (Glenn and Qiu, 2026) does not expose per-instance loss weighting, making the implementation infeasible. Preferring resampling over loss reweighting is also empirically supported: Buda, Maki and Mazurowski (2018) report that oversampling is the dominant resampling strategy for CNNs under class imbalance and does not induce overfitting, addressing the standard objection to with-replacement sampling.

The sampling strategy extends the basic principle of RFS (Gupta, Dollar and Girshick, 2019) to solve the multi-class-per-frame problem, elaborated in Section 4.7.3. It differs, however, in the granularity at which it operates, in the use of searched weights rather than analytical ones (Equation 2.12), and in the aggregation function that determines which class governs the repetition factor of a given image.

The optimisation is conducted within the same proxy framework used in E1, and the resulting stratum weights are applied at full scale. The search is performed using the BO algorithm, with the same surrogate model (multivariate TPE) and evaluation budget (20 evaluations).

The choice of BO follows from the results of E1, which indicate that BO finds configurations that transfer more faithfully to full scale on the validation set, against which the proxy is optimised, in all three seeds (Section 5.1.2). Since E2 is itself a transfer study evaluated against the same validation signal, BO is at least as suitable as ABOHB and directionally preferable, so running both algorithms would double the search cost for a comparison E1 has already provided.

Furthermore, holding the algorithm, surrogate, search budget, and sampling mechanism identical to E1 ensures that the proxy calibration ratio can be compared fairly across the two parameter spaces, isolating the effect of the parameter space itself from differences in the search procedure.

A note on the choice of objective is warranted. The search variables in E2 are the nine per-stratum weights, but the objective the search maximises is aggregate validation mAP@50, the same signal optimised in E1. The sub-type granularity therefore lies in the search variables rather than in the objective: E2 asks whether adjusting the training composition at sub-type level improves overall detection performance, not whether performance can be traded between classes.

Optimising the aggregate is dictated by two requirements. First, it matches the definition of detection performance used throughout the thesis, so that SQ2a is answered on the same terms as SQ1a. Second, and more consequentially, the cross-space calibration comparison (SQ1b) requires E1 and E2 to optimise an identical signal; a per-class objective in E2 would confound the comparison of calibration ratios with a difference in objective.

This choice carries a structural limitation at sub-type granularity. As the unweighted mean of the per-class scores, an aggregate objective cannot register a compositional change that improves one class at another’s expense: an equal gain and loss across two classes leave the mean unmoved. The objective is, in this specific sense, blind to trades between sub-types even though the search acts at sub-type level. This is a property of the aggregate objective and is noted here as a known limitation of the design.

The remainder of this section develops the stratification design, the resampling mechanism, and the search space in turn, before specifying the experimental conditions.

4.7.2 Stratification Design

The dataset audit identified substantial imbalance at the sub-type level defined by damage type and size class, while imbalance at the training-label level remained mild. A natural starting point is therefore the full cross-product of these two dimensions, yielding twenty positive audit cells and a negative group, for twenty-one BO search dimensions in total.

However, there are three problems with this approach.

First, the literature reports that BO generally performs well for a search space of only 10–20 parameters (Moriconi, Deisenroth and Sesh Kumar, 2020). Second, although TPE is more tractable than Gaussian-process-based BO in high-dimensional configuration spaces, this advantage is algorithmic rather than informational: fitting any surrogate that meaningfully concentrates density on promising regions requires sufficient observations relative to dimensionality (Hutter, Kotthoff and Vanschoren, 2019). Under the search budget of twenty evaluations adopted here for parity with E1, a search over twenty-one parameters operates at one observation per dimension, at which point surrogate-guided search degenerates toward random sampling (Moriconi, Deisenroth and Sesh Kumar, 2020; Hutter, Kotthoff and Vanschoren, 2019).

A third consideration concerns the per-cell signal that BO can learn from. Each weight controls the representation of a cell’s ROIs in the training distribution, which in turn determines how strongly the gradient updates of the model are pulled by examples from that cell. For BO to learn whether a cell benefits from up- or down-weighting, varying its weight must produce a proxy mAP differential that exceeds the noise floor of a single training run. While duplication of frames does shape the model’s decision boundary by rebalancing gradient exposure (the standard mechanism by which oversampling addresses class imbalance in CNNs, Buda, Maki and Mazurowski, 2018), the magnitude of this effect is bounded by the visual diversity of the cell: beyond a certain duplication count, the model has fit what the available frames can teach, and further upweighting produces diminishing marginal effect on validation performance. Cells with few unique frames therefore produce little mAP variation across the weight range: the model exhausts what the few available images can teach, and additional upweighting reinforces memorisation (overfitting) rather than learning, leaving BO without a usable signal.

Table 4.6 reports the number of unique frames per audit cell at proxy scale, defined as the number of distinct images containing at least one ROI from that cell. Frame count is the relevant

unit because sampling operates on frames rather than individual ROIs. The ROIs-per-frame column confirms that the two possible units of account do not conflict: with the ratio close to one in nearly all cells (the exception, dent 0–2 cm at 2.42, is not a sparse cell under either count and is not a merge candidate), ranking cells by frames and by ROIs gives the same ordering, so the sparsity identified below is not an artefact of counting frames rather than annotations. Two cells are empty (dent zone, 0–2 cm and dent zone, 2–5 cm), and five further cells contain fewer than 100 unique frames. For these cells the diversity ceiling is reached quickly: pushing their weights high in the search would lead the model to overfit on the small set of available frames rather than learn anything new, and BO has no reliable mAP signal to distinguish good weight values from bad ones across the range.

Table 4.6: Frame and ROI diversity per audit cell for the proxy dataset. Unique frames refers to the number of frames that contain at least one ROI from the given cell. Frames containing ROIs from multiple cells are counted in each, so the column does not sum to the training-set frame total.

Audit cell	Unique frames	Total ROIs	ROIs / frame
dent zone, 0–2 cm	0	0	–
dent zone, 2–5 cm	0	0	–
scratch zone, 0–2 cm	19	20	1.05
dent zone, 5–10 cm	28	28	1.00
dent zone, 10–30 cm	47	48	1.02
dent zone, 30+ cm	49	57	1.16
dent, 30+ cm	75	77	1.03
scratch zone, 2–5 cm	155	184	1.19
scratch, 30+ cm	168	180	1.07
scratch, 0–2 cm	169	239	1.41
dent, 0–2 cm	204	493	2.42
dent, 10–30 cm	207	220	1.06
scratch zone, 30+ cm	284	308	1.08
scratch zone, 5–10 cm	286	344	1.20
scratch zone, 10–30 cm	445	518	1.16
dent, 5–10 cm	490	542	1.11
scratch, 2–5 cm	532	613	1.15
scratch, 5–10 cm	548	629	1.15
scratch, 10–30 cm	566	650	1.15
dent, 2–5 cm	932	1,327	1.42

Dent zone exhibits low frame diversity at all size classes, with two cells empty and the remaining three below 50 unique frames. Scratch zone shows a related pattern at 0–2 cm: only 19 unique frames, consistent with the structural property that zone-level annotations apply to larger affected regions rather than point damage.

As a result, it makes sense to merge these cells into others to increase the pool of frame diversity. Alongside this, to bring the overall number of groups down, the size classes are reduced to small, medium, and large. The twenty audit cells are thereby pooled into nine strata, shown in Table 4.7

Within damage types, the audit’s five size classes are collapsed to three: small (0–2 cm), medium (2–10 cm, combining 2–5 cm and 5–10 cm), and large (10+ cm, combining 10–30 cm and 30+ cm). The breakpoints reflect the size distribution observed in the audit: training mass for both dent and scratch concentrates in the 2–10 cm range and tails off above 10 cm, so the medium and large bins correspond to modal and tail behaviour respectively. The 0–2 cm range is preserved as its own bin. Scratch zone retains two size bins rather than three, since its 0–2 cm cell has too little diversity to stand on its own as a small class.

Note that the formed size bins (small, medium, large) are defined by deterministic aggregation

Table 4.7: E2 stratum definitions with frame diversity at proxy scale.

Stratum	Definition	Unique frames	Total ROIs	ROIs / frame
scratch small	scratch, 0–2 cm	169	239	1.41
dent small	dent $\cup$ dent zone, 0–2 cm	204	493	2.42
dent large	dent $\cup$ dent zone, 10+ cm	324	402	1.24
scratch zone medium	scratch zone, 2–10 cm	414	548	1.32
scratch zone large	scratch zone, 10+ cm	700	826	1.18
scratch large	scratch, 10+ cm	712	830	1.17
scratch medium	scratch, 2–10 cm	989	1,242	1.26
dent medium	dent $\cup$ dent zone, 2–10 cm	1,333	1,897	1.42
negative	frames with no ROIs	1,247	0	–

of the audit’s cm-based size classes and are unrelated to the legacy small/medium/large annotator labels excluded in the audit (Section 3). Those legacy labels were dropped because their cm mapping was inconsistent across annotator sessions.

Each stratum in Table 4.7 receives a weight in the search space that dictates its presence in the training distribution. The optimised weights enter the multinomial resampling procedure described in Section 4.7.3.

Finally, for the nine defined strata, the search budget of twenty evaluations, adopted for parity with E1, operates at roughly two observations per dimension. This is deliberately modest. The budget is not chosen to guarantee convergence but to hold the search procedure identical to E1, without which the cross-space calibration comparison (SQ1b) would be confounded, and to remain within the production-feasible regime that motivates the proxy framework in the first place.

4.7.3 Sampling Mechanism

Following the stratum definition, the sampling mechanism translates a proposed weight vector into a resampled training set in which each stratum is represented in accordance with its assigned weight. A structural feature of the mechanism is that strata are defined at the ROI level, while the sampling operates at the frame level: YOLO trains on complete images and all ROIs in a frame are presented to the model together, so it is frames that are drawn from the training set, not individual ROIs. This level mismatch shapes two design choices, both of which adapt the image-level resampling of RFS (Gupta, Dollar and Girshick, 2019): a multinomial-to-fixed- N draw in place of stochastic rounding, and mean aggregation across a frame’s ROIs in place of maximum aggregation. Each is developed below.

Multinomial-to-fixed- N sampling

The sampling scheme must produce a training set of exactly N frames, where N is the size of the original training set. This is required because the central comparison in E2 isolates the effect of compositional change; if the resampled set size varied with the weight vector, observed differences would conflate composition with training-set size. The mechanism must also operate identically at proxy scale and full scale, with only N differing, so that the proxy-to-full transfer measurement is not confounded by mechanism-induced variation.

A given frame i is assigned a probability

$$p_i = \frac{w_{\text{frame}_i}}{\sum_{j=1}^M w_{\text{frame}_j}}, \quad (4.3)$$

where w_{frame_i} is derived from the ROI stratum weights via the aggregation rule below, and M is the number of frames in the original training set. The resampled training set is then constructed

by drawing

$$(c_1, \dots, c_M) \sim \text{Multinomial}(N; p_1, \dots, p_M), \quad (4.4)$$

where c_i is the number of times frame i appears in the resampled set. By construction $\sum_i c_i = N$ regardless of the weights.

This differs from RFS, which produces a per-epoch training list whose size fluctuates around its expectation due to Bernoulli rounding of fractional repeat factors. The long-tailed dataset for which RFS was developed, LVIS, did not require exact dataset-size control. For this investigation, isolating composition from quantity is essential.

A known consequence of with-replacement sampling under uniform weights is that the resampled set is a bootstrap of the original rather than the original itself. For uniform $p_i = 1/M$ and $N = M$, the count c_i for each frame is approximately Poisson(1) distributed for large M (Novak, 2019), giving $P(c_i = 0) \approx e^{-1}$ and therefore $1 - e^{-1} \approx 63\%$ of frames receiving at least one draw. This bootstrap effect is a property of with-replacement sampling and is constant across any condition that uses the multinomial mechanism, regardless of the weight vector chosen.

Frame-to-weight aggregation

Strata are defined at the ROI level but sampling operates at the frame level, so the stratum weights must be aggregated to produce a single w_{frame_i} per frame. For a positive frame containing ROIs in strata $s_1, \dots, s_k$ with weights $w_{s_1}, \dots, w_{s_k}$:

$$w_{\text{frame}_i} = \frac{1}{k} \sum_{j=1}^k w_{s_j}. \quad (4.5)$$

For a negative frame, $w_{\text{frame}_i} = w_{\text{negative}}$.

RFS uses maximum aggregation, where a frame inherits the repeat factor of its rarest class. This fits the LVIS setting, where the primary axis of imbalance is rare-versus-common (Gupta, Dollar and Girshick, 2019). For this investigation the axis of optimisation is different: the strata partition damage sub-types and size classes, and their relative importance is what BO is searching over rather than a quantity known a priori. Maximum aggregation would commit to a rarest-wins structure that conflicts with the search; mean aggregation is neutral on the ordering and lets BO discover the proportional structure directly.

The empirical viability of mean aggregation rests on most frames containing ROIs from a single stratum, in which case the aggregation rule is the identity. For the proxy dataset, 76.6% of frames contain ROIs from just one stratum, and for the full training dataset this is 78.4%. Pure-stratum frames therefore dominate at both scales, and their proportions align within 1.8 percentage points between proxy and full. As a result, for most of the dataset the probability of drawing a frame is directly proportional to the weight of the stratum of its ROIs, limiting the dilution effect.

It should be noted, however, that a structural limitation of frame-level resampling for object detection is that ROI-level composition cannot be controlled exactly through frame-level weights when frames are mixed.

4.7.4 Search Space

The search space comprises one weight per stratum, sampled independently from a log-uniform distribution over $[0.2, 5.0]$. Three considerations motivate this parameterisation.

First, per-frame weights are normalised to a probability distribution before the multinomial draw (see Equation 4.3), so the sampling distribution depends on the relative weights between frames rather than their absolute values: scaling all weights by a constant factor leaves every frame's sampling probability unchanged. The compositional effect of a weight change is therefore multiplicative rather than additive. Doubling a stratum's weight from 1.0 to 2.0 changes the resampled distribution by the same proportion as doubling it from 2.0 to 4.0, since both moves

double that stratum’s weight relative to the rest. Log-uniform sampling aligns the search geometry with this multiplicative structure, ensuring that TPE’s density estimates are calibrated to equal proportional changes in the sampling distribution rather than to equal absolute changes in weight values. A consequence of this parameterisation is that the neutral weight $w = 1$, at which the multinomial reproduces the bootstrapped natural data distribution (Section 4.7.3), sits at the log-midpoint of the chosen range, with up- and down-weighting explored at equal sampling density on either side.

Second, the bounds permit meaningful redistribution of the training distribution without allowing any stratum to approach degeneracy. The range spans a 5 $\times$ departure from the neutral weight in either direction, which is expected to be sufficient to produce substantial shifts in stratum representation while ensuring that no stratum can reach zero or dominate the resampled set within the chosen bounds. The realised compositional shifts under the frame-mean aggregation mechanism are smaller in magnitude than the nominal 5 $\times$ factor, as mean aggregation across mixed-stratum frames dilutes the effect of any single stratum’s weight.

Third, the bounds comfortably cover the weight range produced by the principled heuristic baseline used in Condition B (Section 4.7.5), which yields normalised weights in the range $[0.56, 1.73]$ across the nine strata. Placing this heuristic point well within the interior of the search space leaves BO room to explore weight configurations substantially more aggressive than the heuristic in either direction.

4.7.5 Experimental Design

To test the effect of data composition optimisation and assess transfer under the proxy framework, the experimental design isolates three distinct effects: the cost of the multinomial resampling mechanism, the compositional benefit of non-uniform stratum rebalancing, and the additional value of BO-optimised weights over a heuristic. A naive comparison between an unmanipulated baseline and BO-optimised weights would conflate all three. The four conditions described below decompose them through pairwise comparisons in which each comparison holds all other mechanisms constant. The full experimental setup is summarised in Table 4.8.

Table 4.8: E2 experimental conditions. All proxy runs use the proxy configuration defined in Section 4.4; all full retrains use the configuration in Section 4.3. Training hyperparameters are locked to the E1 BO-best configuration throughout all phases.

Phase	Model	Condition	Runs	Notes
Search	Proxy	BO weight search	3	20 evaluations per search seed
Train	Proxy	Condition A	3	Reused from E1 (BO-best)
Train	Proxy	Condition A'	3	Independent seeds
Train	Proxy	Condition C	3	Independent seeds
Train	Full	Condition A	3	Reused from E1 (BO-best)
Train	Full	Condition A'	3	Independent seeds
Train	Full	Condition B	3	Independent seeds
Train	Full	Condition C	3	Independent seeds

Condition A: Natural Baseline

Condition A uses the natural training distribution, without any resampling mechanism, and the BO-best training hyperparameters from E1. Since the same experiment was conducted in E1, the existing retrains are reused here for comparison. Condition A therefore serves as the reference against which both the bootstrap effect of the resampling mechanism (isolated by Condition A') and the compositional effects of rebalancing (Conditions B and C) are measured.

Condition A': Uniform Multinomial Baseline

Condition A' isolates the cost of the multinomial-with-replacement mechanism independent of any rebalancing. The multinomial resampling is applied with all nine stratum weights set to 1.0. Under frame-mean aggregation, every frame therefore receives a frame weight of 1.0, and the multinomial draws frames uniformly from the training set. Under this uniform draw, approximately 37% of unique frames are absent from any given draw, with the remaining 63% appearing with duplication. Condition A' measures the performance cost this reduction in effective unique-frame diversity introduces, independent of any compositional change.

Condition B: Square-Root Inverse-Frequency Baseline

Condition B introduces a non-uniform baseline against which BO-optimised weights are compared. The heuristic applies inverse-frequency rebalancing to the stratum weights: for each positive stratum s , the weight is set to $w_s = 1/\sqrt{n_s}$, where n_s is the ROI count for that stratum at full scale; for the negative stratum, n_s is the frame count. Weights are normalised so the mean across the nine strata equals one. The resulting normalised weights span $[0.56, 1.73]$, with smaller strata up-weighted and larger strata down-weighted relative to the neutral value of 1.

Under the multinomial sampling mechanism (Section 4.7.3), each frame is drawn with probability proportional to its frame weight (Equation 4.3). The expected fraction of the resampled training set originating from stratum s is the sum of these frame probabilities across all frames containing an ROI from s . For frames containing ROIs from only one stratum, the contribution depends solely on w_s ; for mixed-stratum frames the contribution is diluted, as the frame weight averages w_s with the weights of other strata present in the frame. Under the simplifying assumption of pure-stratum frames, which holds for 76.6% of positive frames at proxy scale (Section 4.7.2), the expected proportion of stratum s in the resampled training set is approximately:

$$p(s) \approx \frac{w_s \cdot n_s}{\sum_j w_j \cdot n_j}. \quad (4.6)$$

Substituting $w_s = 1/\sqrt{n_s}$ reduces this to square-root sampling (Mahajan et al., 2018):

$$p(s) \approx \frac{\sqrt{n_s}}{\sum_j \sqrt{n_j}}. \quad (4.7)$$

Square-root sampling is the standard moderate rebalancing strategy in the long-tail object detection literature, used to dampen the bias induced by heavy-tailed class distributions without fully equalising class representation (Mahajan et al., 2018). The weight form is used here in preference to specifying probabilities directly because it matches the parameterisation of the BO search space, allowing Condition B to be expressed as a specific point in that search space.

Condition B therefore applies a non-uniform rebalancing strategy grounded in the class-balancing literature, providing a reference point against which the methodological proposition of Condition C can be tested.

Condition C: BO-Optimised Weights

Condition C is the condition in which the stratum weights are optimised by BO over the search space described in Section 4.7.4. The BO search runs at proxy scale, and the resulting BO-best weights from all three proxy searches are then applied at full scale for the retrain.

The proxy calibration ratio ρ for E2 is computed against Condition A' as the denominator baseline. The A'-anchored ratio isolates the transfer effect of compositional optimisation by holding the resampling mechanism constant in both numerator and denominator.

Attribution Decomposition

The four conditions enable a pairwise attribution decomposition in which each comparison isolates a single mechanism, summarised in Table 4.9

Table 4.9: Pairwise comparisons enabled by the four-condition design. Each comparison isolates one mechanism by holding all others constant.

Comparison	Measures
$A \rightarrow A'$	Bootstrap cost of multinomial mechanism
$A' \rightarrow B$	Compositional benefit of heuristic rebalancing
$A' \rightarrow C$	Compositional benefit of BO optimisation
$B \rightarrow C$	Value of automated search over heuristic
$A \rightarrow C$	Total operational benefit

Variance Protocol

Three retrain seeds (42, 43, 44) are used per condition, matched across conditions to enable paired analysis, consistent with the variance protocol used in E1. Each seed jointly randomises weight initialisation, data ordering, augmentation sampling, gradient noise, and (for A' , B, and C) the multinomial draw, following the joint-randomisation protocol of Bouthillier et al. (2021).

Results

This chapter describes the results of the experimental procedure outlined in Chapter 4. Section 5.1 reports results from the training-hyperparameter HPO procedure. Subsection 5.1.1 examines whether HPO in the proxy model produces configurations that improve full-scale performance, and reports the calibration ratio that formalises the transfer effect. Subsection 5.1.2 then analyses whether the choice of HPO algorithm (BO vs ABOHB) affects the strength of that transfer.

Section 5.2 reports the results from the data composition optimisation. The first subsection, 5.2.1, analyses the performance of the proposed procedure against the ablation baselines, while the second subsection, 5.2.2, compares the calibration ratio of the two experimental studies.

5.1 Experimental Study 1 (E1)

5.1.1 Effect of proxy-guided HPO on full-scale performance (SQ1A)

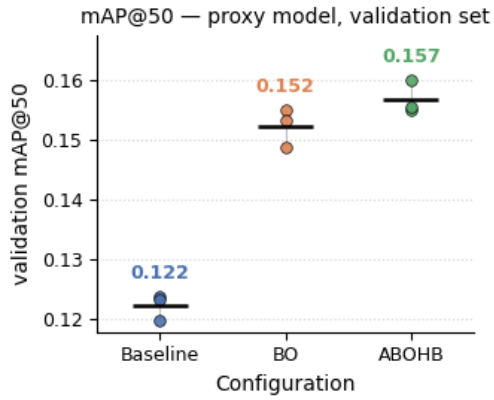
Figure 5.1d shows the mAP@50:95 performance of the full-scale model across three hyperparameter configurations. The baseline configuration corresponds to the untuned hyperparameters; BO-best and ABOHB-best correspond to the best configurations found by the BO and ABOHB search procedures respectively. Full per-seed values are given in Table A.5.

The BO and ABOHB configurations exceed the baseline performance for all three seeds. The configurations resulting from HPO performed on the proxy model therefore led to better performance of the full-scale model on the test set in absolute terms. However, to attribute this improvement specifically to the proxy-constrained HPO pipeline, we must account for the confounding variables that separate Figure 5.1d from the search setting. The full retrain on test differs from the proxy search along three dimensions simultaneously: model scale, evaluation metric, and evaluation set. We isolate each in turn using Figures 5.1a–5.1c.

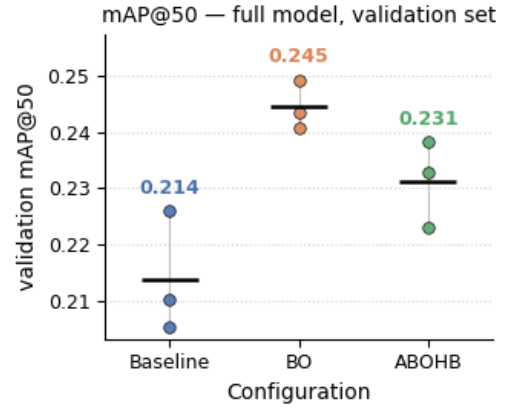
Figure 5.1a displays the proxy model’s mAP@50 on the validation set. Full per-seed values are given in Table A.2.1. This mimics the setting in which configurations were evaluated and ranked during HPO. Note that the proxy mAP@50 values shown here are slightly below those reported in Table 5.2, because the latter reflects search-phase evaluation at the fixed training seed used during HPO (seed 0), while the former are retrains with seeds 42, 43, and 44 used for the variance analysis. Figure 5.1b shows the full model’s mAP@50 on the validation set; since the metric and evaluation set match those of Figure 5.1a, the difference between these two figures isolates the effect of model scale.

In the proxy model, the HPO process produces better mAP@50 across all three seeds for both BO and ABOHB. In the full model, this effect is dampened. The BO configuration outperforms the baseline in all three seeds, but the ABOHB configuration’s performance range overlaps with the baseline.

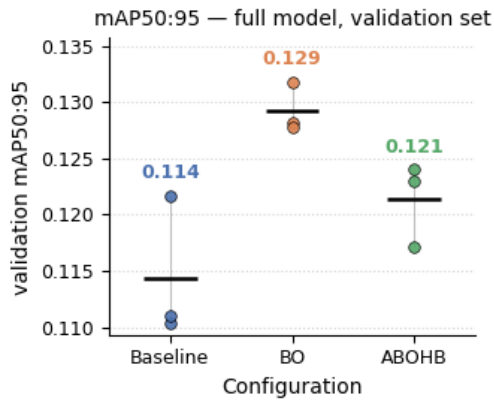
Turning to Figure 5.1c, the same full model is evaluated on the validation set but with the stricter mAP@50:95 metric. Comparing this to Figure 5.1b isolates the metric effect. Absolute values drop substantially, as expected: mAP@50:95 averages AP across multiple IoU thresholds from 0.5 to 0.95 in steps of 0.05, with the strictest threshold requiring near-perfect overlap between predicted and ground-truth bounding boxes. The metric incorporates localisation precision in



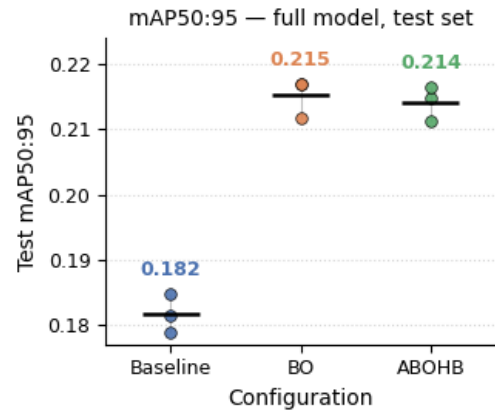
(a) mAP@50 on the proxy model, evaluated on the validation set.



(b) mAP@50 on the full model, evaluated on the validation set.



(c) mAP@50:95 on the full model, evaluated on the validation set.



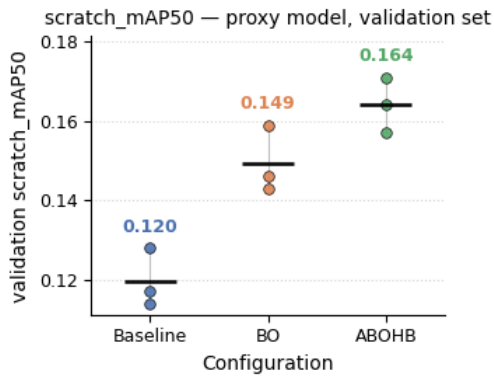
(d) mAP@50:95 on the full model, evaluated on the test set.

Figure 5.1: Overview of confounding effects between the HPO signal (proxy mAP@50) on the validation set and the performance measurement on test at mAP@50:95. Three retrain seeds per condition; horizontal bars indicate the seed mean.

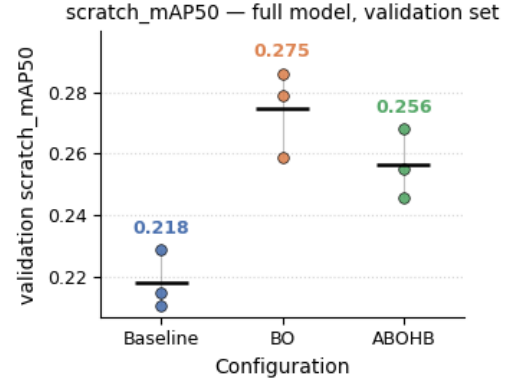
addition to coarse detection capability. The general ordering of configurations is preserved, with the BO and ABOHB tuned configurations producing visibly higher $\text{mAP}@50:95$ than the baseline. The proportional rise from baseline to HPO configurations across both $\text{mAP}@50$ and $\text{mAP}@50:95$ implies that the HPO gain is not confined to coarse detection but includes localisation precision as well.

Finally, returning to Figure 5.1d, the metric and scale are unchanged from Figure 5.1c; only the evaluation set differs. Two observations stand out. First, all absolute $\text{mAP}@50:95$ values are higher on the test set than on the validation set. Second, the relative ordering across configurations differs: the gap between BO and ABOHB visible on validation collapses on test, with the two methods producing nearly identical $\text{mAP}@50:95$.

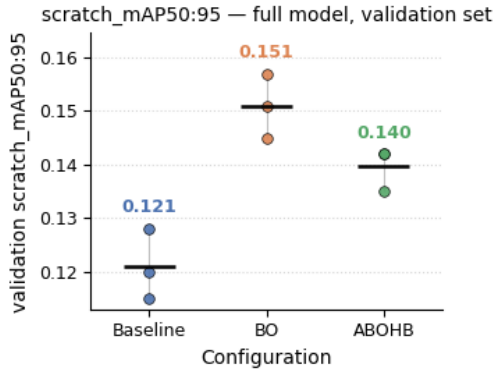
The higher aggregate performance on test may reflect higher within-class AP on test, concentrated in the dent class: at baseline, test dent $\text{mAP}@50:95$ is roughly double its validation counterpart, whereas the scratch class differs far less between the two sets. This within-class gap is examined in the per-class analysis below. To separate the two classes, we reproduce the four-panel analysis for scratch and dent individually.



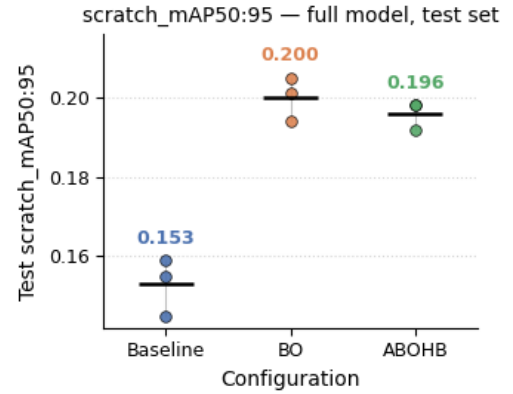
(a) Scratch $\text{mAP}@50$ on the proxy model, evaluated on the validation set.



(b) Scratch $\text{mAP}@50$ on the full model, evaluated on the validation set.



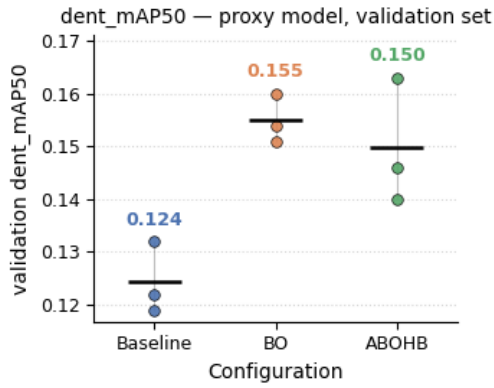
(c) Scratch $\text{mAP}@50:95$ on the full model, evaluated on the validation set.



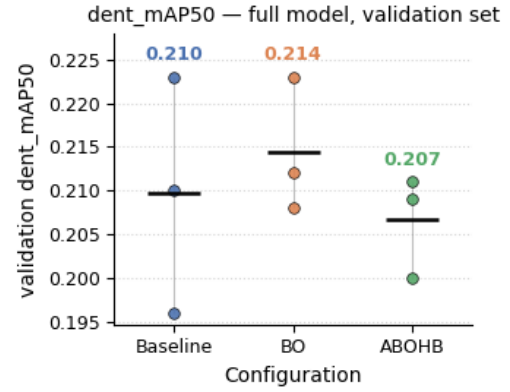
(d) Scratch $\text{mAP}@50:95$ on the full model, evaluated on the test set.

Figure 5.2: Overview of confounding effects between the HPO signal (proxy $\text{mAP}@50$) on the validation set and the performance measurement on test at $\text{mAP}@50:95$ for the scratch class. Three retrain seeds per condition; horizontal bars indicate the seed mean.

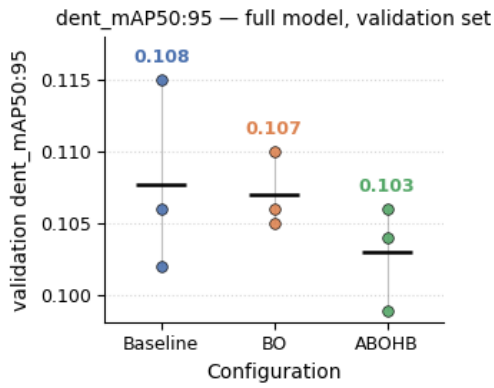
Comparing Figures 5.1, 5.2, and 5.3, the four-panel pattern for scratch closely tracks that of the aggregate, while dent deviates considerably. The similarity between aggregate and scratch arises



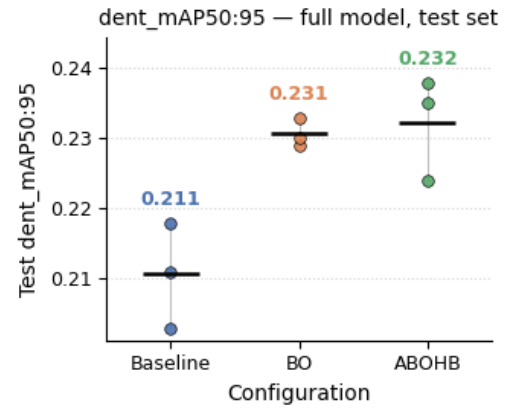
(a) Dent mAP@50 on the proxy model, evaluated on the validation set.



(b) Dent mAP@50 on the full model, evaluated on the validation set.



(c) Dent mAP@50:95 on the full model, evaluated on the validation set.



(d) Dent mAP@50:95 on the full model, evaluated on the test set.

Figure 5.3: Overview of confounding effects between the HPO signal (proxy mAP@50) on the validation set and the performance measurement on test at mAP@50:95 for the dent class. Three retrain seeds per condition; horizontal bars indicate the seed mean.

because the aggregate mAP is the unweighted mean of the two per-class APs, and the scratch class contributes substantially larger gains in absolute terms than dent. Following the line of argumentation above, configurations found on the proxy model therefore transfer well to the full model for the scratch class.

For dent, the picture is more complex. On the proxy model, both BO and ABOHB improve over baseline. On the full model with mAP@50 on validation (Figure 5.3b), the HPO configurations overlap with the baseline: only the BO mean is higher than baseline, while the ABOHB mean is lower. The mAP@50:95 panel on the full model (Figure 5.3c) shows the ABOHB configuration falling clearly below baseline, suggesting that the HPO gain for dent did not extend to localisation precision. On the test set, however, both HPO configurations again beat the baseline.

This pattern indicates that the transfer of HPO-selected configurations is limited and validation-specific for the dent class, while transferring more cleanly for scratch. The recovery of dent performance on test points to within-class distributional differences between validation and test as the operative mechanism. As established in the data audit, 33.9% of validation dent ROIs fall in the 2–5 cm modal size and 20.4% in the 10–30 cm tail, against 45.6% and 11.2% in test. The training set is also concentrated in the modal sizes (35.4% in 2–5 cm, only 9.6% in 30+ cm). A model trained heavily on the modal size band evaluates better on a test set that mirrors this concentration than on a validation set that places more weight on the size tail.

Quantifying transfer with the calibration ratio

To quantify the transfer effect described above, we compute the calibration ratio ρ , defined as the per-seed ratio of full-retrain gain to proxy-retrain gain over the baseline on validation mAP@50.

Table 5.1: Proxy-to-full calibration ratio ρ on the validation set, computed as the per-seed ratio of full-retrain to proxy-retrain mAP@50 gain over baseline.

Class	BO		ABOHB	
	Mean ρ	95% CI	Mean ρ	95% CI
Aggregate	1.026	[0.492, 1.519]	0.491	[0.382, 0.680]
Scratch	1.954	[0.938, 2.731]	0.885	[0.684, 1.000]
Dent	0.146	[−0.063, 0.500]	−0.585	[−1.750, 0.366]

Table 5.1 shows the calibration values across class type and algorithm choice. The ρ values are the mean of three matched-seed ratios with 95% percentile bootstrap confidence intervals (10,000 resamples). Per-seed ρ values are given in Appendix Table A.7. Section 4.2.4 outlines the interpretation of ρ 's value.

Figure 5.4 visualises the results shown in Table 5.1. Several observations can be made. First, both at aggregate level and for the scratch class, $\rho > 0$, indicating that the effect transfers in the same direction at full scale. However, particularly for BO, the confidence intervals are very wide, so the magnitude of the transfer effect cannot be read from them. At aggregate level, for example, the interval for BO, [0.492, 1.519], spans from the proxy overstating the gain to the proxy understating it. Across the three seeds, the per-seed aggregate ρ for BO spans 0.492 to 1.519 (Appendix Table A.7), and this dispersion is the source of the wide bootstrap interval. The dispersion arises from the ratio structure of ρ : each per-seed value divides a small full-retrain gain by a small proxy gain, so modest seed-to-seed variation in either term produces large swings in the ratio. In this case, the variation is a property of the full-retrain gains rather than of the proxy signal, which is comparatively stable across seeds (Figure 5.1a).

For dent, ρ spans negative and near-zero values across both algorithms, indicating transfer that is at best weak and, for ABOHB, directionally reversed. Since the aggregate is the unweighted mean of the two per-class ratios and the scratch class carries the larger, positively-transferring gain, the aggregate transfer effect is primarily scratch-driven.

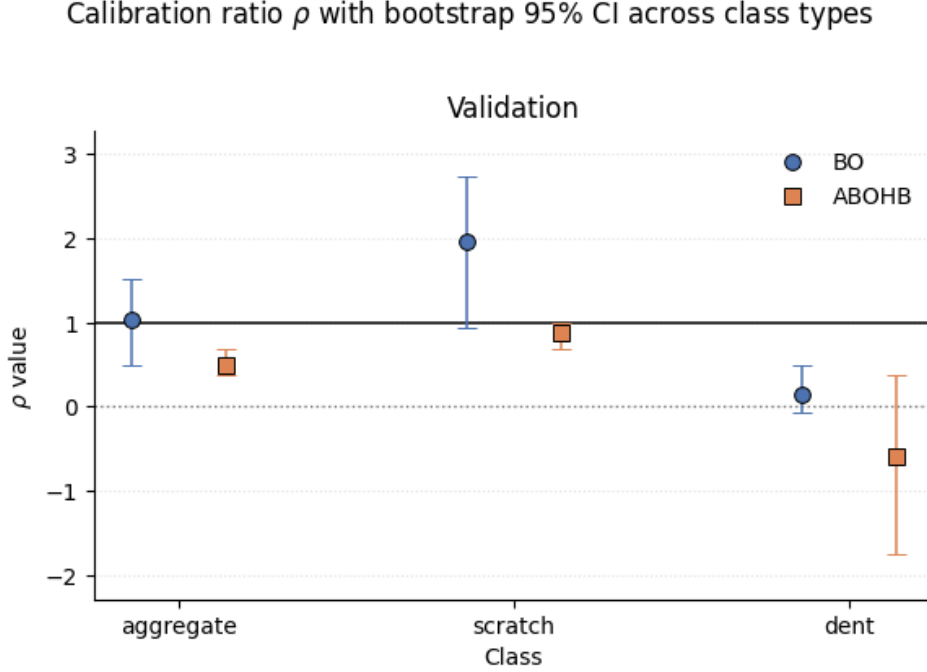


Figure 5.4: Calibration ratio across classes for BO and ABOHB, with 95% bootstrap confidence intervals.

Taken together, the proxy-guided HPO produces configurations whose absolute test-set performance exceeds the baseline in all three seeds, with the scratch class driving most of the aggregate gain. The transfer is class-asymmetric: scratch transfers faithfully, while dent transfer is unreliable on validation and recovers on test, an asymmetry traced to within-class size-distribution differences. The calibration ratio framework formalises these observations but has limited precision at $n = 3$ due to baseline-driven full-retrain stochasticity. We now turn to the question of whether the choice of HPO algorithm affects the strength of transfer.

5.1.2 Effect of HPO algorithm choice on proxy-to-full transfer fidelity (SQ1B)

To compare BO and ABOHB with respect to transfer fidelity, we consult the seed-level ρ values in Appendix Table A.7. For all three seeds, $\rho_{BO} > \rho_{ABOHB}$ at the aggregate level (1.067 vs 0.412, 1.519 vs 0.680, 0.492 vs 0.382). The within-seed paired differences are 0.656, 0.839, and 0.110, which are all positive. This is a robust directional finding: the marginal confidence intervals on ρ_{BO} and ρ_{ABOHB} overlap at aggregate level, but because both ratios are computed against the same baseline run within each seed, the baseline variability of the full retrain on validation, which dominates the absolute estimates, shifts both ratios jointly and largely cancels in the paired difference. The cross-method comparison is therefore more robust than either absolute level. We conclude that, on the validation distribution, BO produces a stronger proxy-to-full transfer effect than ABOHB in all three seeds.

This finding is initially surprising, because ABOHB found a configuration with higher absolute proxy performance during the search itself, as shown in Table 5.2. The same ordering holds in the proxy retrain for aggregate and scratch (Figures 5.1a, 5.2a). The ABOHB configuration produces higher proxy mAP@50 across 2/3 seeds for the aggregate and 3/3 seeds for the scratch class. Yet on the full model, the ordering reverses for aggregate validation mAP@50 (Figure 5.1b) and the gap collapses entirely on test mAP@50:95 (Figure 5.1d).

An interesting note on the best-found configurations by both algorithms relates to the value

Table 5.2: Best-found hyperparameter configurations from the BO and ABOHB proxy searches, compared to the untuned baseline.

Hyperparameter	Untuned default	BO-best	ABOHB-best
lr0	1.000×10^{-3}	2.038×10^{-4}	4.214×10^{-4}
Momentum	0.9370	0.9542	0.9496
Weight decay	5.000×10^{-4}	7.582×10^{-4}	3.596×10^{-3}
Warmup epochs	3	0	0
mAP@50 during search	-	0.162	0.167

of the learning rate. Both selected learning rates (2.038×10^{-4} for BO, 4.214×10^{-4} for ABOHB) sit above the 10^{-4} single-axis optimum suggested by the learning-rate sweep (Section 4.4.3). The sweep varied the learning rate in isolation, with momentum, weight decay, and warmup at their defaults, and so identifies the best learning rate *for that fixed configuration*. The search instead moved all four parameters jointly: both configurations raised momentum, set warmup to zero, and (for ABOHB) substantially increased weight decay. Because these parameters jointly determine the effective optimisation dynamics, the learning rate that is optimal once they have all moved need not match the one that is optimal with them held at their defaults.

To examine why ABOHB obtains higher proxy performance, we plot the per-search trajectories in Figure 5.5.

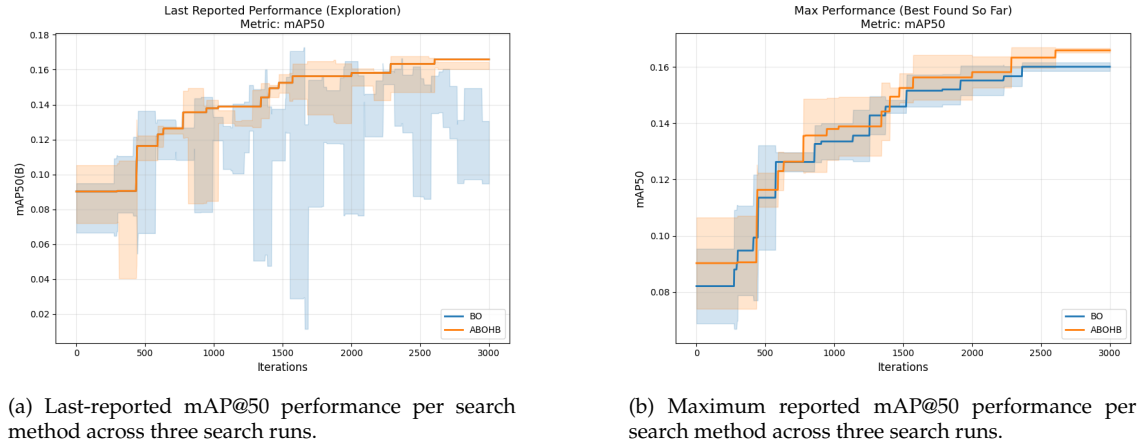


Figure 5.5: Performance across the search trajectory for BO and ABOHB across three seeds. The bold line represents the average, and the shaded band indicates the standard deviation.

Figure 5.5 shows the maximum-so-far (Figure 5.5b) and last-reported (Figure 5.5a) validation mAP@50 over cumulative HPO iterations. Across the three search seeds, ABOHB on average finds a configuration with higher proxy mAP@50 than BO, and its last-reported trajectory rises more steadily. The smoother ABOHB trajectory reflects the pruner: configurations that ABOHB judges as unpromising are stopped early, so the last-reported values only include trials that survived to higher fidelity. BO has no such pruning step, and every trial's last-reported value contributes to the trajectory regardless of how well it performed.

The ability of ABOHB to evaluate more configurations within the same compute budget makes it more likely to find a configuration with an upward-biased proxy mAP@50: with more evaluations on the same noisy validation signal, the best-of-set is more strongly inflated by lucky noise rather than by systematic improvement. This is the selection-bias phenomenon discussed for hyperparameter selection in Cawley and Talbot (2010), sometimes called the winner's curse in HPO. The transferable component of the proxy gain survives to full retrain; the noise component

does not. The ABOHB configuration therefore has a higher proxy mAP@50 but a lower transfer ratio, while the more conservative BO search arrives at a configuration whose proxy gain is more representative of its full-retrain gain.

Looking at the absolute per-seed metrics on the full model rather than ρ , the picture is consistent at the aggregate level: BO outperforms ABOHB on validation full mAP@50 in 3/3 seeds (Appendix Table A.4). On test mAP@50:95, the gap between methods is marginal: BO outperforms ABOHB at the aggregate in 2/3 seeds, on scratch in 2/3 seeds, and on dent in 1/3 seeds. The BO transfer advantage observed on validation therefore does not extend cleanly to test. This is consistent with the validation/test distributional differences documented in Section 5.1.1 which suppress most of the BO-vs-ABOHB difference once the evaluation set shifts.

In summary, ABOHB finds configurations with higher proxy mAP@50 than BO, but the BO configurations transfer more faithfully to full scale on the validation distribution. The mechanism is consistent with a selection-bias effect: the broader search of ABOHB produces noisier maxima whose proxy values overstate the transferable improvement. On test, this advantage is no longer detectable, and the two methods produce broadly comparable absolute performance. At $n = 3$ search seeds and $n = 3$ retrain seeds, this is a directionally consistent rather than statistically significant finding, and a larger sample size would be required to formally characterise the BO-vs-ABOHB calibration gap.

5.2 Experimental Study 2 (E2)

The following section reports results from the data composition optimisation pipeline. Subsection 5.2.1 examines whether Bayesian-optimised sub-type sampling weights (Condition C) improve full-scale performance over the no-resampling baseline (Condition A) and the uniform-weight resampling baseline (Condition A'), and decomposes the observed effects across the four-condition ablation (Conditions A, A', B, C). Subsection 5.2.2 then reports the proxy calibration ratio for Condition C and compares it to the training-hyperparameter calibration ratios from Section 5.1

5.2.1 Performance Improvement with Data Composition Optimisation (SQ2A)

The four experimental conditions (A: no resampling; A': uniform-weight multinomial resampling; B: square-root inverse-frequency heuristic; C: BO-optimised weights) are described in Section 4.7.5. All four conditions inherit the E1 BO-best training hyperparameters as a fixed backbone, so any performance difference between conditions is attributable to data composition alone.

Figure 5.6 presents the four-panel breakdown established in Section 5.1.1: proxy validation mAP@50, full validation mAP@50, full validation mAP@50:95, and full test mAP@50:95. Full numerical results are given in Appendix Tables A.8 and A.9

On the test-set headline metric (Figure 5.6d), the mean ordering is $A (0.215) > B (0.200) > A' (0.197) > C (0.194)$. Two observations follow. First, no resampled condition (A', B, C) improves over the no-resampling baseline A. Second, Condition C does not improve over Condition B or over Condition A' on aggregate test performance; C is the worst condition in all four panels. The same pattern holds on validation (Figures 5.6b, 5.6c), with C consistently below A' and B.

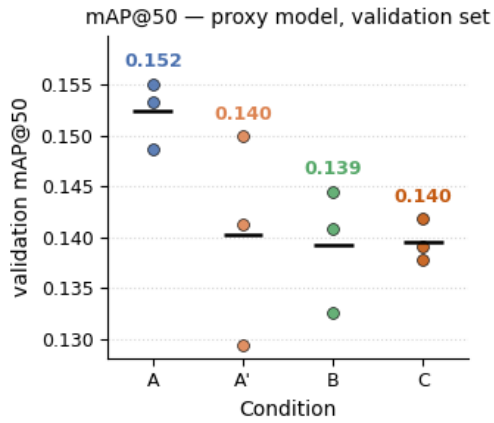
The $A \rightarrow A'$ decline is itself informative: A' shares its training-hyperparameter backbone and its stratum composition with A (Figure 5.9), differing only in that its training set is a multinomial bootstrap of A with roughly 37% of unique frames dropped in expectation. The $A \rightarrow A'$ drop of 0.018 on test mAP@50:95 therefore isolates the diversity cost of the multinomial resampling mechanism itself, independently of any compositional shift. This mechanism cost is quantified formally in the attribution decomposition below.

The per-class breakdown in Figures 5.7 and 5.8 reveals a class-asymmetric structure that the aggregate obscures. On the test set (Figures 5.7d, 5.8d), Condition C is the worst condition for scratch (mean mAP@50:95 0.151 vs. A at 0.200) but the best condition for dent (mean 0.236 vs. A at 0.231). C is the only condition that improves dent over the no-resampling baseline A. Condition B shows the opposite pattern: B is second-best for scratch and worst for dent among the resampled conditions.

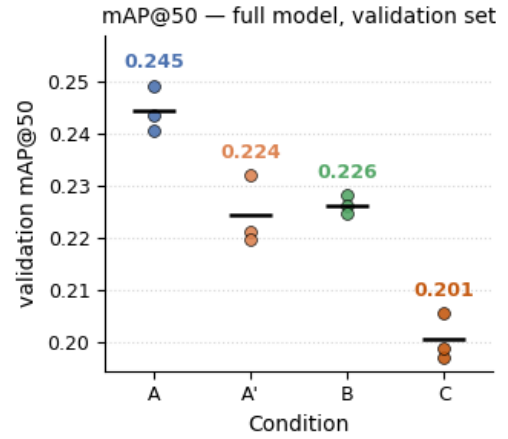
Condition C's aggregate deficit is driven entirely by its scratch performance degradation (-0.049 against A); the dent gain is real but too small to compensate ($+0.006$). Conditions B and C move dent and scratch in opposing directions, which points to their compositional structure as the operative mechanism.

Data Composition

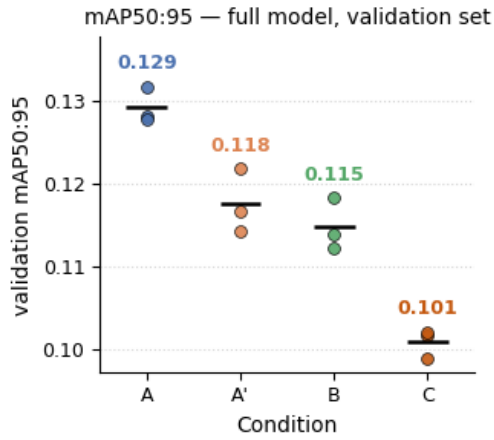
Figure 5.9 reports the realised training composition per stratum under the four conditions. Two observations bear on the interpretation. First, Conditions A and A' produce closely matched compositions across all nine strata, with per-stratum realised proportions of A' within 0.5 percentage points of A on every stratum. This confirms that the uniform-weight multinomial preserves stratum composition in expectation, so the $A \rightarrow A'$ performance drop cannot be attributed to compositional change and is instead attributable to the diversity loss of the multinomial mechanism. Second, the across-seed standard deviations for A', B, and C are small on every stratum (Standard Deviation (SD) < 0.005 throughout), so within-condition multinomial variance is negligible relative to across-condition compositional differences.



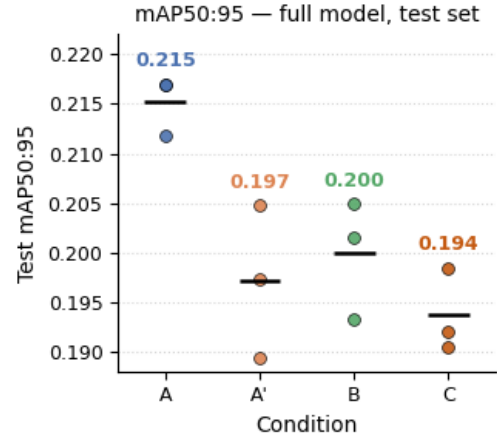
(a) mAP@50 on the proxy model, evaluated on the validation set.



(b) mAP@50 on the full model, evaluated on the validation set.

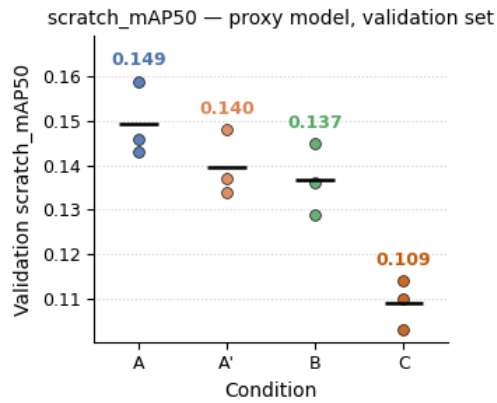


(c) mAP@50:95 on the full model, evaluated on the validation set.

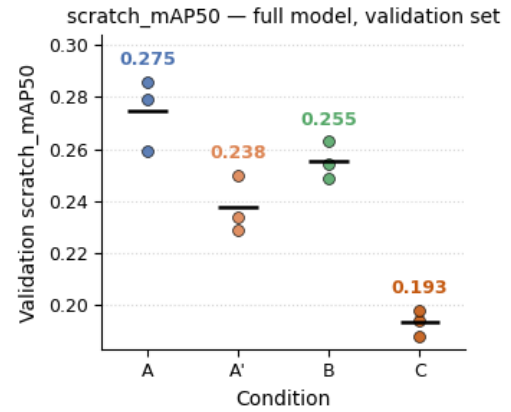


(d) mAP@50:95 on the full model, evaluated on the test set.

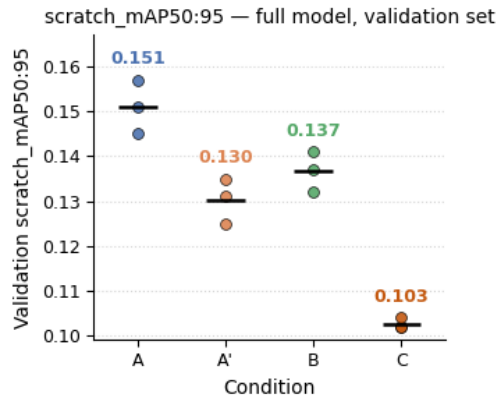
Figure 5.6: Aggregate performance across the four proxy/full and validation/test panels for Conditions A, A', B, and C. Three retrain seeds per condition; horizontal bars indicate the seed mean.



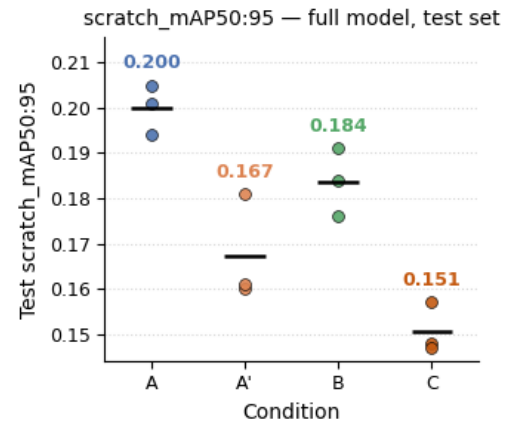
(a) Scratch mAP@50 on the proxy model, evaluated on the validation set.



(b) Scratch mAP@50 on the full model, evaluated on the validation set.

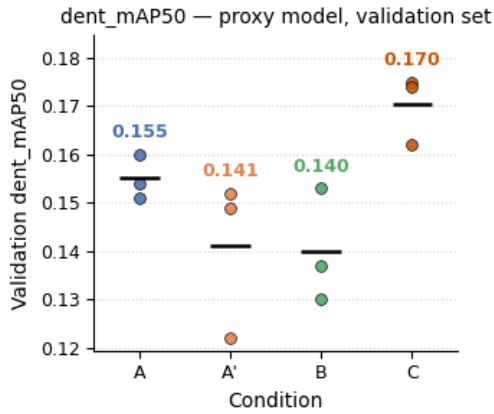


(c) Scratch mAP@50:95 on the full model, evaluated on the validation set.

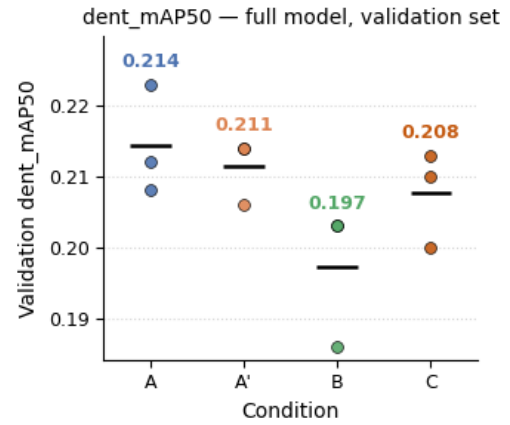


(d) Scratch mAP@50:95 on the full model, evaluated on the test set.

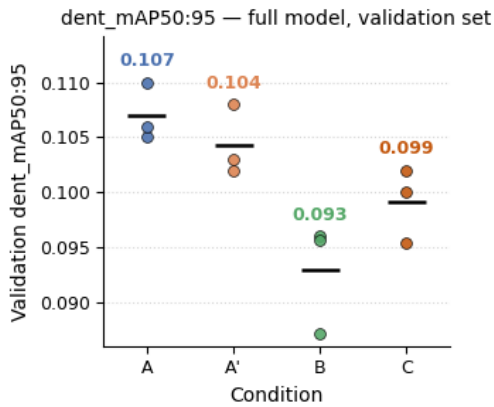
Figure 5.7: Four-panel breakdown for the scratch class, Conditions A, A', B, C. Three retrain seeds per condition; horizontal bars indicate the seed mean.



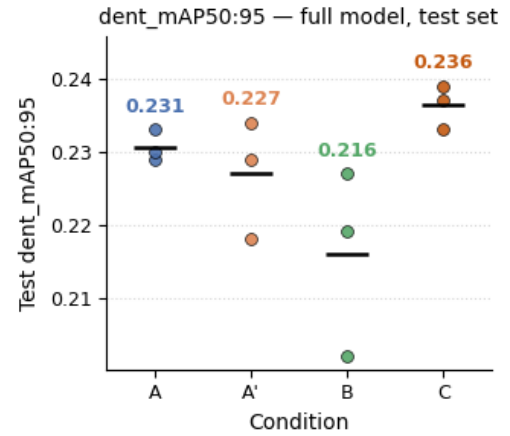
(a) Dent mAP@50 on the proxy model, evaluated on the validation set.



(b) Dent mAP@50 on the full model, evaluated on the validation set.



(c) Dent mAP@50:95 on the full model, evaluated on the validation set.



(d) Dent mAP@50:95 on the full model, evaluated on the test set.

Figure 5.8: Four-panel breakdown for the dent class, Conditions A, A', B, C. Three retrain seeds per condition; horizontal bars indicate the seed mean.

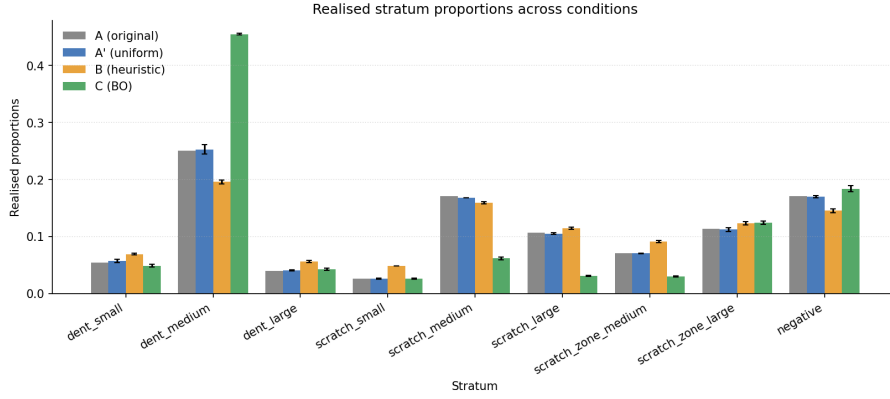


Figure 5.9: Training data composition under the different conditions.

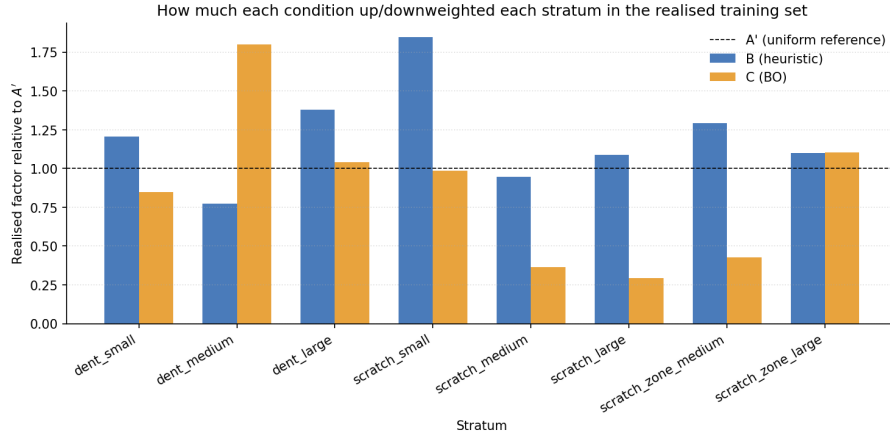


Figure 5.10: Upweight factor of Conditions B and C compared to the multinomial sampling baseline A'.

Figure 5.10 makes the compositional strategies of Conditions B and C directly comparable by normalising each realised proportion to the corresponding A' value. Condition C amplifies `dent_medium` by a factor of 1.80 relative to A', raising its share of the training set from 25.3% (A') to 45.5% (C). At the same time, `scratch_medium` falls to 0.37 $\times$ (from 16.8% to 6.1%), `scratch_large` to 0.29 $\times$ (from 10.5% to 3.1%), and `scratch_zone_medium` to 0.43 $\times$ (from 7.0% to 3.0%). Aggregated to the training-label level, C shifts total dent mass from 35.0% (A') to 54.5%, a 1.56 $\times$ increase, and combined scratch-label mass from 48.1% to 27.2%, a 0.57 $\times$ decrease. In effect, Condition C amplifies the stratum that is already dominant in the training data and starves the majority-class training signal.

Condition B moves in the opposite direction. Its heuristic, square-root inverse-frequency weighting, upweights the rare strata: `scratch_small` by 1.85 $\times$, `dent_large` by 1.38 $\times$, and `scratch_zone_medium` by 1.29 $\times$, while modestly downweighting `dent_medium` to 0.77 $\times$. B produces the classic textbook rebalancing pattern, whereas C produces its inverse.

The per-class performance pattern in Figures 5.7 and 5.8 is the direct image of these compositional shifts. Condition C's expanded dent training mass corresponds to its dent gain, and its reduced scratch training mass corresponds to its scratch loss. Condition B's mild scratch upweighting corresponds to its scratch gain, and its dent downweighting corresponds to its dent loss. The mechanism produces the intended compositional shift, and the composition controls per-class performance in the expected direction; the compositional trade encoded in Condition C is what proves net-harmful on aggregate.

Attribution Decomposition

To quantify the isolated effect of each mechanism step, Figure 5.11 decomposes the total production difference $A \rightarrow C$ into its four constituent single-step contrasts, plus the intermediate $B \rightarrow C$ comparison of BO-guided composition against the heuristic. Bootstrap 95% confidence intervals (Confidence Intervals (CIs)) are computed from the three seed-paired deltas per contrast, with $B = 10,000$ percentile resamples.

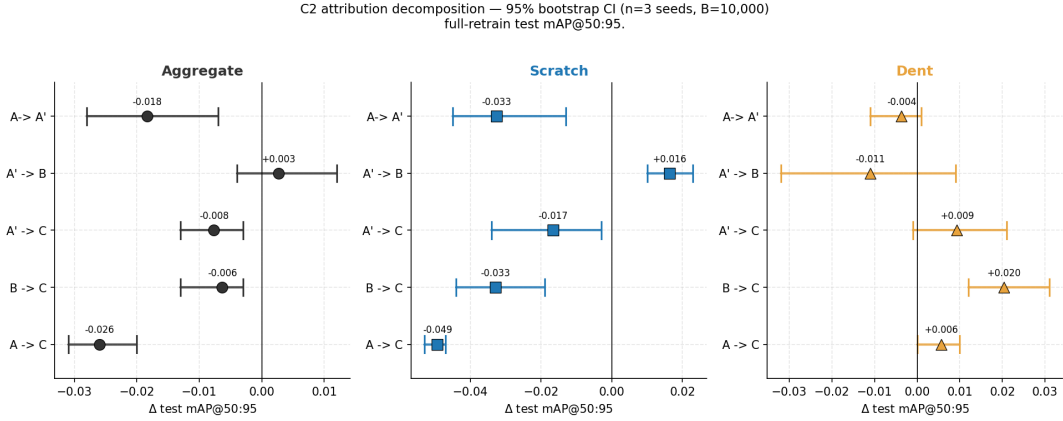


Figure 5.11: Attribution decomposition of test mAP@50:95 differences across Conditions A, A', B, C. Point estimates are seed means; horizontal bars are 95% bootstrap confidence intervals ($n = 3$ paired seeds, $B = 10,000$). Full numerical results in Appendix Table A.13

The $A \rightarrow A'$ contrast isolates the multinomial mechanism cost. On aggregate, the mean delta is -0.018 with 95% CI $[-0.028, -0.007]$, negative in 3 of 3 seeds. This is the largest single negative contribution in the decomposition, and it is driven almost entirely by scratch (-0.033 , CI $[-0.045, -0.013]$, 3/3 negative); the dent effect is small (-0.004 , CI $[-0.011, +0.001]$) and not seed-consistent. The multinomial diversity loss, which the mechanism design anticipated as a small cost, is in practice substantial and asymmetric across classes.

The $A' \rightarrow B$ contrast measures the effect of the heuristic reweighting relative to uniform resampling. On aggregate, the mean is $+0.003$ with CI $[-0.004, +0.012]$, with per-seed deltas of $+0.012$, -0.004 , and 0.000 : no directional consistency across seeds. Hence, there is no aggregate signal at $n = 3$. The scratch effect, in contrast, is directionally consistent and its values exceed zero: a mean of $+0.016$, CI $[+0.010, +0.023]$ and 3/3 seeds positive. The heuristic improves scratch specifically, consistent with its upweighting of the rare scratch strata; the dent effect is directionally negative (-0.011) with a CI that brackets zero.

The $A' \rightarrow C$ contrast measures the effect of BO-optimised weights relative to uniform resampling. On aggregate, the mean is -0.008 with CI $[-0.013, -0.003]$, 3/3 seeds negative: BO-guided composition performs worse than uniform resampling. The scratch loss is clear (-0.017 , CI $[-0.034, -0.003]$, 3/3 negative); the dent gain is directional, but its CI just touches zero ($+0.009$, CI $[-0.001, +0.021]$).

The $B \rightarrow C$ contrast measures whether the BO search outperforms the heuristic. On aggregate, it does not: mean -0.006 , CI $[-0.013, -0.003]$, 3/3 seeds negative. The class-level pattern is again opposing: scratch -0.033 , CI $[-0.044, -0.019]$, 3/3 negative; dent $+0.020$, CI $[+0.012, +0.031]$, 3/3 positive. Condition C is a decisively better bet than Condition B for dent alone and a decisively worse one for scratch and for the aggregate.

The $A \rightarrow C$ contrast reports the total production effect, that is, the outcome a deployment would see if BO-optimised resampling replaced no resampling. On aggregate, the mean is -0.026 with CI $[-0.031, -0.020]$, 3/3 seeds negative. The scratch component (-0.049 , CI $[-0.052, -0.045]$) accounts for essentially the whole aggregate loss; the dent gain ($+0.006$, CI $[+0.000, +0.010]$) is at best marginal in magnitude, and its CI clips zero at the lower bound.

Taken together, the decomposition shows that (i) the multinomial mechanism carries a class-asymmetric cost that is not neutral, (ii) the heuristic recovers some of that cost on scratch specifically, and (iii) Condition C's compositional optimisation produces a scratch-for-dent trade whose net effect on the production distribution is a clear loss. In no aggregate step of the decomposition does BO's optimisation contribute a positive delta.

5.2.2 Proxy-to-Full Transfer Fidelity for Data Optimisation and Training Hyperparameters (SQ2B)

To quantify the transfer effect for the data composition search, the proxy calibration ratio ρ is computed for Condition C on validation mAP@50, with A' as the baseline. As in Section 5.1.1, ρ is defined per seed as the ratio of full-retrain gain to proxy-retrain gain over the baseline and reported as the mean of three matched-seed ratios with 95% percentile bootstrap CIs. A' was chosen as the Condition C baseline because it is the counterfactual the compositional search improves upon: it retains the multinomial mechanism with uninformed weights, isolating the effect of the search itself. This mirrors the role of the untuned defaults baseline in the first contribution and preserves the calibration-of-optimisation interpretation for ρ .

Condition C is compared against Condition A, that is, the BO-tuned training hyperparameter configuration from Section 5.1.1. The Condition A per-class ρ values reported below are identical to those already reported in Table 5.1. Per-seed Condition C ratios and per-seed paired differences are reported in Appendix Tables A.11 and A.12.

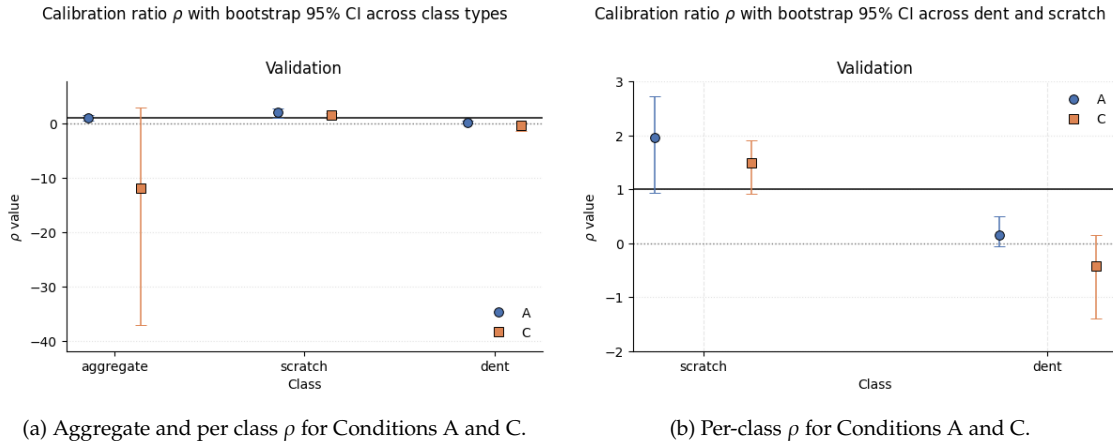


Figure 5.12: Proxy calibration ratio ρ for the training-hyperparameter search (Condition A, from E1) and the data composition search (Condition C, from E2), on validation mAP@50, with 95% bootstrap CI.

Table 5.3: Proxy-to-full calibration ratio ρ for the data composition search on validation mAP@50 (Condition C), and the within-seed paired difference against the training-hyperparameter search (Condition A), $\Delta\rho = \rho_C - \rho_A$. Condition A marginal ρ values are given in Table 5.1. Per-seed values in Appendix Tables A.11 and A.12. [†]The aggregate ρ and $\Delta\rho$ are not interpretable: the aggregate proxy gain is near zero, so the ratio is degenerate (Section 4.2.4). The values are shown for completeness only.

Damage	Condition C		$\Delta\rho = \rho_C - \rho_A$	
	Mean ρ	95% CI	Mean $\Delta\rho$	95% CI
Aggregate [†]	-11.934	[-37.056, 2.870]	-12.960	[-37.548, 1.803]
Scratch	1.492	[0.912, 1.917]	-0.462	[-1.819, 0.979]
Dent	-0.420	[-1.400, 0.160]	-0.565	[-1.337, 0.160]

The aggregate mean ρ for Condition C is -11.934 with CI [-37.056, +2.870]. Because ρ is computed per seed and then averaged, this value is not a single ratio but the mean of three: the per-seed aggregate proxy gains are -0.0122, +0.0097, and +0.0006 (shown here to four decimals to expose the near-zero third seed; elsewhere metrics are reported to three), and the near-zero

denominator of the third seed alone yields a per-seed ratio of approximately -37 (see Appendix Table [A.10](#)). The three-seed mean ρ is therefore dominated by a single degenerate denominator rather than describing a central tendency, which is why the bootstrap CI spans two orders of magnitude and the value is not interpretable at this sample size.

Separately, the smallness of the aggregate proxy gain is itself informative. Averaged across seeds the aggregate proxy delta is -0.001 , arising from C's scratch-for-dent trade producing near-cancelling per-class effects (mean per-class proxy deltas of -0.031 scratch and $+0.029$ dent). The objective BO optimised therefore carried little aggregate gradient over the region the search explored. The aggregate ρ is reported as evidence of this flatness rather than as a calibration measurement, and the analysis proceeds per class.

For scratch, $\rho_C = 1.492$ with CI $[0.912, 1.917]$. The proxy-observed scratch degradation transferred faithfully to full scale; ρ is positive but both deltas in the ratio are negative. The ratio is comparable in magnitude to ρ_A scratch (1.954 , CI $[0.938, 2.731]$); the marginal CIs overlap and do not distinguish the two at $n = 3$.

For dent, $\rho_C = -0.420$ with CI $[-1.400, +0.160]$. The negative sign indicates that the full-scale dent delta reversed sign relative to the proxy delta: the proxy-observed dent gain of $+0.029$ val mAP@50 did not merely fail to transfer to full scale but produced a loss on the validation dent class. This is a stronger form of transfer failure than the near-zero ρ_A dent value (0.146 , CI $[-0.063, +0.500]$), which reflected weak but same-signed transfer.

To leverage the seed pairing across the two conditions, the within-seed paired difference $\Delta\rho_s = \rho_{C,s} - \rho_{A,s}$ is computed per class. Because both ratios share the same training-hyperparameter backbone at retrain time (all Condition C retrains inherit the C1 BO-best hyperparameters, which are the very hyperparameters Condition A was optimised over) and are computed against retrains with matched training seeds, seed-driven full-retrain variance shared between the two ratios cancels in the paired difference. Per-seed values are reported in Appendix Table [A.12](#). For scratch, the mean paired difference is $\Delta\rho = -0.462$ with CI $[-1.819, +0.979]$, negative in 2 of 3 seeds; for dent, $\Delta\rho = -0.565$ with CI $[-1.337, +0.160]$, negative in 2 of 3 seeds. Both paired differences are directionally negative, but their bootstrap CIs bracket zero at $n = 3$, so the pairing does not resolve the marginal-CI overlap. The aggregate paired difference inherits the denominator degeneracy of the marginal aggregate ρ_C and is not interpretable.

Taken together, the paired analysis provides directional evidence that Condition C data-space sampling weights transfer no more faithfully than Condition A training hyperparameters, and for dent transfer less faithfully, but it does not distinguish the two at this sample size.

This section showed that BO selected a corner solution in the compositional search space (dent_medium dominant, most scratch strata suppressed) whose per-class trade did not transfer symmetrically, and whose aggregate proxy gain was too small to support an interpretable calibration ratio. The ρ pattern in Table [5.3](#) is a direct consequence of that solution: a scratch loss that transfers cleanly, a dent gain that reverses at full scale, and an aggregate ratio whose denominator is within the noise floor and therefore not interpretable.

Discussion

Following the results, this chapter seeks to explain the noted observations, discuss limitations, and identify opportunities for further research.

6.1 Interpretation of the findings and limitations

The aim of this thesis was to test the feasibility of proxy-constrained optimisation across two parameter spaces and to quantify the extent of transfer through a newly designed metric, the calibration ratio.

The proxy framework was motivated by the time bottleneck that constrains conventional HPO in production settings. The extent to which this feasibility comes at the cost of outcome quality is the central question the results address.

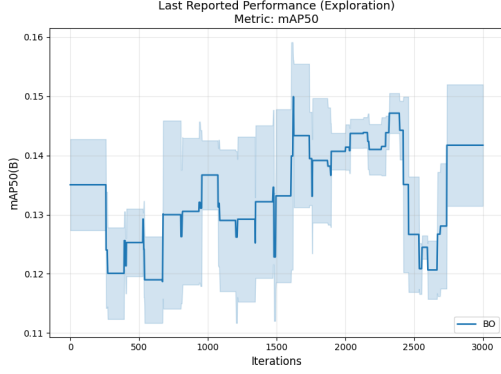
Overall, an aggregate improvement in test performance was observed when optimising the training hyperparameters, Experiment 1 (E1), in the proxy setting, for both choices of optimisation algorithm. In contrast, the data composition optimisation, Experiment 2 (E2), produced no improvement in the test setting, neither over the unsampled baseline (Condition A) nor over the sampling baseline (Condition A').

For E1, the aggregate gain was primarily scratch-driven: scratch realised a larger performance gain over baseline than dent. For E2 the opposite held; a disproportionate performance loss for scratch dragged the across-class average down to an aggregate degradation relative to both baselines, while dent improved only marginally.

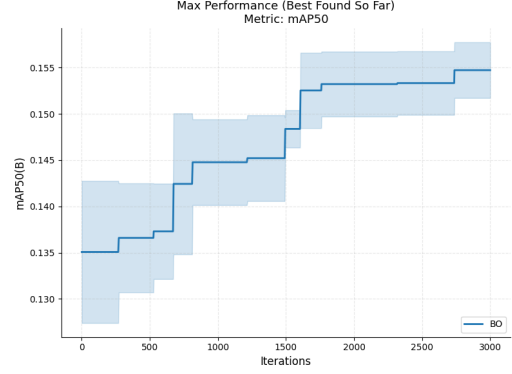
In both cases an asymmetric class outcome is observed, indicating that the best configuration from the proxy search converged to favour a single class. Since aggregate mAP@50 is the unweighted mean of the two per-class APs, this asymmetry is not a property of class proportion but of the magnitude and sign of the per-class changes. The unifying mechanism was the pursuit of marginal returns over a scratch-dominant training landscape. In E1, the configurations steered toward accruing gains to the majority class that supplies most of the gradient signal, and because that gain was additive, scratch improved without dent having to worsen, so the equal-weight mean remained positive. In E2, the marginal proxy returns pointed toward the under-trained dent stratum, but because fixed- N resampling makes composition zero-sum, the dent gain came at the direct expense of scratch mass, and the equal-weight mean turned negative. This exposes the central difficulty of optimising over an aggregate metric: it cannot distinguish a real, transferable gain in one class from a zero-sum trade that cancels across classes.

This difficulty stands on its own, but its effect may have been exacerbated by the search budget constraint. To respect the time-practicality motivation of this research, a search budget of 20 evaluations was adopted. However, the search trajectories for both parameter spaces (Figures 5.5 and 6.1) show no definite signs of convergence. In both cases, and especially for E2 (Figure 6.1b), the maximum reported performance is still climbing at the end of the search. This is consistent with the default TPE configuration, in which the first 10 evaluations are random startup trials used to establish an initial model of the search space (Akiba et al., 2019), meaning over half the budget was spent on pure exploration before the surrogate engaged. While convergence was not

the intention, the aim was to assess whether meaningful optimisation could be achieved within the budget. The evidence suggests instead that the best solutions for both parameter spaces may reflect chance rather than strategic optimisation over a meaningfully explored space.



(a) Last-reported mAP@50 performance for BO across three search runs.



(b) Maximum reported mAP@50 performance for BO across three search runs.

Figure 6.1: Performance across the search trajectory for the BO data composition optimisation across three seeds. The bold line is the mean; the shaded band indicates the standard deviation.

That the non-convergence was more detrimental for E2 may be twofold. First, its parameter space was more than twice the size of the E1 space (nine parameters versus four), while the budget was identical. As the number of possible configurations grows, the budget should grow correspondingly, and in particular should exceed the random-startup period by a sufficient margin to allow the surrogate to explore the space. Second, E2 operated on the data indirectly, with mean frame-weight aggregation diluting the intended per-stratum weight effect. Because the *dent_medium* stratum already held the highest proportion in the training set, its amplification most likely produced the strongest mAP differential per unit of weight, biasing the search toward a corner solution that concentrated on that stratum.

A comparable effect appears in E1, where the higher-performing proxy configuration identified by ABOHB was most likely the product of selection on noisy maxima rather than a systematically superior configuration.

Calibration ratio

The calibration ratio was introduced to quantify the transfer effect across both parameter spaces under the proxy framework.

The results indicate, however, that especially at the aggregate level firm conclusions were difficult to draw because of high variability. This was exacerbated by small denominator (proxy gain) values, as reported in Section 5.2.2. Because ρ is a ratio of small differences of noisy quantities, its variance is dominated by baseline full-retrain stochasticity and grows without bound as the denominator approaches the noise floor.

At the per-class level the ratio was more informative. It identified more faithful transfer for scratch in both parameter spaces, and, for *dent*, weak transfer in E1 and sign reversal in E2.

Scratch transferred more faithfully in both studies because it is the majority training class (roughly 60% of training ROIs) and its representation is preserved across the proxy and full scales; what the proxy learns about scratch detection therefore scales up predictably. The *dent* failures, by contrast, are not the same failure. In E1, *dent* transfer is weak because of the validation-specific size-distribution mismatch established in the data audit: validation *dent* carries a heavier 10cm+ tail on which detection is poorer, so proxy *dent* gains do not translate cleanly to the validation distribution. In E2, the *dent* gain was a corner-solution artefact selected against a near-flat aggregate proxy objective, and it reversed at full scale. The shared observation, faithful

scratch transfer, thus most likely has a single cause, while the dent failures appear to have distinct causes across the two studies.

For the comparison of algorithms, per-seed differential pairing was used, which cancels the shared baseline term. From this, transfer under BO was found to be directionally more faithful than under ABOHB on the validation set, consistent with a winner’s-curse effect in which the broader search of ABOHB inflates a noisy proxy maximum whose noise component does not survive to full retrain. At $n = 3$, and with the effect not detectable on the test set, this is directional evidence rather than established superiority.

Precisely characterising the fidelity of cheap search, however, requires many full retrains, as the current high variance shows. Reducing that variance would require additional expensive full retrains, so the cost of the calibration instrument scales with exactly the budget the proxy framework was designed to avoid. This is a structural limitation of the calibration-ratio paradigm rather than of its application here.

Mechanism bias

Finally, as anticipated, there was a cost associated with the bootstrap effect of the multinomial sampling mechanism, which on its own accounted for a -0.018 change in test mAP@50:95 . Buda, Maki and Mazurowski (2018) found that oversampling does not cause overfitting in CNNs, but that result rests on full data retention. Multinomial-to-fixed- N instead drops roughly 37% of unique frames in expectation. At proxy scale this compounds: beginning from only 25% of the original dataset, a further reduction to roughly 63% of unique frames leaves the model training on approximately 16% of the full dataset. Reduced diversity is particularly detrimental to generalisation, an effect documented empirically for CNN image tasks (Barbedo, 2018), and its impact is therefore amplified in the already data-scarce proxy regime, where it may additionally have degraded the quality of the search itself.

Optimisation versus time trade-off

The proxy framework was designed to yield a hyperparameter optimisation pipeline feasible in wall-clock time; the proxy model ran in approximately 2 hours against approximately 16 for the full retrain.

The framework works when the proxy response exceeds the run-to-run noise floor and is directionally stable across scales. The preliminary learning-rate experiment (Section 4.4.3) suggested that this held for a single-parameter search, though as a single-axis, single-seed sweep it is indicative rather than conclusive.

It appears to work less well for multi-parameter, interacting search spaces whose per-parameter importance and interactions may shift between proxy and full scale. The E1 search sat toward the tractable end of this spectrum, whereas the nine-parameter E2 search sat toward the difficult end.

The stratum framework, in this light, was not given a fair test. It ran under an aggregate objective that could not surface its per-class effects, over a mechanism-biased surface, and within a budget half-consumed by random startup. Whether sub-type strata are an effective optimisation unit therefore remains an open question, pending a per-class objective and an adequate budget. This open question concerns the specific instantiation tested here, not data-composition optimisation as a class of methods; the cross-space transfer comparison the study was designed to make (SQ1b) was itself inconclusive at three seeds, and is treated as such throughout.

6.2 Practical Implications and Recommendations

The findings of this thesis translate into three recommendations for Lensor. They are ordered by immediacy and by the strength of the evidence supporting them: the first is a prerequisite for the other two, the second and third are methodological directions that should only be pursued once the first is in place. Throughout, a distinction is maintained between confidence in *what to do* and

the strength of the underlying evidence: with three retrain seeds, the comparisons in this thesis are directional and descriptive rather than statistically conclusive, and the recommendations are scoped accordingly.

1. Align the validation distribution to the production reference before further optimisation.

The data audit established that the validation set diverged from the test set on negative frame rate and on the within-class size distribution of the dent class (Section 5.1.1). This divergence is not a peripheral issue: it is the mechanism to which the ambiguous dent transfer in E1 was traced, and it is part of why the aggregate composition signal in E2 was difficult to interpret. Because the validation set is the signal every proxy search optimises against, a validation set that does not reflect the production distribution steers the search toward configurations that perform well on an unrepresentative target. Partial remediation was performed here (matching the negative frame rate) but the class-ratio gap was left unresolved. The first recommendation is therefore to complete this remediation, aligning the validation distribution to the production reference on both axes, before any further hyperparameter or data-composition search is undertaken. Every subsequent recommendation depends on this foundation.

2. Treat proxy-guided hyperparameter search as a demonstrated mechanism whose applicability to the production regime remains to be validated.

E1 provides evidence that proxy-constrained hyperparameter optimisation yields configurations that improve full-scale test performance over untuned defaults in all three retrain seeds, with faithful transfer for the majority (scratch) class and unreliable transfer for the minority (dent) class. This establishes the transfer *mechanism*: improvements found cheaply on a reduced proxy can carry to full scale. The improvement is measured relative to the AdamW thesis baseline, not relative to Lensor’s production model, which uses the MuSGD optimiser and, as Appendix A.1 shows, scores substantially higher than the AdamW baseline used throughout this thesis. Whether proxy-guided search yields comparable gains in the production optimiser regime is untested, and this regime gap, rather than hardware alone, is the primary barrier to direct deployment. Two conditions would need to be met to close it: the proxy would need to be run at, or validated against, production resolution and batch size, which under current hardware trades off against the compute saving the proxy exists to provide; and the search would need to be repeated under the MuSGD optimiser to confirm the transfer holds outside the AdamW regime. The recommendation is therefore to treat proxy-guided search as a validated and promising method whose transfer to the production configuration is the next thing to establish.

3. Revisit data-composition optimisation only with a multi-objective formulation, a larger search budget, and a remediated validation set.

E2 returned a diagnosed negative result: Bayesian-optimised sampling weights did not improve full-scale performance, and the calibration ratio did not exceed that of the training-hyperparameter search. The diagnosis identifies three specific conditions under which the approach could be revisited more fairly, each following directly from a cause of the observed failure. First, the aggregate objective could not, by construction, distinguish a genuine improvement from a class trade that nets to zero, and the search converged on exactly such a trade. Because fixed- N resampling makes composition a zero-sum reallocation of training mass, a multi-objective or Pareto formulation is not a refinement but a requirement: it is the only way for the search to see the per-class trade it is making rather than optimise blindly over an averaged signal. Second, the nine-weight space searched at twenty evaluations, with the first ten spent on random startup, left the surrogate little opportunity to concentrate density; a budget scaled to the dimensionality of the composition space would be needed to search it meaningfully. Third, the flatness of the aggregate signal was itself partly an artefact of the validation set, so recommendation 1 is a prerequisite here as well. The recommendation is not to abandon data-composition optimisation but to revisit it only once these three conditions are met; the negative result reported here is a property of this specific instantiation, not a verdict on the broader approach.

Taken together, these recommendations describe a sequence rather than a set of parallel options: remediate the measurement foundation, then build on it. The measurement fix is established and immediate; the two methodological directions are promising but conditional, and neither should be pursued on the current, partially misaligned validation set.

6.3 Limitations and Future Research

Limitations

The claims of this thesis are bound by two main limitations, both discussed where they arose and summarised here.

All comparisons rest on three retrain seeds, a deliberate trade against the compute cost of full retrains. The bootstrap confidence intervals reported throughout are therefore descriptive summaries of per-seed dispersion rather than a basis for formal inference: at three observations, a percentile bootstrap resamples the same three values, so its interval is bounded by their range. Findings are accordingly framed in terms of directional consistency and effect size, with a distinction drawn between unanimous seed agreement, reported as directional evidence, and a bare majority, reported as a lean but not a finding.

The calibration ratio was informative per class but unstable at the aggregate level whenever the proxy gain approached the run-to-run noise floor (Section 6.1). This instability bounds the aggregate transfer claims and is the reason the analysis proceeds per class throughout.

Future Research

The clearest direction this thesis opens is to establish more precisely when the proxy framework yields an interpretable transfer measurement and when it does not. The two studies provide contrasting cases: the four-parameter hyperparameter search produced a clean transfer signal, faithful for the majority class, whereas the nine-parameter composition search produced an aggregate signal too flat to interpret and a per-class comparison that was inconclusive at three seeds. The contrast suggests that the framework’s usefulness depends on whether the proxy gain clears the run-to-run noise floor at the optimised objective; where it does not, transfer cannot be measured, let alone confirmed. Two cases cannot establish where that boundary lies in general, and several directions for testing it follow directly from the present work.

Optimiser sensitivity. Beyond the recommendation to validate the framework under Lensor’s production MuSGD optimiser, the effect of optimiser choice on proxy transfer is itself an open question. Whether different optimisers are better suited at different scales would both delineate the edges of the proxy framework’s viability and inform the broader relationship between scale and optimiser choice.

Intermediate parameter spaces. This thesis compared two parameter spaces: training hyperparameters (E1) and data composition (E2). A natural intermediate is data augmentation, which belongs to model training but acts directly on the data to improve generalisation. Crucially, augmentation avoids the two properties that complicated E2: it applies to every image without reallocating a fixed budget, and it does not dilute control through frame-level aggregation. Testing whether augmentation parameters transfer under the proxy framework would help isolate whether the difficulty seen in E2 is a property of data-space optimisation in general or of the fixed- N resampling mechanism specifically.

Estimators of transfer fidelity. The instability of the calibration ratio near a small denominator is not only a limitation but a prompt. Future work could develop estimators of transfer fidelity that degrade more gracefully as the proxy gain shrinks, or reporting protocols that pair the ratio with its denominator so its reliability is always visible. A practical rule already follows from

the present findings: report the ratio per class, and treat an aggregate ratio as uninterpretable whenever the aggregate proxy gain lies within the noise floor of a single retrain.

Conclusion

This thesis set out to test the feasibility of compute-constrained proxy optimisation for car damage detection across two parameter spaces, and to quantify the resulting transfer through a purpose-built metric, the calibration ratio. The main research question asked: *to what extent do configurations optimised via a compute-constrained proxy model transfer to full-scale detection models, and does optimising training data composition under the same proxy constraints provide more reliable transfer than optimising training hyperparameters?*

One finding frames the answers to both parts. The validation set diverged from the test set on negative frame rate and on the within-class size distribution of the dent class, and because the validation set is the signal every proxy search optimises against, this divergence propagated into both experiments and bounds what any transfer measurement here can conclude. It is a primary result rather than a peripheral limitation: a transfer study is only as reliable as the alignment between the distribution it optimises on and the distribution it is judged against.

On the first part of the research question, proxy-optimised training hyperparameters do transfer. Configurations found by both Bayesian optimisation and ABOHB on the 25% proxy improved full-scale test mAP@50:95 over the untuned baseline in all three seeds. The transfer was not uniform across classes: it was carried by the scratch class, whose proxy gains scaled up predictably because it is the majority training class with a representation preserved between proxy and full scale, while dent transfer was weak and validation-specific, traceable to the within-class size-distribution mismatch between the evaluation sets. Transfer therefore holds, but its reliability is class-dependent, and the aggregate calibration ratio conceals this structure while itself being too imprecise to support a magnitude claim.

On the second part, a distinction must be drawn between two questions the study posed. The first is whether the specific composition pipeline improved performance; the second is whether data-space parameters transfer more faithfully than training-space parameters. On the first, the answer is a firm negative: the Bayesian-optimised composition degraded aggregate test performance relative to both the unsampled and uniform-sampled baselines, and this negative result was fully diagnosed. The search converged to a corner solution that amplified the already-dominant `dent_medium` stratum at the direct expense of scratch training mass, a reallocation that fixed- N resampling makes zero-sum and that an aggregate objective cannot see. On the second question, however, the study is inconclusive. The per-seed paired comparison of calibration ratios was directionally negative in only two of three seeds, with bootstrap intervals bracketing zero, so it does not establish that data-space parameters transfer either more or less faithfully than training-space parameters. This inconclusiveness is not incidental. The composition search operated over a nine-parameter space, more than twice the dimensionality of the hyperparameter search, under an identical budget half-consumed by random startup. The aggregate proxy gains it found were small relative to the run-to-run variation of a single retrain and inconsistent in sign across seeds, so the signal available to guide the search was weak; whether this reflects a genuinely uninformative objective or a search that never reached a more informative region cannot be determined at this budget. The hyperparameter search therefore operated under more favourable conditions than the composition search, and the observed difference reflects those conditions at least as much as any intrinsic property of the parameter spaces. Whether data-

space parameters are inherently more or less transferable remains open, pending a per-class objective and a budget matched to the dimensionality of the search. Critically, this is a negative result for a specific instantiation of composition optimisation, not evidence that data-composition optimisation transfers less faithfully than hyperparameter optimisation as a class of methods.

The calibration ratio, introduced to answer precisely this comparison, was itself only partly able to. At the per-class level it was informative, distinguishing faithful transfer from weak transfer and from sign reversal. At the aggregate level its confidence intervals were too wide to support magnitude claims, because the ratio divides small, noisy full-retrain gains and its variance grows without bound as the denominator approaches the noise floor. Reducing that variance would require additional full retrains, so the instrument’s cost scales with exactly the budget the proxy framework exists to avoid. This tension, that precisely characterising cheap search is expensive, is a structural property of the metric rather than a shortcoming of its application, and constitutes a finding in its own right.

The contributions of this thesis are threefold. First, it demonstrates that proxy-constrained hyperparameter optimisation produces configurations that transfer usefully to full-scale training within the tested optimiser regime, with transfer that is faithful for the majority class and unreliable for the minority class. Second, it introduces the calibration ratio as a diagnostic for proxy-to-full transfer, characterises its behaviour including its high variance at small seed counts, and shows that per-class reporting is necessary because aggregate ratios mask class-asymmetric transfer. Third, it provides a fully diagnosed negative result for data composition optimisation: rather than a bare failure, the outcome is decomposed into a quantified mechanism cost from the multinomial bootstrap, a corner solution induced by an aggregate objective over a mechanism-biased surface, and a startup-dominated search budget, each of which is separately actionable. The attribution decomposition that isolates these effects, and the observation that fixed- N resampling renders compositional optimisation zero-sum across classes, are transferable beyond this dataset.

These findings rest on three retrain seeds, which supports directional and effect-size claims but not significance testing, so the bootstrap intervals reported throughout are descriptive summaries of per-seed dispersion rather than a basis for formal inference. The further limitations that bound these results, the deliberate validation-test divergence retained to preserve comparability between the two studies, the AdamW optimiser regime, and the conditions under which the composition search was run, are set out in Section 6.3.

Taken together, this thesis shows that compute-constrained proxy optimisation is a viable route to feasible hyperparameter tuning for industrial car damage detection, that its value is real but bounded by whether the proxy signal clears the run-to-run noise floor at the optimised objective, and that where the framework did not yield a usable result, applied to training data composition, the reasons were specific and diagnosable rather than opaque. The proxy model is a practical instrument for the time-constrained production setting it was built for; establishing where it works, and where it does not, is the contribution this work makes.

References

- A Ilemobayo, Justus et al. (June 2024). "Hyperparameter Tuning in Machine Learning: A Comprehensive Review". en. In: *Journal of Engineering Research and Reports* 26.6, pp. 388–395. ISSN: 2582-2926. DOI: [10.9734/jerr/2024/v26i61188](https://doi.org/10.9734/jerr/2024/v26i61188) URL: <https://journaljerr.com/index.php/JERR/article/view/1188> (visited on 20/11/2025).
- Akiba, Takuya et al. (July 2019). "Optuna: A Next-generation Hyperparameter Optimization Framework". en. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Anchorage AK USA: ACM, pp. 2623–2631. ISBN: 978-1-4503-6201-6. DOI: [10.1145/3292500.3330701](https://doi.org/10.1145/3292500.3330701) URL: <https://dl.acm.org/doi/10.1145/3292500.3330701> (visited on 20/04/2026).
- Barbedo, Jayme Garcia Arnal (Oct. 2018). "Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification". en. In: *Computers and Electronics in Agriculture* 153, pp. 46–53. ISSN: 01681699. DOI: [10.1016/j.compag.2018.08.013](https://doi.org/10.1016/j.compag.2018.08.013) URL: <https://linkinghub.elsevier.com/retrieve/pii/S0168169918304617> (visited on 03/07/2026).
- Bengio, Yoshua (2012). "Practical Recommendations for Gradient-Based Training of Deep Architectures". en. In: *Neural Networks: Tricks of the Trade*. Ed. by Grégoire Montavon, Geneviève B. Orr and Klaus-Robert Müller. Vol. 7700. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 437–478. ISBN: 978-3-642-35288-1 978-3-642-35289-8. DOI: [10.1007/978-3-642-35289-8_26](https://doi.org/10.1007/978-3-642-35289-8_26) URL: http://link.springer.com/10.1007/978-3-642-35289-8_26 (visited on 08/04/2026).
- Bergstra, James, Rémi Bardenet et al. (2011). "Algorithms for Hyper-Parameter Optimization". In: *Advances in Neural Information Processing Systems*. Ed. by J. Shawe-Taylor et al. Vol. 24. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf
- Bergstra, James and Yoshua Bengio (2012). "Random Search for Hyper-Parameter Optimization". en. In: *Journal of Machine Learning Research* 13, pp. 281–305.
- Bergstra, James, Daniel Yamins and David Cox (June 2013). "Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures". en. In: *ICML'13: Proceedings of the 30th International Conference on International Conference on Machine Learning* 28, pp. 115–123.
- Bhosale, Madhura M. and Yogesh S. Angal (Feb. 2025). "Performance Analysis of Momentum of Adam Optimizer on YOLO-V8 Using Traffic Object Dataset". en. In: *Traitement du Signal* 42.1, pp. 505–518. ISSN: 07650019, 19585608. DOI: [10.18280/ts.420143](https://doi.org/10.18280/ts.420143) URL: <https://www.iieta.org/journals/ts/paper/10.18280/ts.420143> (visited on 08/04/2026).
- Bishop, Christopher (2006). *Pattern recognition and machine learning*. en. Information science and statistics. New York: Springer. ISBN: 978-0-387-31073-2.
- Bochkovskiy, Alexey, Chien-Yao Wang and Hong-Yuan Mark Liao (2020). "YOLOv4: Optimal Speed and Accuracy of Object Detection". en. In: DOI: [10.48550/arXiv.2004.10934](https://arxiv.org/abs/10.48550/arXiv.2004.10934)
- Bouthillier, Xavier et al. (2021). "Accounting for Variance in Machine Learning Benchmarks". In: *Proceedings of Machine Learning and Systems*. Ed. by A. Smola, A. Dimakis and I. Stoica. Vol. 3, pp. 747–769. URL: https://proceedings.mlsys.org/paper_files/paper/2021/file/0184b0cd3cfb185989f858a1d9f5c1eb-Paper.pdf

- Buda, Mateusz, Atsuto Maki and Maciej A. Mazurowski (Oct. 2018). "A systematic study of the class imbalance problem in convolutional neural networks". en. In: *Neural Networks* 106, pp. 249–259. ISSN: 08936080. DOI: [10.1016/j.neunet.2018.07.011](https://doi.org/10.1016/j.neunet.2018.07.011) URL: <https://linkinghub.elsevier.com/retrieve/pii/S0893608018302107> (visited on 18/06/2026).
- Cawley, Gavin C and Nicola L C Talbot (2010). "On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation". en. In: *Journal of Machine Learning Research* 11.
- ClearML (2024). URL: <https://clear.ml/>
- Coleman, Cody et al. (2020). "SELECTION VIA PROXY: EFFICIENT DATA SELECTION FOR DEEP LEARNING". en. In: *ICLR 2020*. DOI: <https://doi.org/10.48550/arXiv.1906.11829>
- DiCiccio, Thomas J. and Bradley Efron (Sept. 1996). "Bootstrap confidence intervals". In: *Statistical Science* 11.3. ISSN: 0883-4237. DOI: [10.1214/ss/1032280214](https://doi.org/10.1214/ss/1032280214) URL: <https://projecteuclid.org/journals/statistical-science/volume-11/issue-3/Bootstrap-confidence-intervals/10.1214/ss/1032280214.full> (visited on 08/06/2026).
- Everingham, Mark et al. (June 2010). "The Pascal Visual Object Classes (VOC) Challenge". en. In: *International Journal of Computer Vision* 88.2, pp. 303–338. ISSN: 0920-5691, 1573-1405. DOI: [10.1007/s11263-009-0275-4](https://doi.org/10.1007/s11263-009-0275-4) URL: <http://link.springer.com/10.1007/s11263-009-0275-4> (visited on 15/04/2026).
- Falkner, Stefan, Aaron Klein and Frank Hutter (2018). "BOHB: Robust and Efficient Hyperparameter Optimization at Scale". en. In.
- Ge, Zheng et al. (Aug. 2021). YOLOX: Exceeding YOLO Series in 2021. en. arXiv:2107.08430 [cs.CV]. DOI: [10.48550/arXiv.2107.08430](https://doi.org/10.48550/arXiv.2107.08430) URL: <http://arxiv.org/abs/2107.08430> (visited on 10/07/2026).
- Glenn, Jocher and Jing Qiu (2026). Ultralytics YOLO26. URL: <https://github.com/ultralytics/ultralytics>
- Gupta, Agrim, Piotr Dollar and Ross Girshick (June 2019). "LVIS: A Dataset for Large Vocabulary Instance Segmentation". en. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA: IEEE, pp. 5351–5359. ISBN: 978-1-7281-3293-8. DOI: [10.1109/CVPR.2019.00550](https://doi.org/10.1109/CVPR.2019.00550) URL: <https://ieeexplore.ieee.org/document/8954457/> (visited on 23/06/2026).
- Hoang, Van-Ha, Jong Weon Lee and Chun-Su Park (2025). "Enhancing Fire Detection with YOLO Models: A Bayesian Hyperparameter Tuning Approach". en. In: *Computers, Materials & Continua* 83.3, pp. 4097–4116. ISSN: 1546-2226. DOI: [10.32604/cmc.2025.063468](https://doi.org/10.32604/cmc.2025.063468) URL: <https://www.techscience.com/cmc/v83n3/61047> (visited on 18/03/2026).
- Huang, Chen et al. (June 2016). "Learning Deep Representation for Imbalanced Classification". en. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, pp. 5375–5384. ISBN: 978-1-4673-8851-1. DOI: [10.1109/CVPR.2016.580](https://doi.org/10.1109/CVPR.2016.580) URL: <https://ieeexplore.ieee.org/document/7780949/> (visited on 23/03/2026).
- Hutter, Frank, Lars Kotthoff and Joaquin Vanschoren, eds. (2019). *Automated Machine Learning: Methods, Systems, Challenges*. en. The Springer Series on Challenges in Machine Learning. Cham: Springer International Publishing. ISBN: 978-3-030-05317-8 978-3-030-05318-5. DOI: [10.1007/978-3-030-05318-5](https://doi.org/10.1007/978-3-030-05318-5) URL: <http://link.springer.com/10.1007/978-3-030-05318-5> (visited on 19/03/2026).
- Jakubik, Johannes et al. (Aug. 2024). "Data-Centric Artificial Intelligence". en. In: *Business & Information Systems Engineering* 66.4, pp. 507–515. ISSN: 2363-7005, 1867-0202. DOI: [10.1007/s12599-024-00857-8](https://doi.org/10.1007/s12599-024-00857-8) URL: <https://link.springer.com/10.1007/s12599-024-00857-8> (visited on 24/02/2026).
- James, Gareth et al. (2013). *An Introduction to Statistical Learning*. en. Vol. 103. Springer Texts in Statistics. New York, NY: Springer New York. ISBN: 978-1-4614-7137-0 978-1-4614-7138-7. DOI: [10.1007/978-1-4614-7138-7](https://doi.org/10.1007/978-1-4614-7138-7) URL: <http://link.springer.com/10.1007/978-1-4614-7138-7> (visited on 25/03/2026).
- Jamieson, Kevin and Ameet Talwalkar (2016). "Non-stochastic Best Arm Identification and Hyperparameter Optimization". en. In: *Journal of Machine Learning Research* 51.

- Jocher, Glenn et al. (2026). *Ultralytics YOLO26: Unified Real-Time End-to-End Vision Models*. Version Number: 1. DOI: [10.48550/ARXIV.2606.03748](https://doi.org/10.48550/ARXIV.2606.03748), URL: <https://arxiv.org/abs/2606.03748> (visited on 10/07/2026).
- Johnson, Olanrewaju V. et al. (2024). "Learning Rate Schedules and Optimizers, A Game Changer for Deep Neural Networks". en. In: *Advances in Intelligent Computing Techniques and Applications*. Ed. by Faisal Saeed, Fathey Mohammed and Yousef Fazea. Vol. 210. Series Title: Lecture Notes on Data Engineering and Communications Technologies. Cham: Springer Nature Switzerland, pp. 327–340. ISBN: 978-3-031-59710-7 978-3-031-59711-4. DOI: [10.1007/978-3-031-59711-4_28](https://doi.org/10.1007/978-3-031-59711-4_28), URL: https://link.springer.com/10.1007/978-3-031-59711-4_28 (visited on 05/06/2026).
- Jones, Donald R and Matthias Schonlau (1998). "Efficient Global Optimization of Expensive Black-Box Functions". en. In: 13, pp. 455–492.
- Kaur, Harsurinder, Husanbir Singh Pannu and Avleen Kaur Malhi (July 2020). "A Systematic Review on Imbalanced Data Challenges in Machine Learning: Applications and Solutions". en. In: *ACM Computing Surveys* 52.4, pp. 1–36. ISSN: 0360-0300, 1557-7341. DOI: [10.1145/3343440](https://doi.org/10.1145/3343440), URL: <https://dl.acm.org/doi/10.1145/3343440> (visited on 23/03/2026).
- Kaur, Ravpreet and Sarbjee Singh (Jan. 2023). "A comprehensive review of object detection with deep learning". en. In: *Digital Signal Processing* 132, p. 103812. ISSN: 10512004. DOI: [10.1016/j.dsp.2022.103812](https://doi.org/10.1016/j.dsp.2022.103812), URL: <https://linkinghub.elsevier.com/retrieve/pii/S1051200422004298> (visited on 14/04/2026).
- Krawczyk, Bartosz (Nov. 2016). "Learning from imbalanced data: open challenges and future directions". en. In: *Progress in Artificial Intelligence* 5.4, pp. 221–232. ISSN: 2192-6352, 2192-6360. DOI: [10.1007/s13748-016-0094-0](https://doi.org/10.1007/s13748-016-0094-0), URL: <http://link.springer.com/10.1007/s13748-016-0094-0> (visited on 04/03/2026).
- Krizhevsky, Alex, Ilya Sutskever and Geoffrey E. Hinton (May 2017). "ImageNet classification with deep convolutional neural networks". en. In: *Communications of the ACM* 60.6, pp. 84–90. ISSN: 0001-0782, 1557-7317. DOI: [10.1145/3065386](https://doi.org/10.1145/3065386), URL: <https://dl.acm.org/doi/10.1145/3065386> (visited on 20/03/2026).
- Kushner, Harold (1964). "A New Method of Locating the Maximum Point of an Arbitrary Multi-peak Curve in the Presence of Noise". In: *Journal of Basic Engineering* 86.1, pp. 97–106. DOI: <https://doi.org/10.1115/1.3653121>
- Li, Lisha et al. (2018). "Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization". en. In: .
- Li, Shuaipeng et al. (2024). "Surge Phenomenon in Optimal Learning Rate and Batch Size Scaling". en. In: *Advances in Neural Information Processing Systems* 37 (NeurIPS 2024).
- Liao, Lizhi et al. (Apr. 2022). "An Empirical Study of the Impact of Hyperparameter Tuning and Model Optimization on the Performance Properties of Deep Neural Networks". In: *ACM Transactions on Software Engineering and Methodology (TOSEM)* 31.3, pp. 1–40. DOI: <https://doi.org/10.1145/3506695>, URL: <https://dl.acm.org/doi/10.1145/3506695>.
- Lin, Tsung-Yi et al. (Feb. 2015). *Microsoft COCO: Common Objects in Context*. en. arXiv:1405.0312 [cs]. DOI: [10.48550/arXiv.1405.0312](https://doi.org/10.48550/arXiv.1405.0312), URL: <http://arxiv.org/abs/1405.0312> (visited on 15/04/2026).
- Loshchilov, Ilya and Frank Hutter (Jan. 2019). *Decoupled Weight Decay Regularization*. en. arXiv:1711.05101 [cs]. DOI: [10.48550/arXiv.1711.05101](https://doi.org/10.48550/arXiv.1711.05101), URL: <http://arxiv.org/abs/1711.05101> (visited on 20/04/2026).
- Mahajan, Dhruv et al. (2018). "Exploring the Limits of Weakly Supervised Pretraining". en. In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari et al. Vol. 11206. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 185–201. ISBN: 978-3-030-01215-1 978-3-030-01216-8. DOI: [10.1007/978-3-030-01216-8_12](https://doi.org/10.1007/978-3-030-01216-8_12), URL: https://link.springer.com/10.1007/978-3-030-01216-8_12 (visited on 24/06/2026).
- Mockus, Jonas (1975). "On Bayesian methods for seeking the extremum". In: *Optimization Techniques IFIP Technical Conference*, pp. 400–404.
- (1989). *Bayesian Approach to global Optimisation*. Vol. 37. Mathematics and its Applications. Kluwer Academic Publishers.

- Montavon, Grégoire, Geneviève B. Orr and Klaus-Robert Müller, eds. (2012). *Neural Networks: Tricks of the Trade: Second Edition*. en. Vol. 7700. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-642-35288-1 978-3-642-35289-8. DOI: [10.1007/978-3-642-35289-8](https://doi.org/10.1007/978-3-642-35289-8). URL: <http://link.springer.com/10.1007/978-3-642-35289-8> (visited on 01/04/2026).
- Moraes, André Magalhães et al. (Mar. 2025). “Effectiveness of YOLO Architectures in Tree Detection: Impact of Hyperparameter Tuning and SGD, Adam, and AdamW Optimizers”. en. In: *Standards* 5.1, p. 9. ISSN: 2305-6703. DOI: [10.3390/standards5010009](https://doi.org/10.3390/standards5010009). URL: <https://www.mdpi.com/2305-6703/5/1/9> (visited on 22/03/2026).
- Moriconi, Riccardo, Marc Peter Deisenroth and K. S. Sesh Kumar (Sept. 2020). “High-dimensional Bayesian optimization using low-dimensional feature spaces”. en. In: *Machine Learning* 109.9-10, pp. 1925–1943. ISSN: 0885-6125, 1573-0565. DOI: [10.1007/s10994-020-05899-z](https://doi.org/10.1007/s10994-020-05899-z). URL: <https://link.springer.com/10.1007/s10994-020-05899-z> (visited on 19/06/2026).
- Neubeck, A. and L. Van Gool (2006). “Efficient Non-Maximum Suppression”. In: *18th International Conference on Pattern Recognition (ICPR'06)*. Hong Kong, China: IEEE, pp. 850–855. ISBN: 978-0-7695-2521-1. DOI: [10.1109/ICPR.2006.479](https://doi.org/10.1109/ICPR.2006.479). URL: <http://ieeexplore.ieee.org/document/1699659/> (visited on 10/07/2026).
- Nguyen, Xuan Dung James and Y.A. Liu (Feb. 2025). “Methodology for hyperparameter tuning of deep neural networks for efficient and accurate molecular property prediction”. en. In: *Computers & Chemical Engineering* 193, p. 108928. ISSN: 00981354. DOI: [10.1016/j.compchemeng.2024.108928](https://doi.org/10.1016/j.compchemeng.2024.108928). URL: <https://linkinghub.elsevier.com/retrieve/pii/S0098135424003466> (visited on 06/05/2026).
- Nogueira-Rodríguez, Alba et al. (Mar. 2023). “Negative Samples for Improving Object Detection—A Case Study in AI-Assisted Colonoscopy for Polyp Detection”. en. In: *Diagnostics* 13.5, p. 966. ISSN: 2075-4418. DOI: [10.3390/diagnostics13050966](https://doi.org/10.3390/diagnostics13050966). URL: <https://www.mdpi.com/2075-4418/13/5/966> (visited on 30/03/2026).
- Novak, S. Y. (Jan. 2019). “Poisson approximation”. In: *Probability Surveys* 16.none. ISSN: 1549-5787. DOI: [10.1214/18-PS318](https://doi.org/10.1214/18-PS318). URL: <https://projecteuclid.org/journals/probability-surveys/volume-16/issue-none/Poisson-approximation/10.1214/18-PS318.full> (visited on 23/06/2026).
- Padilla, Rafael, Sergio L. Netto and Eduardo A. B. Da Silva (July 2020). “A Survey on Performance Metrics for Object-Detection Algorithms”. en. In: *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*. Niterói, Brazil: IEEE, pp. 237–242. ISBN: 978-1-7281-7539-3. DOI: [10.1109/IWSSIP48289.2020.9145130](https://doi.org/10.1109/IWSSIP48289.2020.9145130). URL: <https://ieeexplore.ieee.org/document/9145130/> (visited on 13/04/2026).
- Pascual-González, Mario et al. (Sept. 2025). “Hyperparameter optimization of YOLO models for invasive coronary angiography lesion detection and assessment”. en. In: *Computers in Biology and Medicine* 196, p. 110697. ISSN: 00104825. DOI: [10.1016/j.combiomed.2025.110697](https://doi.org/10.1016/j.combiomed.2025.110697). URL: <https://linkinghub.elsevier.com/retrieve/pii/S0010482525010480> (visited on 02/03/2026).
- Petrak, Johann (2000). “Fast Subsampling Performance Estimates for Classification Algorithm Selection”. en. In: *Proceedings of the ECML-00 Workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, pp. 3–14.
- Redmon, Joseph et al. (June 2016). “You Only Look Once: Unified, Real-Time Object Detection”. en. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, pp. 779–788. ISBN: 978-1-4673-8851-1. DOI: [10.1109/CVPR.2016.91](https://doi.org/10.1109/CVPR.2016.91). URL: <http://ieeexplore.ieee.org/document/7780460/> (visited on 22/03/2026).
- Ren, Mengye et al. (2018). “Learning to Reweight Examples for Robust Deep Learning”. en. In: *Proceedings of the IEEE* 104.1, pp. 148–175. ISSN: 0018-9219, 1558-2256. DOI: [10.1109/JPROC.2015.2494218](https://doi.org/10.1109/JPROC.2015.2494218). URL: <https://ieeexplore.ieee.org/document/7352306/> (visited on 20/03/2026).

- Smith, Leslie N. (Apr. 2018). *A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay*. en. arXiv:1803.09820 [cs]. DOI: [10.48550/arXiv.1803.09820](https://doi.org/10.48550/arXiv.1803.09820) URL: <http://arxiv.org/abs/1803.09820> (visited on 20/04/2026).
- Snoek, Jasper, Hugo Larochelle and Ryan Adams (Dec. 2012). “Practical Bayesian Optimization of Machine Learning Algorithms”. en. In: *NIPS’12: Proceedings of the 26th International Conference on Neural Information Processing Systems 2*, pp. 2951–2959.
- Swersky, Kevin, Jasper Snoek and Ryan P Adams (Dec. 2013). “Multi-Task Bayesian Optimization”. en. In: *NIPS’13: Proceedings of the 27th International Conference on Neural Information Processing Systems 2*, pp. 2004–2012.
- Syamsul, Marshaniswah and Suryo Adhi Wibowo (Dec. 2024). “Optimizers Comparative Analysis on YOLOv8 and YOLOv11 for Small Object Detection”. In: *2024 International Conference on Intelligent Cybernetics Technology & Applications (ICICyTA)*. Bali, Indonesia: IEEE, pp. 978–983. ISBN: 979-8-3315-0649-0. DOI: [10.1109/ICICyTA64807.2024.10912942](https://doi.org/10.1109/ICICyTA64807.2024.10912942) URL: <https://ieeexplore.ieee.org/document/10912942/> (visited on 08/04/2026).
- Wahyudi, Dwi, Indah Soesanti and Hanung Adi Nugroho (Nov. 2023). “Optimizing Hyper-parameters of YOLO to Improve Performance of Brain Tumor Detection in MRI Images”. en. In: *2023 6th International Conference on Information and Communications Technology (ICOI-ACT)*. Yogyakarta, Indonesia: IEEE, pp. 413–418. ISBN: 979-8-3503-1563-9. DOI: [10.1109/ICOIACT59844.2023.10455813](https://doi.org/10.1109/ICOIACT59844.2023.10455813) URL: <https://ieeexplore.ieee.org/document/10455813/> (visited on 23/03/2026).
- Wan, Zhijing et al. (July 2024). “A Survey of Dataset Refinement for Problems in Computer Vision Datasets”. en. In: *ACM Computing Surveys* 56.7, pp. 1–34. ISSN: 0360-0300, 1557-7341. DOI: [10.1145/3627157](https://doi.org/10.1145/3627157) URL: <https://dl.acm.org/doi/10.1145/3627157> (visited on 12/03/2026).
- Wojciuk, Mikolaj et al. (Mar. 2024). “Improving classification accuracy of fine-tuned CNN models: Impact of hyperparameter optimization”. en. In: *Heliyon* 10.5, e26586. ISSN: 24058440. DOI: [10.1016/j.heliyon.2024.e26586](https://doi.org/10.1016/j.heliyon.2024.e26586) URL: <https://linkinghub.elsevier.com/retrieve/pii/S2405844024026173> (visited on 19/03/2026).
- Xu, Feilong et al. (2025). “A Unified Loss for Handling Inter-Class and Intra-Class Imbalance in Medical Image Segmentation”. en. In: *The Thirty-Ninth AAAI Conference on Artificial Intelligence*.
- Yang, Li and Abdallah Shami (Nov. 2020). “On hyperparameter optimization of machine learning algorithms: Theory and practice”. en. In: *Neurocomputing* 415, pp. 295–316. ISSN: 09252312. DOI: [10.1016/j.neucom.2020.07.061](https://doi.org/10.1016/j.neucom.2020.07.061) URL: <https://linkinghub.elsevier.com/retrieve/pii/S0925231220311693> (visited on 19/03/2026).
- Yuan, Wei, Fei Hu and Liangfu Lu (Mar. 2022). “A new non-adaptive optimization method: Stochastic gradient descent with momentum and difference”. en. In: *Applied Intelligence* 52.4, pp. 3939–3953. ISSN: 0924-669X, 1573-7497. DOI: [10.1007/s10489-021-02224-6](https://doi.org/10.1007/s10489-021-02224-6) URL: <https://link.springer.com/10.1007/s10489-021-02224-6> (visited on 05/06/2026).
- Zha, Daochen et al. (May 2025). “Data-centric Artificial Intelligence: A Survey”. en. In: *ACM Computing Surveys* 57.5, pp. 1–42. ISSN: 0360-0300, 1557-7341. DOI: [10.1145/3711118](https://doi.org/10.1145/3711118) URL: <https://dl.acm.org/doi/10.1145/3711118> (visited on 26/02/2026).
- Zhilinskas, A. (1975). “Single-step Bayesian search method for an extremum of functions of a single variable”. In: *Cybernetics and Systems Analysis* 11.1.

Appendix

A.1 Methodology

A.1.1 Proxy configuration selection experiments

Table A.1: Proxy configuration selection experiments. All runs use yolo26s unless noted, with momentum = 0.937, weight decay = 0.0005, warmup epochs = 3, degrees = 5.0, epochs = 100. Proxy runs use 25% of the cleaned training set; baseline and production runs use 100%. mAP@50 is measured on the validation set. Run 16 uses the original uncleaned dataset and non-remediated validation set. Out of memory (OOM) indicates the experiment failed due to insufficient GPU memory.

#	Type	Model	Opt.	Imgsz	Batch	lr0	Mosaic	mAP50	hrs	Note
1	Proxy	s	SGD	640	32	0.010	0.0	0.086	2.14	SGD baseline
2	Proxy	s	AdamW	640	32	0.001	0.0	0.090	2.22	AdamW vs SGD
3	Proxy	s	AdamW	1024	8	0.001	0.0	0.144	5.21	Higher resolution
4	Proxy	s	AdamW	1024	16	0.001	0.0	OOM	–	Batch 16 OOM
5	Proxy	s	SGD	1024	16	0.001	0.0	0.138	5.08	SGD batch 16
6	Proxy	s	AdamW	1024	8	0.001	1.0	0.108	1.33	Mosaic at 1024px
7	Proxy	s	AdamW	1240	4	0.001	1.0	0.123	2.16	Selected (run 1)
8	Proxy	s	AdamW	1240	4	0.001	1.0	0.123	2.18	Selected (run 2)
9	Proxy	s	AdamW	1240	8	0.001	1.0	0.138	1.71	Batch 8
10	Proxy	s	AdamW	1240	8	0.001	0.5	0.135	1.70	Half mosaic
11	Baseline	s	AdamW	1600	8	0.001	1.0	0.231	9.05	Mosaic on
12	Baseline	s	AdamW	1600	8	0.001	0.0	0.224	14.38	Mosaic off
13	Baseline	s	AdamW	1600	16	0.001	0.0	OOM	–	Batch 16 OOM
14	Baseline	m	AdamW	1600	8	0.001	1.0	OOM	–	yolo26m batch 8 OOM
15	Baseline	m	AdamW	1600	4	0.001	1.0	0.214	15.73	Thesis baseline
16	Prod.	m	MuSGD	1600	4	0.010	1.0	0.366	28.56	Lensor production

A.1.2 Preliminary Experiments: Training Plot Proxy Model

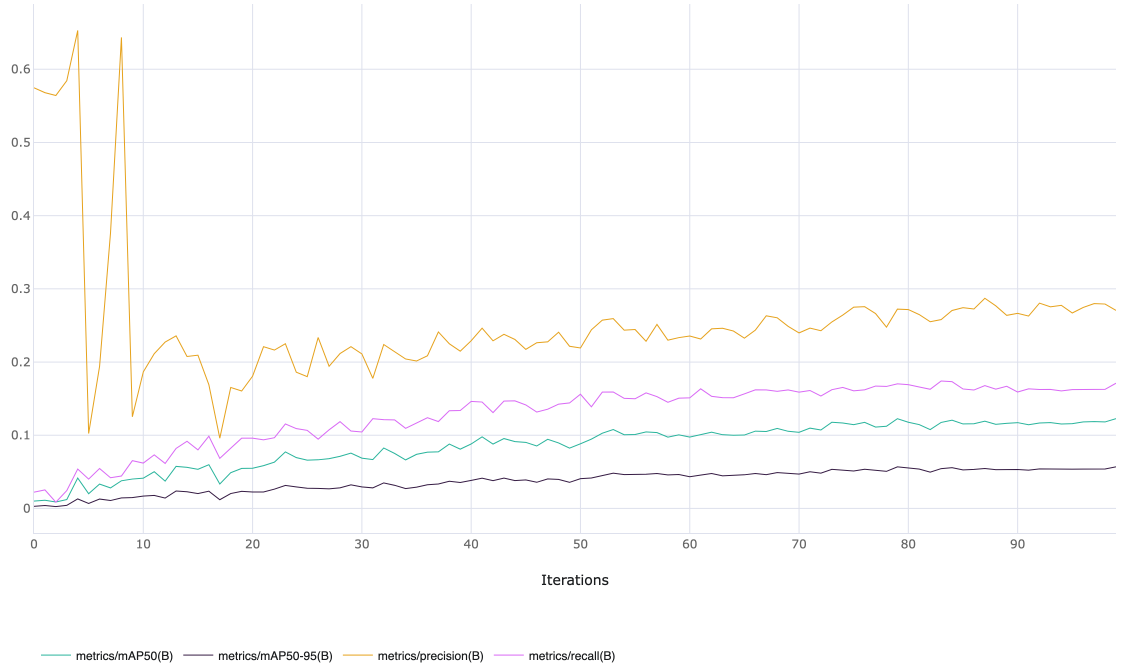


Figure A.1: Validation mAP@50, mAP@50-95, Precision, Recall for a single run of the chosen proxy model.

A.1.3 Frame audit

Table A.2: Frame and ROI diversity per audit cell for the full dataset. Unique frames refers to the number of frames that contain at least one ROI from the given cell. Frames containing ROIs from multiple cells are counted in each, so the column does not sum to the training-set frame total.

Audit cell	Unique frames	Total ROIs	ROIs / frame
dent zone, 2–5 cm	0	0	–
dent zone, 0–2 cm	1	1	1.00
scratch zone, 0–2 cm	56	59	1.05
dent zone, 5–10 cm	62	62	1.00
dent zone, 10–30 cm	88	91	1.03
dent zone, 30+ cm	119	155	1.30
dent, 30+ cm	182	201	1.10
scratch zone, 2–5 cm	545	640	1.17
dent, 10–30 cm	550	597	1.09
scratch, 30+ cm	557	610	1.10
scratch, 0–2 cm	566	691	1.22
dent, 0–2 cm	892	1,438	1.61
scratch zone, 5–10 cm	995	1,156	1.16
scratch zone, 30+ cm	1,081	1,137	1.05
dent, 5–10 cm	1,533	1,659	1.08
scratch zone, 10–30 cm	1,633	1,857	1.14
scratch, 10–30 cm	1,917	2,193	1.14
scratch, 2–5 cm	1,956	2,255	1.15
scratch, 5–10 cm	1,993	2,288	1.15
dent, 2–5 cm	3,585	4,917	1.37

A.1.4 Strata counts on the full training set

Table A.3: E2 stratum definitions with frame diversity at full scale

C2 stratum	Definition	Unique frames	Total ROIs	ROIs / frame
scratch small	scratch, 0–2 cm	566	691	1.22
dent small	dent $\cup$ dent zone, 0–2 cm	893	1,439	1.61
dent large	dent $\cup$ dent zone, 10+ cm	850	1,044	1.23
scratch zone medium	scratch zone, 2–10 cm	1,428	1,855	1.30
scratch zone large	scratch zone, 10+ cm	2,588	2,994	1.16
scratch large	scratch, 10+ cm	2,361	2,803	1.19
scratch medium	scratch, 2–10 cm	3,648	4,543	1.25
dent medium	dent $\cup$ dent zone, 2–10 cm	4,895	6,638	1.36
negative	frames with no ROIs	4,545	0	–

A.2 Results

A.2.1 Experimental Study 1 performance metrics across both evaluation sets

Table A.4: Per-seed mAP@50 on the validation set across configurations, model scales, and damage classes for E1. Means and sample standard deviations are computed over three retrain seeds (42, 43, 44).

Configuration	Scale	Class	Seed 42	Seed 43	Seed 44	Mean	SD
Baseline	Proxy	Aggregate	0.1238	0.1198	0.1232	0.1223	0.0022
		Scratch	0.1280	0.1170	0.1140	0.1197	0.0074
		Dent	0.1190	0.1220	0.1320	0.1243	0.0068
	Full	Aggregate	0.2101	0.2054	0.2259	0.2138	0.0107
		Scratch	0.2110	0.2150	0.2290	0.2183	0.0095
		Dent	0.2100	0.1960	0.2230	0.2097	0.0135
BO-best	Proxy	Aggregate	0.1551	0.1487	0.1533	0.1524	0.0033
		Scratch	0.1590	0.1430	0.1460	0.1493	0.0085
		Dent	0.1510	0.1540	0.1600	0.1550	0.0046
	Full	Aggregate	0.2435	0.2493	0.2407	0.2445	0.0044
		Scratch	0.2790	0.2860	0.2590	0.2747	0.0140
		Dent	0.2080	0.2120	0.2230	0.2143	0.0078
BOHB-best	Proxy	Aggregate	0.1549	0.1601	0.1554	0.1568	0.0029
		Scratch	0.1640	0.1570	0.1710	0.1640	0.0070
		Dent	0.1460	0.1630	0.1400	0.1497	0.0119
	Full	Aggregate	0.2229	0.2328	0.2382	0.2313	0.0078
		Scratch	0.2460	0.2550	0.2680	0.2563	0.0111
		Dent	0.2000	0.2110	0.2090	0.2067	0.0059

Table A.5: Per-seed mAP@50:95 on the test set across configurations, model scales, and damage classes for E1. Means and sample standard deviations are computed over three retrain seeds (42, 43, 44).

Configuration	Scale	Class	Seed 42	Seed 43	Seed 44	Mean	SD
Baseline	Proxy	Aggregate	0.1421	0.1391	0.1302	0.1371	0.0062
		Scratch	0.1040	0.1040	0.0846	0.0975	0.0112
		Dent	0.1810	0.1740	0.1760	0.1770	0.0036
	Full	Aggregate	0.1815	0.1849	0.1790	0.1818	0.0030
		Scratch	0.1450	0.1590	0.1550	0.1530	0.0072
		Dent	0.2180	0.2110	0.2030	0.2107	0.0075
BO-best	Proxy	Aggregate	0.1605	0.1608	0.1664	0.1626	0.0033
		Scratch	0.1300	0.1250	0.1360	0.1303	0.0055
		Dent	0.1910	0.1970	0.1970	0.1950	0.0035
	Full	Aggregate	0.2170	0.2170	0.2118	0.2153	0.0030
		Scratch	0.2050	0.2010	0.1940	0.2000	0.0056
		Dent	0.2290	0.2330	0.2300	0.2307	0.0021
BOHB-best	Proxy	Aggregate	0.1561	0.1592	0.1608	0.1587	0.0024
		Scratch	0.1160	0.1230	0.1270	0.1220	0.0056
		Dent	0.1970	0.1960	0.1940	0.1957	0.0015
	Full	Aggregate	0.2148	0.2113	0.2165	0.2142	0.0027
		Scratch	0.1920	0.1980	0.1980	0.1960	0.0035
		Dent	0.2380	0.2240	0.2350	0.2323	0.0074

A.2.2 Experimental Study 1 validation Δ mAP@50

Table A.6: Validation Δ mAP@50 for Proxy and Full models across methods, damage classes, and seeds. Δ is the difference between the best found configuration and the respective baseline.

Method	Damage	Seed 42		Seed 43		Seed 44	
		Proxy	Full	Proxy	Full	Proxy	Full
BO	Aggregate	0.031	0.033	0.029	0.044	0.030	0.015
	Scratch	0.031	0.068	0.026	0.071	0.032	0.030
	Dent	0.032	-0.002	0.032	0.016	0.028	0.000
BOHB	Aggregate	0.031	0.013	0.040	0.027	0.032	0.012
	Scratch	0.036	0.035	0.040	0.040	0.057	0.039
	Dent	0.027	-0.010	0.041	0.015	0.008	-0.014

A.2.3 Experimental Study 1 full ρ results per seed

Table A.7: Per-seed proxy-to-full calibration ratio ρ on the validation set across damage classes and HPO methods for E1. Mean is computed across the three matched-seed ratios; 95% confidence intervals are percentile bootstrap (10,000 resamples).

Damage	Method	Seed 42	Seed 43	Seed 44	Mean	CI low	CI high
Aggregate	BO	1.067	1.519	0.492	1.026	0.492	1.519
	BOHB	0.412	0.680	0.382	0.491	0.382	0.680
Scratch	BO	2.194	2.731	0.938	1.954	0.938	2.731
	BOHB	0.972	1.000	0.684	0.885	0.684	1.000
Dent	BO	-0.063	0.500	0.000	0.146	-0.063	0.500
	BOHB	-0.370	0.366	-1.750	-0.585	-1.750	0.366

A.2.4 Experimental Study 2 performance metrics across both evaluation sets

Table A.8: Per-seed mAP@50 on the validation set across configurations, model scales, and damage classes for E2. Means and sample standard deviations are computed over three retrain seeds (42, 43, 44).

Condition	Scale	Class	Seed 42	Seed 43	Seed 44	Mean	SD
A	Proxy	Aggregate	0.1551	0.1487	0.1533	0.1524	0.0033
		Scratch	0.1590	0.1430	0.1460	0.1493	0.0085
		Dent	0.1510	0.1540	0.1600	0.1550	0.0046
	Full	Aggregate	0.2435	0.2493	0.2407	0.2445	0.0044
		Scratch	0.2790	0.2860	0.2590	0.2747	0.0140
		Dent	0.2080	0.2120	0.2230	0.2143	0.0078
A'	Proxy	Aggregate	0.1500	0.1294	0.1412	0.1402	0.0103
		Scratch	0.1480	0.1370	0.1340	0.1397	0.0074
		Dent	0.1520	0.1220	0.1490	0.1410	0.0165
	Full	Aggregate	0.2320	0.2213	0.2198	0.2244	0.0067
		Scratch	0.2500	0.2290	0.2340	0.2377	0.0110
		Dent	0.2140	0.2140	0.2060	0.2113	0.0046
B	Proxy	Aggregate	0.1325	0.1445	0.1408	0.1393	0.0061
		Scratch	0.1290	0.1360	0.1450	0.1367	0.0080
		Dent	0.1300	0.1530	0.1370	0.1400	0.0118
	Full	Aggregate	0.2282	0.2261	0.2246	0.2263	0.0018
		Scratch	0.2540	0.2490	0.2630	0.2553	0.0071
		Dent	0.2030	0.2030	0.1860	0.1973	0.0098
C	Proxy	Aggregate	0.1378	0.1391	0.1418	0.1396	0.0021
		Scratch	0.1140	0.1030	0.1100	0.1090	0.0056
		Dent	0.1620	0.1750	0.1740	0.1703	0.0072
	Full	Aggregate	0.1971	0.2056	0.1989	0.2005	0.0045
		Scratch	0.1940	0.1980	0.1880	0.1933	0.0050
		Dent	0.2000	0.2130	0.2100	0.2077	0.0068

Table A.9: Per-seed mAP@50-95 on the test set across configurations, model scales, and damage classes for E2. Means and sample standard deviations are computed over three retrain seeds (42, 43, 44).

Condition	Scale	Class	Seed 42	Seed 43	Seed 44	Mean	SD
A	Proxy	Aggregate	0.1605	0.1608	0.1664	0.1626	0.0033
		Scratch	0.1300	0.1250	0.1360	0.1303	0.0055
		Dent	0.1910	0.1970	0.1970	0.1950	0.0035
	Full	Aggregate	0.2170	0.2170	0.2118	0.2153	0.0030
		Scratch	0.2050	0.2010	0.1940	0.2000	0.0056
		Dent	0.2290	0.2330	0.2300	0.2307	0.0021
A'	Proxy	Aggregate	0.1450	0.1371	0.1369	0.1397	0.0046
		Scratch	0.1080	0.1070	0.1060	0.1070	0.0010
		Dent	0.1820	0.1670	0.1680	0.1723	0.0084
	Full	Aggregate	0.1894	0.1973	0.2048	0.1972	0.0077
		Scratch	0.1600	0.1610	0.1810	0.1673	0.0118
		Dent	0.2180	0.2340	0.2290	0.2270	0.0082
B	Proxy	Aggregate	0.1314	0.1504	0.1388	0.1402	0.0096
		Scratch	0.1110	0.1130	0.1150	0.1130	0.0020
		Dent	0.1520	0.1880	0.1630	0.1677	0.0184
	Full	Aggregate	0.2015	0.1932	0.2049	0.1999	0.0060
		Scratch	0.1760	0.1840	0.1910	0.1837	0.0075
		Dent	0.2270	0.2020	0.2190	0.2160	0.0128
C	Proxy	Aggregate	0.1385	0.1323	0.1338	0.1348	0.0032
		Scratch	0.0717	0.0760	0.0832	0.0770	0.0058
		Dent	0.2050	0.1890	0.1840	0.1927	0.0110
	Full	Aggregate	0.1985	0.1905	0.1921	0.1937	0.0042
		Scratch	0.1570	0.1480	0.1470	0.1507	0.0055
		Dent	0.2390	0.2330	0.2370	0.2363	0.0031

A.2.5 Experimental Study 1 validation $\Delta\text{mAP@50}$

Table A.10: Validation $\Delta\text{mAP@50}$ for Proxy and Full models across conditions, damage classes, and seeds. Δ is the difference between the best found weight configuration from the search space and the baseline (uniform sampling).

Condition	Damage	Seed 42		Seed 43		Seed 44	
		Proxy	Full	Proxy	Full	Proxy	Full
C	Aggregate	-0.012	-0.035	0.010	-0.016	0.001	-0.021
	Scratch	-0.034	-0.056	-0.034	-0.031	-0.024	-0.046
	Dent	0.010	-0.014	0.053	-0.001	0.025	0.004

A.2.6 Experimental Study 2 full ρ results per seed

Table A.11: Per-seed proxy-to-full calibration ratio ρ for the data composition search on the validation set across damage classes. Mean is computed across three matched-seed ratios; 95% CIs are percentile bootstrap ($B = 10,000$).

Damage	Seed 42	Seed 43	Seed 44	Mean	CI low	CI high
Aggregate	2.870	-1.616	-37.056	-11.934	-37.056	2.870
Scratch	1.647	0.912	1.917	1.492	0.912	1.917
Dent	-1.400	-0.019	0.160	-0.420	-1.400	0.160

A.2.7 ρ difference between E1 and E2

Table A.12: Per-seed paired calibration-ratio difference $\Delta\rho_s = \rho_C - \rho_A$ on validation mAP@50. Mean and 95% bootstrap CI on the mean paired difference ($B = 10,000$ percentile resamples).

Damage	Seed 42	Seed 43	Seed 44	Mean	CI low	CI high
Aggregate	1.803	-3.135	-37.548	-12.960	-37.548	1.803
Scratch	-0.547	-1.819	0.979	-0.462	-1.819	0.979
Dent	-1.337	-0.519	0.160	-0.565	-1.337	0.160

A.2.8 Decomposition of costs between different conditions in E2

Table A.13: Difference in mAP@50:95 on the test set between given conditions

Attribution	Damage	Seed 42	Seed 43	Seed 44	Mean	95% CI
A $\rightarrow$ A'	Aggregate	-0.028	-0.020	-0.007	-0.0183	[-0.028, -0.007]
A $\rightarrow$ A'	Scratch	-0.045	-0.040	-0.013	-0.0327	[-0.045, -0.013]
A $\rightarrow$ A'	Dent	-0.011	0.001	-0.001	-0.0037	[-0.011, 0.001]
A' $\rightarrow$ B	Aggregate	0.012	-0.004	0.000	0.0027	[-0.004, 0.012]
A' $\rightarrow$ B	Scratch	0.016	0.023	0.010	0.0163	[0.010, 0.023]
A' $\rightarrow$ B	Dent	0.009	-0.032	-0.010	-0.0110	[-0.032, 0.009]
A' $\rightarrow$ C	Aggregate	-0.003	-0.007	-0.013	-0.0077	[-0.013, -0.003]
A' $\rightarrow$ C	Scratch	-0.003	-0.013	-0.034	-0.0167	[-0.034, -0.003]
A' $\rightarrow$ C	Dent	0.021	-0.001	0.008	0.0093	[-0.001, 0.021]
B $\rightarrow$ C	Aggregate	-0.003	-0.003	-0.013	-0.0063	[-0.013, -0.003]
B $\rightarrow$ C	Scratch	-0.019	-0.036	-0.044	-0.0330	[-0.044, -0.019]
B $\rightarrow$ C	Dent	0.012	0.031	0.018	0.0203	[0.012, 0.031]
A $\rightarrow$ C	Aggregate	-0.019	-0.026	-0.020	-0.0260	[-0.031, -0.020]
A $\rightarrow$ C	Scratch	-0.048	-0.053	-0.047	-0.0493	[-0.053, -0.047]
A $\rightarrow$ C	Dent	0.010	0.000	0.007	0.0057	[0.000, 0.010]