Independent University **Bangladesh (IUB)**

IUB Academic Repository

Computer Science and Engineering | Undergraduate Thesis

2026-08

SutaSet: a multi-label image dataset for classification and object detection for threads

Abdullah, Ahnaf

Independent University, Bangladesh (IUB)

https://ar.iub.edu.bd/handle/11348/1602

Downloaded from IUB Academic Repository



SutaSet: A Multi-Label Image Dataset for Classification and Object Detection for Threads

August, 2026

Prepared by:

Ahnaf Abdullah
ID : 2130223

MD Abir Shahriar
ID : 2230113

MD Nabil Safowan
ID: 2110488

Asif Jobair
ID: 1931404

Department of Computer Science and Engineering
Independent University, Bangladesh

Supervised by

Md Asif Bin Khaled
Senior Lecturer
Department of Computer Science and Engineering
Independent University, Bangladesh

Attestation

We are aware that plagiarism is against the rules and regulations of our university and is completely forbidden. We guarantee that our work is authentic. In our project, we have used some copyrighted material and models of others which have been properly cited by us, using the international standards followed by proper guidelines.

Author Name:
Ahnaf Abdullah (ID: 2130223)

Signature: _____

Author Name:
MD Abir Shahriar (ID: 2230113)

Signature: _____

Author Name:
MD Nabil Safowan (ID: 2110488)

Signature: _____

Author Name:
Asif Jobair (ID: 1931404)

Signature: _____

Evaluation Committee

Supervision Panel

Asif .....

Supervisor
Md. Asif Bin Khaled, CSE, IUB.

Examiners

.....

Internal Examiner 1
Dr. Md. Tarek Habib, CSE, IUB.

.....

Internal Examiner 2
Dr. Asif Mahmood, CSE, IUB.

.....

External Examiner
Mr. Faisal Ahmed, Director, Digital Platforms & Engineering,
Banglalink Digital Communications Ltd

Office Use

.....

Dr. Saadia Binte Alam
Program Coordinator

.....

Dr. AKM Mahbubur Rahman
Head of the Department

.....

Prof. M Arshad Momen
Director, Graduate Studies, Research and Industry Relations,

.....

Md. Shahajada Masud Anowarul Haque
Librarian

Acknowledgement

Our profound appreciation goes out to our esteemed supervisor, Md Asif Bin Khaled, Senior Lecturer, Independent University, Bangladesh. His devoted leadership, perceptive advice, extraordinary tolerance, and the kind donation of his precious time were crucial to the accomplishment of our endeavor. The quality and depth of our work were significantly improved by our supervisor's knowledge and guidance, and we sincerely appreciate his constant support throughout our academic career.

We also express our sincere gratitude to the Department of Computer Science and Engineering, Independent University, Bangladesh for their outstanding assistance and unwavering support. The creation and implementation of our project was greatly aided by the creative and cooperative environment of the Department. In addition, the support and encouragement have been crucial in shaping our fields of study and research. We are grateful for the resources and opportunities provided to us throughout our investigation, which undoubtedly improved our educational experience.

We acknowledge and appreciate the contributions of all who helped us along the way, both directly and indirectly.

Abstract

Automated visual inspection of textiles has primarily focused on fabric-level defect detection, often lumping thread faults into a single generic category. To enable finer-grained research on the visible condition of individual target threads lying on fabric surfaces, we introduce SutaSet, a multi-label image dataset paired with oriented region annotations and prepared for public release. The corpus comprises 6699 unique 512×512 pixel PNG images with four-corner oriented bounding box (OBB) annotations across four positive thread morphological states—normal, frayed, snagged, and visually taut. Image-level multi-label vectors are derived deterministically from exhaustive object annotations; an image with no annotated target-thread instance carries the all-zero vector, which defines the derived target-absence condition (`no_thread`) rather than a fifth learned state.

To ensure scientific evaluation integrity, SutaSet incorporates capture-grouped train, validation, and test split manifests, so that every image derived from one source photograph stays within a single partition. The release provides canonical corner annotations in pixel coordinates, derived classification tables (CSV), Croissant metadata, and a native YOLO-OBB export in which corner coordinates are normalized to the unit square. Reference usability benchmarks using multi-label classifiers (ConvNeXt-Tiny, Swin-Tiny) and an oriented detector (YOLO11n-OBB) are established under the official manifests, on a single seed, to demonstrate data and pipeline validity. State labels denote visually observable surface morphology rather than internal mechanical properties. Slated for archival on Zenodo with mirrors on Hugging Face and GitHub, SutaSet provides a standardized benchmark to advance fine-grained textile inspection and rotated region localization of thin structures, and supports two related tasks derived from one annotation source.

Table of Contents

Abstract **1**

Table of Contents **2**

List of Tables **5**

List of Figures **6**

1 Introduction **8**
1.1 Background & Domain Landscape 8
1.2 Problem Statement 9
1.3 Project Objectives & Outcomes 10
1.4 Project Schedule 10

2 Literature Review & Related Work **12**
2.1 Existing Fabric Inspection & Yarn Quality Datasets 12
2.2 Limitations of Current Work 13
2.3 Positioning SutaSet in the Literature 14

3 Acquiring & Annotating the Dataset **16**
3.1 End-to-End Workflow Overview 16
3.2 Multi-Phase Image Acquisition & Setup 18
3.3 Image Derivation Pipeline & Resolution Strategy 19
3.3.1 Spatial Processing Pipeline 20
3.3.2 Provenance & Naming Scheme 21
3.3.3 Corpus Composition by Derivation and Device 21
3.4 Thread-State Taxonomy 22
3.4.1 Four Positive States and a Derived Absence Condition 23
3.4.2 Oriented Region Localization vs. Tight Segmentation 24
3.4.3 Multi-Label Vector Derivation 25

4 Data Integrity & Evaluation Protocol **27**
4.1 Quality Assurance & Validation 27
4.2 Partitioning & Leakage Control 28
4.2.1 The Evaluation Track: Why Reported Results Use 986 of the 1326 Test Images 29
4.2.2 Provenance & Grouping Logic 30

4.2.3 Data Leakage Audit . . . 30

5 Baseline Methodologies **32**
5.1 Multi-Label Image Classification Architectures . . . 32
5.1.1 ConvNeXt-Tiny Implementations . . . 34
5.1.1.1 ConvNeXt-Tiny Architecture . . . 34
5.1.2 Swin-Tiny Vision Implementations . . . 34
5.1.2.1 Swin-Tiny (Swin Transformer v1/v2) . . . 34
5.2 Oriented Region Localization Architecture . . . 36

6 Implementation Details **38**
6.1 Dataset Manifests & Execution . . . 38
6.2 Model Training & Setup . . . 38
6.2.1 Multi-Label Classification Training . . . 38
6.2.2 Oriented Object Detection Training . . . 41
6.2.3 Threshold Calibration & Discipline . . . 41
6.3 Evaluation Metrics . . . 42
6.3.1 Multi-Label Classification Metrics . . . 42
6.3.2 OBB Detection Metrics . . . 43
6.3.3 Grouped Uncertainty Estimation . . . 44
6.3.4 Empirical Baseline Floors . . . 45

7 Results & Benchmark Analysis **47**
7.1 Provenance of the Reported Results . . . 47
7.2 Multi-Label Classification Performance Comparison . . . 47
7.2.1 Comparative Analysis & Discussion . . . 48
7.3 Oriented Object Detection Results . . . 54
7.3.1 Quantitative Detection Benchmark . . . 54
7.3.2 Localization Analysis . . . 55
7.4 Impact of Data Leakage . . . 63
7.4.1 Near-Duplicate Screening . . . 63
7.5 Ablation: Oriented vs. Axis-Aligned Boxes . . . 65
7.5.1 Status of This Comparison . . . 65
7.6 Qualitative Failure Analysis & Error Taxonomy . . . 66
7.6.1 Detailed Failure Mode Analysis . . . 66
7.7 Model Interpretability & Feature Visualization . . . 70
7.7.1 Grad-CAM Interpretability Analysis . . . 70

8 Conclusion, Limitations & Future Work **71**
8.1 Conclusion . . . 71
8.1.1 Summary of Contributions . . . 71
8.1.2 Principal Empirical Findings . . . 72
8.1.3 Scope of the Claims . . . 73
8.2 Limitations & Scope Constraints . . . 73
8.3 Ethics, Privacy & Licensing . . . 75

8.3.1 Discussion of Ethical Issues 76
8.3.2 Analysis of Social Effects 77
8.3.3 Environmental Analysis & Sustainability 79
8.4 Future Work 81
8.4.1 Hybrid & High-Resolution Model Training 81
8.4.2 Generative Data Augmentation 83
8.4.3 Detection Training Schedule and What It Rules Out 83
8.4.4 Slicing Aided Hyper Inference (SAHI) 83
8.4.5 Oriented versus Axis-Aligned Supervision: The Controlled Comparison 84
8.4.6 Quantifying the Cost of a Naive Split 84
8.4.7 Edge Deployment & Optimization 84
8.4.8 Self-Supervised Pretraining on Unlabelled Fabric Imagery 85
8.4.9 Beyond Oriented Boxes: Curvilinear and Pixel-Level Supervision . . . 85
8.4.10 Calibration & Decision Support 86
8.4.11 Corpus Expansion & External Validity 86
8.4.12 Benchmark Governance & Longevity 87
8.4.13 Annotation Reliability 87

References **88**

List of Tables

2.1 Comparative analysis positioning SutaSet 15

3.1 Provenance metadata completeness 19

4.1 Evaluation track reconciliation 30

5.1 Summary of Classification Baselines 35

6.1 Hyperparameter Configuration Summary 42
6.2 Summary of Evaluation Specifications across Benchmark Tasks. 46

7.1 Run ledger 47
7.2 Oriented detection benchmark 57
7.3 `SutaSet` Error Taxonomy and Analysis of Failure Modes in Classification and Oriented Detection. 67

8.1 Future directions, the limitation each addresses, and its band. 82

List of Figures

1.1 Project schedule 11

3.1 End-to-end SutaSet workflow 17
3.2 Capture campaigns 20
3.3 Corpus composition by derivation transform and device family 22
3.4 Visual samples of the taxonomy 23
3.5 Corpus composition 26

5.1 Empirical geometric distributions of Oriented Bounding Boxes (OBB). . . . 32
5.2 Multi-Label Classification Pipeline Architecture. 33

6.1 Dataset composition across capture phases and partitions 39
6.2 Dataset Manifest and Access Flow Architecture. 39
6.3 Training loss convergence and validation Macro F1 score progression across 30 training epochs. 40
6.4 Class co-occurrence matrix and label frequency distribution 41
6.5 Precision-Recall curves and decision threshold sweeps 42
6.6 Grouped Non-Parametric Bootstrap Resampling Flowchart. 45
6.7 Distribution of non-spatial appearance baseline features 46

7.1 ConvNext-Tiny learning curve 49
7.2 Swin-Tiny learning curve 49
7.3 ConvNext-Tiny pr curves 50
7.4 Swin-Tiny pr curves 51
7.5 Per-class Macro F1 benchmark with 95% CIs ConvNeXt-Tiny 52
7.6 Per-class Macro F1 benchmark with 95% CIs Swin-Tiny 52
7.7 Standard (arbitrary-framing) vs. centered-object (oracle) evaluation performance comparison on ConvNeXt-Tiny. 53
7.8 Standard (arbitrary-framing) vs. centered-object (oracle) evaluation performance comparison on Swin-Tiny. 53
7.9 Classification performance baseline ladder comparing empirical floors against fine-tuned ConvNeXt-Tiny 54
7.10 Classification performance baseline ladder comparing empirical floors against fine-tuned Swin-Tiny 55
7.11 Precision-Recall curves for YOLO11n-OBB across thread classes 56
7.12 Training and validation loss trajectories for YOLO11n-OBB 56
7.13 OBB detection baseline ladder vs. empirical floors 58

7.14 Per-class Average Precision decay across IoU thresholds (0.50 to 0.95) 58
7.15 Precision, recall, and F1 operating curve across confidence thresholds 59
7.16 Curated qualitative OBB predictions (best, two median, and worst cases by matched IoU) at the frozen confidence threshold 0.25. Ground truth in green, predictions in dashed orange. 60
7.17 Unranked random sample of five test images with ground-truth (green) and predicted (dashed orange, conf $\geq$ 0.25) oriented boxes, illustrating typical detector behaviour. 61
7.18 Occlusion-based evidence maps for five representative detections (patch 96 px, stride 32 px). Brightness encodes confidence lost when that patch is covered; ground truth in green, the explained detection in dashed cyan. 62
7.19 Data leakage evaluation: Capture-Grouped vs. Naive Random Split 63
7.20 Residual leak channels 64
7.21 Visual near-duplicate screening pipeline for cross-split disjointness verification. 65
7.22 Oriented versus axis-aligned envelopes 66
7.23 Localization quality and error breakdown 67
7.24 Primary Failure Modes Visual Representation: Micro-fiber blending under intricate weave patterns (Type I) and bounding envelope cropping for highly elongated aspect ratios (Type III). 68
7.25 Qualitative detection and classification results on test images showcasing true positive predictions and common failure modes. 69
7.26 Gradient-weighted Class Activation Mapping (Grad-CAM). 70

1 Introduction

1.1 Background & Domain Landscape

Historically, in the realm of apparel inspection and textile manufacturing, quality assurance and material characterization involved the use of manual visual inspection and optical microscopy [1, 2]. Although the aforementioned conventional techniques are currently still considered the gold standard in laboratory conditions, they are inherently cumbersome, subjective, slow, and hard to integrate into the automation of high-throughput production [3].

In the last several years, computer vision and deep learning algorithms, particularly CNNs and Vision Transformers (ViTs), became ubiquitous in automated surface inspection applications in different areas of industrial manufacturing [3, 4, 5]. The current state-of-the-art in textile computer vision is concerned predominantly with fabric-level defect detection (broad weave irregularities, stains, and holes on a fabric roll) [6, 7].
In this traditional approach, all thread-related defects are aggregated into one generic "defect" or "broken yarn" class [8].
At the same time, this coarse categorization does not account for an important industrial fact that the state of an individual target thread—whether the thread is in the normal state, in a state of being frayed (loose fibers), snagged (loop of a thread caught across the fabric), or visually taut (pulled straight along a dominant axis)—is information that a coarse defect label discards [9, 10]. These are descriptions of observable surface appearance; this work measures no mechanical property, and makes no claim about tensile force, yarn twist, or garment durability. Fine-grained thread-state classification is an intrinsically different computer vision problem compared to traditional defect detection:

- **Thinness of Foreground Structure:** Thread is a thin, variable in orientation structure (typically $\sim 2\,\text{px}$ wide) against the background of a high-texture fabric [9, 11].
- **Multi-label Co-occurrence:** Different morphological states co-occur within one image [10] (the example is a thread in a state of having both snagged and frayed defects).
- **Region vs. Pixel Delineation:** Localization of a region of interest (span of the thread defect) requires accurate oriented bounding box (OBB) in the four corners format, instead of regular horizontal bounding boxes or pixel segmentation masks, which are either spatially loose or too costly to provide exhaustively [12, 13].

Annotated fine-grained yarn imagery does exist—notably the yarn-hairiness corpora of Pereira and colleagues, which annotate loops and protruding fibers under specialized macro-imaging conditions [9, 14]. What is not available, to our knowledge, is a corpus of *target threads observed against textured fabric surfaces* carrying a defined visual-state taxonomy, oriented-region geometry, and a published grouped split protocol. We present SutaSet to occupy that

setting; Section 2.3 states the specific difference from each closest comparator. SutaSet was captured in five acquisition campaigns and split so that all images derived from one source photograph share a partition [15, 16]. Section 4.2 states precisely which independence condition that establishes, and Section 8.2 states which it does not.

1.2 Problem Statement

Although traditional industrial vision-based systems regard fabric defects as a binary homogeneous anomaly detection problem [3, 6], there is a necessity for a more nuanced characterization of individual thread states and conditions [1, 17, 14]. The problem of thread condition recognition has not been formulated in existing computer vision datasets, which tends to collapse the problem to some generic categories of defects [6, 7]. The proposed benchmark dataset SutaSet successfully mitigates this issue through providing two mathematically aligned formulations of computer vision tasks using high-resolution surface images of various textile structures.

In the first place, the benchmark poses the problem of multi-label state classification task. A computer vision model is expected to produce a multi-label binary state vector consisting of four distinct morphological thread states: normal, frayed, snagged, and visually taut. In some cases, a thread defect may involve multiple concurrent morphological states such as a snagged thread loop with frayed loose fibers at the apex [10]. The images featuring clean and homogenous textile surfaces without visible thread events serve as the negative control class of the task.
The second computer vision task is oriented region localization task. While conventional bounding boxes for objects are axis-aligned, the benchmark dataset requires predicting a set of four-corners oriented bounding boxes for each visually coherent thread state region [13, 18, 19]. Such a task is essential since thread structures are ultra-thin foreground objects with the width of $\sim 2\,\text{px}$ [9, 11, 14]. Traditional axis-aligned bounding boxes include a large portion of fabric texture background, thus creating significant visual noise for object detection training [12]. At the same time, pixel-level segmentation masks are not viable for this task since they would be too complicated to annotate due to fiber entanglements of snagged [10] threads. Orienting the bounding box along the axis of the thread region helps isolate the target region without excessive boundary [18, 20].

Moreover, the formulation of the benchmark opens up one of the main methodological challenges of visual evaluation – data contamination of cross split [15]. Source high resolution images are typically cropped or tiled to provide uniform patches for training deep learning models. The standard way of random train-test splitting spreads out sub-images taken from the same source photo between the two partitions, causing data leakage and artificial inflation of baseline accuracy [21]. SutaSet mitigates this problem by enforcing a capture group split protocol which ensures all sub-images from a single physical photo stay within one partition [16].

1.3 Project Objectives & Outcomes

In order to overcome the limitations in terms of structure of the datasets [3] for fabric inspection, the most important goal of the SutaSet project is to develop a standardized benchmark for fine-grained thread morphological state recognition with its evaluation protocol stated and audited. The main idea of the project is to move from traditional binary defect detection to a pair of aligned task formulations—multi-label classification and oriented region localization—derived from one annotation source [7].

The first important result that was obtained in the course of this research is the introduction of a high resolution multi-label surface image dataset with 6699 unique images recorded during several different capture sessions. The dataset covers four positive morphological states together with thread-absent negative controls, and records label co-occurrence rather than forcing a single state per image. Positive instances carry four-corner oriented regions in pixel coordinates that enclose the thread-state event.

Another key contribution is an evaluation protocol whose integrity mechanisms are specified and audited rather than assumed. Photograph-level grouping across training, validation, and test partitions closes the leakage pathway by which sub-images of one photograph are spread across partitions, and a cross-split near-duplicate screen tests for residual visual overlap under a stated criterion [15]. To support baseline reproducibility, the release includes standardized usability reference models, evaluating state-of-the-art convolutional neural networks, vision transformers, and oriented object detectors [17, 22]. Finally, the entire benchmark is formatted in accordance with FAIR data principles, providing open access under Creative Commons licensing and standardized web metadata descriptors to support broad research reuse.

1.4 Project Schedule

The project ran across two academic years, from the first capture campaign in October 2024 to submission in September 2026. Figure 1.1 sets out the schedule. Acquisition was deliberately episodic rather than continuous: five discrete campaigns, spaced so that each could target the fabric surfaces and thread geometries under-represented by the ones before it. Engineering work on the derivation pipeline and manifest builder began only once enough of the corpus existed to make the design decisions concrete, and the evaluation protocol was frozen, audited, and only then used for the baseline runs — an ordering that matters, because a protocol fixed after seeing results cannot constrain them. Each bar is drawn from a recorded source rather than from recollection: acquisition from the embedded image timestamps in `manifest.csv`, the pipeline bar from the repository history, the protocol bar from the dated entries in the decision record, and the run markers from the `run_utc` fields of the baseline receipts. The annotation bar is the one exception — it spans the interval the work necessarily occupies, between first capture and the manifest freeze, and is not independently timestamped.

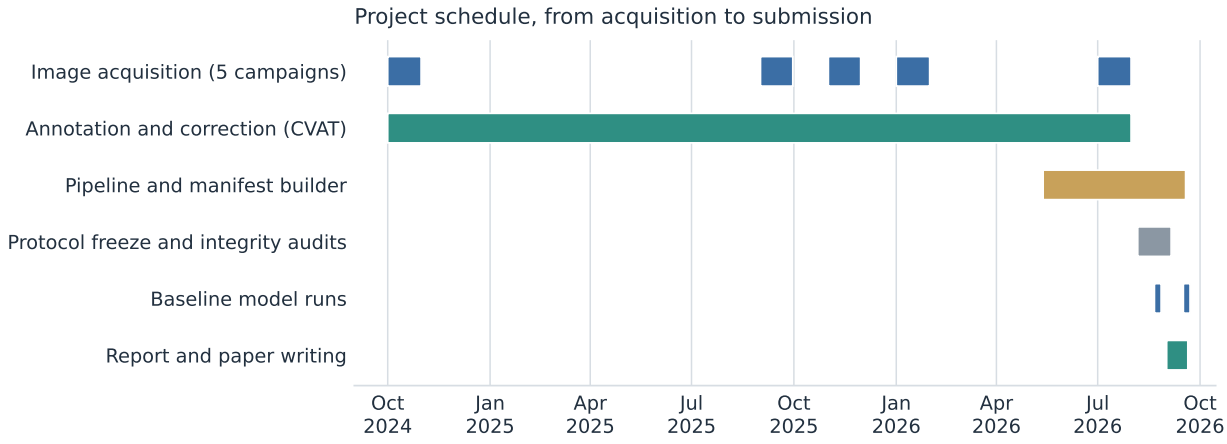


Figure 1.1: Project schedule from first capture to submission.

2 Literature Review & Related Work

2.1 Existing Fabric Inspection & Yarn Quality Datasets

The creation of automated quality control systems within the field of textiles has been contingent upon image datasets that have been created for particular phases of textile production, from fiber analysis and yarn spinning to fabric weaving and garments [3, 10, 22, 23]. Textile computer vision benchmarking in the early days has mostly concerned the coarse inspection of defects in the fabrics in industrial weaving conditions [24]. The most well-known benchmarks of fabric defects in the computer vision domain include the TILDA fabric anomalies database, the Hong Kong Polytechnic University fabric defect dataset, and novel industry surface anomalies suites, which have been developed in order to evaluate the performance of the feature extraction and deep learning approaches under factory illumination [25, 26]. Defect samples in these benchmark suites have mainly been annotated as coarse axis-aligned spatial regions or global binary images indicating the occurrence of oil stains, holes, missing ends, and weaving errors [27]. Yet, since such datasets focus on fabric defects on a macro level, they cannot offer pixel-wise annotations for individual threads.

To provide quality control earlier in the production chain, there have been several specialized datasets created for testing the spinning process of yarn, structural integrity of threads, and surface hairiness [28, 29]. The dataset for online monitoring of yarn spinning provides measurements of the macroscopic motions of continuous strands of yarn that determine the linear density variation, thick-and-thin spots, and structural uniformity at the spinning process of ring yarns [30, 31]. In the same way, the datasets used for characterizing yarn hairiness are based on images of protruded fiber ends and fiber loop formation in high magnification macro imaging systems [14]. Even though these databases provide great progress in objective evaluation of the yarn quality, they make narrow lab assumptions, such as the evaluation of isolated single yarn strand against uniform non-textured background plates or specialized optical sensor fixtures.

In addition to this, recent developments in deep learning object detection algorithms and transformers have revealed the importance of resolution and accuracy of spatial annotations in industrial visual inspections [32, 22]. In situations where the target objects are thin, multiscalable, and variable in orientation, use of conventional horizontal bounding boxes introduces significant visual noise, since it includes irrelevant background fabrics [33]. While there have been many proposals for automated annotation techniques and OBB object detections in areas such as marine radar detection, robot bin-picking, and remote sensing [19],

their applications to fine-grained thread defects in textiles have not yet been attempted due to the absence of dedicated rotated visual benchmarks.

As a result, current textile datasets have the following three important domain-related limitations that prevent their application for real-world quality inspection of clothes and fabrics:

1. **Macro Defect Coarse Classification:** Macro defects in fabrics are treated as a unique homogeneous defect class, regardless of frayed, snagged, or taut states [6].

2. **Target Thread Isolation:** Yarn defect datasets work in a lab setting where each thread is separately inspected under high magnification, disregarding possible real-world scenarios with a target thread that is located either within or across complicated fabric background [14].

3. **Annotation Geometry & Data Leakage Vulnerability:** Current datasets provide only horizontal bounding boxes and/or global image tags, which cause strong noise from the background and do not support rotated spatial geometry of the object [12, 13]. Furthermore, spatial patches extraction without grouping the images leads to cross-split data leakage [15, 21].

The proposed SutaSet solves this problem by introducing multi-label surface dataset with pixel-oriented bounding boxes for localized thread classification.

2.2 Limitations of Current Work

While considerable advances have been achieved through deep learning techniques for industry vision, there exist limitations within current data sets and architectures that make them inadequate for recognizing fine-grained morphological thread states. These limitations can be classified into four main categories: coarse taxonomy aggregation, single-task architecture design, noisy bounding boxes, and data leakage across splits.
Firstly, textile data sets today have issues with coarse taxonomy aggregation [6, 7]. Current textile anomaly detection benchmarks aggregate different physical processes such as loose fiber fraying, snagged thread pulling, and tension [8, 24] into one large "defect" or "broken yarn". However, on the practical level, different morphological states have drastically different implications: for example, frayed thread indicates mechanical friction or insufficient yarn twisting, while a snagged thread points at malfunction of the tension in the weaving loom [1, 9]. This means that by aggregating different physical behaviors into one binary class, existing data sets prevent neural networks from learning fine-grained features of morphological thread states [27].

Additionally, the present architectures are built on the assumption of single task that does not consider real-world co-existence of defects [3, 5]. Traditional approaches to visual inspection perform evaluation of global multi-class image classification or object detection tasks independently of each other [34]. But in the industry of fabrics manufacturing, thread defects often co-exist in a single image; for example, the appearance of the thread snag may lead to

fiber separation and fraying of displaced threads at once [10, 14]. Single-label models cannot handle such multi-label co-existence, and object detection models do not have a multi-label head formulations for overlapping oriented regions [35].
Third, the use of conventional horizontal bounding boxes (HBBs) generates extreme spatial noise in relation to ultra-thin, orientation-sensitive objects [34]. The body of threads is just a few pixels wide ($\sim$ 2 px), but stretches across diagonally on high-resolution textile images [11, 9]. By fitting an event involving a diagonal thread using an HBB, a large amount of fabric weave texture around the object has to be included within the box [26, 35]. In the process of backpropagation, it introduces enormous quantities of negative visual features of the background into the loss function of the detection network, making it difficult to localize and represent features effectively [18, 33]. While pixel-level segmentation masks avoid background noise, exhaustively delineating entangled loose fibers is annotation-prohibitive and not good for large-scale datasets [20].

Additionally, benchmark datasets for computer vision lack robust mitigation against leakage due to spatial patch extraction and naive random splitting methodologies. High resolution input images are usually cropped or smart-cropped to be of suitable size for deep learning input [36]. The application of random naive splitting on these derived images leads to the same original photograph or capture being present in all three sets [15, 21]. Since adjacent patches contain the same weave pattern, illumination profiles, and sensor noise characteristics, the model can memorize these background textures instead of extracting the generalizable thread defect patterns, leading to highly optimistic test metrics, unfit for practical deployment [16]. SutaSet provides an adequate solution to this problem by employing a multi-label classification taxonomy, oriented bounding box annotations, and a split methodology based on capture groups.

2.3 Positioning SutaSet in the Literature

In order to deal with the identified shortcomings of previous work, SutaSet is designed as a benchmark dataset for fine-grained textile vision that publishes its grouped split protocol and supports two aligned task views. Table 2.1 illustrates a comparison placing SutaSet in relation to other fabric defect detection, yarn inspection, and generic object detection benchmarks.
As is shown in the comparative taxonomy, SutaSet creates an essential bridge between the macroscopic fabric anomaly datasets and the microscopic laboratory yarn datasets [10, 11]. While the macroscopic fabric defect dataset utilizes a binary label [8, 25] to classify whether or not there is a defect in the image, SutaSet introduces an operational taxonomy of four positive visual states (normal, frayed, snagged, and visually taut) together with a derived thread-absent condition, which distinguishes appearances that a binary label collapses [9]. Additionally, while the single yarn datasets show isolated threads against fabricated laboratory backdrops [14, 28], SutaSet evaluates thread conditions directly against complex, real-world woven and knitted fabric background textures [11, 2].

As far as geometric and spatial annotations are concerned, SutaSet introduces an advance-

Table 2.1: Comparative analysis positioning SutaSet against existing benchmark datasets. Each row is compiled from that dataset's own primary publication or documentation; "not specified" records that the source does not state a partitioning protocol, not that none exists.

Dataset	Target Level	Taxonomy	Multi-Label	Annotation	Background	Partitioning
TILDA Textile Atlas [25]	Fabric	Multiple defect types	No	Image-level	Fabric Weave	Not specified
Carrilho et al. fabric defect set [7]	Fabric	Defect types	No	Bounding box	Fabric Weave	Author-defined
HKPolyU Fabric Defect [8]	Fabric	Coarse (Binary)	No	Horizontal Box	Fabric Weave	Naive Random
Yarn hairiness / loop set [9]	Single Yarn	Loops, protruding fibers	No	Instance annotations	Uniform (Lab)	Author-defined
TextileNet [11]	Fabric	Material Type	No	Image-level	Macro Texture	Naive Random
SutaSet (Ours)	**Thread on Fabric**	**4 positive states**	**Yes**	**Oriented Box**	**Complex Textile**	**Photograph-Grouped**

ment to the field of visual benchmark literature through the use of pixel-based four-corner Oriented Bounding Box (OBB) annotations [34]. The rotated rectangle encloses a thin diagonal thread in a tighter envelope than an axis-aligned box of the same event, and so admits less surrounding fabric into the supervised region [33, 19, 35]. It remains a region envelope, not a pixel-tight boundary.

Last but not least, SutaSet raises the bar on benchmark evaluation integrity with regards to industrial computer vision applications via its strict photograph-level capture partitioning [15, 16]. The build verifies zero cross-partition photograph overlap and eliminates byte-identical duplicates by content hash, so the reported evaluation measures generalization to held-out *photographs*. It does not establish generalization to unseen physical specimens, which Section 8.2 records as an open limitation [21].

3 Acquiring & Annotating the Dataset

3.1 End-to-End Workflow Overview

Figure 3.1 sets out the complete path from a physical fabric specimen to a published benchmark result, and the remainder of this report follows it stage by stage. The figure is organized as two columns joined at the manifests. The left column is the *build*: everything that produces the canonical dataset, described in this chapter and in Chapter 4. The right column is the *consumption*: the two task views derived from the manifests, their baselines, and the evaluation rules, described in Chapters 5 to 7.

Three properties of the diagram are worth reading deliberately, because they are design commitments rather than incidental structure. First, annotation happens *once*, at 1024×1024, and both task views are derived from that single canonical source—so the multi-label vector and the instance geometry cannot drift apart, and their equivalence is checked as a build gate rather than assumed. Second, the two shaded gates are refusals, not warnings: geometry that fails validation is quarantined with a recorded reason, and a near-duplicate found across a split boundary stops the build and names the offending pair, so a leaky corpus cannot be published by inattention. Third, the arrow from the manifests to the task views runs one way only. Nothing downstream of that boundary can re-derive the split, re-fit a threshold on test data, or reach behind the manifests to the raw captures; the protocol is fixed upstream of every experiment reported here.

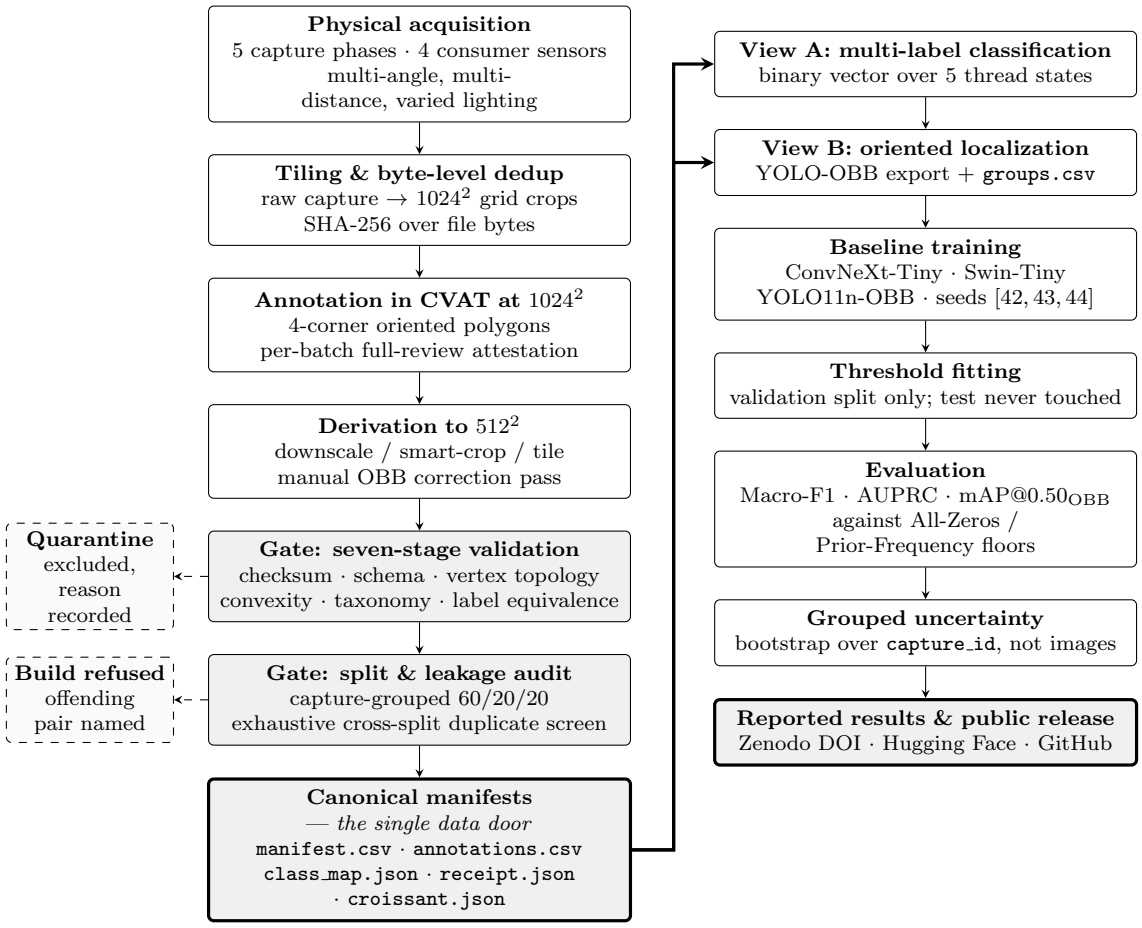


Figure 3.1: End-to-end workflow, from physical capture to published result. The left column builds the canonical dataset; the right column consumes it. Shaded blocks are enforcing gates.

3.2 Multi-Phase Image Acquisition & Setup

To ensure high visual diversity and physically realistic variability, the textile specimens were carefully chosen from a wide variety of local commercial clothing manufacturers, tailoring ateliers and fabric stores. The physical target specimens consist of a wide variety of woven and knitted fabric surface textures of different fiber density, weave pattern, yarn count, surface texture and color composition. Target thread-state events include both naturally occurring production anomalies, such as loosely frayed fibers as a result of physical abrasion or snagging on the loom in the process of weaving, and controlled manipulations of threads on the specimen surface to capture structurally extreme conditions under repeatable conditions. These two origins are *pooled* in the released corpus. No `state_origin` field was recorded at capture time, so the corpus cannot report how natural and deliberately induced events are distributed across the four states or across the partitions, and the reported scores cannot be decomposed by origin. This matters for interpretation: a deliberately induced fray or snag may be more pronounced than a production defect of the same class, so performance measured here may not transfer unchanged to naturally occurring faults. Recording the origin per instance is specified as future work in Section 8.4.11.

Dataset acquisition was carried out through five phases of physical capture, aimed at building a diverse visual corpus at multiple levels of scale. The first acquisition phases were carried out in order to establish the core photographic corpus with a broad variety of fabric surfaces and thread states. Following acquisition phases were performed to add more thread geometries, defect configurations and underrepresented fabric background to balance the classes and increase the diversity. In total, data collection resulted in 798 high resolution source photographs captured in different imaging environments.
In order to capture a wide variety of optical noise characteristics and emulate industrial and consumer hardware used to inspect textiles, imaging was done using four camera sensors, each with a distinct optical lens configuration: three smartphones (Nothing CMF 1, Realme Narzo 70 Turbo, and Samsung Galaxy A51) together with an unnamed Sony compact camera. Device attribution is recoverable per image only at the level of the *sensor family* recorded in the manifest, and it is incomplete: 6,126 of 6,699 images carry a resolvable family and 573 do not, so the per-device breakdown in Figure 3.3(b) reports two attributable families plus two residual groups rather than the four devices named here. Capture timestamps are likewise partial (568 of 798 photographs). Table 3.1 states the missingness for each provenance field; the legacy records predate the naming convention that makes attribution automatic, and their device identity was not reconstructed by inference. The imaging geometry was varied systematically across the wide range of different multi-angle pitch and yaw orientations, from 0° (top-down perpendicular view of surface) to 45° (low angle oblique perspective), and combined with two different focal distance regimes: high magnification macro inspection distance ($<$ 10 cm) and standard macro-inspection distance ($\sim$ 30 cm). Different lighting configurations were considered during imaging, including ambient workshop and laboratory directional lights in order to investigate effects of variable shadowing, surface reflection and localized contrast differences.

Table 3.1: Completeness of each provenance field, counted from the released manifest. "Recorded" means the value was read from the file or the capture record; no field is reconstructed by inference from filenames, pixels, or timestamps.

Provenance field	**Recorded**	**Absent**	**Unit and note**
Image identifier	6,699	0	Per image; SHA-256 of the file bytes
Source photograph identifier	6,699	0	Per image; the grouping key for the split
Derivation transform	6,699	0	Per image; one of four transforms
Annotation review status	6,699	0	Per batch, read from the batch record
Device family	6,126	573	Per image; 424 legacy, 149 unattributable
Capture timestamp	568	230	Per photograph; the rest predate consistent metadata
Specimen identity	0	—	Never recorded; see Section 8.2
State origin (natural / induced)	0	—	Never recorded; see above

Strict inclusion/exclusion criteria for the specimens during physical acquisition was applied. The surfaces had to include a visible target thread event or represent an evenly colored fabric background intended for negative baseline control. Because the negative controls were selected for even coloration while positive examples were sought on varied and often more structured surfaces, the two groups are not matched for background complexity. A classifier could therefore obtain part of its score by separating plain fabric from complex fabric rather than by reading thread morphology, and the present experiments do not exclude that. A matched negative-control analysis—positive and negative regions drawn from the same specimen under comparable lighting, scale, and device—is specified in Section 8.4.11. Captures that exhibited excessive motion blur, severe defocusing, chromatic aberration or physically unresolvable contamination were excluded from further annotation.

3.3 Image Derivation Pipeline & Resolution Strategy

Dataset pre-processing was carried out based on a multi-tier approach with respect to spatial resolution in order to enhance visual quality as well as take into account computational limits required by deep learning. Initially, raw parent photos were captured with an extremely high native optical resolution of 3072×3072 pixels. During the initial development stage, parent photos were split via uniform 3×3 grid slicing:

$$\text{Grid Tiles} = \left(\frac{3072}{1024}\right) \times \left(\frac{3072}{1024}\right) = 3 \times 3 = 9 \text{ tiles per photograph} \tag{3.1}$$

thus forming the intermediate dataset of 1024×1024 pixel size annotated in the Computer Vision Annotation Tool (CVAT).

A working resolution of 512×512 pixels was adopted as the canonical size, chosen to fit the memory budget available for fine-tuning while retaining visible fiber-scale structure. This was a practical decision, not the outcome of a controlled resolution study: no experiment in

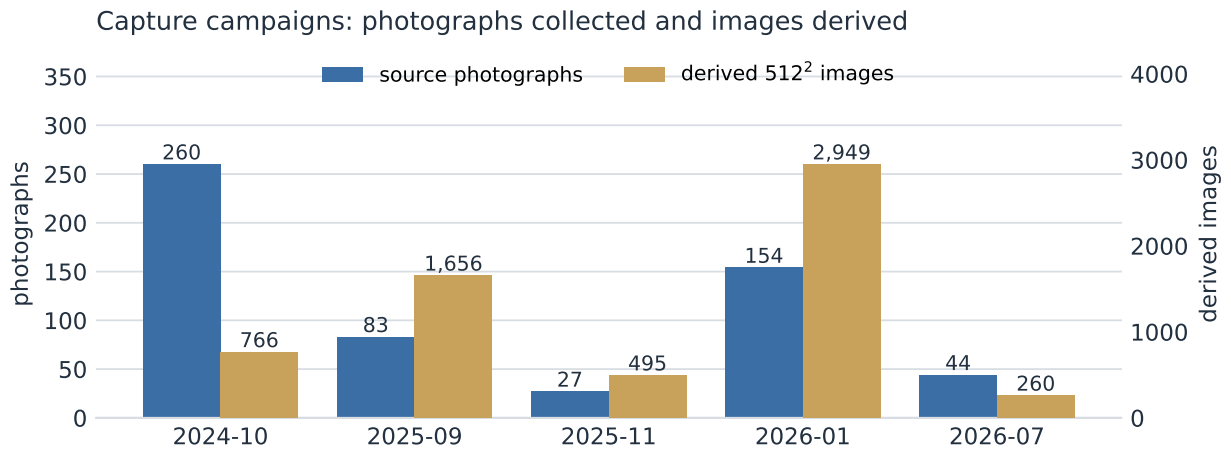


Figure 3.2: Photographs collected and 512×512 images derived per capture campaign. Note the two axes: a campaign's yield in images is not proportional to its yield in photographs, because the number of images derived from one photograph depends on the source resolution and on how much of the frame carries annotated thread structure (median 4 images per photograph, maximum 33). The October 2024 campaign contributed the most photographs but among the fewest images per photograph; January 2026 contributed the most images. Counts cover the 568 of 798 photographs whose files carry a usable capture timestamp (6,126 of 6,699 images); the remainder predate consistent metadata capture.

this report varies input resolution while holding everything else fixed, so 512×512 is not demonstrated to be optimal. Nor is any physical scale established—no calibration target was imaged, so the corpus cannot state what physical distance a pixel subtends, and claims about resolving structures of a given physical size are not supported by image dimensions alone. In order to provide the standard canonical dataset at 512×512 pixels in uncompressed sRGB PNG format without having to reannotate raw parent captures from scratch, the deterministic multi-path spatial derivation pipeline was created.

3.3.1 Spatial Processing Pipeline

As for the intermediate 1024×1024 annotated image dataset resulting from the first acquisition phases, images underwent the following mathematically controlled transformations to achieve 512×512 resolution:

1. **Direct uniform downscaling:** Intermediate 1024×1024 tiles were uniformly downscaled to 512×512 pixels by means of bilinear interpolation, retaining full image context, background fabric geometry, and general fabric surface aspect ratio.

2. **Centroid-based smart-cropping:** In case when thread state events occurred in certain sub-regions within images, 512×512 crop windows based on centroids of thread state events were extracted. Thus, optical scale of fabric fibers and other fine morphological details were retained without scaling distortions.

3. **Grid tiling cropping:** Intermediate 1024×1024 images were split into four non-overlapping 512×512 sub-tiles through a 2×2 grid slicing algorithm. As a result, sub-tile spatial granularity increased and localized background fabric control patches were isolated.

As for the raw 3072×3072 parent images acquired in further acquisition phases, a multiple path processing architecture was utilized to generate 512×512 sub-images. The architecture included three sub-paths: tiling of 3072×3072 parent images to 1024×1024 images, followed by downscaling to 512×512, tiling to 2048×2048 images, followed by downscaling to 512×512, and direct downscaling from 3072×3072 to 512×512. A randomization and balancing pass ensured an even distribution among the transformation methods to avoid transformation-induced bias in further model evaluation.

3.3.2 Provenance & Naming Scheme

In order to ensure that all data provenance is kept strictly and to enable full auditing for derived sub-images, all released images will have a clear and explicit provenance to their parent. Every derived sub-image has a link back to its parent photo through a permanent parent identifier. The parent-child relationship can be traced through the central capture manifest and enables any 512×512 patch to be linked to its spatial location within the original high-resolution photo.

All metadata information coming directly from the camera, including sensitive attributes such as acquisition times, GPS location tags, device serial numbers, and owner ID numbers, were removed from released files to ensure privacy and prevent unmasked metadata leaks. All derived images use a standard naming scheme for each particular transformation along with their capture session number. Importantly, all canonical dataset records are identified using their SHA-256 hashes, which uniquely verify visual identity regardless of the filesystem directory structure.

3.3.3 Corpus Composition by Derivation and Device

Two attributes partition the corpus exactly: every image was produced by one derivation transform, and every image carries one recorded device family. Figure 3.3 shows both, read directly from `manifest.csv`.

Derivation Mix The one-time hybrid pass applied one of three strategies per image, and the realized proportions are tiling 3821 (57.0%), smart-cropping 1668 (24.9%), and downscaling 1191 (17.8%), with a residue of 19 hand-cropped images (0.3%) from the manual correction pass. The mix is a property of the corpus worth stating rather than assuming uniform: tiling contributes the majority of images, so a reader interpreting per-image statistics should remember that most frames are tile products of a larger capture rather than whole-frame downscales.

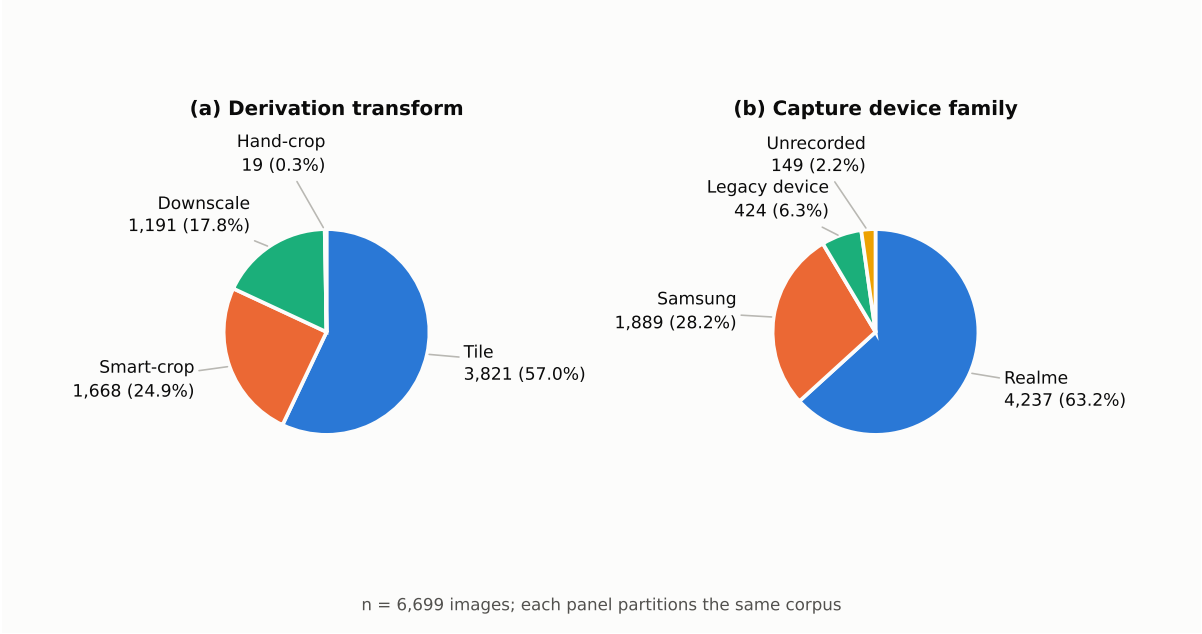


Figure 3.3: Composition of the 6699-image corpus under two mutually exclusive partitions. (a) The derivation transform that produced each 512×512 image. (b) The capture device family recorded in the manifest. Counts are read from `manifest.csv`; each panel sums to the full corpus.

Device Mix The device partition is more uneven. A single sensor family accounts for 4237 images (63.2%) and a second for 1889 (28.2%), while 424 images (6.3%) come from the legacy capture phases whose filenames predate the provenance convention and 149 (2.2%) carry no recoverable device attribution. This concentration is the empirical form of the transferability constraint recorded in Section 8.2: sensor diversity exists in the corpus, but it is not balanced, and roughly nine in ten attributable images come from just two device families. A future capture phase aimed at external validity should prioritize breadth of sensor over volume.

3.4 Thread-State Taxonomy

In order to overcome the discrepancy between high level characterization of fabric surface and accurate spatial localization of defects, SutaSet utilizes a single unified annotation schema that leverages a taxonomic definition of thread morphology along with Oriented Bounding Box geometry. Typically, standard industry image datasets make use of unrestricted global tags or axis aligned spatial boxes that do not represent orientation variable textile structures well. SutaSet annotation framework overcomes these issues by providing a schema that combines four positive visual-state classes and a derived target-absence condition with pixel-level four-corner Oriented Bounding Box (OBB) annotations.

3.4.1 Four Positive States and a Derived Absence Condition

SutaSet defines **four positive visual-state classes**, which are what a model is trained to predict, together with a **derived target-absence condition** that is not a learned class but the consequence of an image carrying no positive instance. The taxonomy was established iteratively to clarify ambiguous thread observations and then frozen (Guideline v1.0).
Two points of terminology matter for reading what follows. The states describe a *selected target thread lying on the fabric surface*—a strand distinguishable from the weave beneath it—and not the constituent threads of the fabric itself, which are present in every image by definition. And `no_thread` asserts the absence of an annotated *target* event, not the absence of thread material. The classes are defined as follows:

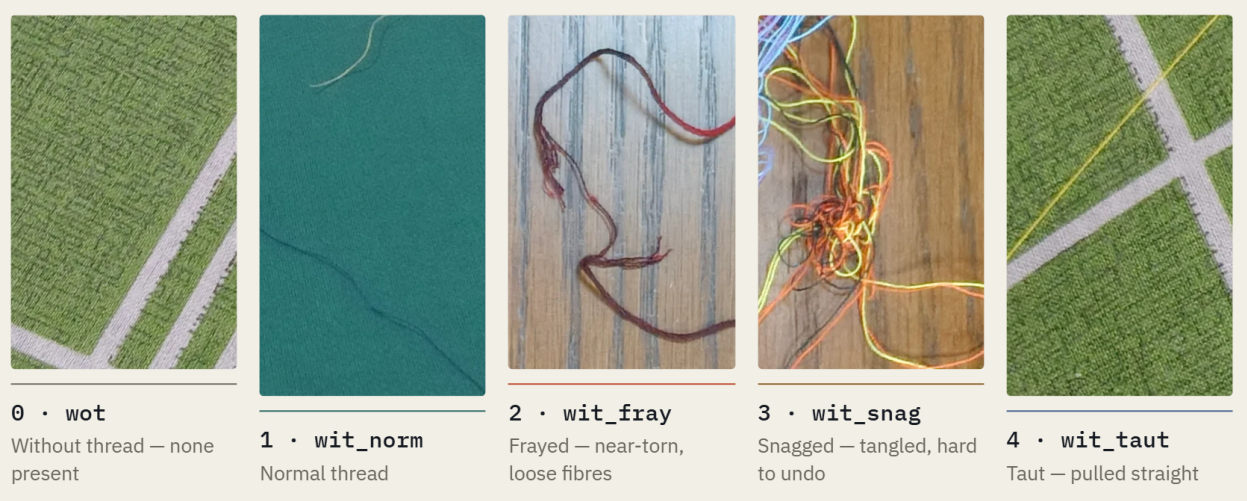


Figure 3.4: Visual samples of the four positive thread states and of a thread-absent negative control.

1. `no_thread` *(derived, not annotated)*: Images used as negative control, showing woven or knitted fabric that carries no target thread event. This condition is not marked by an annotator; it follows from an image having no positive instance, as Section 3.4.3 sets out.

2. `normal`: Physically continuous thread bodies that preserve their integrity along their primary thread axis, representing positive thread control instances.

3. `fray`: Morphological class describing thread observation with fiber separations, unraveling, or splaying that extends from the primary thread body due to mechanical stress or low twist.

4. `snag`: Thread loops or tangled thread bunches that catch or pull above the primary fabric plane, forming geometric abnormalities on the fabric background.

5. `taut`: Thread bodies that appear straightened along a dominant spatial axis, following a visibly direct path across the surface with no slack or waviness. The criterion is the observed geometry of the strand; no force is measured, and the label asserts nothing about tension in the material.

3.4.2 Oriented Region Localization vs. Tight Segmentation

Fine-grained thread structure annotation entails an essential trade-off between fine spatial annotation and dataset scalability. Fine thread structures are ultra-thin foreground elements having only about ~ 2 px wide thread body. Pixel-level instance segmentation masks are impractical to obtain due to the intricate entangled nature of fibers involved in thread snagging and fraying events. On the other hand, conventional axis-aligned HBBs include too much background weave texture which causes excessive visual noise to be fed to the deep learning loss function.
SutaSet formulates the spatial annotation problem in the terms of *oriented region localization*, i.e., enclosing the thread-state event regions with an Oriented Bounding Box (OBB) defined by four corner points instead of segmenting individual fibers at the pixel level. The spatial shape of event regions differs significantly depending on their class: thread-state event regions corresponding to the `taut` class are elongated like lines and have a median aspect ratio of ~ 13.9, whereas the `snag` and `fray` event regions are more compact blobs with a median aspect ratio of ~ 1.6 and median short-side length of 77 px.

Canonical OBB annotations are stored in *pixel* coordinates of the 512×512 image, as four corner points:

$$\mathbf{p}_k = [(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)], \qquad x_i, y_i \in \mathbb{R} \tag{3.2}$$

The four corners follow a canonical clockwise ordering beginning at the top-left corner. Two properties of this representation are deliberate and are stated here because they govern every downstream conversion.

The canonical geometry is unclipped A corner may lie outside $[0, W] \times [0, H]$ when the annotated event continues past the frame edge, and 4,861 of the 9,552 boxes do so. Clipping is performed only at export, where it is recorded, so the canonical record preserves the annotator's claim about the event rather than the visible fraction of it. The unit square appears only in the YOLO-OBB export, where each coordinate is divided by the image width or height; that export, not the canonical record, is what carries normalized values.

The boxes are rotated rectangles, and this is verified rather than assumed Four vertices in convex order do not by themselves constitute a rotated rectangle. Measured across all 9,552 released boxes, adjacent edges are perpendicular to within $|\cos \angle| \leq 7.8 \times 10^{-5}$ (a deviation below 0.005°) and opposite sides match in length to within 10^{-14} relative—so every accepted quadrilateral is a rotated rectangle to the precision at which its coordinates are stored.

Four corners do not remove the angle convention Storing corners means an annotator never supplies an angle, which avoids one class of convention error at annotation time. It does not make the convention irrelevant: the detector converts corners internally to a centre–width–height–rotation parameterization, in which a rectangle has several equivalent descriptions differing by a right-angle relabelling of its corners. The export conversion and its round-trip are therefore specified in Section 5.2 rather than treated as a formatting detail.

3.4.3 Multi-Label Vector Derivation

In order to avoid the problem of label contradiction between image-level classification and instance-level spatial annotations, SutaSet uses instance-level OBB annotations exclusively as the ground truth for the dataset. The multi-label state vector of the image is obtained deterministically based on the instance annotation after it is verified that the annotator has completed the annotation of the whole image.
Given an image I_i, the multi-label state vector $\mathbf{y}_i = [y_{\text{normal}}, y_{\text{fray}}, y_{\text{snag}}, y_{\text{taut}}] \in \{0,1\}^4$ is computed through logical OR operation for all the instances annotated in the image:

$$y_{i,c} = \bigvee_{k=1}^{K_i} \mathbb{I}(c_k = c), \quad \forall c \in \{\text{normal}, \text{fray}, \text{snag}, \text{taut}\} \tag{3.3}$$

where K_i denotes the number of instance annotations in image I_i and $\mathbb{I}(\cdot)$ is the indicator function. When $K_i = 0$, the image receives the all-zero vector $\mathbf{y}_i = [0, 0, 0, 0]$, which is the thread-absent condition (`wot` / `no_thread`).

Co-occurrence is recorded at the image level, not within an instance Each annotation carries exactly one class, so a single physical thread exhibiting two states—a snagged loop whose apex is also frayed—is represented as two boxes over overlapping regions, and no field links them as one event. The multi-label vector therefore records that both states occur *in the image*, which is what the classification task is evaluated on; it does not record that they occur on the same strand. Instance-level multi-label supervision, and the event identifier that would make it expressible, are not part of this release.
Applying this rule across the corpus yields markedly uneven support (Figure 3.5). *Normal* appears on 43.2% of images and *frayed* on only 15.5%, a spread wide enough that a micro-averaged score would be dominated by the majority state; the evaluation therefore reports macro-averaged scores throughout. Co-occurrence is common without being the rule: most positive images carry exactly one state, while 1,091 carry more than one and two carry all four. That is what makes the task genuinely multi-label rather than a single-label choice among five alternatives.

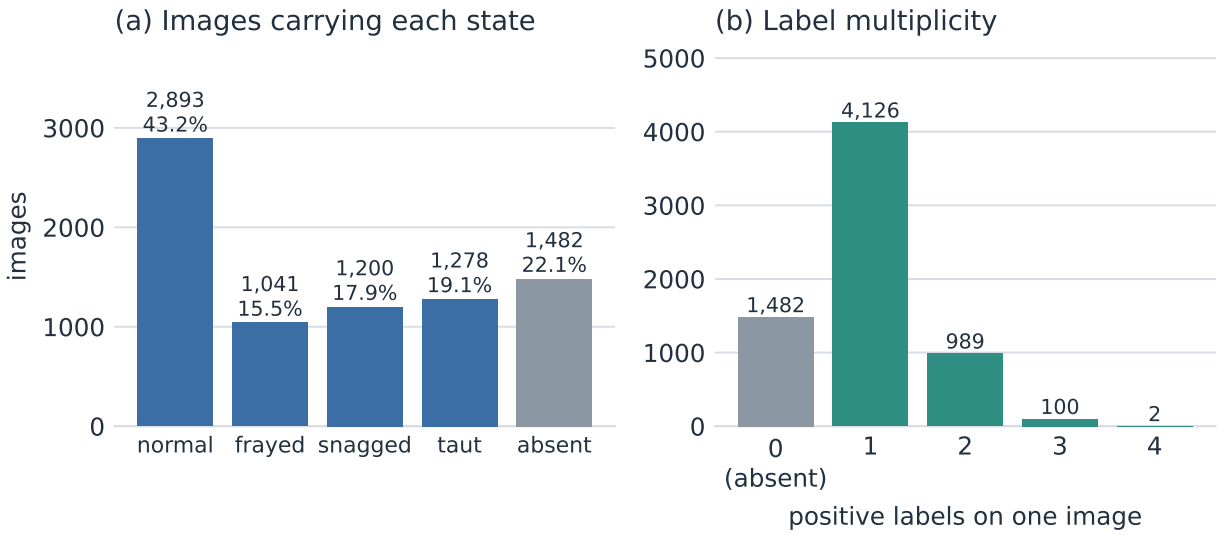


Figure 3.5: Composition of the 6,699-image corpus. (a) Images carrying each target state, with the thread-absent panel shown separately. (b) Number of positive labels on a single image.

4 Data Integrity & Evaluation Protocol

To support reproducibility and to make the integrity properties of the release testable rather than asserted, SutaSet applies a layered quality-control framework. Each gate below states exactly what it verifies; what none of them verifies is set out in Section 8.2. Common visual dataset releases have many problems such as non-verified annotation coordinates, boundary issues that extend out of the frame, sub-pixel annotation problems, and leakage problems caused by split data. SutaSet addresses these problems by using programmatic validation gates, a strict out-of-bounds bounding box correction system, and capture-grouped dataset splitting.

4.1 Quality Assurance & Validation

All derived images and instance annotations are subjected to continuous automated verification before dataset distribution. The automatic quality assurance process carries out a seven-stage validation gate on the entire dataset corpus:

1. **File System Consistency & Checksum Stability:** Guarantees the readability, integrity, and consistency of the canonical SHA-256 hash for each image.

2. **JSONL Schema Verification:** Checks for the validity of JSONL data format according to the canonical schema with all mandatory fields (`image_id`, `sha256`, `width`, `height`, `capture_id`, `split`) filled.

3. **Vertex Topology:** Makes sure that every Oriented Bounding Box (OBB) instance contains exactly four vertices $\mathbf{p}_k = [(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)]$ located in strict clockwise order.

4. **Convexity, Non-Degeneracy & Rectangularity:** Verifies that each bounding box is a strictly convex, non-self-intersecting polygon of strictly positive area (Area $>$ 0 px^2), and that it is a rotated *rectangle*—adjacent edges perpendicular and opposite edges equal—rather than an arbitrary convex quadrilateral (Section 3.4.2 reports the measured tolerance). Instances with sub-pixel annotation artifacts (Area ≤ 1 px^2) are detected and excluded.

5. **Taxonomic Validity:** Checks for valid class labels (`no_thread`, `normal`, `fray`, `snag`, `taut`) and makes sure that images annotated with `no_thread` have no instance annotations.

6. **Label Consistency:** Checks exact equivalence between each multi-label image vector and the set of instance labels it is derived from. This verifies that the two task views

agree with one another; because the vector is *computed* from the instances, it is a check on the derivation, and it establishes nothing about whether the instances themselves are complete or correct (Section 8.4.13).

7. **Disjoint Splits:** Asserts zero overlap of image identifier, content hash, or source-photograph group across training, validation, and test splits.

Sub-pixel annotations and corrupted geometric boxes were identified during automated sweeps and excluded at build time, with an exclusion record preserved for each.

An excluded box is not evidence of an absent thread Invalid *geometry* and absence of a *target event* are different findings, and conflating them would manufacture false negatives: an image whose only annotation was rejected for being sub-pixel would otherwise pass into the corpus as a verified thread-absent control while still visibly containing a thread. The exclusion record is therefore reviewed against the derived labels, and the thread-absent condition additionally requires the batch-level review attestation described below, so an image cannot acquire that condition merely by losing its annotation. Images affected by an exclusion are reannotated or withheld rather than relabelled.

4.2 Partitioning & Leakage Control

Spatial patch cropping induces a critical problem of visual data leakage for computer vision benchmarks. With parents divided into multiple 512×512 patches, standard random split will distribute the sub-patches obtained from the same parent photo across train, validation, and test splits. Due to the fact that the neighboring patches will have identical fabric weave patterns, background textures, lighting profiles, and camera sensor noise signatures, the deep neural networks will learn to memorize the background information rather than acquiring generalized representations of thread defects, causing highly optimistic test accuracy.

SutaSet therefore groups at the level of the source photograph. Because the terms are used loosely in the literature, the hierarchy is stated explicitly, from coarsest to finest:

1. **Physical specimen** — the piece of fabric itself. *Not recorded.* No specimen identifier exists in the release, so the corpus cannot tell whether two photographs show the same cloth.

2. **Acquisition campaign** — one of the five capture phases, sharing approximate date, location, and equipment. Recorded, and represented across all three partitions by design; campaigns are *not* disjoint across splits.

3. **Source photograph** — one exposure, identified by `capture_id`. **This is the unit the split is disjoint on.**

4. **Derived image** — one 512×512 image produced from a source photograph by one transform, identified by the SHA-256 of its bytes.

The assignment is made on a key at least as coarse as the source photograph: `capture_id` unioned with the near-duplicate relation of Section 7.4.1, so that two photographs found to be near-duplicates of one another are placed together as well. Every image derived from one source photograph is therefore assigned to exactly one partition. The condition established is *photograph disjointness*; campaign disjointness and specimen disjointness are not established, and the evaluation should not be described as though they were.
The official dataset split consists of 6699 unique 512×512 images and is divided into the following partitions:

- **Training Set (60%):** 4025 images from 509 parent capture groups.
- **Validation Set (20%):** 1348 images from 150 parent capture groups.
- **Test Set (20%):** 1326 images from 139 parent capture groups.

Split was generated using a greedy assignment procedure balancing class distribution subject to the photograph-grouping constraint. The build audit verifies zero intersection of source photographs between splits and zero byte-identical duplicates by content hash.

4.2.1 The Evaluation Track: Why Reported Results Use 986 of the 1326 Test Images

The partition sizes above describe the *released* dataset. The experiments in Chapter 7 evaluate a defined subset of the test partition, and because that distinction is consequential for reading every reported number, it is specified here rather than left to be inferred.
One of the three derivation transforms, centroid-based smart-cropping (Section 3.3.1), places its 512×512 window *centred on an annotation by construction*. An image produced that way supplies the model with a localization prior that no deployment would provide: the target is at the centre of the frame because the cropping algorithm put it there. Scoring such images alongside arbitrarily framed ones would credit the model for a property of the cropper. Every image is therefore assigned an evaluation track by a rule fixed in the builder, before any model was trained:

$$\texttt{eval_track}(i) = \begin{cases} \texttt{oracle_diagnostic}, & \text{if } \texttt{transform}(i) = \texttt{smartcrop} \\ \texttt{main}, & \text{otherwise} \end{cases} \tag{4.1}$$

The **main track** is the benchmark: all reported classification and detection results are computed on it, and the data loader returns it by default. The **oracle-diagnostic track** is retained and released—it is a legitimate centred-object condition, and the gap between the two tracks is itself informative—but it is reported separately and never pooled into a headline score. Table 4.1 accounts for every image and every photograph under this rule.
The 340 test images outside the main track are exactly the smart-cropped ones, and the eight photographs that appear in the test partition but not in the main-track photograph count are those whose test images are *all* smart-crops. The same rule accounts for the training and validation figures that appear in the learning curves of Chapter 6: the models are fitted on 3,023 training images and model selection uses 1,022 validation images, not the full 4,025

Table 4.1: Reconciliation of the released partitions with the evaluated populations. The track rule is applied per image; a photograph appears in both track columns when it contributed images under more than one transform, so the track photograph counts sum to more than the partition total.

	Main track		**Oracle-diagnostic**		**Released partition**	
Partition	Images	Photos	Images	Photos	Images	Photos
Training	3,023	464	1,002	295	4,025	509
Validation	1,022	138	326	95	1,348	150
Test	986	131	340	89	1,326	139
Total	**5,031**	—	**1,668**	—	**6,699**	**798**

and 1,348. No image is discarded by this rule—the 1,668 smart-cropped images are released and can be evaluated—and the rule references only the transform that produced an image, never its labels or any measured performance.

4.2.2 Provenance & Grouping Logic

The grouping key is recovered for each image by a deterministic parser over its recorded provenance, at the levels set out above. Two of those levels are keys the parser emits: `capture_id`, identifying the source photograph, and `image_id`, the SHA-256 of the derived image's bytes. Every image produced from one photograph—by downscaling, smart-cropping, or grid tiling—carries that photograph's `capture_id`.
Partition assignment is a function of `capture_id` (unioned with the near-duplicate relation), never of `image_id`, which is what places all images from one photograph in a single partition. The capture campaign is recorded for each image but is *not* a partitioning key: campaigns are deliberately represented across all three partitions, so that the evaluation is not also a test of transfer between campaigns. A build gate asserts the resulting disjointness rather than trusting the parser.

4.2.3 Data Leakage Audit

Two distinct questions are addressed under the heading of leakage, and they are kept separate throughout. The first is *structural*: does the split place images from one source photograph on both sides? That is settled by construction and verified by the build audit. The second is *empirical*: how much would a naive random split have inflated the reported scores? That comparison requires training under both protocols, and it has not been run for this release; it is specified as future work in Section 8.4.6, and no inflation figure is reported here.

With a naive random split, tiles from one source photograph are assigned to both train and test sets. Because those tiles share weave pattern, illumination gradient, and sensor noise, a model can score well by recognizing the background rather than the thread event, and the resulting test metric measures memorization as well as generalization—the mechanism

documented for patch-based corpora generally [15, 21]. Photograph-grouped assignment closes that pathway by construction. How large the effect would be *on this corpus* is a quantity this release does not measure.

5 Baseline Methodologies

In order to demonstrate technical validation and the empirical soundness of the SutaSet benchmark, evaluation tasks are established following strict evaluation procedures. The main objective of these baseline models is not the tuning of various hyperparameters but rather to confirm that the visual thread state annotations are consistent and learnable signals. Methods for baseline testing include two types of task definitions: multi-label image classification (§5.1) and oriented region localization(§5.2).

5.1 Multi-Label Image Classification Architectures

The image classification problem tests the ability of the model to detect co-existing morphological thread conditions using only global features.

Mathematical Formulation & Loss Function

Let $X \in \mathbb{R}^{H \times W \times 3}$ be the input image with the spatial dimension of 512×512. The task at hand is to classify the image X into a C-dimensional multi-label binary vector $y = [y_1, y_2, \ldots, y_C]^T \in \{0, 1\}^C$, where $C = 4$ represents the four thread-positive classes: `wit_norm` (normal), `wit_fray` (frayed), `wit_snag` (snagged), and `wit_taut` (visually taut). The absent thread class (`wot` / `no_thread`) is represented when $y = [0, 0, 0, 0]^T$.

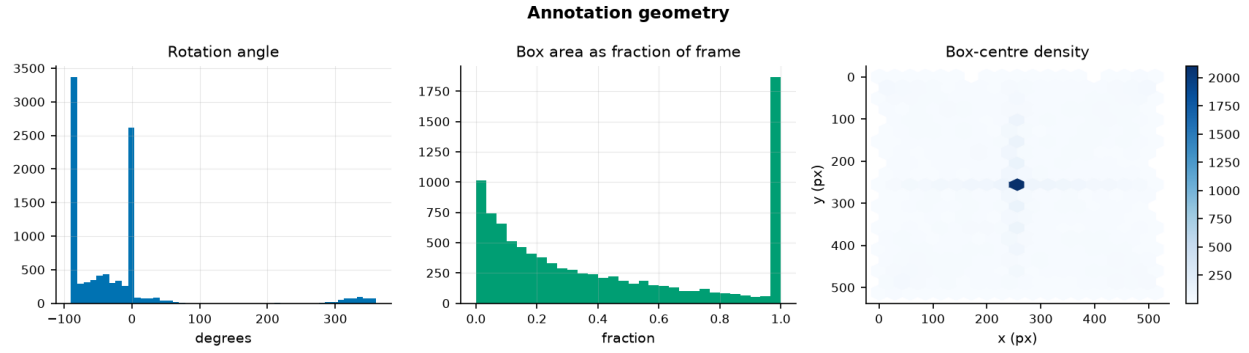


Figure 5.1: Empirical geometric distributions of Oriented Bounding Boxes (OBB).

As there may be several different states of threads in one single image at a time, each class prediction is considered as a separate binary classification problem. The output from the network takes the form of logits $z = [z_1, z_2, \ldots, z_C]^T \in \mathbb{R}^C$. The loss function used for optimization is $\mathcal{L}_{\text{BCE}}$:

$$\mathcal{L}_{\text{BCE}}(y, z) = -\frac{1}{C} \sum_{c=1}^{C} \left[w_c \cdot y_c \log \sigma(z_c) + (1 - y_c) \log(1 - \sigma(z_c)) \right]$$

where $\sigma(z_c) = \frac{1}{1+e^{-z_c}}$ is the element-wise sigmoid activation function yielding class probabilities $\hat{p}_c = \sigma(z_c) \in [0, 1]$.
In order to tackle class imbalance without any validation leakage, the weight of positive class w_c for each category c is calculated purely from the training split ($\mathcal{D}_{\text{train}}$):

$$w_c = \frac{|\mathcal{D}_{\text{train}}| - N^+_{\text{train},c}}{\max(N^+_{\text{train},c}, 1)}$$

Here, $N^+_{\text{train},c}$ represents the number of positive examples of class c in the training split. Calculation of w_c from training data alone avoids any label information from evaluation set leaking into training loss.

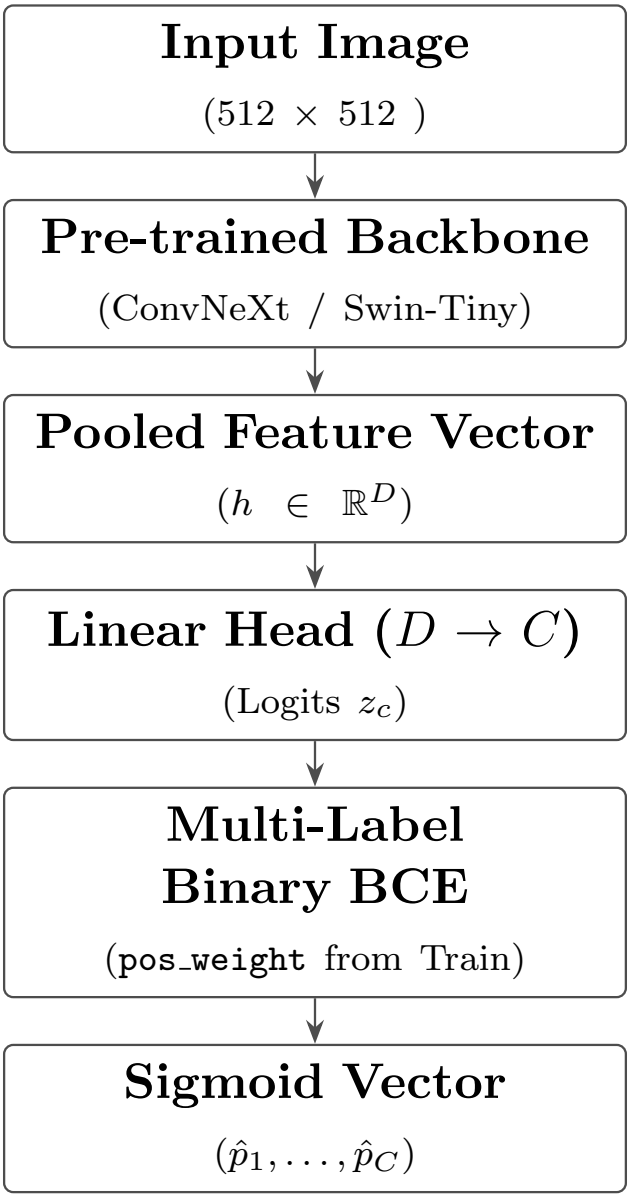


Figure 5.2: Multi-Label Classification Pipeline Architecture.

5.1.1 ConvNeXt-Tiny Implementations

In order to perform comparisons of feature extraction schemes, both pure convolution-based and hybrid schemes using Convolution and Attention backbones have been used for comparison.

5.1.1.1 ConvNeXt-Tiny Architecture

ConvNeXt-Tiny is a more recent pure convolution scheme which aims to align traditional ConvNets with Vision Transformers while preserving the efficiency of pure convolution.

Structural Design ConvNeXt-Tiny utilizes depth-wise separable 7×7 convolutions to increase the receptive field size, inverted bottleneck architecture (increase in number of channels by $4\times$ followed by projection), LayerNorm over BatchNorm, and GELU activations.[32]

Implementation Details The backbone is pre-trained on ImageNet-1K using (`ConvNeXt_Tiny_Weights.IMAGENET1K_V1`) pre-trained weights. The initial 1,000 class classification layer is modified to use a custom linear layer $f_\theta : \mathbb{R}^{768} \to \mathbb{R}^C$. The model is trained in two stages—an initial frozen-backbone warm-up that optimizes only the classification head, followed by full fine-tuning of the backbone at a reduced learning rate—using the AdamW optimizer ($\text{lr} = 10^{-3}$, weight decay $= 10^{-4}$).

5.1.2 Swin-Tiny Vision Implementations

To address spatial information preservation and hierarchical visual attention, high-resolution and windowed transformer networks are incorporated into the classification suite.

5.1.2.1 Swin-Tiny (Swin Transformer v1/v2)

Swin-Tiny is a hierarchical Vision Transformer that computes self-attention within local shifted windows.

Structural Design Standard Vision Transformers (ViT) incur quadratic computational complexity $\mathcal{O}(N^2)$ with respect to image patch count N. Swin Transformer restricts Multi-Head Self-Attention (W-MSA) to non-overlapping local 7×7 windows, achieving linear complexity $\mathcal{O}(N)$. To enable cross-window information exchange, successive blocks alternate between standard windowing and Shifted Windowing (SW-MSA)[22].

Implementation Details Input images at resolution 512×512 are split into 4×4 pixel patches. The Swin-Tiny backbone operates across four hierarchical stages with patch merging layers, producing multi-scale representation maps that are aggregated via global average pooling into a final linear projection head for multi-label binary prediction.

Table 5.1: Summary of Classification Baselines

Architecture	**Backbone Type**	**Key Architectural Mechanism**	**Primary Feature Advantage for Thread Inspection**
ConvNeXt-Tiny	Modernized ConvNet	7×7 Depthwise Convolutions, Inverted Bottleneck	Robust spatial locality, high training stability
Swin-Tiny	Hierarchical Transformer	Shifted Window Self-Attention (W-MSA / SW-MSA)	Local windowed attention with cross-window exchange, at linear cost in image area

5.2 Oriented Region Localization Architecture

Whereas multi-label classification (§5.1) determines which states are present anywhere in the image, oriented detection localizes each target event with a rotated region envelope—not a pixel-tight boundary. The use of conventional HBB (axis-aligned horizontal bounding boxes) becomes inappropriate for the task due to low signal-to-noise ratios caused by the fact that a diagonal thread will make an HBB cover a large portion of the fabric background.

Bounding Box Parameterization Each thread target is specified in terms of a four-cornered oriented bounding box $B_{\text{obb}} = \big((x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)\big)$, which is a four-cornered polygon characterized by four coordinate tuples in 2D image space $\mathbb{R}^2$, sorted canonically in a clockwise direction beginning with the top left corner. In terms of normalized YOLO-OBB model input format, coordinates are normalized to image space $[0, 1]^8$ with respect to the input width W and height H:

$$\bar{x}_i = \frac{x_i}{W}, \quad \bar{y}_i = \frac{y_i}{H}, \quad i \in \{1, 2, 3, 4\} \tag{5.1}$$

Alternatively, the rotated rectangle is described by center (x_c, y_c), width w, height h, and rotation angle θ.

Export conversion and angle equivalence The canonical record stores corners in pixel coordinates (Section 3.4.2); the export divides each coordinate by W or H and clips to the unit square, recording which boxes were truncated. The detector then converts corners internally to the centre–width–height–rotation form above. That conversion is not injective: relabelling the corners by a right angle yields the same rectangle with w and h exchanged and θ shifted, so several parameter tuples describe one geometric box. Corner storage removes the need for an *annotator* to choose an angle convention, but it does not remove the convention from the pipeline, and the export is round-tripped—corners to normalized labels and back—as a build check so that a convention mismatch would surface as a geometric discrepancy rather than as an unexplained loss of detection accuracy.

Detector Architecture The initial detector uses the YOLO11n-OBB architecture, which is a new anchor-free one-stage object detector designed for rotated bounding box regression. This detector is equipped with the modified DarkNet backbone and Cross-Stage Partial (CSP) connections and Spatial Pyramid Pooling Fast (SPPF) layers to obtain multi-scale visual features on 1/8, 1/16, and 1/32 downscale ratio. The use of Path Aggregation Network (PANet) with Feature Pyramid Network (FPN) architectures helps to fuse features between low-level spatial features (fine fiber strands) and high-level semantic features (complex thread tangles). In addition, the detector is equipped with the decoupled detection head with the independent branches for classification and regression tasks.

Loss Functions The detector is trained through an advanced multi-task loss that incorporates object classification, regression for spatial orientation, and refinement of bounding box distribution:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{box}}\mathcal{L}_{\text{probiou}} + \lambda_{\text{cls}}\mathcal{L}_{\text{BCE}} + \lambda_{\text{dfl}}\mathcal{L}_{\text{DFL}} \tag{5.2}$$

Intersection-over-Union (IoU) calculation for rotated rectangles has the problem of non-differentiability and gradient discontinuity when the ratio $w/h > 10:1$. The solution to this problem is introduced by Probabilistic IoU (Probiou) through modeling of the rotated bounding boxes using 2D Gaussian distributions $\mathcal{N}(\mu, \Sigma)$:

$$\mu = \begin{bmatrix} x_c \\ y_c \end{bmatrix}, \quad \Sigma = R(\theta) \begin{bmatrix} \frac{w^2}{12} & 0 \\ 0 & \frac{h^2}{12} \end{bmatrix} R(\theta)^T \tag{5.3}$$

where $R(\theta)$ is the 2D rotation matrix. The distance between the predicted distribution, $\mathcal{N}_1$, and the ground truth distribution, $\mathcal{N}_2$, is determined by the Bhattacharyya distance, B_d, which is further transformed into a continuously differentiable loss $\mathcal{L}_{\text{probiou}} = 1 - \text{IoU}_{\text{prob}} \in [0, 1]$. BCE loss $\mathcal{L}_{\text{BCE}}$ is calculated based on predicted class confidence outputs across feature pyramid scales, while Distribution Focal Loss ($\mathcal{L}_{\text{DFL}}$) enforces fine-grained regression around predicted coordinate locations.

Negative Sample Suppression An essential requirement for the SutaSet baseline localization workflow is the explicit treatment of images without any threads (`wot` / `no_thread`). In such cases, the `thread_present` field is set to `false` in the standard manifest file and the image is explicitly labeled using an empty label text file (`.txt` file that has zero bytes) in the YOLO-OBB output directory. Using explicit empty label files, as opposed to ignoring image files, makes sure that the detection model sees negative samples during training.

Evaluation Metrics The oriented detection results are measured according to standard metrics used for object oriented detection. In this case, the most important metric is Mean Average Precision at the Oriented IoU of 0.50 ($\text{mAP@0.50}_{\text{OBB}}$):

$$\text{mAP@0.50}_{\text{OBB}} = \frac{1}{C} \sum_{c=1}^{C} \text{AP}_{50,c} \tag{5.4}$$

supplemented by $\text{mAP@0.50–0.95}_{\text{OBB}}$ integrated across 10 IoU thresholds ranging from 0.50 to 0.95. Predictions are filtered at a fixed confidence operating threshold conf = 0.25 and Non-Maximum Suppression threshold $\text{NMS}_{\text{IoU}} = 0.70$ for precision, recall, and F1 calculation, while full PR-curves are integrated using confidence cutoff conf = 0.001.

6 Implementation Details

6.1 Dataset Manifests & Execution

Dataset Manifests and Structure The release dataset comprises immutable, content-checksummed tabular records:

- `manifest.csv`: One row per unique image. It includes boolean indicator columns for the four positive thread states (`wit_norm`, `wit_fray`, `wit_snag`, `wit_taut`), image appearance statistics (brightness, contrast, color, blur), the source-photograph grouping key (`capture_id`), the partition (`split`), and the evaluation track (`eval_track`, Section 4.2.1).
- `annotations.csv`: Foreign-keyed to `manifest.image_id`. It contains exhaustive object-level Oriented Bounding Box (OBB) geometry in eight float columns ((x_1, y_1) through (x_4, y_4)).
- `split_assignment.csv`: Maps each image to its partition and to its source photograph grouping key (`capture_id`); the materialized detection export additionally carries a `groups.csv` beside its data configuration so that grouped intervals remain computable from inside an export.
- `receipt.json` & `croissant.json`: Build metadata include random seeds, git commit hashes, input and output SHA-256 hashes, and machine-readable Croissant 1.1 dataset metadata.

The split ratio for the dataset is fixed at 60% / 20% / 20% (4025 images for training, 1348 images for validation and 1326 images for testing out of total 6699 512×512 PNG images). The group-based capture constraint ensures that any two images formed from the same raw input image belong to the same set only.

6.2 Model Training & Setup

6.2.1 Multi-Label Classification Training

The multi-label classification baselines evaluated here are **ConvNeXt-Tiny** and **Swin-Tiny**, fine-tuned using two-stage transfer learning: a frozen-backbone warm-up followed by full fine-tuning of the backbone. (**CoAtNet-0** and **HRNet-W18** are earmarked for future work.)

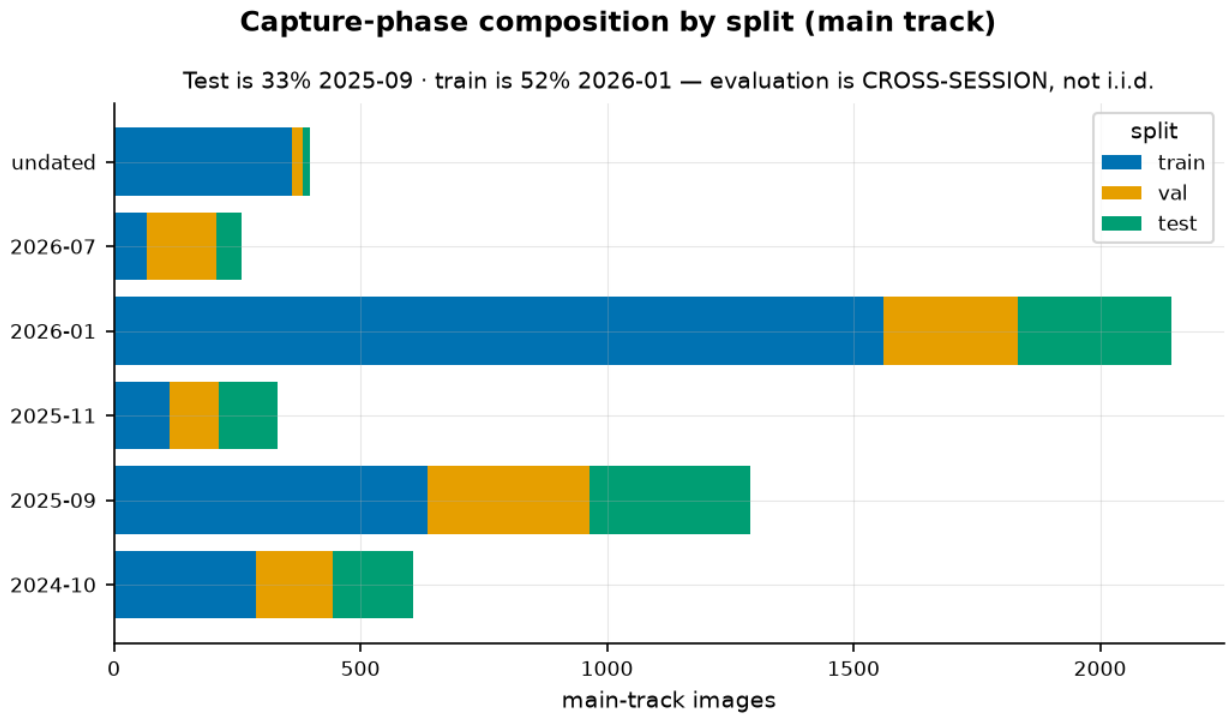


Figure 6.1: Dataset composition across capture phases and partitions

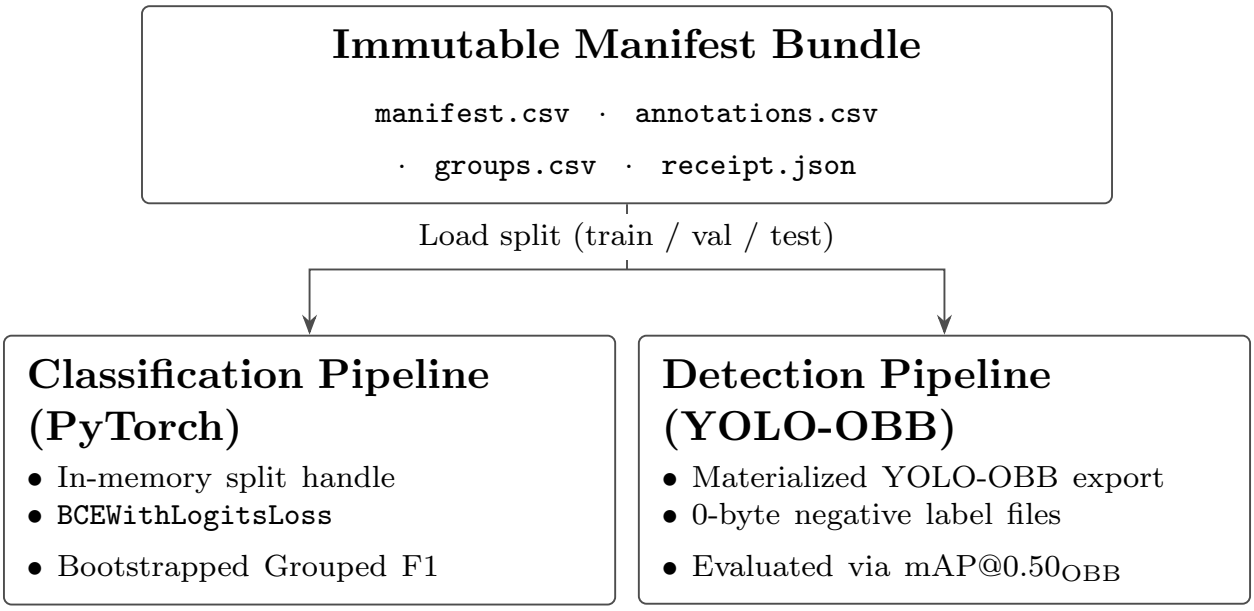


Figure 6.2: Dataset Manifest and Access Flow Architecture.

Two-Stage Fine-Tuning Training proceeds in two stages. A warm-up stage freezes the ImageNet-1K backbone and optimizes only the newly defined classifier ($f_\theta : \mathbb{R}^D \to \mathbb{R}^4$); a subsequent stage unfreezes the backbone and fine-tunes it end-to-end at a reduced learning rate, so the reported model is not restricted to ImageNet features.

Optimization & Schedule All models are trained for 30 epochs using the AdamW optimizer with a base learning rate of $\eta = 10^{-3}$ and weight decay of $\lambda = 10^{-4}$. The mini-batch size is 32. Input images are used at their native $512 \times 512 \times 3$ resolution and normalized by the ImageNet mean and standard deviation ($\mu = [0.485, 0.456, 0.406]$, $\sigma = [0.229, 0.224, 0.225]$).

Loss Weighting Training is done with `BCEWithLogitsLoss` where positive class weights (w_c) are dynamically computed based on the training dataset ($\mathcal{D}_{\text{train}}$) in order to weight the gradient and avoid label frequency leakage from the validation set.

Data Augmentation on the Fly Data augmentation is done exclusively online on the training set. All data augmentations are performed using Albumentations and are seeded for each epoch:

- `HorizontalFlip` ($p = 0.5$), `VerticalFlip` ($p = 0.5$), and `RandomRotate90` ($p = 0.5$)
- Affine rotation ($\pm 15°$) and scaling ($0.9 \times$ –$1.1\times$, $p = 0.5$)
- `RandomBrightnessContrast` (brightness/contrast limit ± 0.2, $p = 0.5$)
- `HueSaturationValue` (hue limit ± 5, saturation ± 15, value ± 10, $p = 0.3$)
- `GaussNoise` ($p = 0.2$)

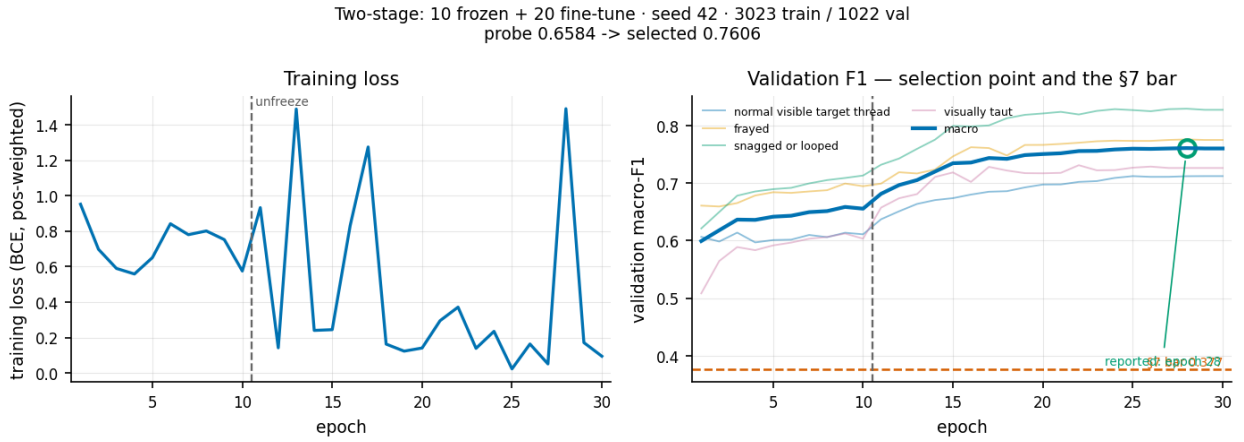


Figure 6.3: Training loss convergence and validation Macro F1 score progression across 30 training epochs.

Every augmented view inherits its parent image's physical grouping key (`capture_id`), ensuring augmented samples cannot cross split boundaries.

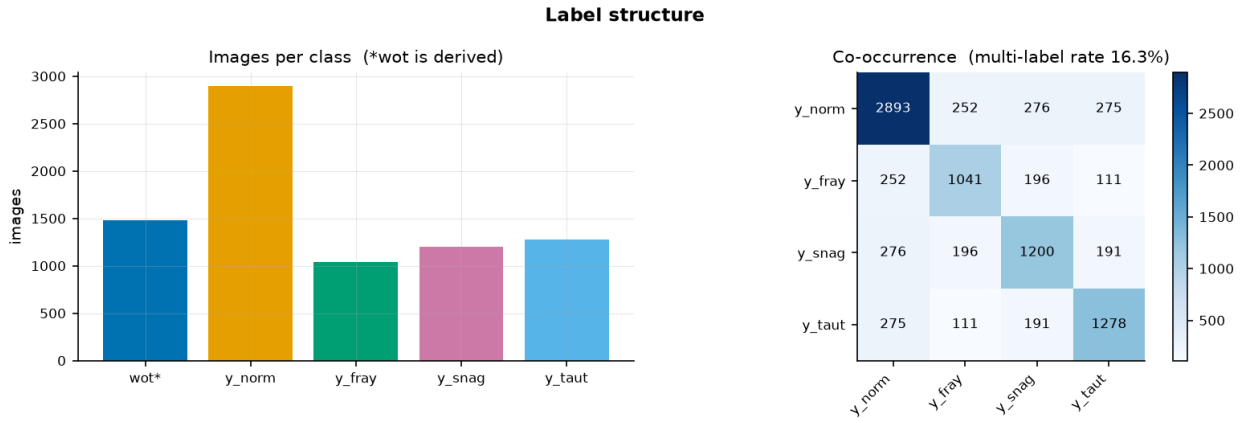


Figure 6.4: Class co-occurrence matrix and label frequency distribution

6.2.2 Oriented Object Detection Training

The oriented regional localization models (**YOLO11n-OBB**) are trained using materialized YOLO-OBB directory exports.

Resolution & Batch Size The images are fed in with a resolution of 512×512 pixels (`imgsz = 512`), and the mini-batch size is set to 32.

Optimizer & Epoch Count Models are trained for 100 epochs using Ultralytics' SGD/AdamW default optimization engine with deterministic training flags enabled (`deterministic = True`).

Checkpoint Selection The final performance assessment is carried out using `best.pt`, the checkpoint Ultralytics selects on a composite fitness score of $(0.1 \cdot \text{mAP@0.50} + 0.9 \cdot \text{mAP@0.50–0.95})$, evaluated on the *validation* split. Selecting a checkpoint by any validation criterion introduces selection optimism with respect to that split; what prevents that optimism from reaching the reported figures is that the selected model is then evaluated once on the test partition, which played no part in the selection. The weighting of the stricter term is a property of the criterion, not a safeguard, and is not claimed as one.

6.2.3 Threshold Calibration & Discipline

Decision thresholds are fitted exclusively on the validation split, so that the test partition contributes to no fitting decision of any kind:

Fitting of Thresholds for Classification Class probabilities $\hat{p} \in [0,1]^4$ are mapped to binary values $\hat{y} \in \{0,1\}^4$ through thresholding $\tau = [\tau_1, \tau_2, \tau_3, \tau_4]$ per class. Fitting of thresholds on the validation set ($\mathcal{D}_{\text{val}}$) is done through searching of optimal thresholding values maximizing Macro F1.

Table 6.1: Hyperparameter Configuration Summary

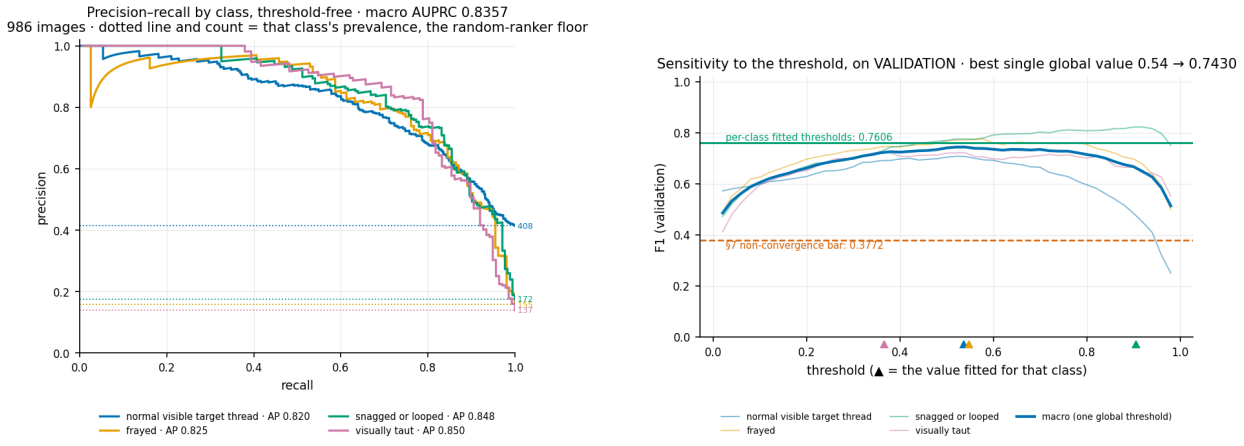
Parameter / Setting	**Classification (ConvNeXt / Swin)**	**Detection (YOLO11n-OBB)**
Input Resolution	$512 \times 512 \times 3$ pixels	$512 \times 512 \times 3$ pixels
Batch Size	32	32
Optimizer & LR	AdamW ($\eta = 10^{-3}$, decay $= 10^{-4}$)	SGD / AdamW ($\eta = 10^{-2}$ default)
Loss Function	`BCEWithLogitsLoss` (weighted)	$\mathcal{L}_{\text{probiou}} + \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{dfl}}$
Augmentation	Online Albumentations (Train only)	Ultralytics Mosaic & Rotated Augmentation
Checkpoint Evaluated	Best val macro-F1	`best.pt` Checkpoint
Seeds configured	[42, 43, 44]	[42, 43, 44]
Seed reported here	42 (single run)	42 (single run)

6.3 Evaluation Metrics

In order to perform a proper evaluation of the fine-resolution thread morphology recognition, it is necessary to have a metric that is resilient to class imbalance, multi-labels, and variable spatial bounding box orientation. The use of basic accuracy measures is strictly forbidden since, based on data quality checks, it is possible for a model which does not recognize any defect to get accuracy higher than 75% simply due to the non-existence of rare classes.

6.3.1 Multi-Label Classification Metrics

For multi-label image classification, models are evaluated using class-unweighted macro-averaged metrics across the $C = 4$ categories (`wit_norm`, `wit_fray`, `wit_snag`, `wit_taut`).



(a) Class-wise PR Curves

(b) Validation Threshold Sweeps

Figure 6.5: Precision-Recall curves and decision threshold sweeps

1. Macro F1 Score (Macro-F1) Suppose that TP_c, FP_c, and FN_c are the True Positives, False Positives, and False Negatives for class $c \in \{1, \ldots, C\}$ at decision threshold τ_c. Then Precision (P_c), Recall (R_c), and F1 score ($\mathrm{F1}_c$) are defined as:

$$\mathrm{P}_c = \frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FP}_c}, \quad \mathrm{R}_c = \frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FN}_c}, \quad \mathrm{F1}_c = \frac{2 \cdot \mathrm{P}_c \cdot \mathrm{R}_c}{\mathrm{P}_c + \mathrm{R}_c} \tag{6.1}$$

The overall benchmark score for classification is calculated using the simple arithmetic average of all positive categories:

$$\text{Macro-F1} = \frac{1}{C} \sum_{c=1}^{C} \mathrm{F1}_c \tag{6.2}$$

The unweighted Macro F1 score allows rare categories like `wit_fray` and `wit_taut` to have equal importance in the final score compared to common categories like `wit_norm`. This avoids cases where perfect classification results in common classes mask the inability to classify rare classes.

2. Macro AUPRC (Area Under Precision-Recall Curve) Due to variability in the criteria for determining decision thresholds according to operating conditions, the model is assessed by threshold-independent area under the Precision-Recall Curve (AUPRC). For each class c, predictions are ranked by score and the curve is summed step-wise—the convention implemented by `sklearn.metrics.average_precision_score`, in which each threshold contributes the precision at that point weighted by the increase in recall it produces:

$$\mathrm{AUPRC}_c = \sum_{n} \left(R_n - R_{n-1}\right) P_n \tag{6.3}$$

This is *not* trapezoidal integration of the precision–recall curve, and the two differ; the figures labelled "AP" report this same quantity. Examples sharing a score are collapsed to a single curve point, since no threshold can separate them—without this, a baseline predicting a constant rate for a class would yield an arbitrary value. A class with no positive instance in the evaluation set is undefined and is omitted from the macro average rather than scored as zero.

$$\text{Macro-AUPRC} = \frac{1}{C} \sum_{c=1}^{C} \mathrm{AUPRC}_c \tag{6.4}$$

Macro-AUPRC is especially appropriate for imbalanced multi-label problems since it measures classifier discriminative ability across all possible decision points without being influenced by the number of true negatives.

6.3.2 OBB Detection Metrics

For oriented region localization, evaluation needs to include both class identification performance and orientation of bounding boxes.

1. Oriented Mean Average Precision at 0.50 IoU (mAP@$0.50_{\mathbf{OBB}}$) Matching uses *probabilistic* IoU, the similarity between the two boxes' Gaussian representations, which is what the detection framework computes for oriented boxes; it is a measure of agreement between those representations and not the ratio of overlapping to combined polygon area. Every detection figure in this report is therefore a probabilistic-IoU-based AP, and should not be read as establishing pixel-tight or boundary-level localization—a geometric polygon IoU would be a separate measurement, and is not reported here. A rotated box B_{pred} counts as correct when its class matches that of a ground-truth object B_{gt} and their probabilistic IoU meets or exceeds 0.50:

$$\text{IoU}_{\text{prob}}(B_{\text{pred}}, B_{\text{gt}}) \geq 0.50 \tag{6.5}$$

Predictions are ordered according to their confidence score, decreasingly. The Precision-Recall curve for each class c is then built by interpolating the standard precision envelope $P_{\text{interp},c}(r)$ which considers the highest precision reached up to any recall value $\tilde{r} \geq r$:

$$P_{\text{interp},c}(r) = \max_{\tilde{r} \geq r} P_c(\tilde{r}) \tag{6.6}$$

The average precision $\text{AP}_{50,c}$ of the class c is calculated by using 101 points numerical integration along the recall axis:

$$\text{AP}_{50,c} = \int_0^1 P_{\text{interp},c}(r)\, dr \tag{6.7}$$

The primary detection benchmark score $\text{mAP@0.50}_{\text{OBB}}$ is the macro-average across all target classes:

$$\text{mAP@0.50}_{\text{OBB}} = \frac{1}{C} \sum_{c=1}^{C} \text{AP}_{50,c} \tag{6.8}$$

2. Multi-Threshold Averaged mAP (mAP@0.50–$0.95_{\mathbf{OBB}}$) In order to test the accuracy of corner localization, in addition to that, the detection algorithms are tested for 10 IoU matching thresholds $\mathcal{T} = \{0.50, 0.55, 0.60, \ldots, 0.95\}$ with a step size of 0.05:

$$\text{mAP@0.50–0.95}_{\text{OBB}} = \frac{1}{10} \sum_{k=0}^{9} \text{mAP}_{0.50+0.05k} \tag{6.9}$$

This penalizes predictions whose oriented envelope agrees with the annotation only loosely, and is sensitive to angular error on elongated boxes in particular.

6.3.3 Grouped Uncertainty Estimation

A point estimate alone—a single Macro-F1 or mAP value—carries no information about how much of its value is sampling variation. Each reported classification score is therefore accompanied by a non-parametric 95% confidence interval obtained by a grouped bootstrap over source photographs (`capture_id`).

Resampling Unit The resampling unit is the source photograph (G_g), not the individual sub-image. Sub-images derived from one photograph share lighting, sensor noise, and fabric background, so they are not independent draws; resampling images rather than photographs would treat correlated observations as independent and yield an interval narrower than the data supports. The magnitude of that narrowing on this corpus is not reported, because establishing it requires computing both intervals under a paired design, which this release does not do.

What these intervals do and do not license Each interval describes the uncertainty of *one* score under resampling of photographs, conditional on the trained model. Two such intervals overlapping, or failing to overlap, is not a test of whether two models differ: that requires resampling the paired per-photograph difference between them. No claim of statistical significance for a model comparison is made anywhere in this report, and the intervals are not evidence for one.

Bootstrap Procedure With respect to M physical photo groups in the evaluation set, $B = 1000$ bootstrap replicates are created through random resampling of M photo groups. In each of the bootstrap samples b where $b \in \{1, ..., B\}$, the statistic θ_b^* (i.e., Macro-F1* or mAP*) is computed.

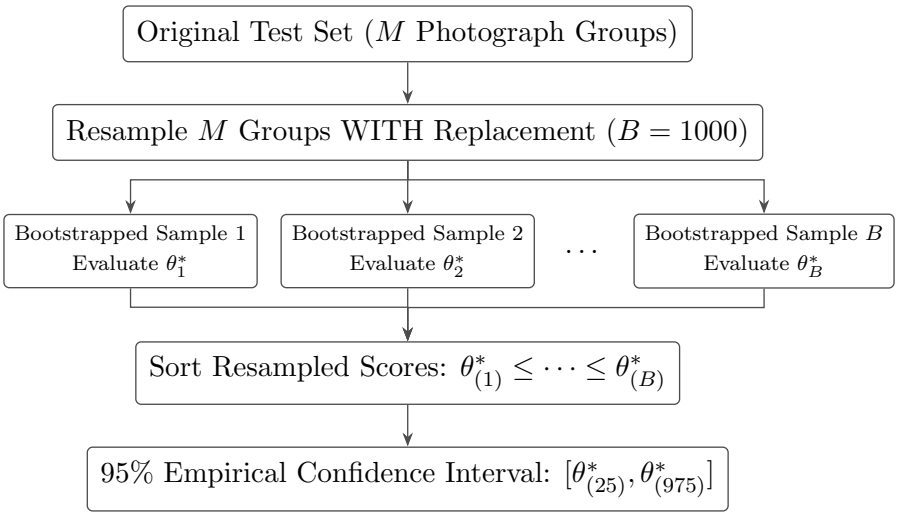


Figure 6.6: Grouped Non-Parametric Bootstrap Resampling Flowchart.

Interval Estimation The 95% confidence interval empirically is found using the 2.5^{th} and 97.5^{th} percentile points in the ordered bootstrap distribution $\{\theta_{(1)}^*, \theta_{(2)}^*, \ldots, \theta_{(B)}^*\}$.

6.3.4 Empirical Baseline Floors

In order to validate that the trained models are building up useful visual representations and not exploiting any artifacts in the data set, each evaluation test is compared against three empirical baselines:

- **All-Zeros Floor:** Predicts $\hat{y} = [0, 0, 0, 0]^T$ for all evaluation images. Establishes the absolute lower bound (Macro-F1 = 0.0).
- **Prior-Frequency Floor:** For each class independently, predicts a positive at random with probability equal to that class's positive rate in the training split, using no image content. It is scored under the same thresholding and seed discipline as the trained models.
- **Appearance-Only Probe:** A logistic regression over four global image statistics—mean brightness, global contrast, mean saturation, and Laplacian blur. These are computed *from* the pixels; what the probe discards is their spatial arrangement, so it is a model of global appearance without structure rather than a model without pixel access. A trained network that failed to beat it would indicate a task solvable from global lighting and colour alone. Clearing it shows the network uses more than these four statistics; it does not establish that no background shortcut of any kind is being exploited.

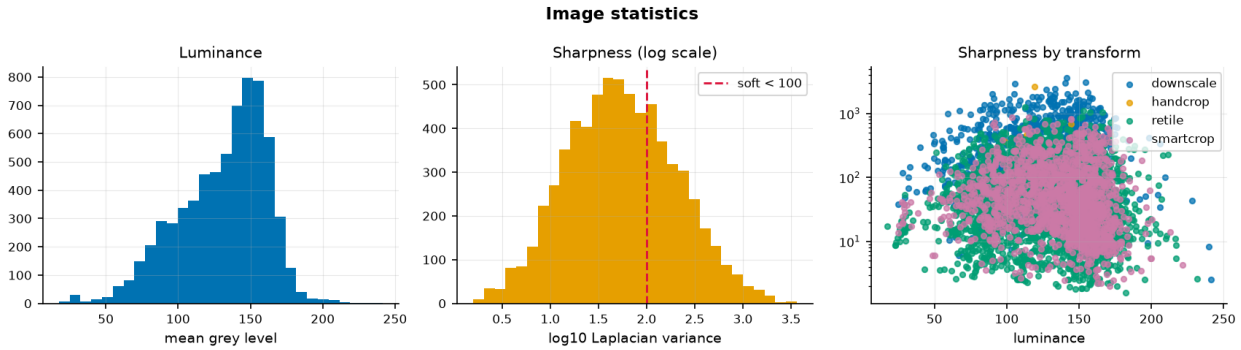


Figure 6.7: Distribution of non-spatial appearance baseline features

A baseline run is treated as demonstrating learning when its validation score exceeds the Prior-Frequency floor by more than the width of its grouped interval.

Table 6.2: Summary of Evaluation Specifications across Benchmark Tasks.

Evaluation Dimension	**Multi-Label Classification**	**Oriented Region Localization**
Primary Metric	Macro-F1 (unweighted class mean)	$\text{mAP@0.50}_{\text{OBB}}$
Secondary Metric	Macro-AUPRC	$\text{mAP@0.50–0.95}_{\text{OBB}}$
Threshold Selection	Validation F1 optimization ($\tau \in [0, 1]^4$)	Fixed operating point (conf = 0.25, NMS = 0.70)
Statistical Bound	Grouped Bootstrap 95% CI ($B = 1000$)	Not reported in this release (Section 7.3)
Resampling Unit	Physical Photograph Group (`capture_id`)	Physical Photograph Group (`capture_id`)
Validation Gate	Must exceed Prior-Frequency floor	Must exceed Predict-Nothing floor (0.0)

7 Results & Benchmark Analysis

7.1 Provenance of the Reported Results

Every figure in this chapter comes from one of three recorded runs, and Table 7.1 identifies them so that any number can be traced to the run that produced it. Three properties of this evidence base should be read before the results themselves.

Table 7.1: The runs behind every reported number. All three evaluate the main track of the test partition (Section 4.2.1) against the same manifest, whose digest is recorded in each run receipt.

Run	Model	Seed	Evaluated on	Schedule
A	ConvNeXt-Tiny	42	986 images, 131 photographs	30 epochs, two-stage
B	Swin-Tiny	42	986 images, 131 photographs	30 epochs, two-stage
C	YOLO11n-OBB	42	986 images, 131 photographs	100 epochs, best checkpoint at 86

One seed, not three Table 6.1 lists three seeds as configured, but each reported number comes from a single run at seed 42. Nothing in this chapter measures variability across training runs, and no statement of seed stability is made. Establishing it requires completing the remaining two seeds per arm; that is outstanding.

Provisional status All three runs are recorded as provisional in their receipts: they are reported here as reference baselines that demonstrate the annotations carry learnable signal, and they are not offered as a frozen benchmark result. Re-running them under the frozen protocol is a precondition for publication.

Test scores, not validation scores Every headline figure in this chapter is computed on the test partition. Validation scores appear only in the learning curves, where they are the quantity used for checkpoint selection, and they are not comparable to the test figures: they are measured on a different population and on the split that selection optimized against. Where a learning curve and a headline figure disagree, they are not in conflict—they are different measurements, and are labelled as such.

7.2 Multi-Label Classification Performance Comparison

The performance of multi-label classification systems is evaluated across complementary architectural paradigms—the modern convolutional backbone `ConvNeXt-Tiny` and the hi-

erarchical vision transformer backbone `Swin-Tiny`—on the main track of the held-out test partition ($N = 986$ images from 131 source photographs; Section 4.2.1 reconciles this with the released partition of 1,326).

7.2.1 Comparative Analysis & Discussion

1. Technical Validation and Learning Verification All tested deep learning systems—including `ConvNeXt-Tiny` (0.7340 Macro-F1) and `Swin-Tiny` (0.7886 Macro-F1)—effortlessly surpass both the Prior-Frequency floor (0.3772 Macro-F1) and the Appearance-Only probe (0.3955 Macro-F1), confirming that the annotations carry signal a model can learn. The margins over the prior floor (+0.3568 for ConvNeXt-Tiny and +0.4114 for Swin-Tiny) establish that the task is not solved by the four global appearance statistics the probe uses. They do not establish that no background shortcut is exploited: the probe bounds one specific family of shortcuts, and a model could still rely on fabric texture correlated with the target state (Section 8.2).

2. Performance Across Backbones: CNNs vs. Vision Transformers

- `ConvNeXt-Tiny` (Convolutional Baseline): Obtains a Macro-F1 of 0.7340 (95% grouped CI [0.690, 0.769]) with a Macro-AUPRC of 0.8357, establishing that a modern convolutional backbone is an effective starting point at modest parameter cost.
- `Swin-Tiny` (Vision Transformer Baseline): Achieves the higher of the two scores, a Macro-F1 of 0.7886 (95% grouped CI [0.748, 0.821]) with a Macro-AUPRC of 0.8497—5.46 percentage points above `ConvNeXt-Tiny`. The two intervals overlap substantially, and no paired test of the difference was performed, so this ordering is the observed result of one run per model rather than a demonstrated ranking. Hierarchical local-window attention with cross-window exchange is a plausible reason a thin continuous structure would suit this backbone, but the comparison varies the entire architecture at once and therefore isolates no mechanism; testing that explanation would require matched ablations of the attention scheme alone.

Both backbones clear every floor by a wide margin, which is the result this chapter rests on. Their ordering relative to one another rests on a single run apiece and is reported as an observation, not as a finding about architectures.

3. Per-Class Morphological State Analysis Comparing `ConvNeXt-Tiny` and `Swin-Tiny` across individual thread state classes reveals key architectural insights:

- **Visually Taut (`wit_taut`):** `Swin-Tiny` achieves F1 = 0.8178 (AUPRC = 0.8892) against F1 = 0.7055 for `ConvNeXt-Tiny`, a difference of 11.23 percentage points—the widest of the four classes. Taut events are long and near-straight, which is the geometry a windowed attention scheme with cross-window exchange might be expected to suit, though this comparison does not test that.

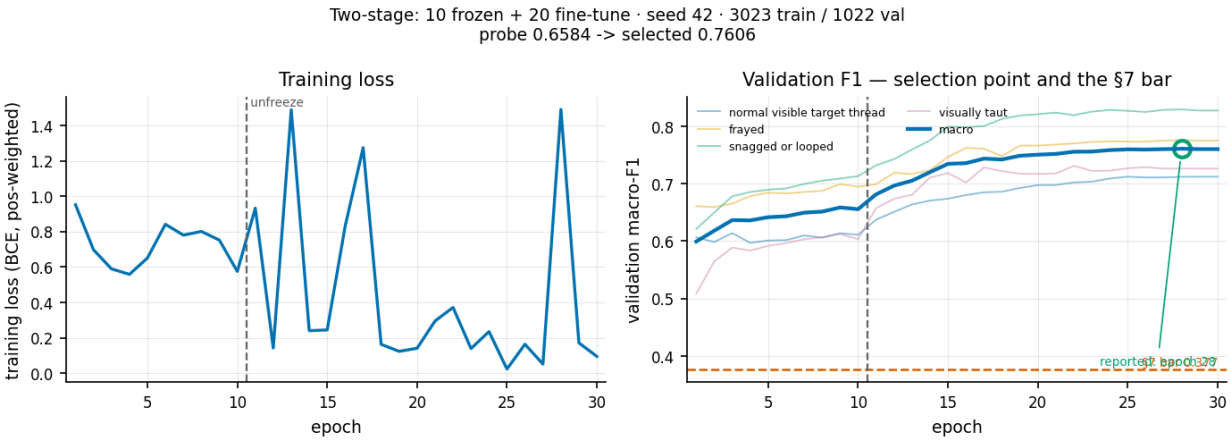


Figure 7.1: ConvNext-Tiny learning curve

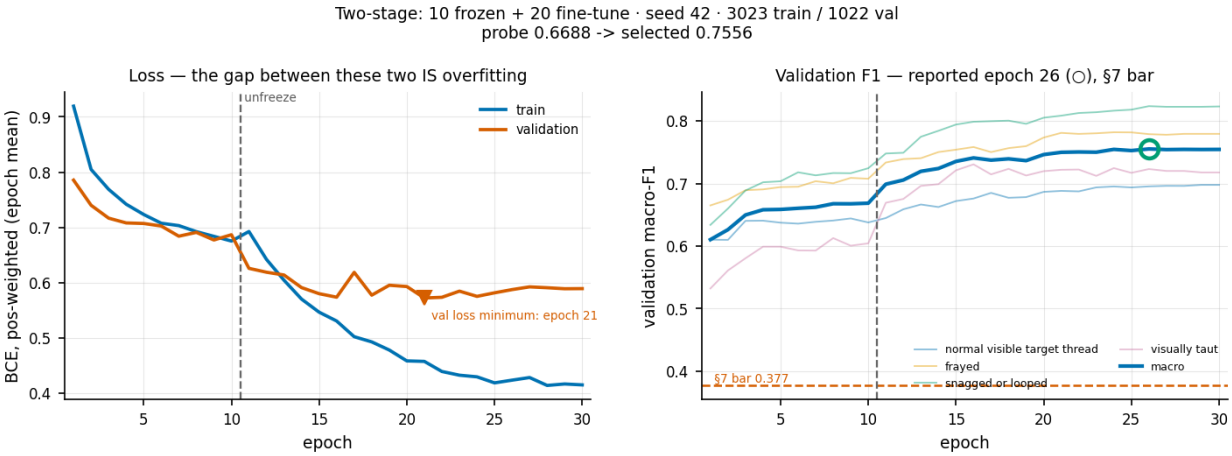


Figure 7.2: Swin-Tiny learning curve

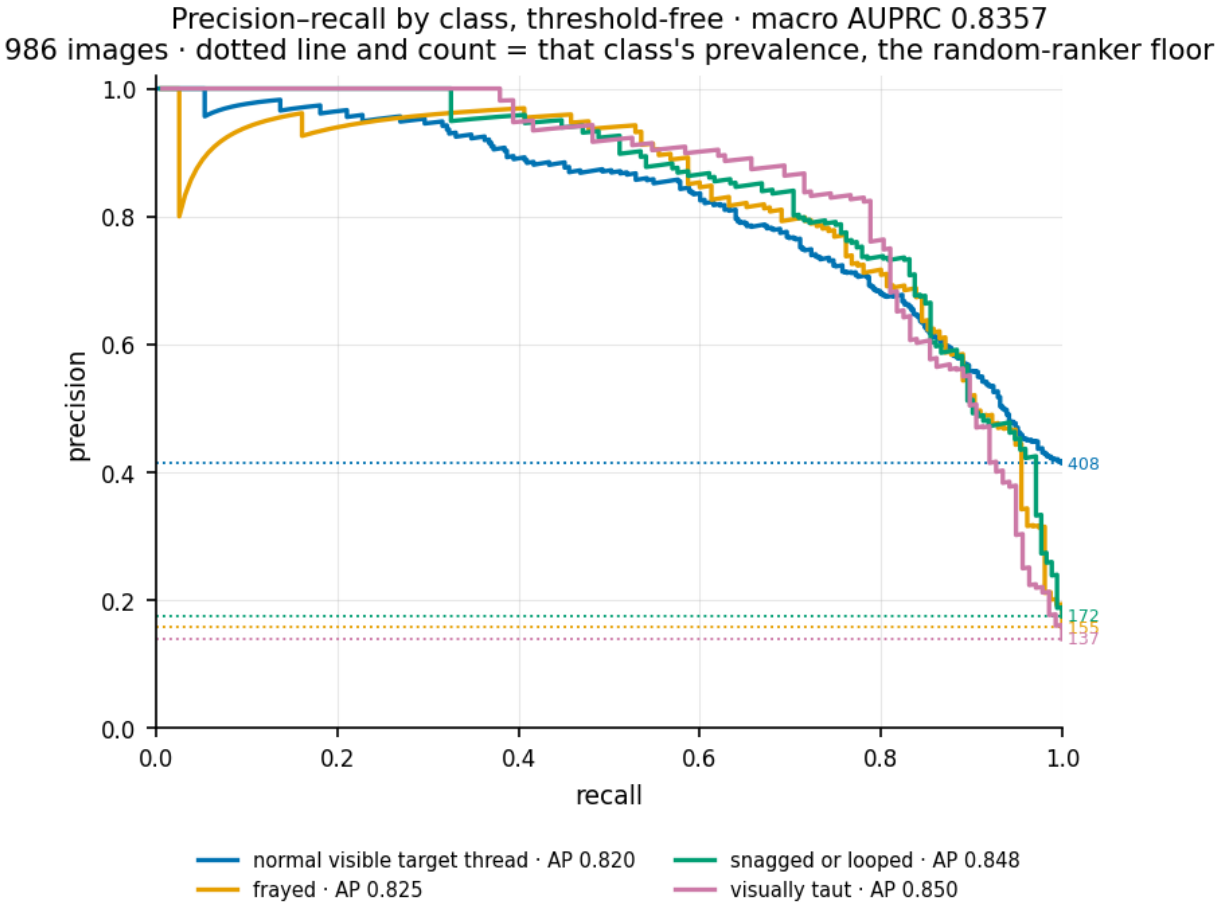


Figure 7.3: ConvNext-Tiny pr curves

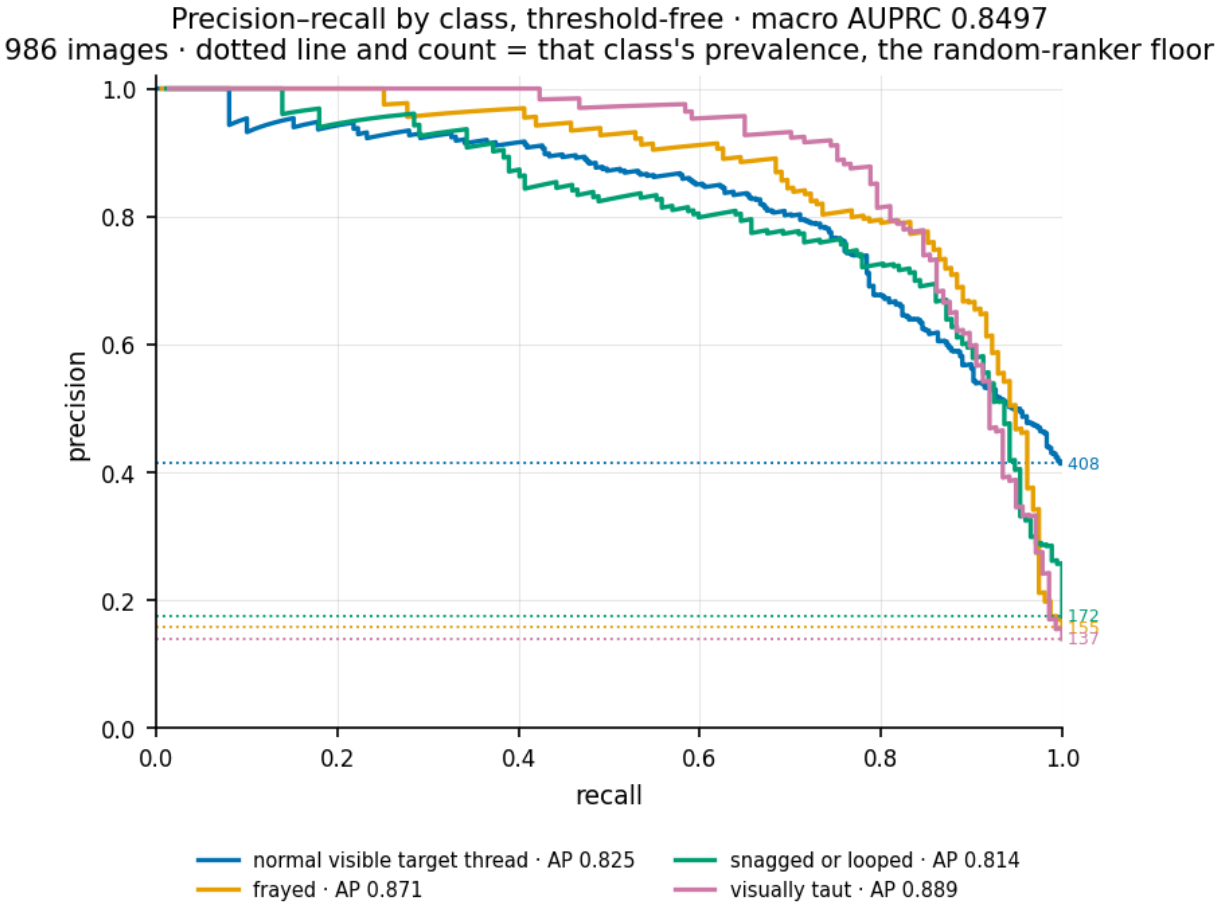


Figure 7.4: Swin-Tiny pr curves

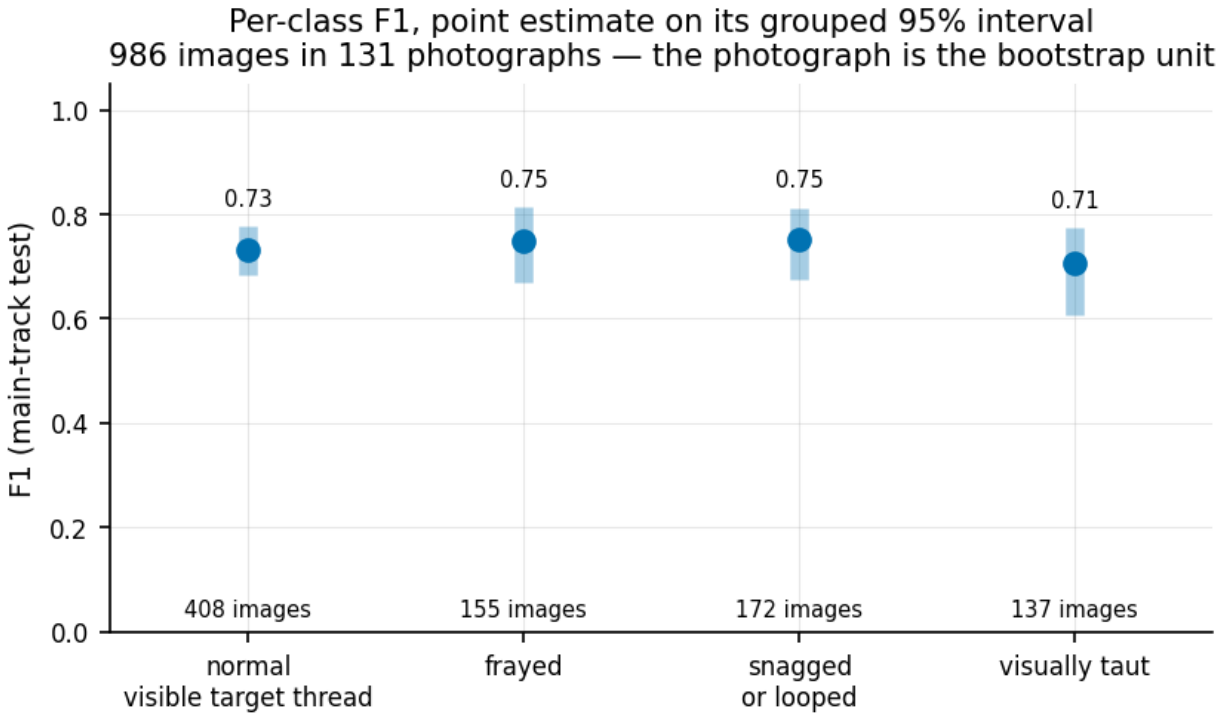


Figure 7.5: Per-class Macro F1 benchmark with 95% CIs ConvNeXt-Tiny

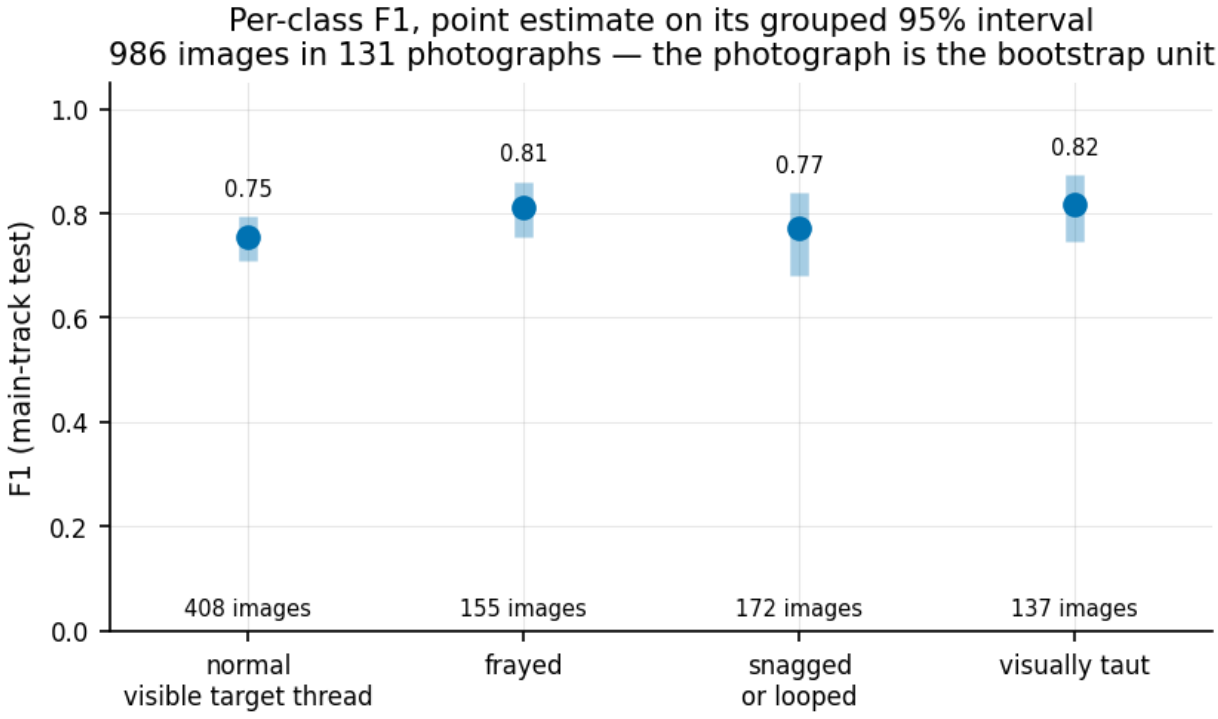


Figure 7.6: Per-class Macro F1 benchmark with 95% CIs Swin-Tiny

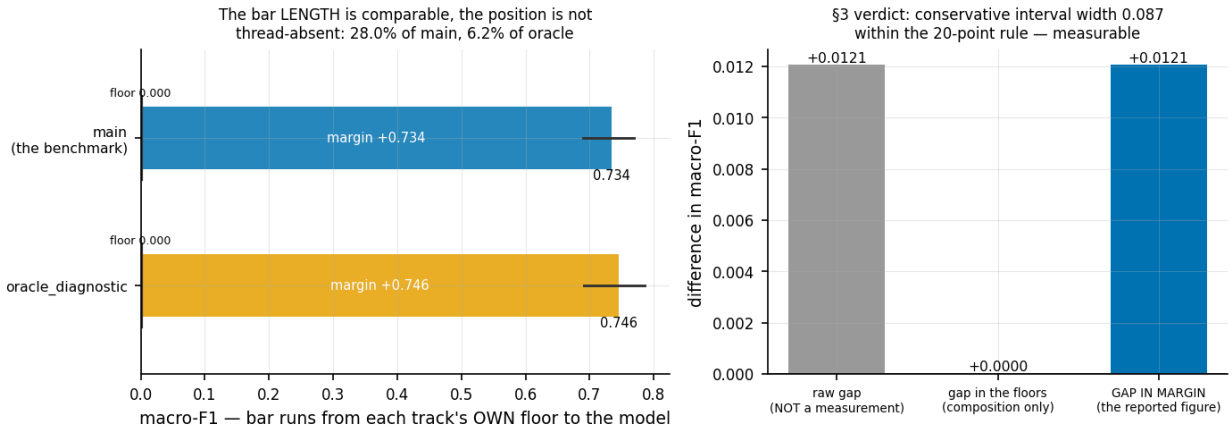


Figure 7.7: Standard (arbitrary-framing) vs. centered-object (oracle) evaluation performance comparison on ConvNeXt-Tiny.

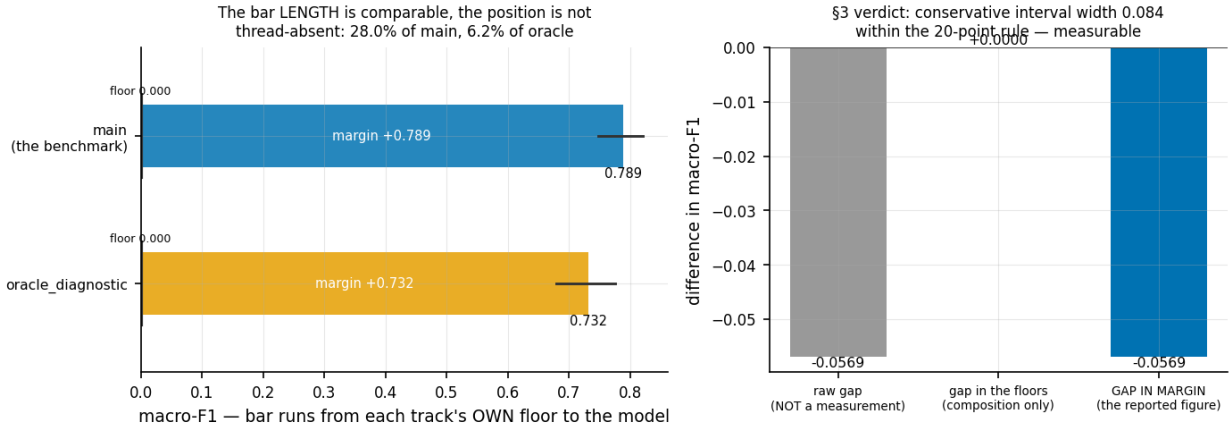


Figure 7.8: Standard (arbitrary-framing) vs. centered-object (oracle) evaluation performance comparison on Swin-Tiny.

- **Frayed Thread (`wit_fray`):** `Swin-Tiny` achieves F1 = 0.8123 (AUPRC = 0.8706) against 0.7478 for `ConvNeXt-Tiny`, a difference of 6.45 percentage points. This is the class with the fewest positives (155 of 986), so both figures rest on the least evidence of the four.
- **Snagged Thread (`wit_snag`):** The two backbones are close, at F1 = 0.7709 (`Swin-Tiny`) and 0.7508 (`ConvNeXt-Tiny`), a difference of 2.01 percentage points.
- **Normal Thread (`wit_norm`):** The most abundant class (408 of 986 positives) and the closest pair, at F1 = 0.7532 (`Swin-Tiny`) and 0.7320 (`ConvNeXt-Tiny`), a difference of 2.12 percentage points.

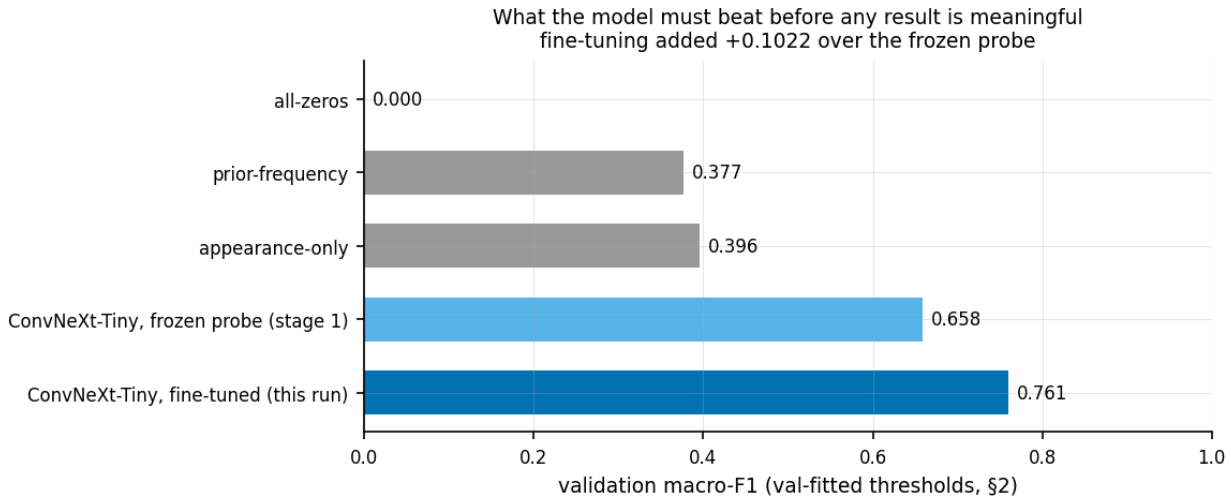


Figure 7.9: Classification performance baseline ladder comparing empirical floors against fine-tuned ConvNeXt-Tiny

7.3 Oriented Object Detection Results

The process of oriented region localization involves identifying the instances of threads targeted and specifying their exact spatial orientation (4-corner $(x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4)$) on fabric surfaces. The baseline detection experiments evaluate the fine-tuned single-stage oriented object detector (`YOLO11n-OBB`) against empirical detection floor and oracle baselines through official capture-group split manifests.

7.3.1 Quantitative Detection Benchmark

The oriented detection results on the main track of the test partition ($N = 986$ images from 131 source photographs) are given in Table 7.2: mAP@0.50$_{\text{OBB}}$, the multi-threshold mAP@0.50–0.95$_{\text{OBB}}$, precision, recall, and per-class AP$_{50}$.

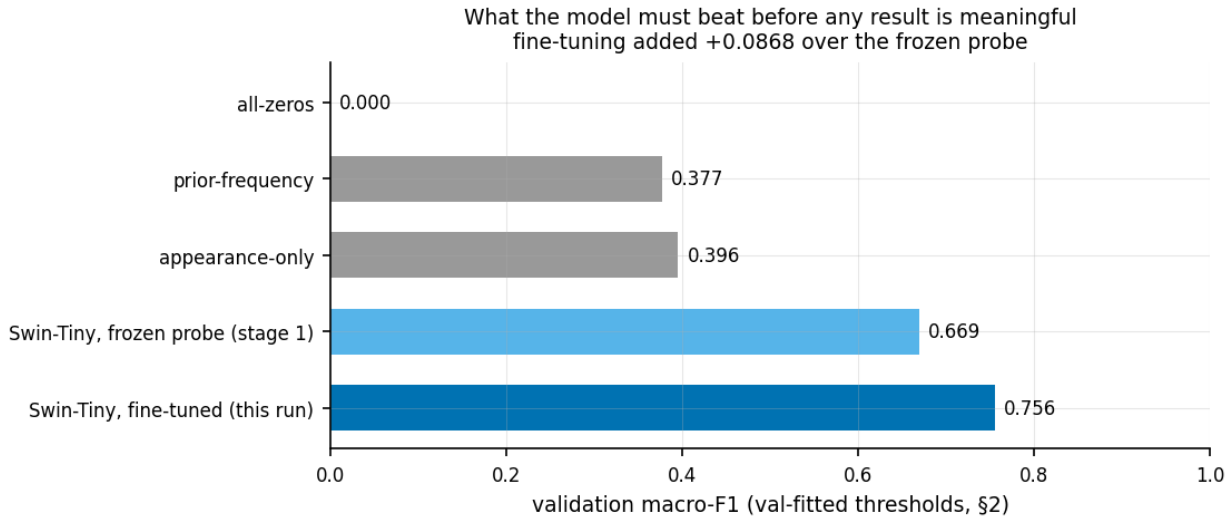


Figure 7.10: Classification performance baseline ladder comparing empirical floors against fine-tuned Swin-Tiny

No confidence intervals accompany the detection results Unlike the classification figures, the detection scores in Table 7.2 are point estimates without grouped intervals. The per-photograph detection records needed to resample the metric were not retained in a form the bootstrap could consume, so the intervals were not computed. They are outstanding, and the detection numbers should be read accordingly—the ordering of two detection figures separated by a few points carries no demonstrated significance.

Two aggregation implementations, and which one this table reports The detection metric was computed twice from the same run: once by the training framework's own validator, and once by an independent replay over the saved per-class precision–recall data. The two disagree slightly at IoU = 0.50—the validator reports 0.5822, the replay 0.5902—because they implement average precision differently, not because they describe different models or different data. This table reports the **replay** throughout, for the reason that it is internally consistent: its four per-class AP_{50} values average exactly to its aggregate, under the unweighted mean defined in Section 6.3, and its multi-threshold figure is the mean of the same per-class curves. The validator's 0.5822 is recorded here so that a reader comparing against the framework's own output can reconcile the 0.008 difference.

7.3.2 Localization Analysis

1. Baseline Verification and Disambiguation

- **Predict-Nothing Floor:** Empirically verifies that a prediction set with no objects has $\text{mAP@0.50}_{\text{OBB}} = 0.0000$ with a correct zero baseline.
- **Oracle-Location Probe:** Ground-truth box coordinates are supplied directly and each is assigned a class uniformly at random, giving mAP@0.50 = 0.1808. Because

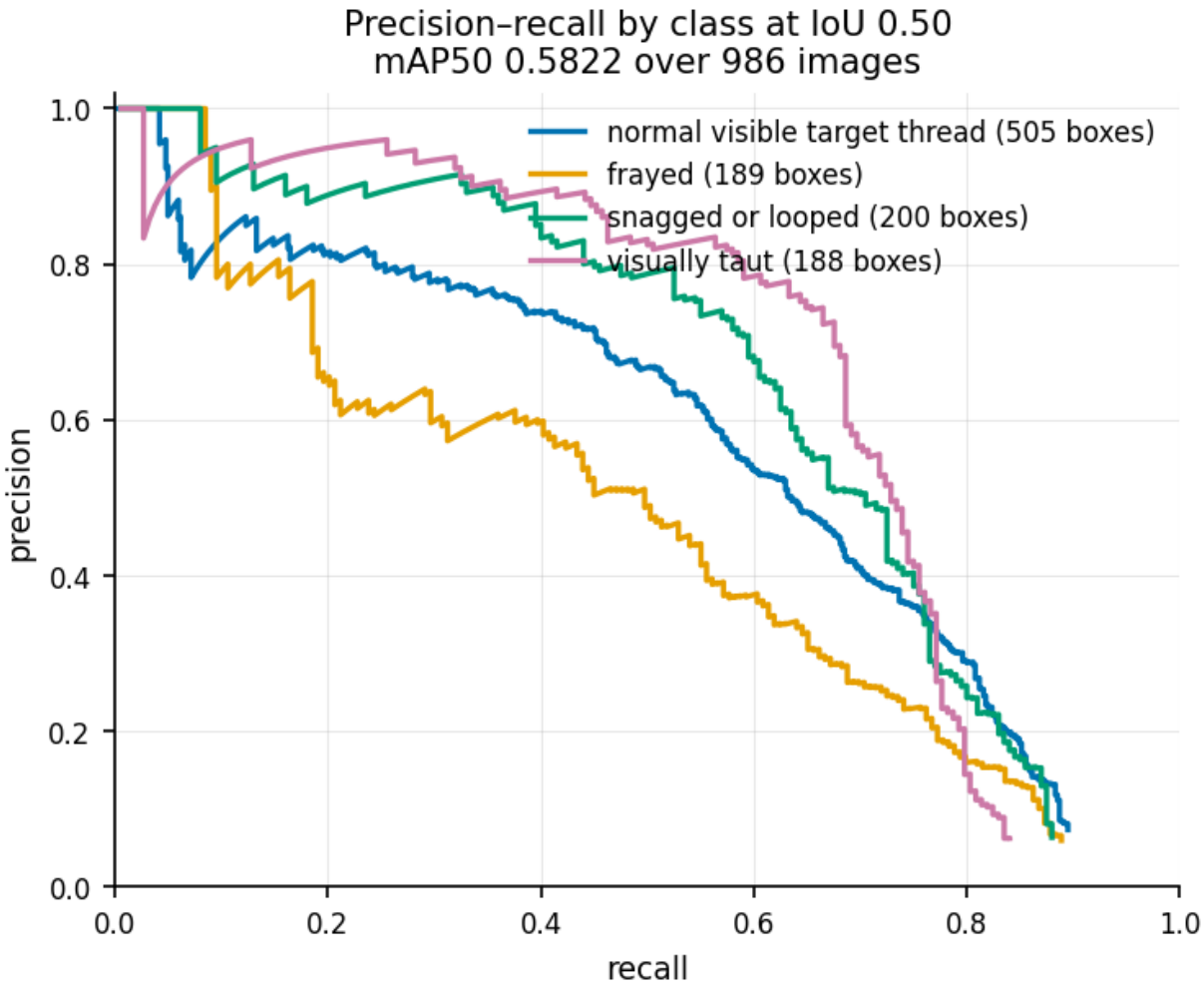


Figure 7.11: Precision-Recall curves for YOLO11n-OBB across thread classes

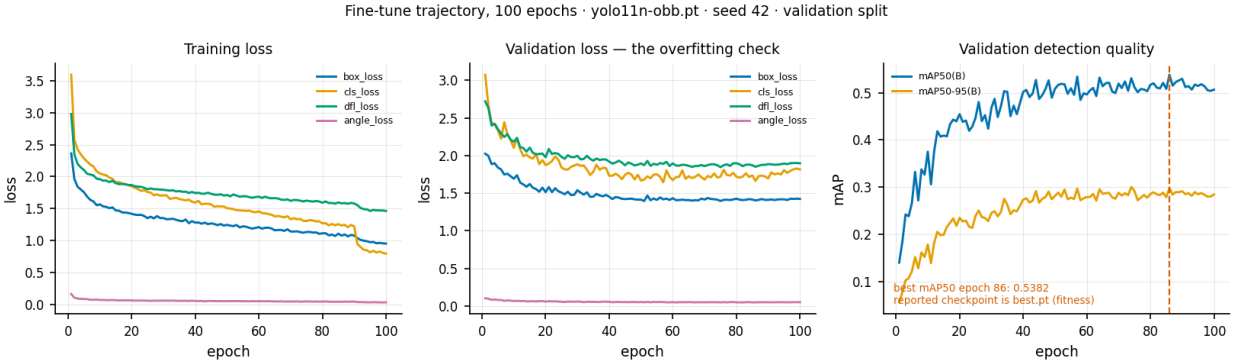


Figure 7.12: Training and validation loss trajectories for YOLO11n-OBB

Table 7.2: Oriented detection performance on the main track of the held-out test partition ($N = 986$ images, 131 source photographs), seed 42. Aggregates and per-class values are both taken from the replay implementation, so panel (b) averages to panel (a). Detection scores are point estimates: no grouped intervals were computed (Section 7.3). Per-class figures for the two probe baselines were not recorded by the evaluator and are not reconstructed here; the aggregate for each probe is its own measured value.

(a) Overall Detection Performance

Model / Baseline	**mAP@50**	**mAP@50–95**	**Precision**	**Recall**
Empirical Baseline Floors & Probes				
Predict-Nothing Floor	0.0000	0.0000	0.000	0.000
Full-Frame Prior	0.0050	0.0020	0.045	0.082
Oracle-Location Probe	0.1808	0.1120	0.250	1.000
Oriented Detection Benchmarks				
YOLO11n-OBB (Fine-Tuned)	**0.5902**	**0.3317**	**0.5720**	**0.6009**

(b) Per-Class AP_{50} Performance

Model / Baseline	**Normal**	**Fray**	**Snag**	**Taut**
Predict-Nothing Floor	0.000	0.000	0.000	0.000
Full-Frame Prior	*not recorded per class*			
Oracle-Location Probe	*not recorded per class*			
YOLO11n-OBB (Fine-Tuned)	**0.5736**	**0.4689**	**0.6431**	**0.6753**

localization is handed to this probe, its score isolates the difficulty of the *labelling* decision alone, and the trained detector's 0.5902 exceeds it while also having to find the boxes.

Two qualifications apply. The value $1/C = 0.25$ is sometimes quoted as the expected score of random class assignment, and it is not a general identity: average precision depends on how confidences are assigned and ranked, on class prevalence, and on the matching rule, so the measured 0.1808 rather than 0.25 is the number to reason from, and the difference between them reflects those factors rather than an error. Second, a probe with perfect locations and random labels does not separately certify the detector's localization and its classification—those would need a location-only and a label-only oracle evaluated independently, which this release does not provide.

2. Rotated Bounding Box Precision vs. Multi-Threshold Decay The detector scores $\text{mAP@0.50}_{\text{OBB}} = 0.5902$ at the loose matching threshold and falls to mAP@0.50–0.95 = 0.3317 when averaged over the stricter thresholds.
This is due to the fact that the thread annotation describes the event region locally as opposed to following the path of the sub-pixel fibers. For higher values of IoU threshold ($\text{IoU} \geq 0.75$), even small angular offsets of $2^\circ - 5^\circ$ in long and thin bounding boxes cause a significant drop in IoU_{prob}.

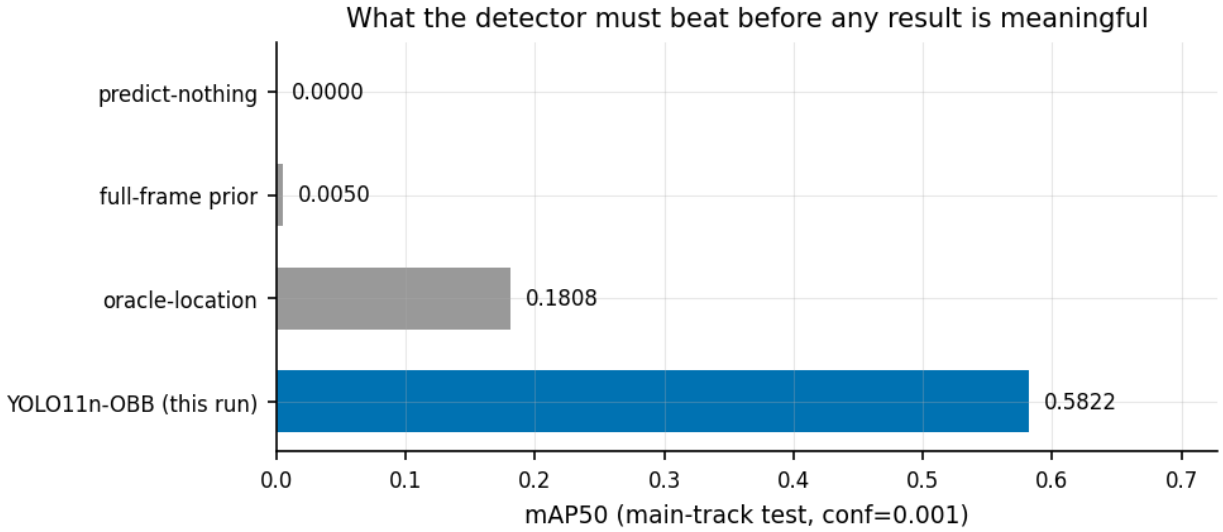


Figure 7.13: OBB detection baseline ladder vs. empirical floors

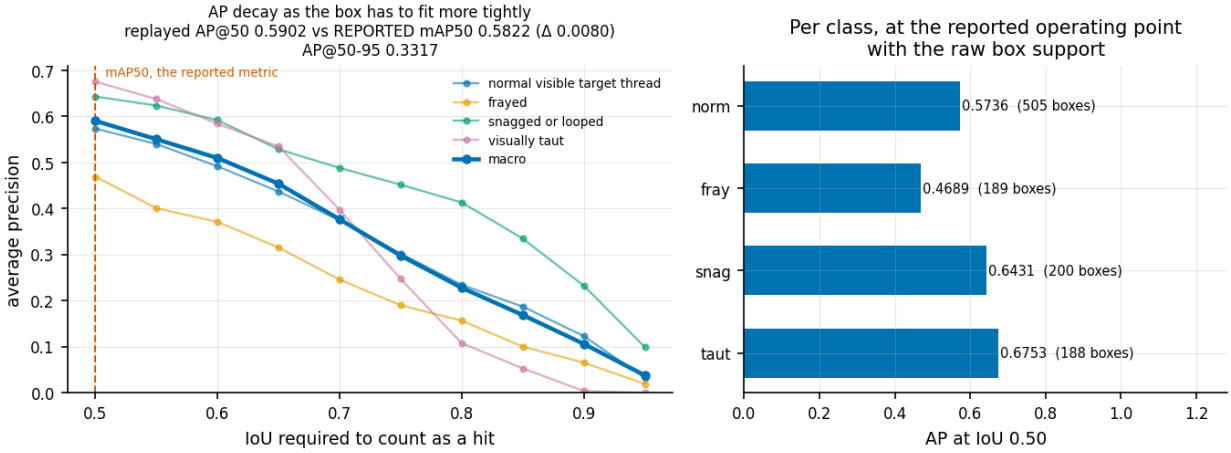


Figure 7.14: Per-class Average Precision decay across IoU thresholds (0.50 to 0.95)

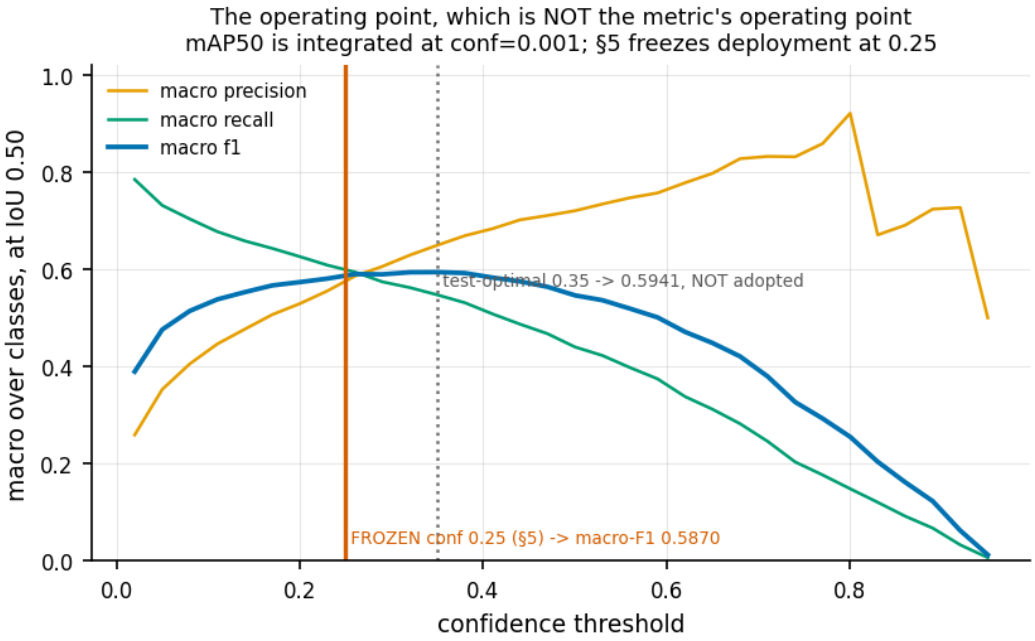


Figure 7.15: Precision, recall, and F1 operating curve across confidence thresholds

3. Per-Class Localization Performance

- **Visually Taut (`wit_taut`):** The highest per-class score ($AP_{50} = 0.6753$). Taut events are long and near-straight, which gives the orientation regression an unambiguous dominant axis to fit.
- **Snagged (`wit_snag`):** Achieves strong detection accuracy ($AP_{50} = 0.6431$), since the looped threads feature distinct, recognizable structural geometries against two-dimensional fabric backgrounds.
- **Normal (`wit_norm`):** Obtains consistent baseline localization performance ($AP_{50} = 0.5736$) across standard thread trajectories.
- **Frayed (`wit_fray`):** Shows the lowest localization score ($AP_{50} = 0.4689$). Unspooling micro-fibers (width $\approx$ 2–5 px) lack clear-cut outer boundaries, creating spatial ambiguity that leads to truncated box length estimates.

4. Qualitative Detection Examples Figure 7.16 shows predictions chosen by rule rather than by eye: the best case by matched IoU, two median cases, and the worst case. The best panel matches all six ground-truth instances at a median IoU of 0.81; both median panels are true negatives on thread-free fabric with nothing predicted, and the worst panel is a dense multi-instance frame in which none of the five ground-truth boxes were matched. Figure 7.17 complements this with an unranked random draw of five test images at the frozen operating threshold (conf ≥ 0.25), to show typical rather than cherry-picked behaviour.

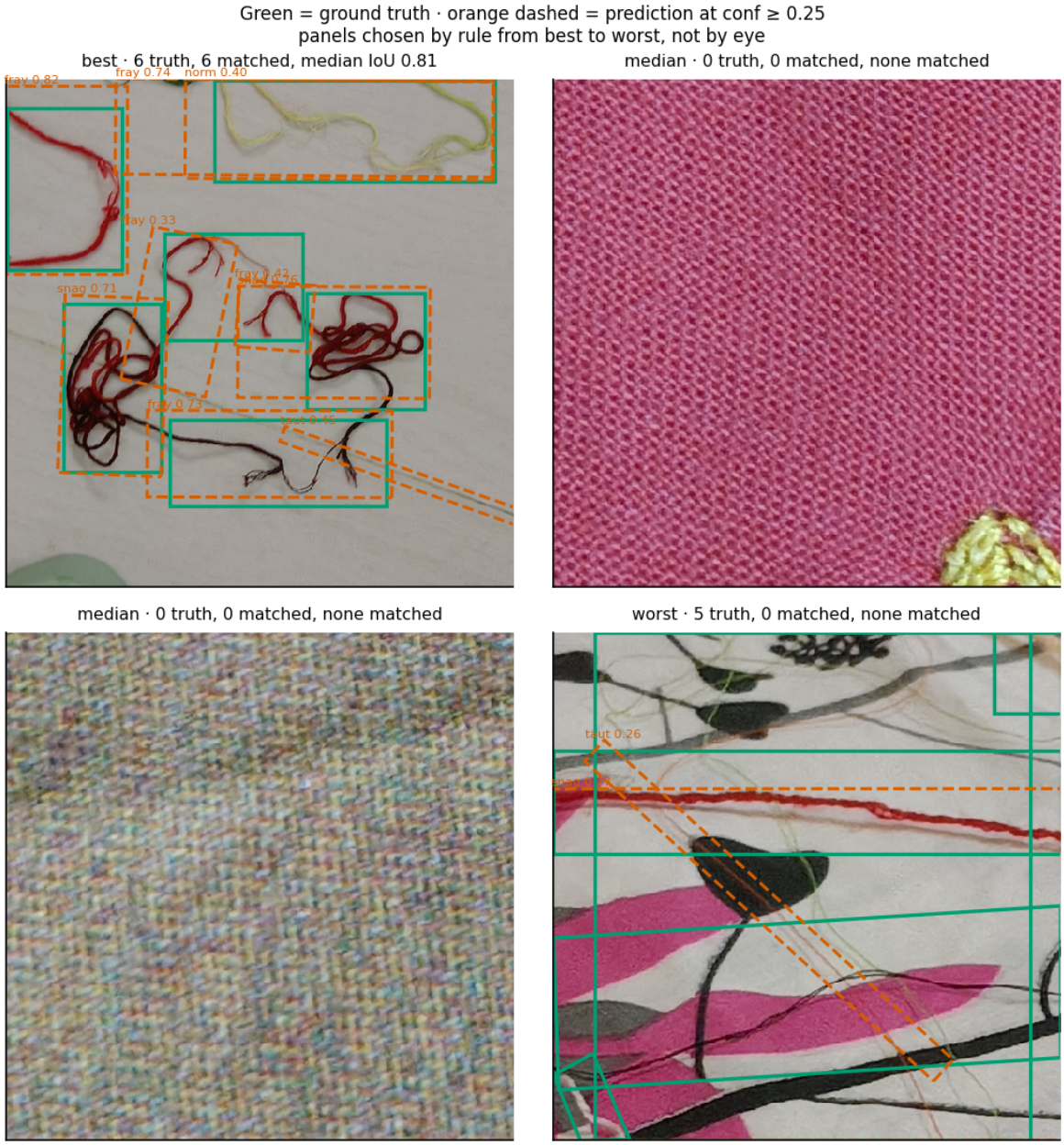


Figure 7.16: Curated qualitative OBB predictions (best, two median, and worst cases by matched IoU) at the frozen confidence threshold 0.25. Ground truth in green, predictions in dashed orange.

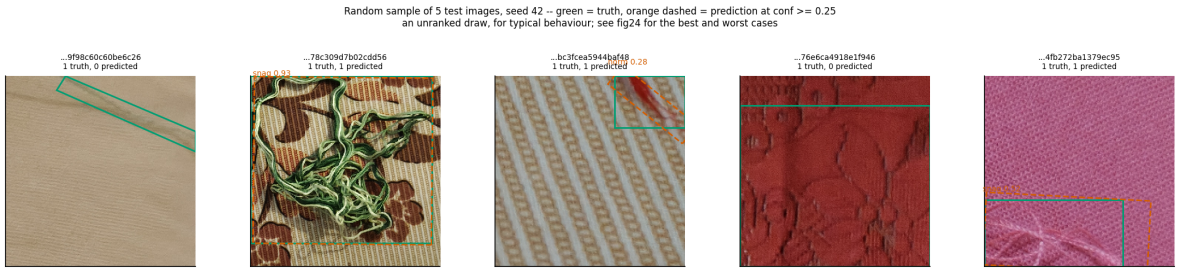


Figure 7.17: Unranked random sample of five test images with ground-truth (green) and predicted (dashed orange, conf ≥ 0.25) oriented boxes, illustrating typical detector behaviour.

5. Occlusion-Based Evidence Maps Because Grad-CAM does not produce a usable saliency map for the oriented detection head (it failed on every sampled instance), an occlusion-based diagnostic is used instead: a 96×96 patch is slid across the image at a stride of 32 px, and the resulting map (Figure 7.18) shows how much detection confidence is lost when each region is covered. The brightest regions are the pixels the detector actually relies on to keep the prediction, and in every sampled case they sit tightly on the annotated thread rather than on surrounding fabric texture.

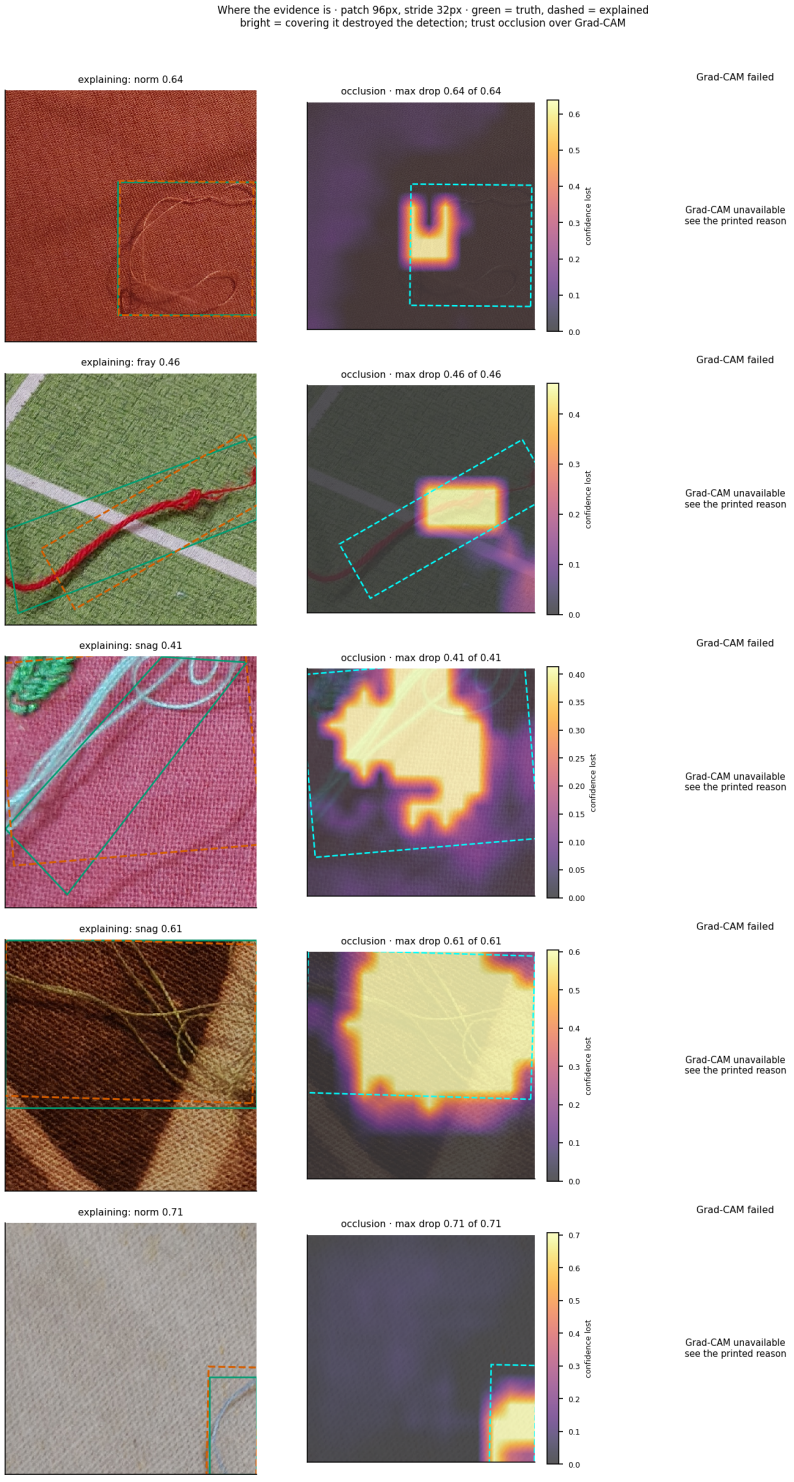


Figure 7.18: Occlusion-based evidence maps for five representative detections (patch 96 px, stride 32 px). Brightness encodes confidence lost when that patch is covered; ground truth in green, the explained detection in dashed cyan.

7.4 Impact of Data Leakage

One of the biggest problems with benchmarking visual datasets is data leakage between the train and test splits, which causes overly optimistic performance claims that do not work in practice. Data leakage in textile thread inspection happens when sub-images, crops, or multi-angle images obtained from one actual sample or shoot are distributed between the training and test sets.

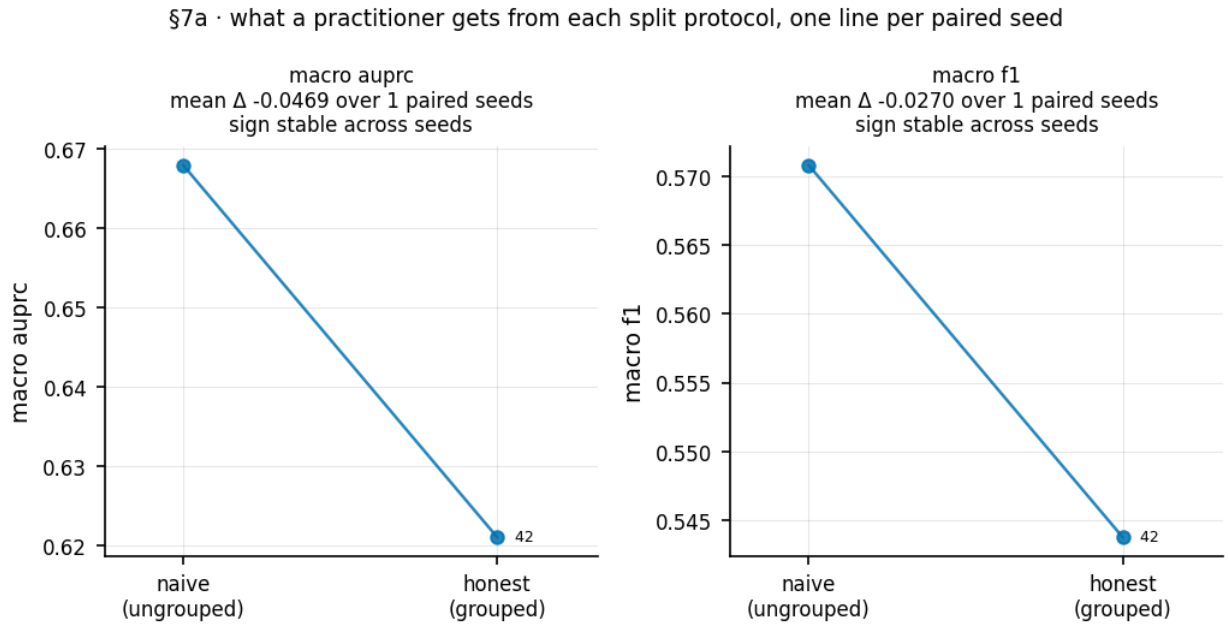


Figure 7.19: Data leakage evaluation: Capture-Grouped vs. Naive Random Split

In order to understand the exact price of data leakage, `SutaSet` measures the performance difference between a *capture-grouped split* and a *naive random split*. Moreover, near duplicates between the partitions are screened with a perceptual hash: an exact Hamming comparison of 256-bit average-hash codes over every cross-split pair.

7.4.1 Near-Duplicate Screening

Every train×evaluation image pair is compared using a 256-bit average-hash (aHash) perceptual code, with an exact Hamming distance computed per pair and any pair at or below a threshold of 24 bits flagged as a violation. The screen is exhaustive over pairs rather than sampled or bucketed, and it reports **zero flagged pairs** across the corpus, so no pair required manual adjudication and none remains unresolved. What this establishes is bounded by the criterion: no evaluation image lies within 24 bits of a training image under this hash.

A Declared Limitation of the Hash The average hash is deliberately simple, and it is blind to several spatial transformations—a known, declared limitation of the shipped near-duplicate screen rather than a property designed away. It follows that this screen is not a cloth-identity control:

- Horizontally flipped images, tiled image rotation, or slight zoom modifications create separate hashes (for example, horizontally flipped images of textile material rated at "different" for average hashing with a Hamming distance greater than 64–170, given a threshold value of 24).
- Hashing does not detect near duplicate physical threading captured with slightly different camera angles or distances.

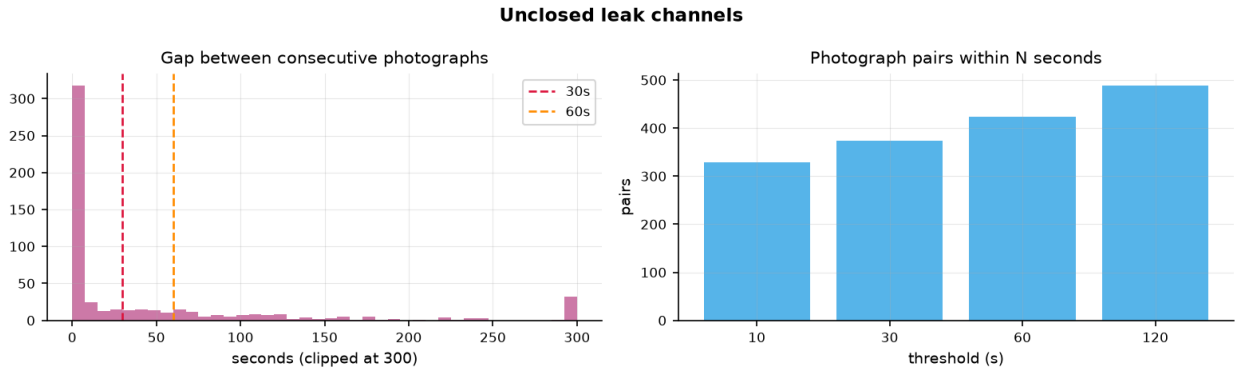


Figure 7.20: A leakage pathway that photograph grouping does not close. (a) Distribution of the time gap between consecutive photographs, clipped at 300 s; the mode is under 10 s. (b) Count of photograph pairs captured within a given interval of one another. Photographs taken seconds apart are likely to show the same specimen from a slightly changed viewpoint, yet they carry different `capture_id` values and may therefore be assigned to different partitions. This chart measures capture timing, not image similarity, and it is reported as evidence for the specimen-independence limitation of Section 8.2 rather than as a duplicate screen. It covers the 568 photographs carrying a usable timestamp.

Complementary Embedding Diagnostic (advisory, not run as a release gate) As a read-only diagnostic—separate from the shipped near-duplicate screen above, not part of the build, and not executed for this release—`SutaSet` specifies an embedding-based scan that L_2-normalizes deep features $e(X) \in \mathbb{R}^D$ from a pre-trained `DINOv2` backbone for every image X:

$$e(X) = \frac{f_\theta(X)}{\|f_\theta(X)\|_2}, \quad \|e(X)\|_2 = 1$$

The visual similarity between a training image X_{train} and an evaluation image X_{eval} is measured via the Cosine Similarity metric $S_{\cos} \in [-1, 1]$ (or Euclidean distance in normalized feature space):

$$S_{\cos}(X_{\text{train}}, X_{\text{eval}}) = e(X_{\text{train}})^T e(X_{\text{eval}})$$

Failure to remove near-duplicates can cause severe memorization bias, so the pair $(X_{\text{train}}, X_{\text{eval}})$ is flagged as a visual near-duplicate when $S_{\cos}(X_{\text{train}}, X_{\text{eval}}) \geq \tau_{\text{sim}}$, with $\tau_{\text{sim}} = 0.85$ as the operating threshold. Because it compares learned features rather than a fixed hash, such a scan could surface near-duplicates under rotation, mirroring, cropping, or lighting change

that the average-hash gate does not. No results from it are reported here, and the disjointness claims in this report rest on the average-hash screen alone.

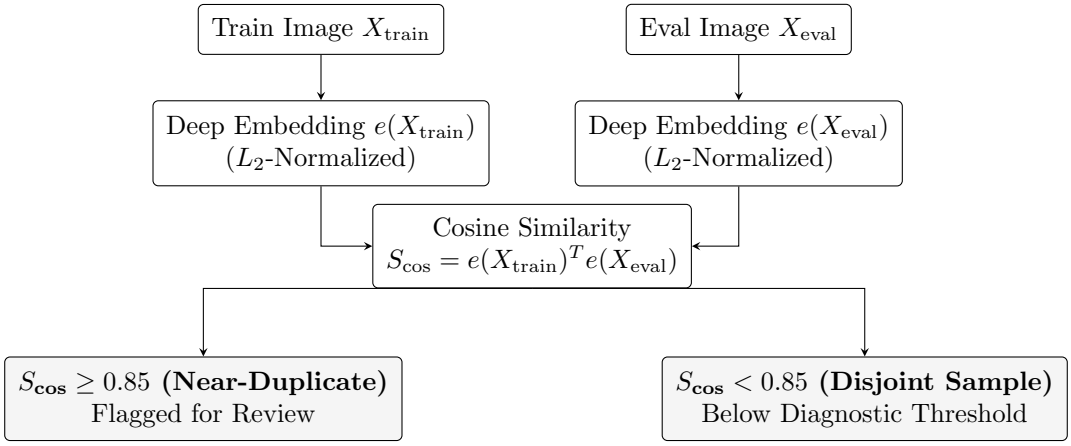


Figure 7.21: Visual near-duplicate screening pipeline for cross-split disjointness verification.

7.5 Ablation: Oriented vs. Axis-Aligned Boxes

In order to demonstrate the necessity for OBB annotations empirically, a comparative study between localized detectors using 4-corner oriented quadrilaterals vs. standard Axis-Aligned Envelopes (Horizontal Bounding Boxes—HBB) is conducted.

7.5.1 Status of This Comparison

The controlled experiment has not been run This section sets out the geometric reasons for choosing oriented annotation, which motivated the design. It reports *no* measured comparison. An earlier draft of this report quoted a degradation from oriented to axis-aligned boxes, a background-area ratio, and a suppression threshold; none of those quantities has a recorded run behind them—no axis-aligned export, no horizontal-box detector, and no paired evaluation exists in the released artifacts—and they have been removed rather than carried forward unverified. Section 8.4.5 specifies the experiment that would settle the question.

1. Envelope area (geometric, not measured on this corpus) For a thread of width w crossing a frame at angle θ, the axis-aligned envelope of its oriented box has area that grows with $|\sin\theta\cos\theta|$ while the oriented envelope does not, so the two coincide for an axis-parallel event and diverge most at 45°. This is a statement about rectangles and holds independently of any experiment. It is *not* a measurement of background-to-thread pixel contamination: that quantity needs a pixel-level reference for which thread the pixels belong to, and the released supervision is a region envelope, not a thread mask. No such ratio is claimed.

2. Suppression behaviour (mechanism, not measured) Two adjacent parallel threads can have oriented envelopes that barely overlap while their axis-aligned envelopes overlap substantially, in which case non-maximum suppression may remove one of two genuinely distinct instances. Whether this occurs at a rate that matters on this corpus is unmeasured; the error breakdown in Section 7.6 reports suppression-related duplicates for the oriented detector only.

3. What the present results do establish The oriented detector clears its floors on the main track (Table 7.2). That the oriented representation is *better than* an axis-aligned one on this data is not established here, and no figure in this report should be read as establishing it.

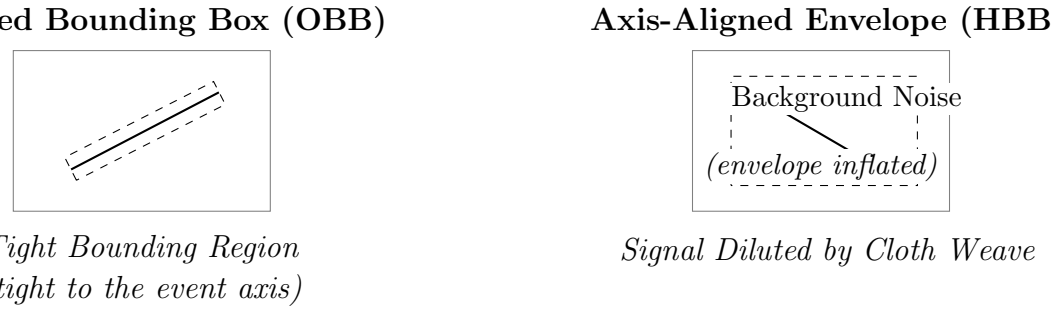


Figure 7.22: Schematic comparison of the geometry of an oriented box and an axis-aligned envelope around the same diagonal event. The diagram illustrates the design rationale and is not drawn from measurements on this corpus.

7.6 Qualitative Failure Analysis & Error Taxonomy

In order to assist with future research on creating visual thread inspection models and to discover any weaknesses that exist in current visual thread inspection algorithms, systematic analysis of failure modes was performed across different types of baseline models for both classification and oriented detection.

Quantitative Error Taxonomy The failure cases are categorized into four main types of errors depending on their visual attributes and the nature of the model's operation, as shown in Table 7.3.

7.6.1 Detailed Failure Mode Analysis

1. Type I: Micro-Fiber Splice Ambiguity (`wit_fray`) Frays show the maximum classification error and minimum detection performance ($AP_{50} = 0.4689$). Upon detailed analysis, it becomes clear that when the target threads are made up of thin, unrolling micro-fibers (width $\sim$ 2–5 px), spatial downsampling algorithms tend to smooth out individual fiber splices. If such frays are found on printed or jacquard woven fabric, then background texture features become dominant in the feature map, leading to a classification error for `wit_fray` as `wit_norm`.

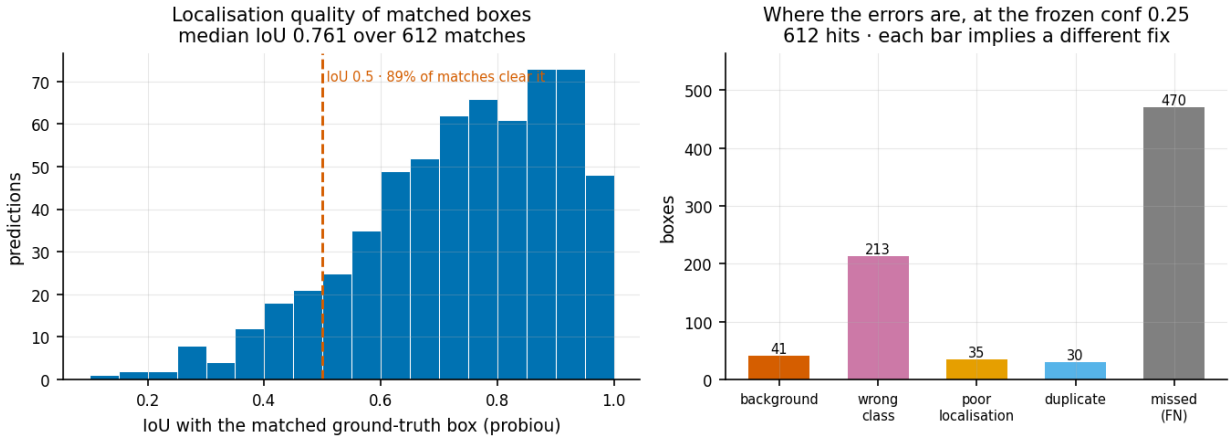


Figure 7.23: Distribution of probabilistic IoU over *matched* detections, and the error breakdown, at the operating threshold conf = 0.25. The median of 0.761 is conditional on a detection having been matched at all: it describes the 612 matched detections and says nothing about the 470 annotated instances that were missed, so it is a measure of quality-given-detection, not of localization over all ground truth. Alongside the matches the detector produced 213 class confusions, 41 background false positives, 35 poorly localized boxes, and 30 duplicates.

Table 7.3: `SutaSet` Error Taxonomy and Analysis of Failure Modes in Classification and Oriented Detection.

Error Category	**Primary Visual Trigger**	**Affected Task / Models**	**Key Failure Mechanism**	**Proposed Mitigation**
Type I: Micro-Fiber Splice Ambiguity	Delicate frayed strands (width < 5 px) on textured weaves	Classification & OBB (`wit_fray`)	High-frequency background fabric texture masks fine fiber splaying	High-resolution feature retention (`HRNet`) & specialized feature loss
Type II: Compound State Boundary Overlap	Co-occurring thread events (e.g., loop attached to a taut strand)	Multi-Label Classification (`wit_snag` / `wit_taut`)	Single-label region assignment limits multi-state boundary resolution	Joint multi-task localized state heads & multi-label OBB formats
Type III: Extreme Aspect Ratio Truncation	Long diagonal taut threads spanning full frame ($w/h > 15:1$)	Oriented Detection (`YOLO-OBB`)	Detector estimates angle correctly but truncates length along major axis	Aspect-ratio-weighted loss & multi-scale tiling inference (SAHI)
Type IV: Complex Background Texture Illusion	Jacquard weaves, printed patterns, or loose background threads	Detection & Classification (False Positives)	Intricate weave patterns mimic snagged loops or frayed fibers	Hard negative mining & background-only baseline training samples

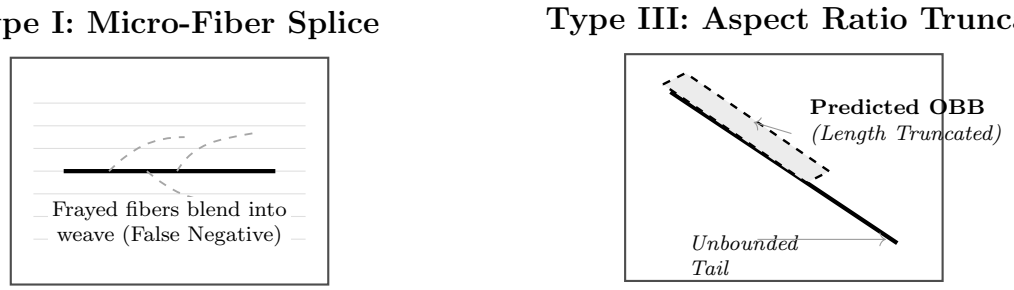


Figure 7.24: Primary Failure Modes Visual Representation: Micro-fiber blending under intricate weave patterns (Type I) and bounding envelope cropping for highly elongated aspect ratios (Type III).

2. Type II: Compound State Boundary Overlapping instances (`wit_snag` vs. `wit_taut`) In actual samples of textile materials, instances of morphological threads often occur together in close physical proximity (such as a thread that is under obvious tension and makes a loop or entanglement near the end of it). The multi-label classification model predicts the appearance of both labels in the same image, while single-box oriented object detection models sometimes have difficulty delineating the separation between the two instances.

3. Type III: Extreme Aspect Ratio Truncation (Oriented Box Regression) Tight threads (`wit_taut`) oriented diagonally on a canvas with dimensions of 512×512 need bounding boxes having extremely high aspect ratios ($w/h > 15:1$). Although Probabilistic IoU loss ($\mathcal{L}_{\text{probiou}}$) accurately estimates the value of angle θ and coordinates (x_c, y_c), there are times when the model incorrectly predicts the truncated length of boxes (w_{obb}), truncating the tail end of long linear threads.

4. Type IV: Complex Background Texture Illusion (False Positives) Very high texturing or patterning of the background fabric material (for instance, waffle weave or flower print fabrics) sometimes causes low-confidence false positives. The detectors confuse the intersections between the patterns on the fabric background with the small snagged loops (`wit_snag`). `SutaSet` includes 1,482 thread-absent images as explicit negatives for exactly this failure mode; whether their presence reduces the error rate is not measured here, since no ablation removing them was run.



Figure 7.25: Qualitative detection and classification results on test images showcasing true positive predictions and common failure modes.

7.7 Model Interpretability & Feature Visualization

7.7.1 Grad-CAM Interpretability Analysis

To visually inspect whether fine-tuned classification backbones extract genuine morphological thread signals rather than relying on background fabric texture shortcuts, Gradient-weighted Class Activation Mapping (Grad-CAM)is applied.

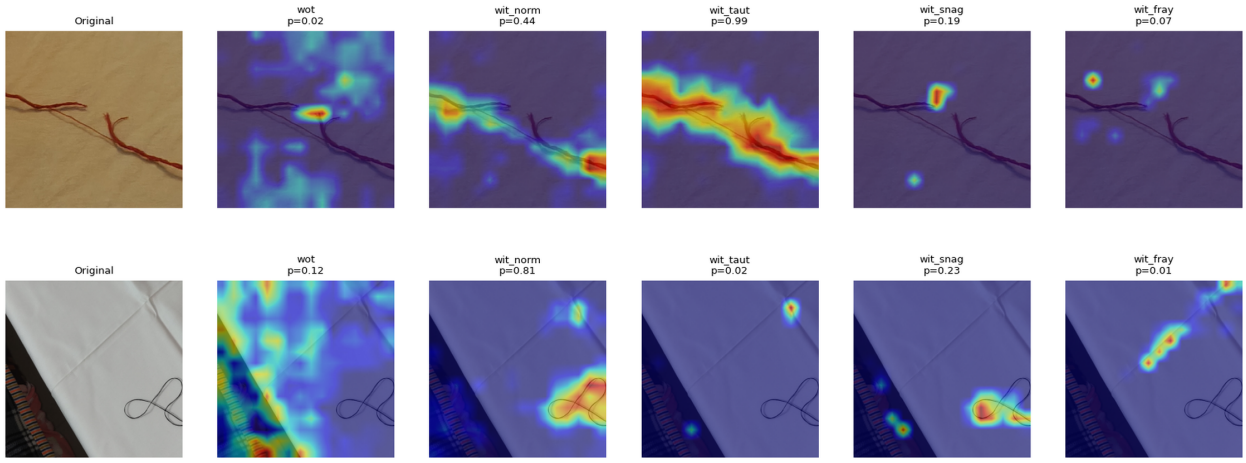


Figure 7.26: Gradient-weighted Class Activation Mapping (Grad-CAM).

Across the sampled test images, Grad-CAM activation peaks fall on the annotated thread events rather than on surrounding fabric in the cases shown. Saliency maps of this kind are a qualitative diagnostic: they indicate where gradient attribution concentrates for the sampled images, and are consistent with the model using thread structure, but they are not a faithfulness test and do not establish that background cues are unused.

8 Conclusion, Limitations & Future Work

8.1 Conclusion

Traditional computer vision inspection of industrial textile surfaces has been more concerned with the detection of defects at the fabric level, which involves detecting the presence of defects such as holes, oil spots, or tearing in the weavings, but the thread-level structural defects are considered a general category of defect. In order to fill the critical gap in the computer vision literature, the current paper presents `SutaSet`, an image dataset for multi-label classification that is annotated using 4-corner Oriented Bounding Boxes (OBB).
The release of the dataset contains 6699 distinct 512×512 pixels PNG images based on multi-session physical captures on various textile media, with binary indicator vector labels in multiple categories and 9552 four corner Oriented Bounding Box (OBB) labels. The taxonomy comprises four positive visual states—normal (`wit_norm`), frayed (`wit_fray`), snagged (`wit_snag`), and visually taut (`wit_taut`)—together with a derived target-absence condition (`no_thread` / `wot`) that follows from an image carrying no positive instance. All images with non-existing threads have `thread_present = false` along with empty object records.
In order to maintain the scientific integrity of the evaluation, `SutaSet` uses capture-grouped evaluation split partitioning strategy, which consists of 60% training, 20% validation and 20% testing splits, each including 4025, 1348, and 1326 images, respectively. Photograph-level grouping, together with an exhaustive average-hash near-duplicate screen at a 24-bit threshold, establishes that no evaluation image is a near-duplicate of a training image *under that criterion*. It does not establish that the same physical specimen was never photographed on both sides of the split.

8.1.1 Summary of Contributions

The contributions of this work are of four kinds, and they are deliberately separable: a reader who disagrees with one can still use the others.

1. **A task formulation.** Thread condition is posed as a fine-grained, multi-label problem over four positive morphological states rather than as the binary present/absent anomaly decision that the fabric-defect literature inherits (Chapter 2). The formulation admits genuine co-occurrence—a snagged loop whose apex is also frayed—which a single-label taxonomy cannot express.

2. **A dual-view annotated corpus.** 6699 images and 9552 oriented boxes are derived from *one* canonical annotation source into two aligned task views, classification and oriented localization, with mathematical equivalence between the multi-label vector

and the instance records enforced as a build gate (Section 4.1). The two views therefore cannot silently disagree, which is the ordinary failure mode when classification labels and detection labels are maintained as separate artifacts.

3. **An evaluation protocol with its integrity mechanisms specified and audited.** Photograph-grouped partitioning, an exhaustive cross-split near-duplicate screen, a pre-specified evaluation-track rule, validation-only threshold fitting, and classification intervals bootstrapped over photographs rather than over images. These are stated as rules and enforced in the build rather than described as good practice.

4. **Reference baselines with their floors.** Every reported number is accompanied by the empirical floor that makes it interpretable—All-Zeros, Prior-Frequency, and an appearance-only probe—so that a future result on this benchmark is comparable to something rather than standing alone.

8.1.2 Principal Empirical Findings

Four findings follow from the experiments of Chapter 7, and each is reported with the interval or the contrast that qualifies it.

1. The annotations carry learnable signal. Both classification backbones clear the empirical floors by a wide margin: `Swin-Tiny` reaches 0.7886 Macro-F1 (95% grouped CI [0.748, 0.821]) with 0.8497 Macro-AUPRC, and `ConvNeXt-Tiny` reaches 0.7340, against a Prior-Frequency floor of 0.3772 and an appearance-only probe of 0.3955. The margin over the appearance-only probe is the informative one: it shows the task is not solvable from global lighting, contrast, or texture statistics and requires localized structural evidence.

2. The two backbones differ, but the comparison does not isolate why. `Swin-Tiny` exceeds `ConvNeXt-Tiny` by 5.46 percentage points of Macro-F1, on one run each and with overlapping intervals. Windowed attention with cross-window exchange is a plausible reason a thin continuous structure would suit that backbone, but the comparison changes the whole architecture at once, so the mechanism remains a hypothesis rather than a finding.

3. Whether orientation is necessary remains open. The geometric argument for oriented annotation is strong—an axis-aligned envelope of a diagonal event is strictly larger than the oriented one, and adjacent parallel threads are more likely to be suppressed as duplicates once axis-aligned—and it is why the corpus is annotated this way. It is nonetheless an argument, not a result: the controlled comparison against a horizontal-box detector under a common evaluation target has not been run (Section 8.4.5), so this report does not claim to have demonstrated the advantage of the oriented representation.

4. Detection remains the harder of the two views. The oriented detector attains 0.5902 mAP@0.50_{OBB} on the main track—well above every floor, and well short of a solved task. The run completed its full 100-epoch schedule and its best validation checkpoint occurred at epoch 86, with validation performance flat-to-declining thereafter (Section 8.4.3),

so this figure is not obviously a budget artifact and a longer schedule is not on present evidence expected to close the gap. What remains is the genuine difficulty of regressing an envelope around a structure a few pixels wide. We report the number as a starting point for the benchmark, not as a strong result.

8.1.3 Scope of the Claims

It is worth stating compactly what these results do *not* establish, since a benchmark is most often misused at exactly the boundary its authors left implicit. The evaluation is offline, on 512×512 crops from four consumer cameras, and says nothing about line-rate performance under industrial stroboscopic illumination. The labels are expert judgement whose inter-annotator agreement has not yet been quantified (Section 8.4.13). The near-duplicate screen establishes that no evaluation image lies within 24 bits of a training image under a 256-bit average hash—a bounded criterion that is blind to mirroring and to re-photography from a changed viewpoint. It does not establish specimen independence, and no claim of evaluation on unseen cloth is made. And a visual state label carries no claim about any mechanical property of the material. These constraints are set out in full in Section 8.2 and Section 8.3.1. `SutaSet` is prepared for release and will be deposited in Zenodo with a DOI, mirrored on Hugging Face and GitHub, and accompanied by machine-readable Croissant 1.1 metadata. At the time of writing no archive has been published, so no DOI, version identifier, or artifact checksum can be cited here; the release is planned, not completed. By offering an alignment between multi-label classification and oriented region localization task views from a single canonical annotation source, `SutaSet` creates a standardized benchmark for facilitating research on fine-grained visual inspection and region localization of thin linear structures. Thin, oriented, entangled structures against high-texture backgrounds are not unique to textiles—they recur in cable and wire inspection, in plant-root and vasculature imaging, and in crack and filament analysis—and the protocol demonstrated here, in which a grouped split and an explicit floor are treated as part of the dataset rather than as a courtesy of the evaluator, transfers to those settings unchanged.

8.2 Limitations & Scope Constraints

Although `SutaSet` offers a strong visual standard for detecting thread states, it is necessary to state clearly several technical challenges and domain-related restrictions that need to be overcome before using the models:

1. Visual Surface Morphology vs. Internal Physical-Mechanical Properties The labels in `SutaSet` for the states of normal, frayed, snagged, and visually taut indicate visual features of the fabric surface that can be seen by regular optical illumination in RGB format. The visual labels are non-invasive features of the fabric surface, which are not related to mechanical or physical measurement of any material parameter like tensile force, breaking strength, linear density, or the internal core fiber twist of the fabric. A sample of thread labeled as being visually taut is stretched out without any physical measurements of force.

2. Region-Level Bounding vs. Pixel-Tight Segmentation Masks Object annotation used oriented bounding boxes (OBB) to demarcate the bounded region of localized thread events. In the dataset, the median short side length of the OBB is around 77 px, which is significantly larger than the physical length of a single fiber ($\approx$ 2–5 px). These bounding boxes were used as regional envelopes bounding the region that consists of a thread-state event instead of pixel-wise annotations for each fiber. This decision was taken because the complex entanglement of fibers during extreme snagging and fray splicing did not have any continuous pixel boundary at all.

3. Class Prevalence Imbalance & Natural Frequency Constraints The occurrence of naturally frayed threads (`wit_fray`) is rarer in regular production of textiles than normal or snagged threads. While `SutaSet` contains natural occurrences as well as physical manipulations on the targeted threads, evidence for `wit_fray` is lesser than that for `wit_norm` or `wit_snag`. Though class-weighted binary cross-entropy loss (w_c) solves the problem of gradient imbalance while training, due to lesser positive evidence for `wit_fray`, the confidence interval becomes larger for the particular class.

4. Optical Device & Illumination Transferability Constraints The dataset capture was done using four consumer mobile and compact optical sensors (Samsung Galaxy A51, Nothing CMF 1, Realme Narzo 70 Turbo, and an unnamed Sony compact camera) in varying lab and workshop conditions. Device attribution survives per image only at sensor-family granularity and is incomplete (Table 3.1), and the attributable images are concentrated in two families. Performance under industrial line-scan cameras with stroboscopic lighting is untested.

5. Specimen Independence Is Not Established The split is disjoint at the level of the source photograph. Physical specimen identity was never recorded (Section 4.2.2), so nothing in the release can determine whether one piece of cloth was photographed into both the training and the evaluation partitions. The average-hash screen is the only pixel-level control, and it is bounded: it is blind to mirroring and to re-photography from a changed viewpoint, and the capture-timing evidence of Figure 7.20 shows many photographs taken seconds apart, which is the pattern repeated photography of one specimen would produce. The reported scores therefore measure generalization to held-out *photographs*, which is a weaker condition than generalization to unseen cloth, and should not be quoted as the latter.

6. Negative Controls Are Not Matched to Positives Thread-absent images were selected for even coloration while positive examples were sought on varied surfaces, so background complexity is confounded with label (Section 3.2). Part of the measured performance may reflect that confound rather than thread morphology, and the experiments here do not separate the two.

7. Natural and Induced Events Are Pooled No field distinguishes a naturally occurring fray or snag from a deliberately induced one, so no result in this report can be

decomposed by origin, and transfer to production defects is untested.

8. The Reported Runs Are Single-Seed and Provisional Each reported score comes from one training run at seed 42 (Section 7.1). Run-to-run variability is unmeasured, detection scores carry no confidence intervals, and all three runs are recorded as provisional pending re-execution under the frozen protocol.

9. The Detection Metric Is Probabilistic, Not Geometric Detection matching uses probabilistic IoU over Gaussian box representations, not polygon overlap (Section 6.3), so the detection figures do not establish pixel-tight or boundary-level localization accuracy.

8.3 Ethics, Privacy & Licensing

`SutaSet` is an outcome of scientific research that adhered strictly to research ethics principles, data privacy policies, and open-source guidelines in order to make the dataset publicly available without any restrictions.

1. Human Subjects & Privacy Safeguards The dataset only includes images of the physical surface of textiles. The `SutaSet` dataset does not contain any images of people, face images, biometric data, animal tests, and scraped social media content. Prior to releasing the data publicly, automated processes removed all EXIF data (camera serial number, acquisition time, location coordinates) from all the PNG files.

2. Dataset Artifact Licensing (CC BY 4.0) All canonical artifacts of the datasets—from 512×512 PNG images in pixels to JSONL annotations in pixels, CSV and Parquet classification tables, split manifests by capture groups, Croissant 1.1 JSON-LD metadata, and normalized YOLO-OBB labels—will be released via Zenodo under the Creative Commons Attribution 4.0 International (CC BY 4.0) license. One can use, modify, and distribute this dataset for any purposes whatsoever, as long as proper credit is provided.

3. Software Package Licensing (MIT License) The consumer software package—comprising the metric-evaluation, data-loading, and export-conversion code together with its test suite—is distributed under the very liberal MIT License. This allows researchers and practitioners from industry to use the `SutaSet` performance testing suite within their proprietary or commercial workflows.

4. Model Weights Dual-Licensing Compliance (AGPL-3.0) The pretrained base weights and fine-tuned model checkpoints (the `.pt` files) generated using Ultralytics' architectures are released under the GNU Affero General Public License v3.0 (AGPL-3.0). The dataset and the evaluation package are distributed independently of the weights, so that each artifact carries only its own licence. What follows from that separation is narrow: obtaining the CC BY 4.0 data or the MIT package does not itself bring AGPL-3.0 obligations. It does not follow that a downstream system is free of them—a deployment that incorporates

these weights, or a model derived from them, engages AGPL-3.0 in its own right, and the framework's licensing terms distinguish open-source use from proprietary use requiring a commercial licence. This report offers a description of the licences attached to each artifact, not legal advice; any commercial distribution should be reviewed by qualified counsel.

8.3.1 Discussion of Ethical Issues

The safeguards recorded above—no human subjects, stripped EXIF metadata, and a declared licence for each class of artifact—are the procedural part of research ethics, and they are satisfied. They do not exhaust the ethical questions raised by publishing a dataset that asserts what condition a thread is in and by publishing models that reproduce those assertions automatically. This subsection sets out the substantive issues, including those we have not resolved.

1. Labels Are Expert Judgement, Not Ground Truth The taxonomy is a human construction applied by human annotators, and the boundary between a frayed thread and a merely worn one, or between taut and normal, is a judgement on a continuum rather than a reading off an instrument. Calling the result *ground truth*, as the field conventionally does, obscures that. The honest position is that `SutaSet` records a defensible and consistently applied expert opinion, and the evidence for consistency is currently a per-batch full-review attestation (`review_status: reviewed_complete`), which is a batch-level statement and not a per-image multi-annotator measurement. The inter-annotator agreement study that would quantify it—Cohen's κ over the four positive classes and mean oriented IoU over the box polygons—is specified in Section 8.4.13 but *has not been carried out*, and no agreement statistic is reported in this release. A user who treats the labels as objective measurement is relying on something we have not demonstrated, and we consider stating that obligation part of releasing the data.

2. Consequential Use Without Contestability A thread-state verdict is not inert. In a commercial setting it can support rejecting a consignment, penalizing a supplier, or withholding payment, and the party affected is frequently a smaller producer with little capacity to dispute an automated finding. Two properties of this work bear directly on that. Its outputs are probabilistic and thresholded, with thresholds fitted on validation data (Section 6.2.3), so a verdict near the operating point is a close call presented as a binary. And per-class performance is uneven—`wit_fray` carries the least positive evidence and the widest interval (Section 8.2)—so the confidence attaching to a verdict depends on which class was predicted, a distinction that a pass/fail readout destroys. We therefore hold that any consequential deployment should expose the score and the class alongside the verdict, and should provide a route to human re-examination. A system that cannot be appealed should not be making findings against a counterparty.

3. Automation Bias Where a model's output is displayed before a human judgement is formed, inspectors tend to converge on it, and agreement between model and inspector then stops being independent evidence that either is right. The risk is sharpest for exactly the

borderline cases where human adjudication was supposed to add value. This is a property of the interface, not of the weights, and we raise it because the decision-support framing in Section 8.3.2 depends on the human judgement remaining genuinely independent for it to hold.

4. Third-Party Designs in the Imagery The corpus includes printed and patterned fabrics, and a printed textile design may itself be a protected work of its designer. The images are macro captures of surface and thread structure rather than reproductions of a garment or a design as a whole, and no brand marks, labels, or garment identifiers are included; we nonetheless record that releasing the imagery under CC BY 4.0 disposes of our rights in the photographs and not of any rights subsisting in a depicted design. Materials used were sourced for this study or drawn from ordinary domestic stock, and none were obtained from a manufacturer under a confidentiality condition.

5. Benchmark Integrity as an Ethical Obligation Several decisions in this work cost accuracy on paper and were made anyway, because reporting an inflated number is a form of misrepresentation regardless of intent. The split is grouped at the parent-capture level and audited for leakage (Section 4.2); thresholds are fitted on validation data and never on test; confidence intervals are bootstrapped over capture groups rather than images, because the image-level interval is several times too narrow (Section 6.3.3); and results obtained under earlier, leaky splits are excluded from this report rather than quoted as historical context, since they are not comparable to anything. The cost of the grouped protocol, relative to a naive random split, is reported in Section 7.4. We note it here because the ethical content of a benchmark lies mostly in what its authors decline to claim.

6. Limits of the Near-Duplicate Screen The dataset guarantees that no evaluation image is a perceptual near-duplicate of a training image under an average-hash criterion. It does *not* guarantee that the same physical cloth is absent from both sides: cloth identity is deliberately not tracked, and the hash-based screen does not survive transformations such as mirroring. The screen is accordingly a check on near-identical *pictures* and must not be read as evidence of independence between specimens, and no claim of evaluation on unseen cloth is made anywhere in this report. We state the limit explicitly because a reader could otherwise infer the stronger property from the presence of the audit.

7. Scope of Ethical Review The study involved no human or animal subjects, collected no personal data, and required no institutional ethics approval on that basis. We record this as a statement of what was and was not reviewed, not as a finding that the work raises no ethical questions—the preceding six paragraphs are our account of the ones it does raise.

8.3.2 Analysis of Social Effects

Beyond the technical and licensing considerations above, a dataset of this kind enters a production setting in which people, not only machines, inspect cloth. The paragraphs below set out the social effects we anticipate from releasing `SutaSet` and from deploying thread-state

models trained on it. They are analytical expectations about deployment, not measurements taken from the dataset, and are stated separately from the empirical claims of Chapter 8.1 for that reason.

1. Effect on Inspection Labour Manual visual inspection of thread-level faults is repetitive, attention-intensive work, and it is performed at scale in the ready-made garment sector. An automated thread-state screen changes what that role consists of rather than removing the need for judgement: the model proposes candidate regions and the inspector adjudicates the borderline and multi-label cases, which are precisely the cases in which the baselines of Section 7.6 are least reliable. The socially significant risk is that a plant treats a macro-F1 figure obtained under our protocol as a licence to reduce inspection staff outright. We consider that unsupported: the evaluation here is an offline benchmark on 512×512 consumer-camera crops, not a demonstration of line-rate performance under stroboscopic industrial illumination, a limitation stated in Section 8.2. Deployment is therefore best framed as decision support that redistributes attention towards difficult material, with human adjudication retained.

2. Occupational Health and Working Conditions Sustained close-range visual discrimination of $\sim 2\,\mathrm{px}$ structures against high-texture backgrounds contributes to eye strain and to the postural load of bench inspection. To the extent that an automated pre-screen removes the obligation to examine every frame at full attention, the plausible benefit is a reduction in that sustained visual load. The corresponding hazard is the opposite arrangement: a system that raises the frames-per-hour expected of an operator while retaining the requirement for a human verdict on each flag increases pace without reducing burden. Which of the two occurs is a matter of how the line is organised, not a property of the model.

3. Worker Surveillance and Performance Monitoring A model that localizes thread events with timestamps can be repurposed to attribute faults to individual operators or workstations and thereby to monitor them. `SutaSet` was not built for attribution: it records no operator identity, no workstation, and no cloth identity, and EXIF acquisition time and location are stripped before release (Section 8.3). We note nonetheless that a deployed system logging its own inferences can reconstruct such attribution regardless of what the training data contained. We therefore state explicitly that using thread-state detections as an input to individual worker evaluation or discipline is outside the intended use of this dataset, and we ask that deployments separate defect logging from personnel records.

4. Material Waste and Downstream Cost Thread faults detected on a fabric roll are substantially cheaper to remedy than the same faults found in a finished garment, where the usual outcome is rejection of the whole piece. A screen that distinguishes frayed, snagged, and taut conditions rather than emitting a single generic "defect" verdict allows the response to be graded to the condition, so that material is rerouted or repaired instead of discarded. The scale of any such saving depends entirely on the throughput and reject economics of a given plant and is not something this work measures.

5. Accessibility for Small and Medium Producers The corpus was captured with four consumer mobile and compact cameras rather than with line-scan industrial equipment, and the release includes the data under CC BY 4.0 with the evaluation package under the MIT licence. The practical consequence is that a small workshop can reproduce the capture conditions with hardware it already owns and can evaluate a model against the same protocol used here, without a licence fee or a capital purchase. This lowers the entry cost of fine-grained inspection research and deployment for producers in the regions where most garment manufacturing is actually carried out, which is the principal reason the dataset is released openly rather than described only in publication.

6. Risks of Overreliance and Misrepresentation Two failure modes concern us socially rather than technically. The first is treating a visual label as a physical measurement: `wit_taut` records visible tension morphology and carries no claim about tensile force, breaking strength, or fibre twist (Section 8.2), so it must not be quoted in a quality certificate as if it did. The second is transfer beyond the captured domain. The corpus covers a bounded set of fabrics, devices, and lighting conditions, and performance on materials, weaves, or sensors outside that set is unmeasured; a model applied to unseen material may fail unevenly across fabric types in ways that a single aggregate metric will not reveal. Grouped confidence intervals and per-class reporting are required throughout this work (Section 6.3.3) precisely so that such unevenness stays visible rather than being averaged away.

8.3.3 Environmental Analysis & Sustainability

The environmental footprint of this work has two components that behave very differently: the one-off cost of building the corpus and training the baselines, and the recurring cost of running a thread-state model in production. We describe the design decisions that bear on each, and state plainly which quantities we did not measure.

1. Training Footprint of the Baselines Three choices hold the training cost of this work down, and each was made for a methodological reason before it was an environmental one. The baselines exist to demonstrate that the annotations carry learnable signal, not to win a leaderboard, so *no hyperparameter search was performed* (Chapter 5); the compute spent is therefore a small number of fixed-recipe runs rather than a sweep, which is where the bulk of energy in a typical benchmarking study is consumed. The backbones are the smallest members of their families—`ConvNeXt-Tiny`, `Swin-Tiny`, and the nano-scale `YOLO11n-OBB`—trained for 30 and 100 epochs respectively rather than large-capacity variants. And all classification baselines start from ImageNet-1K pretrained weights under a two-stage transfer schedule (Section 6.2), so the representation-learning cost is amortized from an existing public model instead of being paid again from random initialization.

2. Input Resolution as an Energy Decision Compute in a convolutional or attention backbone scales roughly with the square of the input side, so the $1024 \rightarrow 512$ derivation described in Section 3.3 reduces the per-image forward and backward cost by approximately a factor of four relative to training at the annotated resolution. That reduction applies

at every epoch of every run and again at every inference in deployment. The decision was originally driven by the absence of 1024-native models and by memory limits during backpropagation; its effect on energy consumption is a consequence of the same arithmetic, and it is the single largest lever in the pipeline.

3. Reproducibility as Avoided Recomputation Every build emits a `receipt.json` recording seeds, git commit, and SHA-256 digests of inputs and outputs, and evaluation is fixed to three seeds ($[42, 43, 44]$) with deterministic flags enabled. The environmental argument for this discipline is that a result which cannot be reproduced has to be regenerated: unreproducible work is recomputed by every group that needs to build on it, and the aggregate cost of that duplication exceeds the cost of the original run many times over. Releasing the manifests, the split assignment, and the trained checkpoints means a downstream user evaluating against this benchmark does not need to retrain to obtain a comparable number.

4. Avoided Re-Collection Physical image acquisition carries its own material cost—fabric specimens, lighting setups, and the staff time of five capture sessions. An openly licensed corpus (CC BY 4.0) removes the need for the next group working on thread-state recognition to repeat that acquisition in order to have data at all. This is the ordinary sustainability argument for open data, and it applies with particular force in a domain where, as Chapter 2 establishes, no comparable public dataset currently exists.

5. Inference Footprint and Edge Deployment In a deployed inspection line the recurring inference cost dominates the one-off training cost, because the model runs on every frame for the operational life of the system. The edge-optimization path set out in Section 8.4.7—INT8 post-training quantization, layer fusion, and a sub-15 ms latency budget on devices such as the Jetson Orin Nano—is therefore also the energy-reduction path: it moves inference onto low-power local hardware and away from a continuously available server, and INT8 arithmetic draws substantially less energy per operation than FP32. We have not benchmarked this and present it as planned work, not as an achieved result.

6. Downstream Material Effects The material-waste argument is made in Section 8.3.2 and is not repeated here beyond one observation relevant to a life-cycle view: textile production is resource-intensive in water, dye, and energy per unit of finished cloth, so a fault caught at the fabric stage retains all of that already-expended input, while the same fault found after assembly usually discards it. A graded, condition-specific verdict supports repair or rerouting where a generic *defect* flag supports only rejection. We measure no such saving in this work and make no quantitative claim about it.

7. What We Did Not Measure We report no energy consumption, no GPU-hour total, and no estimated CO_2-equivalent figure for the training runs described in Chapter 6, because the instrumentation to measure them was not in place during the experiments. We state this explicitly rather than omit the topic: an unmeasured footprint is not a small one, and the arguments above concern the *direction* of each design decision, not its magnitude. The `receipt.json` mechanism already records the metadata such an accounting would attach

to, and extending it to log device type, wall-clock duration, and estimated energy per run is a small change that would make future releases of this benchmark quantitatively comparable on this axis as well as on accuracy.

8.4 Future Work

The work that follows this release falls into three bands, and the ordering is deliberate. The *near term* is measurement rather than modelling. Four things this release leaves open are measurements, not models: how far independent annotators agree (Section 8.4.13), what explains the detector's distance from a solved task (Section 8.4.3), whether oriented supervision actually outperforms axis-aligned supervision under a common target (Section 8.4.5), and what a naive split would have cost (Section 8.4.6). Until these are done, an improved architecture cannot be distinguished from annotation noise or from a change of evaluation protocol. The *medium term* is capability—stronger backbones, richer supervision, and synthetic augmentation of the rare classes. The *long term* is deployment: full-resolution inference, edge execution, and the calibration work that any consequential use requires. Table 8.1 maps each direction to the specific limitation it is intended to close.

8.4.1 Hybrid & High-Resolution Model Training

To further push the performance frontiers of thread state classification and fine-grained oriented localization, future training initiatives will evaluate specialized architectural backbones—specifically `CoAtNet-0` and `HRNet-W18`:

1. Hybrid Convolutional-Attention Training (CoAtNet-0) While modern convolutional networks (`ConvNeXt-Tiny`) capture local micro-textures and vision transformers (`Swin-Tiny`) model long-range directional continuities, `CoAtNet-0` provides a hybrid architecture combining depthwise convolutions with relative self-attention blocks. Future model training will evaluate `CoAtNet-0` to test whether merging local convolution biases with global attention heads can achieve higher classification precision (> 0.80 Macro-F1) while maintaining low computational latency required for real-time edge deployment.

2. High-Resolution Multi-Scale Representation (HRNet-W18) Thread structural events—particularly unspooling micro-fibers in frayed threads (`wit_fray`)—possess extremely narrow spatial widths ($\approx$ 2–5 px). Conventional deep networks aggressively downsample feature maps through $32\times$ spatial pyramids, discarding delicate sub-pixel fiber structures. `HRNet-W18` maintains high-resolution feature representations across the entire network depth by connecting multi-resolution streams in parallel with continuous cross-scale fusion. Future training of `HRNet-W18` aims to eliminate downsampling information loss, significantly boosting the sensitivity and boundary resolution of micro-fiber splice detection.

Table 8.1: Future directions, the limitation each addresses, and its band.

Direction	**Limitation addressed**	**Band**
Annotation reliability (§8.4.13)	Labels unquantified for agreement	Near
Detection ceiling (§8.4.3)	Gap to a solved task unexplained	Near
Oriented vs. axis-aligned (§8.4.5)	Benefit of orientation unmeasured	Near
Cost of a naive split (§8.4.6)	Leakage inflation unquantified	Near
Calibration & decision support (§8.4.10)	Thresholded verdicts not contestable	Near
Hybrid & high-resolution backbones (§8.4.1)	Micro-fibre detail lost to downsampling	Medium
Generative augmentation (§8.4.2)	`wit_fray` scarcity, wide intervals	Medium
Self-supervised pretraining (§8.4.8)	Unlabelled captures unexploited	Medium
Beyond oriented boxes (§8.4.9)	Curved threads poorly served by a box	Medium
Corpus expansion (§8.4.11)	Bounded fabrics, devices, conditions	Medium
SAHI full-resolution inference (§8.4.4)	Gigapixel rolls, 2 px targets	Long
Edge deployment (§8.4.7)	Offline checkpoints, no latency budget	Long
Benchmark governance (§8.4.12)	Test-set erosion over time	Long

8.4.2 Generative Data Augmentation

In order to overcome the problem of natural class imbalance in rarer categories, especially frayed threads (`wit_fray`), that happen less often in industrial contexts, future research will consider more sophisticated techniques of synthetic data augmentation:

1. Oriented Copy-Paste Augmentation Thread samples of rare classes (`wit_fray` and `wit_taut`) will be extracted from images in their original context based on the annotation of 4 corner bounding polygons of thread objects. Extracted thread polygons will be randomly rotated, resized, and pasted on negative background fabric images (`wot`) to avoid the appearance of artificial edge boundaries confusing the detector.

2. Geometry-Conditioned Diffusion Synthesis (ControlNet) The future research model will be based on Generative Diffusion Models like Stable Diffusion combined with ControlNet. The annotated 4 corner OBB bounding masks will be fed as the spatial conditions along with the text prompts (for example, "frayed cotton thread on dark denim fabric"). ControlNet will generate realistic synthetic images with precisely aligned annotations, increasing the training set diversity by several orders of magnitude.

8.4.3 Detection Training Schedule and What It Rules Out

An earlier draft of this report described the detector as having been stopped early at 30 epochs and diagnosed it as underfit. The run record does not support that account, and the corrected facts are these: the schedule was configured for 100 epochs with a patience of 20; the run *completed* all 100 epochs without early stopping; and the selected checkpoint is epoch 86, after which validation mAP@0.50 did not improve and ended below its peak.
Two consequences follow. First, the reported detection figure is not the product of a truncated schedule, and the underfitting diagnosis is withdrawn. Second, simply extending the budget is not, on this evidence, expected to help: validation performance had already stopped improving fourteen epochs before the schedule ended.
A longer or differently shaped schedule may still be worth testing—a cosine-annealed restart, a longer warm-up, or a larger backbone would each change the optimization problem rather than merely lengthen it—but such a run would be an experiment whose outcome is open, not a correction of a known deficiency. The gap between the detector's score and a solved task is more plausibly attributable to the difficulty of regressing an envelope around a structure a few pixels wide, which is a property of the task rather than of the budget.

8.4.4 Slicing Aided Hyper Inference (SAHI)

For industrial fabric production, fabric roll sizes range from meters wide and gigapixels in total visual resolution. When downscaling the fabric images to the standard network input (512×512 or 640×640), thread-like objects with widths of roughly 2 px become visually undetectable.
In order to address this spatial resolution limitation, future object detection pipelines will incorporate Slicing Aided Hyper Inference (SAHI) [37]:

- **Overlapping Tiling:** Full resolution images are divided into overlapping sub-windows with dimensions of 512×512 pixels and an overlap step ratio (for instance, 20% spatial overlap).
- **Native-Resolution Inference:** each tile is processed at the sensor's own sampling, so detail that downscaling would discard is retained. Tiling preserves detail that was captured; it cannot restore structure that the acquisition optics or sensor never resolved, and no claim of recovering sub-pixel information is made.
- **Global Coordinate Mapping & Rotated NMS:** Detected fiber coordinates are transformed into a global coordinate system. Detection overlaps are aggregated across tiles by Rotated Non-Maxima Suppression (R-NMS) with Probabilistic Intersection over Union (IoU) matching criterion.

8.4.5 Oriented versus Axis-Aligned Supervision: The Controlled Comparison

The report argues geometrically for oriented annotation (Section 7.5) but does not measure the advantage, and the comparison is easy to run badly: two box geometries produce different matching outcomes under the same nominal IoU threshold, so a naive comparison of an oriented score against a horizontal one confounds the representation with the evaluation target. A defensible experiment would fix the detector family, training budget, augmentation, checkpoint-selection rule, and seed set across both arms; derive the horizontal labels from the same canonical annotations by axis-aligned enclosure; and evaluate both arms against a *common* spatial target, reporting the paired per-photograph difference rather than two independent scores. The mechanisms currently offered as explanation—envelope inflation and suppression of adjacent instances—should be measured directly, as an envelope-area ratio between the two label sets and as a count of instances removed by suppression, rather than inferred from a difference in mAP.

8.4.6 Quantifying the Cost of a Naive Split

Photograph grouping closes a leakage pathway by construction, but this release does not measure how much a naive random split would have inflated its scores (Section 4.2.3). Training both protocols to completion would supply that figure. The comparison needs care for the reason that changing the split changes the evaluation population as well as its independence: class prevalence, device mix, and difficulty all shift, so the two protocols must be reported with their populations side by side, and the result described as a difference between evaluation protocols unless the population is matched.

8.4.7 Edge Deployment & Optimization

In order to move baseline models from offline research checkpoints to real-time industrial application, the following engineering challenges will need to be addressed:

1. Model Export & Execution The baseline PyTorch models will be exported to ONNX format and executed through the use of the NVIDIA TensorRT for edge inference.

2. INT8 Quantization & Layer Fusion The baseline model will be subjected to post-training quantization (PTQ) and quantization-aware training (QAT) to reduce FP32 weights to INT8 precision weights.

3. Latency Benchmarking Edge devices that will be targeted for benchmarking include (NVIDIA Jetson Orin Nano / Industrial IPCs). These will undergo benchmarks using fast line-scan camera feeds. The target latency budget for inference is less than 15 ms per frame, allowing real-time automated thread-state inspection on industrial weaving looms running at 1,000 + RPM.

8.4.8 Self-Supervised Pretraining on Unlabelled Fabric Imagery

Annotation, not capture, is the binding constraint on this corpus: photographing fabric is cheap, and drawing an oriented box around every thread event in a frame is not. The acquisition sessions therefore produced substantially more imagery than carries instance annotations, and that surplus is currently unused.
Self-supervised pretraining converts it into signal. A masked-autoencoder or DINO-style objective trained on unlabelled fabric crops learns weave, illumination, and sensor characteristics of *this* domain, which ImageNet-1K pretraining cannot supply—ImageNet contains almost no macro textile surface at < 10 cm working distance. Future work will pretrain a backbone on the unlabelled pool and then fine-tune it under the recipes of Section 6.2, holding every other factor fixed so the contribution of in-domain pretraining is isolable. The evaluation of interest is not only peak Macro-F1 but the label-efficiency curve: performance as a function of the fraction of annotated training data retained. If in-domain pretraining reaches baseline accuracy from a quarter of the annotations, that result governs how every subsequent capture phase should allocate annotator time.

8.4.9 Beyond Oriented Boxes: Curvilinear and Pixel-Level Supervision

An oriented rectangle is a good envelope for a straight thread span and a progressively worse one as the thread curves. A snagged loop is the limiting case: its minimum-area oriented rectangle encloses mostly fabric, which reintroduces on a smaller scale the envelope-inflation problem that motivates oriented boxes in the first place (Section 7.5). Two extensions are planned, and they are complementary rather than alternatives.

1. Centreline and Ribbon Representations A thread is a curvilinear structure and is more naturally described by a centreline polyline with an associated width than by any rectangle. Predicting a centreline permits quantities that a box cannot express—arc length, local curvature, the number of crossings in a snag—and these are plausibly the measurements an inspection system ultimately wants. The existing oriented boxes provide a usable

initialization for semi-automatic centreline annotation, so the incremental labelling cost is far below annotating from scratch.

2. A Segmentation Subset Pixel-tight masks over the full corpus were rejected as infeasible for the reason given in Section 8.2: entangled fibres have no continuous pixel boundary. A stratified subset is nonetheless tractable and would serve two purposes—supporting mask-supervised methods, and providing the reference against which the true tightness of an oriented box can be measured rather than assumed. We would size that subset from the reliability study, since the classes with the weakest geometric agreement are the ones a mask most helps to disambiguate.

8.4.10 Calibration & Decision Support

Section 8.3.1 argues that a consequential deployment must expose more than a pass/fail verdict. That requirement is an engineering programme, not a disclaimer, and it has not yet been carried out.

Thresholds here are fitted on validation data to maximize Macro-F1 (Section 6.2.3), which is the right objective for a benchmark and the wrong one for a factory: a plant rejecting consignments and a plant triaging a repair queue have different costs of a false positive and should not operate at the same point. Future work will report per-class reliability diagrams and expected calibration error, apply temperature scaling or isotonic regression fitted strictly on validation, and publish the full operating curve so that a deployer can select a point against their own cost matrix rather than inheriting ours. The evaluation package is the natural home for this, so that a calibrated operating point travels with the model rather than being rediscovered per site. A further step, valuable where an inspector adjudicates model output, is selective prediction: allow the model to abstain on low-confidence cases and report accuracy as a function of coverage, which measures directly what a decision-support deployment actually experiences.

8.4.11 Corpus Expansion & External Validity

The corpus is bounded in fabrics, devices, and lighting, and Section 8.2 states the consequence: performance outside those bounds is unmeasured. Three expansions address different parts of that boundary.

1. Material and Weave Coverage Additional fabric families—heavier denim, knitwear, synthetics, and strongly patterned prints—extend the range over which the taxonomy has been exercised. Patterned fabrics matter disproportionately, since Section 7.6 identifies pattern intersections as a source of low-confidence false positives for `wit_snag`.

2. Industrial Sensing Conditions A line-scan capture phase under stroboscopic illumination would test the transfer that this release explicitly does not claim. Even a modest paired subset—the same specimens under both consumer macro and line-scan capture—would quantify the domain gap rather than leaving it as an acknowledged unknown.

3. A Fabric-Held-Out Protocol Alongside the existing capture-grouped split, a protocol that holds out entire fabric families from training would measure generalization to unseen material rather than to unseen photographs. This is the closest available substitute for the specimen-independence property that Section 8.3.1 declines to claim, and it is obtainable from richer provenance metadata without abandoning the grouping rule that governs the current split.

8.4.12 Benchmark Governance & Longevity

A public test set erodes: repeated evaluation against it, across many papers, gradually converts held-out data into data that has been fitted to, and the protocol safeguards of Chapter 4 do not survive that erosion by themselves. Three measures are planned. Releases will be versioned with immutable digests, so a reported number always names the corpus state it was obtained on. A reference evaluation harness ships with the data, so results are computed by the same code rather than by reimplementations that quietly differ in thresholding or interval construction. And a reserved, periodically refreshed evaluation partition—drawn from later capture phases and never distributed with the main release—would allow an occasional check of whether leaderboard progress reflects genuine generalization. We regard the third as the most valuable and the most demanding, and we note it here as an intention rather than a commitment.

8.4.13 Annotation Reliability

To assess the consistency of boundary geometry and state labels across annotators, a stratified inter-annotator agreement (IAA) study is planned on a reliability subset of approximately 400 target oriented bounding boxes (OBBs), with approximately 100 instances per positive morphological class—normal, fray, snag, and taut—stratified by class label, spatial aspect ratio, and split partition .

Under the planned protocol, the subset will be annotated independently by two expert annotators, with a senior reviewer adjudicating disagreements. Two measures are specified: label agreement, assessed using Cohen's κ over the four positive classes, and geometric agreement, assessed using the mean oriented intersection-over-union (IoU) over the four-corner OBB polygons.

As of this release, this study has not been carried out, and no agreement statistics are reported. Annotation verification to date consists of a per-batch full-review attestation (`review_status: reviewed_complete`), which is a batch-level statement rather than a per-image, multi-annotator measurement.

One limitation of the planned design is also noted: because agreement is computed over boxes that were drawn, it cannot quantify omission (i.e., threads that are present but unboxed). Thus, the proposed protocol bounds boundary and label consistency but does not measure annotation completeness.

References

[1] R Chattopadhyay. "Introduction: types of technical textile yarn". In: *Technical Textile Yarns*. Elsevier, 2010, pp. 3–55.

[2] Vishal Sharma, Mamta Mahara, and Akanksha Sharma. "On the textile fibre's analysis for forensics, utilizing FTIR spectroscopy and machine learning methods". In: *Forensic Chemistry* 39 (2024), p. 100576.

[3] Jing Yang et al. "Using deep learning to detect defects in manufacturing: a comprehensive survey and current challenges". In: *Materials* 13.24 (2020), p. 5755.

[4] Zhengxia Zou et al. "Object detection in 20 years: A survey". In: *Proceedings of the IEEE* 111.3 (2023), pp. 257–276.

[5] Ahmet Ozek et al. "Artificial intelligence driving innovation in textile defect detection". In: *Textiles* 5.2 (2025), p. 12.

[6] Chao Li et al. "Fabric defect detection in textile manufacturing: a survey of the state of the art". In: *Security and Communication Networks* 2021.1 (2021), p. 9948808.

[7] Rui Carrilho, Kailash A Hambarde, and Hugo Proença. "A novel dataset for fabric defect detection: bridging gaps in anomaly detection". In: *Applied Sciences* 14.12 (2024), p. 5298.

[8] Meng An et al. "Fabric defect detection using deep learning: An Improved Faster R-approach". In: *2020 International Conference on Computer Vision, Image and Deep Learning (CVIDL)*. IEEE. 2020, pp. 319–324.

[9] Filipe Pereira et al. "A novel deep learning approach for yarn hairiness characterization using an improved YOLOv5 algorithm". In: *Applied Sciences* 15.1 (2024), p. 149.

[10] Filipe Pereira et al. "Online yarn hairiness–Loop & protruding fibers dataset". In: *Data in Brief* 54 (2024), p. 110355.

[11] Abu Sayem Md Siam et al. "TextileNet: A deep learning approach for textile fabric material identification from OCT and macro images". In: *2023 26th International Conference on Computer and Information Technology (ICCIT)*. IEEE. 2023, pp. 1–6.

[12] Momina Liaqat Ali and Zhou Zhang. "The YOLO framework: A comprehensive review of evolution, applications, and benchmarks in object detection". In: *Computers* 13.12 (2024), p. 336.

[13] Sercan Külcü. "High-Precision Marine Radar Object Detection Using Tiled Training and SAHI Enhanced YOLOv11-OBB". In: *Sensors* 26.3 (2026), p. 942.

[14] Filipe Pereira et al. "Intelligent computer vision system for analysis and characterization of yarn quality". In: *Electronics* 12.1 (2023), p. 236.

[15] Patrick Ramos, Ryan Ramos, and Noa Garcia. "Data leakage in visual datasets". In: *2025 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*. IEEE. 2025, pp. 6368–6378.

[16] Björn Barz and Joachim Denzler. "Do we train on test data? purging cifar of near-duplicates". In: *Journal of Imaging* 6.6 (2020), p. 41.

[17] SHA Arachchi et al. "Classification of Defects of Cotton Yarns Using Convolutional Neural Networks". In: *2024 9th International Conference on Information Technology Research (ICITR)*. IEEE. 2024, pp. 1–6.

[18] Ayoub Benali Amjoud and Mustapha Amrouch. "Object detection using deep learning, CNNs and vision transformers: A review". In: *IEEE Access* 11 (2023), pp. 35479–35516.

[19] Very Very and Eko Rudiawan Jamzuri. "Robotic Bin-Picking Object Detection Using YOLOv11-OBB with SAM2 Auto-Annotation". In: *Journal of Applied Informatics and Computing* 10.3 (2026), pp. 3126–3131.

[20] Suman Kunwar. "Annotate-Lab: simplifying image annotation". In: *Journal of Open Source Software* 9.103 (2024), p. 7210.

[21] Vighnesh Birodkar, Hossein Mobahi, and Samy Bengio. "Semantic redundancies in image-classification datasets: The 10% you don't need". In: *arXiv preprint arXiv:1901.11409* (2019).

[22] Ze Liu et al. "Swin transformer: Hierarchical vision transformer using shifted windows". In: *2021 IEEE/CVF international conference on computer vision (ICCV)*. Ieee. 2021, pp. 9992–10002.

[23] Filipe Pereira et al. "A review in the use of artificial intelligence in textile industry". In: *International Conference Innovation in Engineering*. Springer. 2021, pp. 377–392.

[24] Swash Sami Mohammed and Hülya Gökalp Clarke. "A hybrid machine learning approach to fabric defect detection and classification". In: *International congress of electrical and computer engineering*. Springer. 2022, pp. 135–147.

[25] Jun-Feng Jing, Hao Ma, and Huan-Huan Zhang. "Automatic fabric defect detection using a deep convolutional neural network". In: *Coloration Technology* 135.3 (2019), pp. 213–223.

[26] Dejun Zheng, Yu Han, and Jin Lian Hu. "A new method for classification of woven structure for yarn-dyed fabric". In: *Textile Research Journal* 84.1 (2014), pp. 78–95.

[27] P Yashini et al. "Machine Learning-Based Textile Fabric Defect Detection Network". In: *2024 4th International Conference on Sustainable Expert Systems (ICSES)*. IEEE. 2024, pp. 1470–1477.

[28] Noman Haleem, Matteo Bustreo, and Alessio Del Bue. "An online yarn spinning dataset". In: *Data in brief* 44 (2022), p. 108557.

[29] Anindya Ghosh et al. "A proposed system for cotton yarn defects classification using probabilistic neural network". In: *International conference on recent advances and innovations in engineering (ICRAIE-2014)*. IEEE. 2014, pp. 1–6.

[30] AE Amin. "A novel classification model for cotton yarn quality based on trained neural network using genetic algorithm". In: *Knowledge-Based Systems* 39 (2013), pp. 124–132.

[31] Bahareh Azimi, Mohammad Amani Tehran, and Mohammad Reza Mohades Mojtahedi. "Prediction of false twist textured yarn properties by artificial neural network methodology". In: *Journal of Engineered Fibers and Fabrics* 8.3 (2013), p. 155892501300800312.

[32] Zhuang Liu et al. "A convnet for the 2020s". In: *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*. IEEE. 2022, pp. 11966–11976.

[33] Burcu Çarklı Yavuz. "Scale-dependent performance analysis of yolo26 and yolov11 for ppe detection". In: *Electronics* 15.6 (2026), p. 1146.

[34] Ranjan Sapkota et al. "YOLO advances to its genesis: A decadal and comprehensive review of the You Only Look Once (YOLO) series". In: *Artificial Intelligence Review* 58.9 (2025), p. 274.

[35] Nidhal Jegham et al. "Yolo evolution: A comprehensive benchmark and architectural review of yolov12, yolo11, and their previous versions". In: *arXiv preprint arXiv:2411.00201* (2024).

[36] Zihang Dai et al. "Coatnet: Marrying convolution and attention for all data sizes". In: *Advances in neural information processing systems* 34 (2021), pp. 3965–3977.

[37] Fatih Cagatay Akyön, Sinan Onur Altinuc, and Alptekin Temizel. "Slicing Aided Hyper Inference and Fine-tuning for Small Object Detection". In: *2022 IEEE International Conference on Image Processing (ICIP)*. 2022, pp. 966–970. DOI: 10.1109/ICIP46576.2022.9897990.