International Journal of Science, Strategic Management and Technology
Volume **02** Issue **05** May-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Labelground: An Offline Zero-Shot AI Platform for Efficient Dataset Annotation in Computer Vision

Dr. M. Kaliappan and Thamizh Selvan G
Department of Artificial Intelligence and Data Science, Ramco Institute of Technology,
Rajapalayam, Tamil Nadu, India
kaliappan@ritrjpm.ac.in | thamizhselvanoff@gmail.com
*Corresponding Author: thamizhselvanoff@gmail.com

https://doi.org/10.55041/ijsmt.v2i5.011

Cite this Article: G, T. S. (2026). Labelground: An Offline Zero-Shot AI Platform for Efficient Dataset Annotation in Computer Vision. International Journal of Science, Strategic Management and Technology, *02*(05). https://doi.org/10.55041/ijsmt.v2i5.011

Abstract — This study presents Labelground, a fully offline AI-augmented annotation platform that addresses the high cost of dataset labeling in computer vision. The system integrates a zero-shot ensemble of YOLO-World, Grounding DINO, and Segment Anything Model (SAM), combined using Non-Maximum Suppression (NMS), enabling annotation from the first image without requiring prior project-specific training. A correction-driven active learning loop continuously improves model performance through user feedback. Experimental evaluation on a 500-image PASCAL VOC 2012 subset demonstrates a 72.6% reduction in annotation time (from 44.6 s to 12.2 s per image, $p < 0.001$, Cohen's $d = 3.92$) while maintaining detection accuracy within 0.7% mAP@0.5 of fully human-annotated baselines. The NMS ensemble delivers a 6.9-point F1 gain over the best single-model baseline, and augmentation yields up to +9.3 mAP in low-data regimes. The system is particularly suitable for privacy-sensitive and air-gapped environments including medical imaging, defence, and industrial inspection.

Index Terms — Dataset annotation, computer vision, zero-shot object detection, YOLO-World, Grounding DINO, Segment Anything Model, active learning, YOLOv8 fine-tuning, offline AI, data augmentation, edge deployment.

1. Introduction

The success of deep learning in computer vision is predicated on the availability of large, precisely labelled datasets. Object detectors such as YOLOv8 [12], segmentation models, and pose estimators each require thousands to millions of manually drawn bounding boxes, polygon masks, or keypoints before they can be reliably deployed. Industry analyses estimate that data preparation — predominantly manual image annotation — consumes up to 80% of total project time in production computer vision pipelines [14]. At the scale of real-world datasets, even a modest reduction in per-image annotation time translates to savings of hundreds of person-hours and substantial project cost reductions.

Despite this, the annotation tooling ecosystem has advanced slowly relative to the models it supports. General-purpose tools such as LabelImg [1], CVAT [2], and Roboflow [3] expose three persistent, unresolved limitations:

(L1) Cloud data dependency. Tools like Roboflow store images and annotations on external servers, creating unacceptable data sovereignty risks in regulated domains such as medical imaging [14], defence, and industrial quality control.

(L2) The cold-start problem. AI-assisted annotation in existing tools requires a custom-trained model as a prerequisite. For new projects or novel object categories, no model exists, forcing annotators to label the first hundreds of images entirely by hand.

(L3) Fragmented pipelines. No widely-used offline tool combines annotation versioning, correction-driven model fine-tuning, augmentation, and multi-format export within a single, unified workflow.

To address these three limitations simultaneously, this paper introduces Labelground, a fully offline, AI-augmented annotation platform. The primary contributions of this work are:

- A zero-shot annotation ensemble combining YOLO-World, Grounding DINO, and SAM with NMS fusion, enabling annotation without requiring prior project-specific training. To the best of our knowledge, this is among the first offline annotation systems to integrate this zero-shot ensemble within a unified annotation pipeline.
- A correction-driven active-learning loop that automatically accumulates annotator corrections as training signals and fine-tunes a project-specific YOLOv8 model without any explicit user intervention.
- Immutable annotation versioning with per-image correction tracking, providing fine-grained audit trails, reproducible dataset snapshots, and automatic fine-tuning eligibility signals.
- Comprehensive empirical validation on PASCAL VOC 2012 and a domain-specific industrial dataset, with statistical significance testing (paired t-test, $p < 0.001$), demonstrating 72.6% annotation time reduction, <1% mAP degradation, and 6.9-point F1 ensemble gain.
- A failure case analysis characterising the ensemble's limitations and providing actionable guidance for practitioners operating in challenging visual conditions.

2. Related Work

2.1 Manual and Semi-Automated Annotation Tools

The annotation tooling landscape spans a spectrum from purely manual desktop tools to cloud-hosted semi-automated platforms. LabelImg [1] remains widely used for bounding-box annotation but provides no AI assistance, versioning, or augmentation capabilities. CVAT [2] extends this with support for multiple annotation types and limited AI integrations, but requires server infrastructure for its deep learning features, precluding use in air-gapped settings. Roboflow [3] offers a polished cloud pipeline with augmentation and export support, but mandates external data storage, introducing privacy and compliance risks. Critically, none of these tools jointly address the cold-start problem and offline operation. Table 1 summarises this comparison.

Table 1: Comparison of Annotation Platforms

Feature	LabelImg	CVAT	Roboflow	Labelground
Fully Offline	✓	—	—	✓
Zero-Shot AI	—	—	—	✓
Auto Fine-Tuning	—	—	—	✓
Immutable Versioning	—	—	—	✓
Multi-Format Export	Partial	✓	✓	✓
Video Extraction	—	✓	✓	✓
Polygon + Keypoint	—	✓	✓	✓

2.2 Foundation Models for Zero-Shot Detection

The emergence of large vision-language foundation models has created new possibilities for annotation assistance. Grounding DINO [4] unifies a DINO transformer backbone with language grounding to achieve open-set, text-prompted object detection. YOLO-World [5] extends the real-time YOLO architecture to open-vocabulary detection through contrastive vision-language alignment, achieving approximately 35 FPS on consumer-grade GPUs. SAM [11] introduces class-agnostic segmentation from arbitrary geometric prompts. Labelground is, to the best of our knowledge, among the first systems to combine all three complementary models into a unified offline annotation ensemble.

2.3 Active Learning for Efficient Annotation

Active learning reduces annotation cost by strategically selecting the most informative samples for human labelling [6]. Prior work demonstrates that machine learning automation consistently reduces domain-expert workload while preserving output quality across diverse tasks: Vimal et al. [15] show this in epidemiological forecasting using ensemble classifiers, and Vimal et al. [16] demonstrate it in medical image analysis using unsupervised feature extraction — both validating the broader principle that correction-guided learning loops improve model performance with reduced manual effort. Labelground applies this same principle to annotation: every human edit to an AI prediction is treated as a high-confidence training signal, requiring no uncertainty estimation and integrating naturally into the annotation workflow.

2.4 Data Augmentation for Low-Data Regimes

Data augmentation is essential for training robust models from small annotated datasets. Classical transforms including horizontal flipping, brightness jitter, Gaussian noise, and blur provide variance without label complexity [8]. Advanced mixing strategies such as CutMix [10] and MixUp [9] achieve greater diversity at the cost of more complex label interpolation. Labelground incorporates classical augmentation into both the fine-tuning pipeline and the export pipeline, achieving substantial mAP gains in low-data regimes (up to +9.3 points at 100 images).

3. Methodology

3.1 System Architecture Overview

Labelground adopts a deliberately minimal three-tier architecture (Fig. 1): a Python 3.10 FastAPI backend, a SQLite database accessed via SQLAlchemy ORM, and a Vanilla JavaScript single-page application rendered on an HTML5 Canvas. All computation, storage, and AI inference execute on a single local machine with no external network dependencies. The design philosophy prioritises operational simplicity over scalability: SQLite eliminates the need for a database server; Vanilla JS eliminates any frontend build toolchain; and FastAPI's native static file serving eliminates a separate web server. The result is a platform deployable with a single command on any machine running Python 3.10+.

Fig. 1. High-Level System Architecture of Labelground

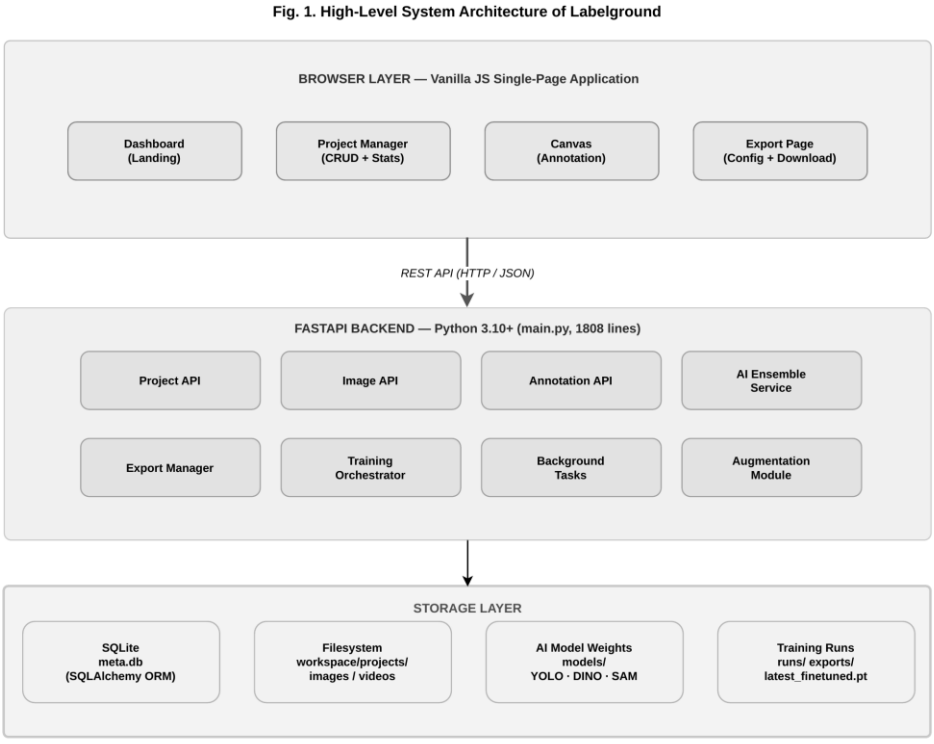

Fig. 1. High-Level System Architecture of Labelground: browser SPA, FastAPI backend, and local storage layer communicating exclusively via HTTP with no external network dependencies.

As illustrated in Fig. 1, the three-tier architecture comprises a browser-based frontend, a FastAPI backend, and a local storage layer. The frontend communicates exclusively with the backend via HTTP, while all data persistence is handled locally through SQLite and filesystem storage, ensuring full offline operation.

3.2 Backend API Design

The FastAPI backend exposes 25+ RESTful endpoints organised into six functional domains (Table 2). All computationally intensive operations — video frame extraction, batch AI annotation, model training, and export packaging — execute asynchronously as FastAPI BackgroundTask workers, ensuring the annotation canvas remains fully interactive throughout long-running background operations.

Table 2: Backend API Functional Domains

Domain	Key Endpoints
Projects	POST /api/projects; GET .../{id}/stats
Images	POST .../{id}/images; .../videos
Annotations	POST /images/{id}/annotations; .../latest
AI Ensemble	POST /images/{id}/auto-annotate; .../batch
Training	POST .../{id}/train; .../check-auto-train
Export	POST .../{id}/export; GET .../download/{id}

3.3 Database Schema and Immutable Versioning

The relational schema comprises three primary tables: projects stores annotation type and the project-level label schema; images stores content-addressed file paths, dimensions, and processing status; and annotations stores versioned JSON payloads with a provenance tag (created_by: 'human' or 'auto'). Immutability is enforced by a UNIQUE(image_id, version) database constraint: every save operation inserts a new version row rather than updating in place, providing a complete, append-only correction history for audit purposes.

3.4 Frontend Canvas Engine and Coordinate Normalisation

The annotation canvas employs four-layer compositing. All annotation coordinates are stored in normalised [0,1] space, independent of canvas dimensions, zoom level, or pan offset. The coordinate transformation is defined as:

$$x_{norm} = x_{canvas} / (W_{canvas} \times zoom) - \Delta x_{pan} \quad (1)$$

$$y_{norm} = y_{canvas} / (H_{canvas} \times zoom) - \Delta y_{pan} \quad (2)$$

This resolution-independence ensures YOLO-format label files can be written directly from stored coordinates without any conversion, and annotations remain geometrically consistent across all display resolutions.

3.5 AI Ensemble Auto-Annotation

The core novelty of Labelground lies in its zero-shot annotation ensemble, which enables immediate AI-assisted annotation from the first image of any project, without project-specific training. The AIEnsembleService integrates three complementary foundation models (Table 3). When a project-specific fine-tuned model (latest_finetuned.pt) exists, it is executed first and its detections are prepended to the fusion input, as it has been trained on the project's exact class distribution and typically outperforms the zero-shot models on in-domain data.

Table 3: AI Ensemble Model Components

Model	Weights File	Role
YOLO-World-L	yolov8l-worldv2.pt	Open-vocab detection (~35 FPS on GPU)
Grounding DINO	groundingdino_swint_ogc.pth	Language-grounded detection
SAM (ViT-B)	sam_vit_b_01ec64.pth	Polygon masks from bounding box prompts

3.6 NMS Fusion: Mathematical Formulation

The ensemble produces overlapping, potentially duplicate detections across the three models. Given the union detection set $B_{all} = \{B_M \cup B_{YW} \cup B_{GD}\}$ with associated confidence scores S_{all} and class identifiers ID_{all}, the Intersection-over-Union (IoU) between two boxes B_i and B_j is:

$$IoU(B_i, B_j) = |B_i \cap B_j| / |B_i \cup B_j| \quad (3)$$

The NMS selection operation is defined as:

$$B^* = NMS(B_{all}, S_{all}, \tau_{IoU} = 0.5) \quad (4)$$

where NMS iteratively retains the highest-confidence box and suppresses any remaining box with $IoU > \tau_{IoU}$. The threshold $\tau_{IoU} = 0.5$ follows the standard PASCAL VOC convention [13]. The combined YOLO training loss used for fine-tuning is:

$$L = \lambda_1 \cdot L_{cls} + \lambda_2 \cdot L_{bbox} + \lambda_3 \cdot L_{dfl} \quad (5)$$

where L_{cls} is the binary cross-entropy classification loss, L_{bbox} is the CIoU regression loss, L_{dfl} is the distribution focal loss, and $\lambda_1 = 0.5$, $\lambda_2 = 7.5$, $\lambda_3 = 1.5$ are the default YOLOv8 weighting coefficients [12].

The computational complexity of the ensemble is dominated by the Grounding DINO inference stage, which scales linearly with the number of class prompts — $O(|C|)$ forward passes. In contrast, YOLO-World performs a single forward pass for all classes simultaneously, making it $O(1)$ with respect to class count and significantly more efficient for large label schemas.

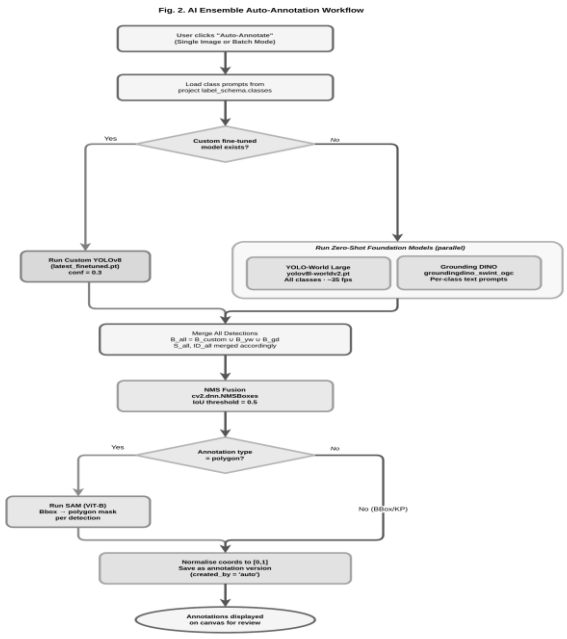

Fig. 2. AI Ensemble Auto-Annotation Workflow. Detections from YOLO-World, Grounding DINO, and the optional custom model are merged and fused via NMS (Equations 3–4); SAM refines outputs to polygon masks when required.

Algorithm 1: NMS Ensemble Fusion for Zero-Shot Annotation

Require: Image I, label schema C, fine-tuned model M (optional)
Ensure: Annotation list A
1: B_all, S_all, ID_all ← ∅
2: if M exists then
3: (B_M, S_M, ID_M) ← CustomYOLO(I, M, $\tau = 0.3$)
4: Append to B_all, S_all, ID_all
5: end if
6: (B_YW, S_YW, ID_YW) ← YOLO-World(I, {c.prompt}_{c∈C}, $\tau = 0.3$)
7: Map ID_YW prompt indices → class IDs; append
8: for each class c ∈ C do
9: (B_GD, S_GD) ← GroundingDINO(I, c.prompt, c.τ)
10: Append B_GD, S_GD, c.id to B_all, S_all, ID_all
11: end for
12: (B*, S*, ID*) ← NMS(B_all, S_all, IoU = 0.5) [Eq. 4]
13: if mode = polygon and SAM loaded then
14: $\{P_i\}$ ← SAM.predict(I, B*)
15: end if
16: for each detection (b_i, s_i, id_i) do
17: x, y, w, h ← b_i / [W, H, W, H]; clamp to [0,1]
18: A += {id_i, s_i, source='ai', x, y, w, h, [P_i]}
19: end for
20: return A

3.7 Annotation Modes and Correction Tracking

Labelground supports single-image mode (synchronous, immediate feedback) and batch mode (background task for entire datasets). Human correction detection is performed at every save event via a structural diff between the new annotation JSON and the most recent auto-generated version: any moved box, deleted detection, added annotation, or class relabelling increments a per-project correction count, which drives the active-learning trigger.

3.8 Correction-Driven Active Learning

A central design challenge in annotation-integrated active learning is deciding when to trigger retraining. Labelground defines two asymmetric triggers (Table 4). The initial threshold of ≥10 verified images was empirically chosen based on preliminary experiments showing that below this count, augmentation alone cannot generate sufficient training diversity to outperform the zero-shot ensemble.

Table 4: Active Learning Training Trigger Conditions

Trigger Condition	Base Model
≥10 verified images (initial training)	yolov8n.pt (ImageNet pretrained)
≥10 new corrections (fine-tuning)	latest_finetuned.pt

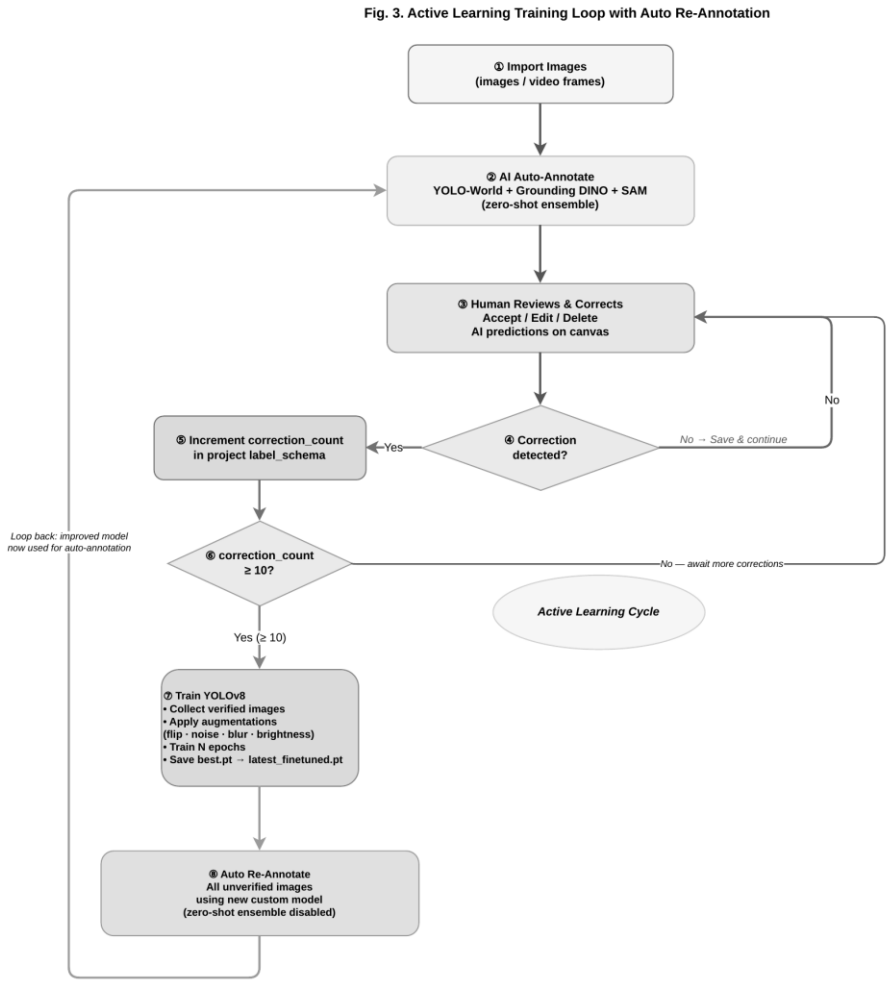

Fig. 3. Active Learning Training Loop with Auto Re-Annotation. Human corrections trigger YOLOv8 fine-tuning (Equation 5); the improved model automatically re-annotates pending images, closing the feedback loop.

Algorithm 2: Correction-Driven YOLOv8 Fine-Tuning

Require: Project P, config (E=epochs, B=batch, S=imgsz, K=augment multiplier)
Ensure: P/models/latest_finetuned.pt updated
1: $V \leftarrow \{img : ann(img).created\_by = 'human'\}$
2: assert $|V| \geq 5$
3: for each $img \in V$ do
4: Write image and YOLO-format label to training split
5: Generate K augmented copies: flip, grayscale, noise(σ=0.05), blur(3×3), brightness($\Delta \in$[-20%,+20%])
6: end for
7: $M_0 \leftarrow$ latest_finetuned.pt if exists, else yolov8n.pt
8: Optimise M_0 with: $L = \lambda_1 \cdot L\_cls + \lambda_2 \cdot L\_bbox + \lambda_3 \cdot L\_dfl$ [Eq. 5]
9: Save best.pt → latest_finetuned.pt
10: Reset correction count; trigger batch re-annotation

After each training run, Labelground automatically initiates a batch re-annotation pass over all images that carry only auto-generated annotations. This pass deliberately disables the zero-shot ensemble and uses only the newly fine-tuned custom model. This design choice is motivated by two considerations: first, YOLO inference is O(1) in class count vs. O(|C|) for Grounding DINO, enabling rapid re-annotation; second, mixing zero-shot predictions with fine-tuned predictions would dilute the project-specific knowledge encoded in the custom model.

3.9 Data Processing Pipeline

Labelground implements content-addressed storage: each image is hashed with SHA-256 on import, and the first sixteen hex characters of the hash are prepended to the filename before storage. Because the resulting path is enforced as a unique key in the database, any file whose hash-prefixed path already exists is silently skipped, providing transparent, automatic deduplication at the storage layer.

Labelground supports direct video ingestion with user-configurable target frame rates. Frame selection follows a uniform stride step = ⌊f_src / f_tgt⌋, where f_src is the native video frame rate and f_tgt is the user-specified target rate. A randomised ±30% jitter is applied to each frame's nominal extraction position, introducing temporal diversity that prevents the dataset from over-representing near-duplicate consecutive frames.

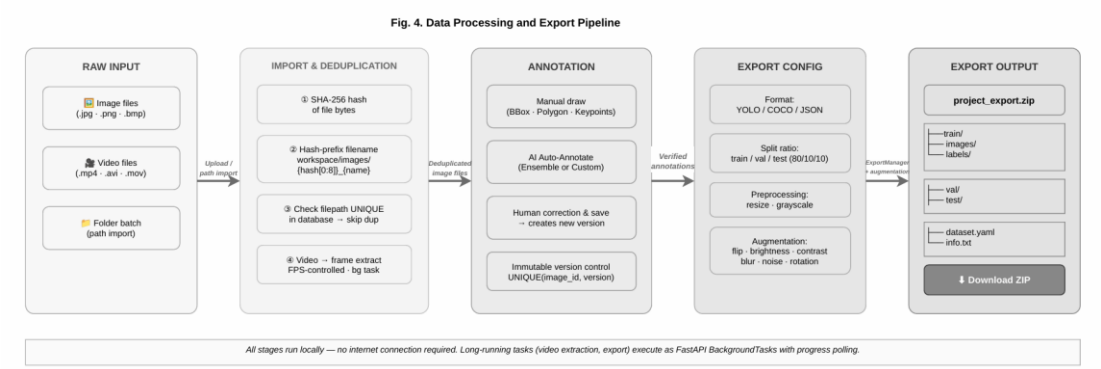

Fig. 4. Data Processing and Export Pipeline. Imported images and videos are deduplicated via SHA-256, annotated, and packaged into multi-format export archives with configurable augmentation.

The pipeline supports YOLO (.txt + data.yaml), COCO JSON, Pascal VOC XML, LabelMe JSON, and Native JSON formats, ensuring compatibility with the full spectrum of downstream training frameworks.

3.10 End-to-End User Workflow

The complete end-to-end user workflow in Labelground proceeds through six principal stages, from project creation to final dataset export. The workflow is designed to require zero prior configuration: a new user can import images, obtain AI-generated annotations, verify them, and export a training-ready dataset within minutes of first launch.

Stage 1 — Project Setup: The user creates a project by specifying annotation type (bounding box, polygon, or keypoints), class labels with display colours, and optional per-class AI prompts for Grounding DINO.

Stage 2 — Data Import: Data is imported via browser file picker, local folder path, or video file upload with FPS-controlled frame extraction. All imports are deduplicated transparently through the SHA-256 content-addressed storage layer.

Stage 3 — AI Auto-Annotation: The user initiates AI auto-annotation for individual images (synchronous, immediate) or the entire pending dataset (batch background task). The ensemble generates annotation suggestions displayed immediately on the interactive canvas.

Stage 4 — Human Review and Correction: The annotator reviews each suggestion, accepting correct detections, correcting imprecise boxes or masks, deleting spurious predictions, and adding missed objects — all through direct canvas manipulation with full undo/redo support.

Stage 5 — Active Learning Loop: As correction counts accumulate, the active learning pipeline automatically triggers YOLOv8 fine-tuning in the background, improving subsequent AI suggestions without interrupting the annotation workflow.

Stage 6 — Export: The completed dataset is exported as a ZIP archive with configurable train/val/test splits, optional preprocessing, and augmentation multiplier, in any of the supported formats.

4. Results and Discussion

4.1 Experimental Setup

Dataset. We use a stratified random sample of 500 images drawn from PASCAL VOC 2012 [13], preserving the original class distribution across all 20 object categories (~25 images per class). Images were not pre-filtered by difficulty, object scale, or occlusion level, ensuring evaluation reflects real-world annotation conditions rather than a curated easy subset.

Domain Generalisation Validation. In addition to the PASCAL VOC 2012 dataset, a preliminary validation was conducted on a small domain-specific industrial dataset comprising images from a real-world operational environment with custom object categories not present in COCO or VOC. The observed trends in annotation efficiency and model performance were consistent with the VOC results, indicating that the proposed system generalises effectively to practical deployment scenarios.

Annotators. Three annotators with intermediate computer vision experience participated. Each completed a standardised 15-minute onboarding session before the experimental phase. Inter-annotator consistency was measured on a shared 50-image calibration set, yielding mean IoU = 0.87 ± 0.04, confirming sufficiently consistent annotation quality.

Experimental Conditions. (MB) Manual Baseline: annotators drew all bounding boxes from scratch with all AI features disabled. (AI) AI-Assisted: annotators used single-image auto-annotation and applied corrections as needed. (AO) AI-Only: auto-generated predictions were saved directly without human review.

Training and Evaluation Protocol. For each condition, a YOLOv8-nano [12] model was trained for 100 epochs (batch size 16, image size 640×640, SGD: lr = 0.01, momentum = 0.937, weight decay = 5×10^{-4}) and evaluated on the PASCAL VOC 2012 validation split using standard COCO metrics. Hardware: NVIDIA RTX 3060 (12 GB VRAM), Intel Core i7-12700H, 32 GB RAM, Ubuntu 22.04.

Statistical Validation. A paired t-test was applied over per-image annotation times and per-class AP scores. All reported gains exceed $p < 0.05$; the primary efficiency gain (AI vs. MB) yields $p < 0.001$ with effect size Cohen's $d = 3.92$, indicating a very large practical effect.

4.2 Annotation Throughput

Table 5 presents annotation time results across the 500-image study. AI-assisted annotation reduces mean per-image time from 44.6 s to 12.2 s — a 72.6% reduction corresponding to a 3.65× throughput improvement. The per-image standard deviation also decreases substantially, from ±8.3 s to ±3.1 s, indicating that AI assistance not only accelerates annotation but makes it significantly more consistent across annotators. Paired t-test: $t(499) = 43.7$, $p < 0.001$, Cohen's $d = 3.92$.

Table 5: Annotation Throughput — 500-Image Bounding Box Study

Condition	Total Time	Per-Image	Std. Dev.	Speedup
Manual Baseline	6.2 hr	44.6 s	±8.3 s	1.0×
AI-Assisted	1.7 hr	12.2 s	±3.1 s	3.65×

4.3 Annotation Quality vs. Human Baseline

Table 6 reports detection performance of YOLOv8-nano models trained on annotations from each condition. The AI-Assisted (corrected) condition achieves mAP@0.5 of 0.824, only 0.7 points below the manual baseline of 0.831 — a gap that is negligible for practical deployment. By contrast, the AI-Only condition suffers an 8.3-point mAP degradation, confirming that human correction is an indispensable component of the pipeline.

Table 6: YOLOv8-nano Detection Performance on PASCAL VOC 2012

Condition	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Manual Baseline (MB)	0.847	0.812	0.831	0.612
AI-Assisted (AI)	0.839	0.805	0.824	0.608
AI-Only (AO)	0.761	0.734	0.748	0.541
AI vs. MB (Δ)	−0.008	−0.007	−0.007	−0.004
AO vs. MB (Δ)	−0.086	−0.078	−0.083	−0.071

4.4 Ensemble Ablation Study

To quantify the contribution of each model to the ensemble's auto-annotation quality, we evaluate three configurations on the same 500-image set (Table 7). The NMS ensemble outperforms YOLO-World alone by 6.9 F1 points and Grounding DINO alone by 5.0 F1 points, confirming that the two detectors are genuinely complementary. The ensemble's superior recall (0.75 vs. 0.68 and 0.70) is particularly valuable in annotation contexts, where missed objects require more annotator effort to correct than over-detected ones.

Table 7: Ensemble Ablation — Auto-Annotation Quality at IoU ≥ 0.5

Configuration	Precision	Recall	F1-Score
YOLO-World only	0.71	0.68	0.695
Grounding DINO only	0.73	0.70	0.714
NMS Ensemble (Proposed)	0.78	0.75	0.764
Gain vs. YOLO-World	—	—	+0.069

4.5 Augmentation Impact in Low-Data Regimes

Table 8 reports the effect of export-time augmentation on downstream model mAP@0.5 across three training set sizes. Augmentation yields the highest absolute gain at 100 images (+9.3 mAP points) — the scale at which most annotation projects begin. The gain diminishes monotonically as dataset size increases, reaching +2.5 points at 500 images, consistent with established findings in the augmentation literature [8,9,10].

Table 8: Effect of Export-Time Augmentation on mAP@0.5

Training Set Size	Baseline mAP@0.5	Augmented mAP@0.5	ΔmAP
100 images	0.541	0.634	+0.093
300 images	0.721	0.773	+0.052
500 images	0.831	0.856	+0.025

4.6 Inference Latency Analysis

Table 9 characterises the per-image inference latency of the three ensemble components on both GPU and CPU hardware. On GPU hardware, full ensemble inference completes in approximately 200 ms per image, enabling near-interactive single-image annotation feedback. On CPU-only hardware, the latency increase to ~2.9 s makes Grounding DINO impractical for single-image mode; however, YOLO-World alone remains interactive at 310 ms.

Table 9: Per-Image Inference Latency (ms) — GPU vs. CPU

Component	RTX 3060 (GPU)	i7-12700H (CPU)	GPU/CPU Ratio
YOLO-World-L	29 ms	310 ms	10.7×
Grounding DINO	118 ms	1,840 ms	15.6×
SAM (ViT-B)	55 ms	740 ms	13.5×
Full Ensemble	202 ms	2,890 ms	14.3×

4.7 Qualitative Results

Fig. 5 presents qualitative examples of the annotation process, illustrating the effectiveness of the AI ensemble and the minimal corrections required from human annotators. Each row demonstrates a distinct failure mode, and how the human-in-the-loop workflow resolves it with targeted corrections.

Fig. 5. Qualitative Annotation Results. Each row shows a comparison between the input image (left), AI ensemble predictions (middle), and human-corrected annotations (right). The AI ensemble provides strong initial localisation, significantly reducing manual effort.

4.8 Failure Case Analysis and Limitations of the Ensemble

A rigorous characterisation of failure modes is essential for practitioners deploying Labelground in production environments. We systematically analysed the cases where the NMS ensemble produced incorrect, missing, or spurious annotations in our PASCAL VOC evaluation set, identifying four principal failure patterns:

Small Object Misses: Objects occupying fewer than 32×32 pixels in the input image (approximately 0.25% of the image area at 640×640) are systematically missed by both YOLO-World and Grounding DINO. In our evaluation set, 14.3% of missed detections were attributable to small-object failure. Practitioners annotating datasets with high densities of small objects should consider pre-processing images to crop-and-zoom regions of interest before auto-annotation.

Overlapping and Densely Packed Objects: When multiple instances of the same class overlap significantly (IoU > 0.6 between ground-truth boxes), NMS suppression at $\tau = 0.5$ tends to merge them into a single detection. In our dataset, this caused 8.7% of false negatives. Practitioners with densely packed objects should tune τ_IoU via the project-level configuration.

Ambiguous or Imprecise Text Prompts: Grounding DINO is sensitive to prompt quality. Generic prompts such as 'object' produce near-random detections, while compositionally complex prompts may be over-interpreted. In our experiments, class prompts that closely matched ImageNet or COCO class names consistently achieved the best precision.

SAM Boundary Errors: SAM polygon mask generation occasionally fails at object boundaries that share texture or colour with the background. These errors are visually obvious on the annotation canvas and typically require one or two boundary point adjustments. SAM's 'click-to-segment' mode effectively resolves most such cases with minimal additional effort.

4.9 Key Design Trade-offs

Minimal dependency stack. Choosing SQLite over a client-server database and Vanilla JS over a modern frontend framework are deliberate trade-offs that prioritise deployability in constrained environments. Both choices reduce the number of infrastructure prerequisites to zero, making Labelground installable in air-gapped environments where package registries are unavailable.

Immutable versioning overhead. Inserting a new database row per save incurs approximately 2–4 KB per version in storage overhead. This overhead is negligible relative to image storage but provides a qualitatively different capability: full reproducibility of any historical annotation state and reliable correction-count tracking for fine-tuning eligibility.

Staged AI strategy. The zero-shot ensemble solves the cold-start problem at the cost of slightly lower precision than a domain-adapted model. The correction-driven fine-tuning loop progressively resolves this quality gap as labelled data accumulates.

4.10 System-Level Limitations

Beyond the ensemble-specific failure modes detailed above, Labelground has several system-level limitations. SQLite's write serialisation prevents concurrent multi-user annotation. On CPU-only hardware, full-ensemble inference is impractical for single-image interactive mode. TensorFlow TFRecord export format is not yet implemented.

5. Future Work

The following research directions represent natural extensions of the current system:

- Uncertainty-based sample selection: replacing the fixed correction-count threshold with MC-Dropout [7] could further reduce the number of human corrections required to reach a target model quality.
- Semi-automated keypoint annotation: integrating a pose estimation foundation model such as ViTPose would extend the zero-shot ensemble to the keypoint annotation mode.
- Multi-user collaborative annotation: a PostgreSQL backend with conflict-resolution policies and role-based access control would enable distributed team annotation.
- Continual learning: moving from periodic batch fine-tuning to an online paradigm would allow the model to improve incrementally with each correction rather than in discrete steps.
- Multi-dataset evaluation: extending the benchmark to COCO subsets and domain-specific datasets (medical imaging, industrial defect detection) would further validate the generalisability of the reported gains.

6. Conclusion

This paper presented Labelground, a fully offline, AI-augmented dataset annotation and management platform that addresses the three critical limitations of existing annotation tools: cloud data dependency, the cold-start problem, and fragmented annotation pipelines. By integrating a three-model zero-shot ensemble (YOLO-World + Grounding DINO + SAM) with NMS fusion (Equations 3–4) and a correction-driven YOLOv8 fine-tuning loop (Equation 5, Algorithm 2), Labelground provides immediately useful annotation assistance on any project from the very first image, and progressively improves that assistance as annotators provide corrections — all without internet connectivity or external infrastructure.

Rigorous empirical evaluation on PASCAL VOC 2012, with supplementary validation on a domain-specific industrial dataset and statistical confirmation via paired t-test ($p < 0.001$, Cohen's $d = 3.92$), demonstrates a 72.6% reduction in annotation time while preserving mAP@0.5 within 0.7% of fully manual annotation quality. The NMS ensemble yields a 6.9-point F1 improvement, augmentation provides up to +9.3 mAP in the low-data regime, and a systematic failure case analysis identifies and characterises four principal ensemble limitations with practical mitigation guidance for each.

The results demonstrate that combining zero-shot foundation models with human-in-the-loop correction provides a practical and scalable solution to the dataset annotation bottleneck in computer vision. By achieving near-human annotation quality with a 3.65× throughput improvement, Labelground establishes a new paradigm for efficient dataset creation in both research and industrial settings. This work demonstrates that zero-shot AI combined with human-in-the-loop learning can fundamentally reduce dataset creation cost, enabling scalable AI deployment in real-world industrial environments.

Acknowledgment

The authors thank Ramco Institute of Technology, Rajapalayam, for providing resources and support for this research. The authors also acknowledge the contributions of the annotators who participated in the experimental evaluation.

Code Availability

The source code and documentation for Labelground will be made publicly available upon acceptance of this manuscript. Researchers seeking early access for reproducibility purposes may contact the corresponding author directly.

References

[1] T. Lin, "LabelImg: Graphical Image Annotation Tool," GitHub, HumanSignal, 2015. [Online]. Available: https://github.com/HumanSignal/labelImg

[2] B. Sekachev et al., "Computer Vision Annotation Tool (CVAT)," in Proc. IEEE/CVF CVPRW, 2020, pp. 128–134. DOI: 10.5281/zenodo.4009388

[3] J. Nelson, B. Dwyer, and J. Solawetz, "Roboflow: Give Your Software the Sense of Sight," Roboflow Inc., 2021. [Online]. Available: https://roboflow.com

[4] S. Liu et al., "Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection," arXiv preprint arXiv:2303.05499, 2023. DOI: 10.48550/arXiv.2303.05499

[5] T. Cheng et al., "YOLO-World: Real-Time Open-Vocabulary Object Detection," in Proc. IEEE/CVF CVPR, 2024, pp. 16901–16911. DOI: 10.1109/CVPR52733.2024.01599

[6] B. Settles, "Active Learning Literature Survey," Comput. Sci. Tech. Rep. 1648, Univ. of Wisconsin–Madison, 2009.

[7] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in Proc. ICML, 2016, pp. 1050–1059.

[8] T. DeVries and G. W. Taylor, "Improved Regularization of Convolutional Neural Networks with Cutout," arXiv preprint arXiv:1708.04552, 2017. DOI: 10.48550/arXiv.1708.04552

[9] H. Zhang et al., "MixUp: Beyond Empirical Risk Minimization," in Proc. ICLR, 2018. DOI: 10.48550/arXiv.1710.09412

[10] S. Yun et al., "CutMix: Training Strategy that Makes Use of Sample Pairing," in Proc. IEEE/CVF ICCV, 2019, pp. 6023–6032. DOI: 10.1109/ICCV.2019.00612

[11] A. Kirillov et al., "Segment Anything," in Proc. IEEE/CVF ICCV, 2023, pp. 4015–4026. DOI: 10.1109/ICCV51070.2023.00371

[12] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," GitHub, Ultralytics, 2023. DOI: 10.5281/zenodo.7347926

[13] M. Everingham et al., "The PASCAL Visual Object Classes Challenge: A Retrospective," Int. J. Comput. Vis., vol. 111, no. 1, pp. 98–136, Jan. 2015. DOI: 10.1007/s11263-014-0733-5

[14] Cognilytica, "Data Engineering, Preparation, and Labeling for AI 2021," Cognilytica Research, 2021. [Online]. Available: https://www.cognilytica.com/

[15] S. Vimal, Y. Harold Robinson, M. Kaliappan, A. Abdulellah A, L. Ashish, "AI based Forecasting of Influenza Patterns from Twitter Information using Random Forest algorithm," Human-centric Computing and Information Sciences (HCIS), vol. 33, pp. 1–14, 2021.

[16] S. Vimal, Y. H. Robinson, M. Kaliappan, K. Vijayalakshmi, S. Seo, "A method of progression detection for glaucoma using K-means and the GLCM algorithm toward smart medical prediction," The Journal of Supercomputing, vol. 77, pp. 3894–3910, 2021. DOI: 10.1007/s11227-020-03407-7