Gradient-Free Few-Shot Cross-Domain Transfer for Visual Regression with Evolutionary Strategies

Siri Sri Churakanti
Dept. Mathematics and Computer Science
Lawrence Technological University Southfield, USA
schurakan@ltu.edu

Chan Jin Chung
Dept. Mathematics and Computer Science Lawrence Technological University Southfield, USA
cchung@ltu.edu

***Abstract*—Accurate visual estimation of liquid volume is important in industrial settings such as chemical processing, beverage filling, and pharmaceutical dispensing. Traditional deep learning approaches hundreds of labeled datasets (800+ images) for each liquid type, and rely on gradient-based fine-tuning, making them difficult to deploy when conditions or liquid appearance change. In this project, we first trained single-domain volume regression models using CNN backbones (ResNet50 and EfficientNetB0) and evaluated their within-domain performance. We then quantified the domain gap using a zero-adaptation transfer baseline (e.g., applying an orange-trained model directly to red images), which produced large cross-domain error (214.59 mL MAE for OR→RED).To address this, we explored gradient-free few-shot weight adaptation using population Evolution Strategies ES(μ + λ) with the 1/5 success rule, optimizing on a 65-image mixed dataset (50 target + 15 source for retention) without backpropagation.ES(10+80) reduced MAE from 214.59 mL to 38.73 mL (81.95% reduction) and can run within about one hour. These results suggest that gradient-free few-shot adaptation is a practical option for resource-constrained deployment settings where backpropagation-based fine-tuning is costly or infeasible.**

Keywords—Evolutionary Strategy, Cross-Domain Transfer, Visual Regression, Industrial Automation, Volume Measurement, Gradient-Free Optimization, Weight Optimization, Transfer Learning, Few-Shot Learning.

I. Introduction

Estimating the volume of a liquid from an image of a measuring vessel is an essential task in industrial automation and quality assurance. Applications include bottling lines, chemical dispensing and robotic pouring, where knowing the volume without contact improves efficiency and hygiene. Deep CNNs have enabled accurate visual regression on such tasks, but they are often trained on a fixed liquid appearance and require large labelled datasets. When the liquid's colour or transparency changes—e.g. from orange juice to red juice—models generalize poorly, and re-training with hundreds of new samples and gradient-based fine-tuning is expensive and slow.

A. Industrial Need for Adaptive Volume Measurement

Automated liquid volume measurement offers advantages over mechanical or ultrasonic sensors because it provides flexible, camera-based, non-contact estimation. In real production environments, liquids vary widely in appearance across product lines and even across batches (color, foam, opacity, lighting reflections, and meniscus shape). A bottling plant producing dozens of beverage varieties would need either (i) a separate model per liquid domain or (ii) a reliable adaptation method. Training a dedicated model for each variant can require hundreds of labeled images per liquid, significant annotation effort, and repeated gradient-based training runs, which is not ideal when new products must be introduced quickly. This motivates methods that can adapt a pre-trained model to a new liquid domain using minimal labeled data and limited compute.

B. Key Terminology and Concepts

This work uses standard *transfer learning* terminology. Transfer learning uses knowledge from a source domain to improve learning in a related target domain, typically by starting from pre-trained weights rather than random initialization. *Cross-domain transfer* refers to adapting a model trained on one visual domain (e.g., orange liquid in a measuring cup) to perform well on a different domain (e.g., red liquid) while the underlying task remains the same. In our setting, both domains share the same vessel geometry and the same volume labels (100–610 mL across 22 classes), but differ in appearance and image statistics. The main challenge is the domain gap between source and target feature distributions.

A practical difficulty in adaptation is *catastrophic forgetting*: gradient-based fine-tuning updates weights to reduce target loss, but these updates can overwrite weight patterns that were important for the source domain. As a result, performance on the original domain can degrade even while target performance improves. This stability–plasticity tradeoff is central to real deployment: adaptation must improve target accuracy without destroying useful prior knowledge.

Why Evolutionary Algorithms Instead of Backpropagation?

In this project we focus on gradient-free adaptation using *Evolutionary Strategies* (ES). ES optimizes model parameters by evaluating candidate solutions (forward passes only) instead of computing gradients. This is attractive for industrial deployment for three reasons.

- First, in few-shot settings (e.g., 50–65images), gradient estimates can be noisy and can overfit quickly, whereas ES can search for weights that improve end-to-end performance.
- Second, gradient-free optimization reduces implementation complexity and memory requirements because it avoids backpropagation and gradient storage, making it more suitable for edge devices.
- Third, ES can naturally incorporate mixed-domain evaluation by defining fitness on a small dataset that includes both target samples (for adaptation) and a small set of source samples (to reduce forgetting).

We use population-based ES, written *as ES(μ+λ), together with the 1/5 success rule (Rechenberg*) is a heuristic for adapting the mutation step size σ in evolutionary strategies:

- If more than 20% of recent mutations lead to improvement → increase σ (explore more broadly)
- If less than 20% of recent mutations lead to improvement → decrease σ (exploit current region). This 20% success rate represents the optimal balance between exploration (searching new areas) and exploitation (refining promising solutions)
- This provides a simple, effective mechanism to balance exploration and refinement without gradients.

In *Hyperparameter Optimization (HPO*), Hyperparameters are settings that control the learning process but are not learned from data:

- Architecture: Number of neurons, layers, dropout rates
- Training: Learning rate, batch size, optimizer choice.
- Transfer learning: Freeze ratio (proportional layers to freeze)

HPO searches for hyperparameter combinations that maximize model performance

C. *Problem Statement*

Given a source domain model trained on liquid type with full dataset, and a target domain with limited samples, our goal is to adapt the model to predict liquid volume accurately in the target domain while maintaining reasonable performance on the source domain.
(i) No gradient computation during adaptation. (ii) Minimal target domain data requirement (<100 images). (iii) Adaptation time: minutes to ~1 hour depending on ES configuration and hardware.

D. *Contribution:*

This work makes two contributions.

❖ First, we train strong single-domain visual regression models using common CNN backbones (ResNet50 and EfficientNetB0), establishing reliable within-domain baselines.

❖ Second, we propose few-shot cross-domain weight adaptation using population ES(μ+λ) on a small mixed dataset (50 target images + 15 source images), without backpropagation. On a controlled orange-to-red domain shift, the zero-adaptation transfer baseline produces very high error (214.59 mL MAE), demonstrating a large domain gap. After ES-based few-shot weight optimization, cross-domain error is reduced substantially: ES(10+80) improves OR→RED MAE from 214.59 mL to 38.73 mL (81.95% reduction), and improves RED→OR MAE from 17.53 mL to 11.84 mL (32.47% reduction).

E. *Paper Organization:*

Section II reviews related work. Section III presents our methodology. Section IV describes the experimental setup. Section V presents results. Section VI discusses industrial applications. Section VII concludes the paper.

II. Related Work

A. *Transfer Learning in Computer Vision*

Transfer learning has become foundational in computer vision. Yosinski et al. [1] demonstrated that features learned in early CNN layers (edges, textures) transfer well across tasks, while later layers become task-specific. The standard pipeline involves loading ImageNet-pretrained weights, freezing early layers, and fine-tuning later layers on target data. In our setting, liquids share the same regression task and container geometry but differ in appearance (color/opacity/reflectivity), producing a domain gap that is difficult to address with limited target labels.

B. *Few-Shot Learning*

Meta-learning approaches address few-shot classification: Model-Agnostic Meta-Learning (MAML) [2] optimizes for initialization weights that enable fast adaptation; Prototypical Networks [3] learn metric spaces for distance-based classification. These approaches are mainly designed for classification and still depend on gradient-based adaptation, whereas our goal is few-shot regression with gradient-free adaptation.

C. *Evolutionary Strategies for Deep Learning*

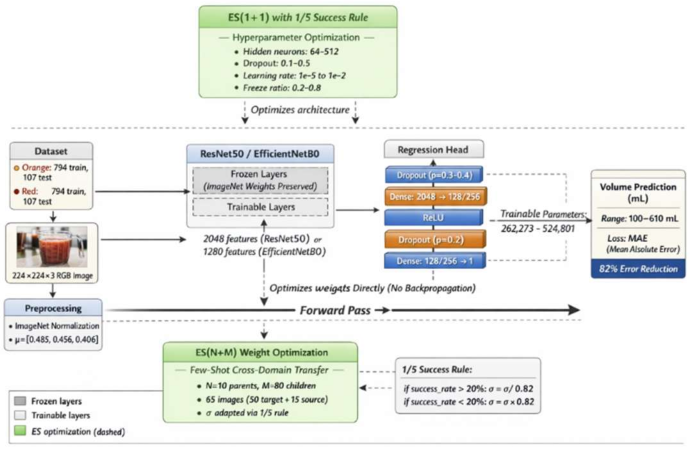
OpenAI demonstrated that ES can compete with reinforcement learning algorithms [4], being simpler and highly parallelizable. EvoLearn [5] applied evolutionary weight optimization to forecasting models. In our prior work, we applied ES(1+1) with the 1/5 success rule for hyperparameter optimization and achieved strong performance on a controlled classification benchmark, motivating ES as a practical search method in our workflow [8].

D. *The 1/5 Success Rule*

The 1/5 success rule (Rechenberg) [6] adapts mutation step size based on the recent improvement rate; modern analysis is provided by Biedrzycki [7]. During domain adaptation, catastrophic forgetting is a known challenge [9]. Self-adaptation and covariance-based ES ideas (e.g., CMA-style methods) have been studied by Hansen and Ostermeier [10] and motivate future extensions beyond isotropic mutations. The adaptation rule uses factor $c_d \approx 0.82$:

```
if success_rate > 0.2:
    $\sigma\_new = \sigma / c$     # Increase step size (~1.22×)
else:
    $\sigma\_new = \sigma \times c$     # Decrease step size (0.82×)
```

III. Neural Network Architecture



ES(1+1) with 1/5 Success Rule
— Hyperparameter Optimization —
• Hidden neurons: 64-512
• Dropout: 0.1-0.5
• Learning rate: 1e-5 to 1e-2
• Freeze ratio: 0.2-0.8
Optimizes architecture
Dataset
Orange: 794 train, 107 test
Red: 794 train, 107 test
224×224×3 RGB Image
Preprocessing
• ImageNet Normalization
• μ=[0.485, 0.456, 0.406]
ResNet50 / EfficientNetB0
Frozen Layers (ImageNet Weights Preserved)
Trainable Layers
2048 features (ResNet50) or 1280 features (EfficientNetB0)
Regression Head
Dropout (p=0.3-0.4)
Dense: 2048 → 128/256
ReLU
Dropout (p=0.2)
Dense: 128/256 → 1
Trainable Parameters: 262,273 - 524,801
Volume Prediction (mL)
Range: 100–610 mL
Loss: MAE (Mean Absolute Error)
82% Error Reduction
Optimizes weigqts Directly (No Backpropagation)
Forward Pass →
ES(N+M) Weight Optimization
— Few-Shot Cross-Domain Transfer —
• N=10 parents, M=80 children
• 65 images (50 target + 15 source)
• σ adapted via 1/5 rule
1/5 Success Rule:
if success_rate > 20%: σ = σ/ 0.82
if success_rate < 20%: σ = σ × 0.82
Frozen layers
Trainable layers
ES optimization (dashed)

IV. Data Collection and Organization

A. *Equipment and Capture Setup*

Images were collected using a transparent plastic **measuring cup** with a maximum capacity of 600 mL and printed volume markings. A smartphone camera was used for all captures to keep the imaging device consistent across datasets. To create a controlled domain shift while keeping the underlying task identical, two liquid domains were recorded: OR (orange liquid) and RED (red liquid). Lighting and camera placement were kept as consistent as possible across sessions (Fig. 1).

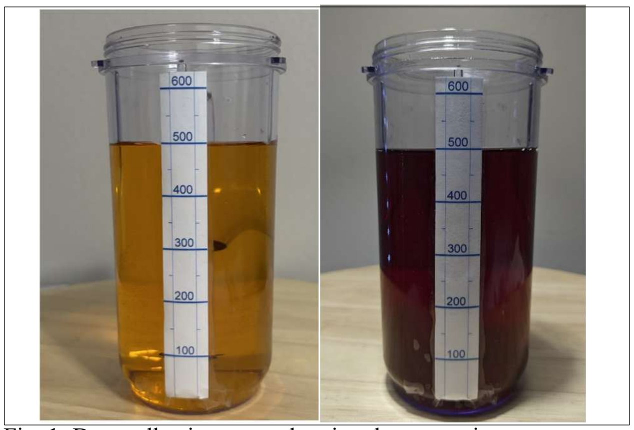


Fig. 1: Data collection setup showing the measuring cup, camera position, and controlled lighting environment used for capturing both OR (orange juice) and RED (red juice) datasets.

B. *Data Collection Protocol:*

For each labeled volume level, the measuring cup was filled to the target marking and the ground-truth volume (mL) was recorded. Multiple images were captured per volume under two viewpoints to increase viewpoint variation while keeping the setup controlled:

- **Front View (FV):** camera approximately at liquid level
- **Bottom View (BV):** camera positioned lower to view the meniscus and liquid surface from below

This procedure was repeated for all volume classes.

C. *Dataset Description:*

Two datasets were created with identical label structure (same volume classes) but different visual appearance:

- **OR Dataset (Orange Juice)**
  - Liquid : Orange juice (food colour/consistent brand)
  - Training images: 794
  - Testing images: 107
  - Folder structure: Flat hierarchy
  - Visual properties: Semi-transparent, orange color spectrum (590-620nm)
- **RED Dataset (Red Juice)**
  - Liquid: Red-colored juice (cranberry/fruit punch)
  - Training images: 794
  - Testing images: 107
  - Folder structure: Nested hierarchy with view subfolders
  - Visual properties: More opaque, red color spectrum (620-750nm)

D. *Labels and Class Structure:*

Both datasets(Table1) use **22 discrete volume labels** spanning 100–610 mL: 100, 105, 150, 155, 200, 205, 250, 260, 300, 310, 350, 355, 400, 410, 450, 470, 500, 505, 550, 560, 600, 610.

TABLE I: Dataset Statistics

Data Base	Domain	Training	Testing	Total_Classes
OR	Orange juice	794	107	22
RED	Red juice	794	107	22

E. *Image Preprocessing*

All images processed using the same pipeline:

- Resize: Scale to 224×224 pixels
- Normalize: Apply ImageNet statistics

$$x' = \frac{x - \mu}{\sigma}$$

Where μ = 0.485, 0.456, 0.406] and σ = [0.229, 0.224, 0.225]

V. METHODOLOGY

A. *Problem Formulation and Evaluation Metric*

Given an input image xof a measuring cup, the model predicts liquid volume ŷin milliliters. We treat this as a visual regression problem and evaluate performance using Mean Absolute Error (MAE)

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

B. *Base Models and Transfer Learning Setup*

We train single-domain volume regression models using ImageNet-pretrained CNN backbones:

- **ResNet50** (pretrained on ImageNet)

ResNet50 regression head : The original fc layer is replaced with an MLP head:

Dropout(p_1)→Linear($d \to h$) →ReLU→ Dropout(p_2) →Linear($h \to 1$) where $d$is the ResNet feature dimension and h("hidden_units") is a tunable hyperparameter.

- **EfficientNetB0** (pretrained on ImageNet)

EfficientNetB0 regression head: The classifier is replaced with an MLP head constructed from configurable parameters (number of hidden layers, hidden width, dropout, activation), ending in a final Linear layer that outputs one scalar volume prediction.

For each backbone, we replace the original classification layer with a regression head and train on labeled cup images. During standard supervised training (baseline training and ES-HPO evaluations), we apply a freeze ratio that freezes an initial portion of backbone parameters and trains the remaining parameters plus the regression head.

C. *Application 1: ES(1+1) Hyperparameter Optimization for Single-Domain Training (ES-HPO)*

To improve single-domain model performance without manual tuning, we use Evolution Strategy ES(1+1**)** to search training/model hyperparameters. ES(1+1) maintains one "parent" configuration and proposes one "child" configuration per generation via mutation. If the child improves validation MAE, it replaces the parent. Search Space:

TABLE II: Hyperparameter Search Space for ES(1+1) HPO:

Parameters	**Min**	**Max**	**Mutation**
Hidden neurons	32	512	Gaussian(σ = 30)
Dropout rate	0.1	0.5	Gaussian(σ = 0.05)
Learning rate	10^{-5}	10^{-2}	Log-scale Gaussian
Freeze ratio	0.0	0.7	Gaussian(σ = 0.05)
Batch size	{8, 16,	24, 32}	20/% random switch

ES-HPO configuration (from your HPO code):

- Max generations: 30
- Success window for the 1/5 rule: 10
- Each evaluation trains for 20 epochs with early stopping patience 6
- Step adaptation factor: $c = 0.82$

Fitness for ES-HPO: validation MAE after training with the candidate hyperparameters (lower is better).

This HPO stage is performed once per domain (OR and RED) to establish strong within-domain models before transfer experiments.

Algorithm 1 ES(1+1) Hyperparameter Optimization

```
Require: Training data Dtrain, validation data Dval
Require: Search space S, σinit, max gen, cd = 0.82, w = 10
  Ensure: Optimized hyperparameters θ*
  1: θ_parent ← BaselineInit(S)
  2: f_parent ← TrainAndEvaluate(θ_parent, D_train, D_val)
  3: σ ← σ_init; successes ← []
  4: for gen = 1 to max_gen do
  5:    θ_child ← Mutate(θ_parent, σ, S)
  6:    f_child ← TrainAndEvaluate(θ_child, D_train, D_val)
  7:    if f_child < f_parent then
  8:       θ_parent ← θ_child; f_parent ← f_child
  9:       successes.append(1)
  10:    else
  11:       successes.append(0)
  12:    end if
  13:    // Apply 1/5 Success Rule
  14:    if gen mod window = 0 then
  15:       p_s ← mean(successes[-window:])
  16:       if p_s > 0.2 then
  17:          σ ← σ / c     // Increase (×1.22)
  18:       else
  19:          σ ← σ × c     // Decrease (×0.82)
  20:       end if
  21:    end if
  22: end for
23: return θ_parent
```

D. *Application 2 — Few-Shot Cross-Domain Weight Adaptation Using Population ES(μ+λ)*

After training a source-domain model (e.g., OR), we measure cross-domain degradation by evaluating it directly on the target domain (e.g., RED). We then adapt the model using population-based Evolution Strategies ES(μ+λ) with a small labeled dataset.

1) Few-shot mixed dataset (fitness set)

For adaptation, we build a 65-image mixed dataset:

- 50 target-domain images (few-shot adaptation signal)
- 15 source-domain images(retention signal to reduce forgetting)

This mixed set is used to compute ES fitness and encourage improvements on the target without completely destroying source performance.

2) ES(μ+λ) weight optimization loop

We treat the entire model parameter vector as the optimization variable. Each generation:

1. Initialize μ parents from the same pre-trained weights (copied μ times).
2. Mutation: generate λ children by selecting a random parent and adding Gaussian noise to its parameters:
$$Wchild = Wparent + \epsilon, \epsilon \sim N(0, \sigma^2)$$
3. Evaluation (fitness): compute MAE on the 65-image mixed dataset (forward-pass only).
4. Selection ("plus" strategy): combine parents and children (μ+λ), then keep the best μ individuals (lowest MAE) as next-generation parents.
5. Track whether the generation improved the best fitness and update step size using the 1/5 rule.

E. 1/5 Success Rule for Step-Size Adaptation

To control mutation strength during search, we use the 1/5 success rule. Let p_s be the success rate over a recent window of generations (success = best MAE improved).

$$\mathrm{p_s} = \left(\frac{1}{\mathrm{w}}\right) * \mathrm{sum(improved_i)}$$

where w is the window size and improved_i indicates whether generation i achieved improvement. The step-size update rule is

$$\sigma_{t+1} = \begin{cases} \frac{\sigma_t}{0.82}, & if\ p_s > 0.2 \\ \sigma_t \cdot 0.82, & if\ p_s < 0.2 \\ \sigma_t, & otherwise \end{cases}$$

This mechanism reduces manual tuning and helps balance exploration vs. exploitation.

V. Results

A. *Evaluation Metrics*

We report Mean Absolute Error (MAE) in milliliters as a primary metrics

$$MAE = np.mean\,(np.abs(act - pred))$$

And prediction of Average Accuracy as a secondary relative-error score, as defined in lower MAE and higher Avg Acc indicate better performance.

$$Avg\ Accuracy = np.mean(100 * (1 - np.abs(act - pred)\,/\,act))$$

B. Single-Domain HPO Results

Table III summarizes within-domain performance for OR and RED using ResNet50 and EfficientNetB0. ES(1+1) is used here only for hyperparameter optimization (ES-HPO) to reduce manual tuning. EfficientNetB0 with ES-HPO (BI) achieved the best single-domain results, and these trained models serve as starting checkpoints for cross-domain evaluation.

TABLE III: ES(1+1) Hyperparameter Optimization Results

Experiments	OR			RED		
	MAE	Acc%	Time	MAE	Acc%	Time
RN50 (Base)	9.64	96%	1hr	10.88	94.68%	1hr
RN50 + ES (BI)	5.52	98.23%	6.5 hrs	17.2	93.48%	5hr
RN50 + ES (RI)	6.71	97.52%	0.5hr	16.44	91.63%	0.5hr
ENBO(Base)	3.56	98.84%	1h	10.36	94.39%	0.5hr
ENB0 + ES (BI)	3.47	98.97%	0.5hr	9.53	95.13%	1hr
ENB0 + ES (RI)	10.53	96.74%	0.5hr	15.99	93.11%	0.5hr

Note: RN50 = ResNet50, ENB0 = EfficientNetB0; ES refers to ES(1+1) used for hyperparameter optimization, BI is Baseline Init, RI is Random Init

C. Cross-Domain Baseline (No Transfer)

Before adaptation, we evaluate each source-trained model directly on the other domain's test set (**no transfer / no fine-tuning**). Table IV shows a large domain gap in OR→RED, where the OR-trained model's MAE increases from 9.55 mL (OR test) to 214.59 mL (RED test).

TABLE IV: Cross-Domain Performance Without Adaptation (No Transfer Baseline)

Configuration	**MAE**	**Accuracy**
OR model on OR test	9.55	97.2%
OR model on Red test	214.59	47.6%
Red model on Red test	9.69	95.1%
Red model on OR test	17.53	91.3%

D. *Few-Shot Cross-Domain Weight Adaptation (Population ES(μ+λ))*

Here Few-shot adaptation using population Evolution Strategies ES(μ+λ) with the 1/5 success rule, optimizing on a 65-image mixed fitness set (50 target + 15 source for retention).

TABLE V: ES(1+1) Cross-Domain Transfer Results Using Population ES(μ+λ) on Mixed Set

Transfer	Before (MAE)	After (MAE)	improv	Time
OR to RED	214.59 ml	38.7 ml	81.9 %	1 hr
RED to OR	17.53 ml	11.8 ml	32.4%	1hr

Note: "Before" denotes the baseline cross-domain performance when applying the source-trained model directly to the target domain with no adaptation. "After" denotes performance post ES(N+M) few-shot weight optimization. Accuracy refers to 100 × (1 − |y_true − y_pred|/y_true). We report Δ Accuracy in percentage points (pp) relative to the "Before" accuracy.

E. Volume prediction Results

1. Volume prediction on Orange Juice results:

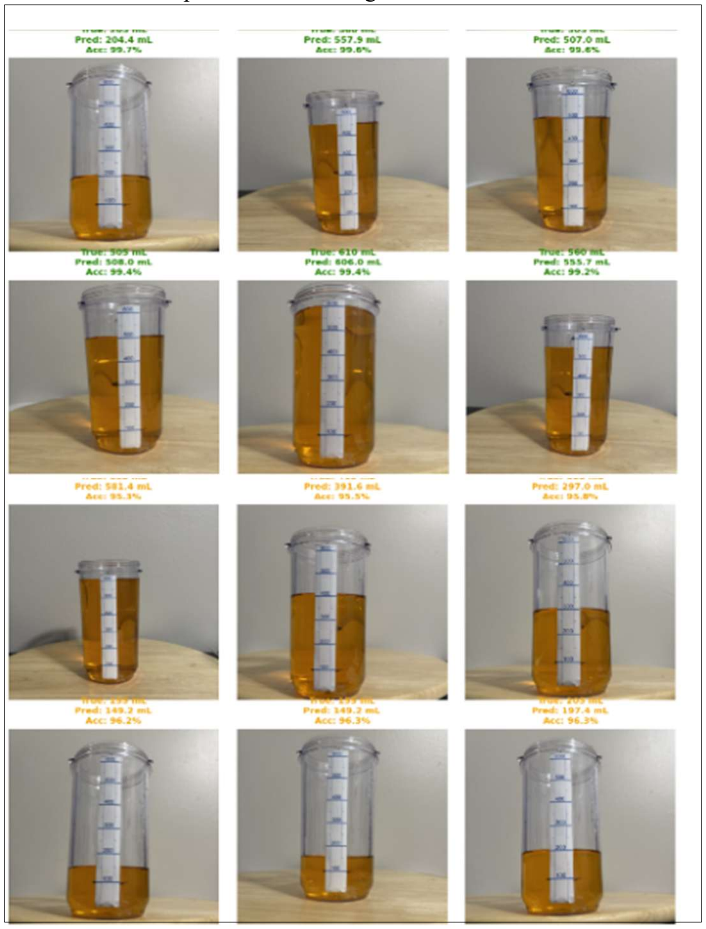


Fig. 2. OR test predictions showing ground truth (True), predicted volume (Pred), and Acc.

2. Volume prediction on RED Juice results:

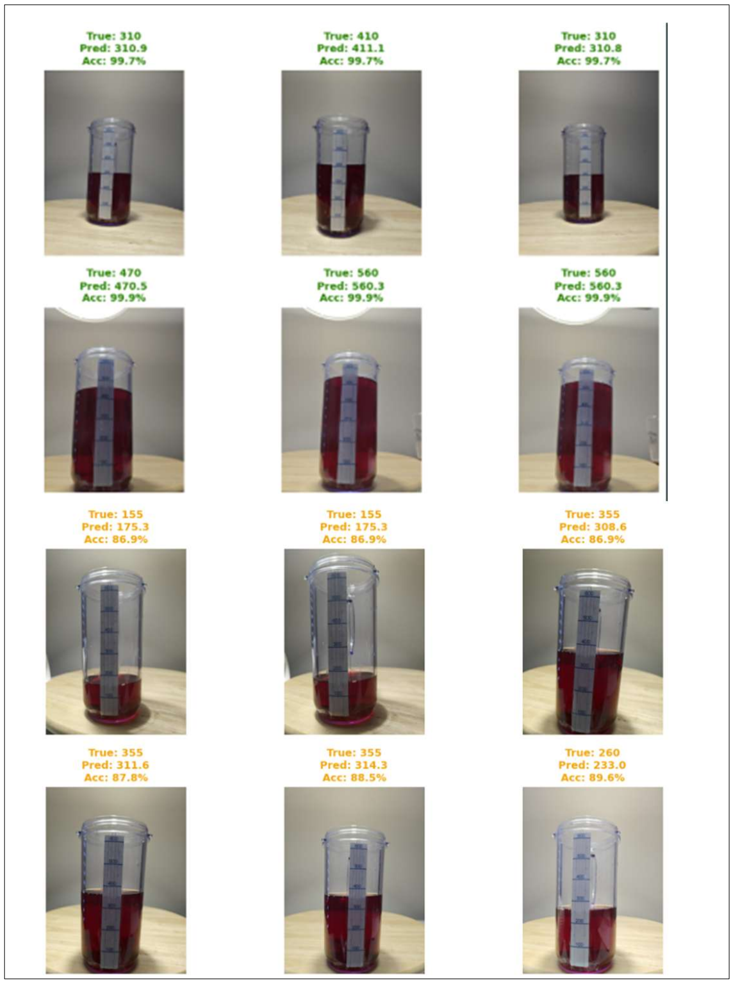


Fig. 3. Example RED test predictions after ES(μ+λ) adaptation. Predictions track the correct volume range more closely than the no-adaptation baseline, consistent with the MAE reductions in Table V.

VI. Future Work

1) **Population-based ES**: he ES(1+1) algorithm presented represents the simplest evolutionary strategy. Extending to ES(N+M) with multiple parents (N > 1) and children (M > 1) can maintain population diversity and improve exploration of the weight space. Preliminary experiments with ES(4+4) and ES(10+80) suggest potential for greater error reduction. Additionally, Covariance Matrix Adaptation (CMA-ES) could replace isotropic Gaussian mutations with learned covariance structures that better capture weight correlations.
2) **CMA-ES Integration**: Replace isotropic Gaussian mutation with Covariance Matrix Adaptation for correlated mutations that respect loss landscape geometry.
3) **Elastic Weight Consolidation**: Combine Fisher Information-based importance weighting with ES mutation to protect critical weights.
4) **Multi-domain transfer**: Extend to sequential adaptation across multiple domains (e.g., orange → red → green). To investigate continual learning strategies that accumulate knowledge across domains without catastrophic forgetting of previously adapted domains
5) **Edge deployment**: he gradient-free nature of ES makes it suitable for resource constrained environments. Validation on embedded platforms (Raspberry Pi, NVIDIA Jetson) will confirm practical deployment viability.

VII. Conclusion

We investigated gradient-free cross-domain transfer for visual liquid volume regression using Evolutionary Strategies with the 1/5 success rule. Our workflow includes (i) training strong single-domain regression models using ImageNet-pretrained CNN backbones and (ii) adapting a source-domain model to a new liquid domain using few-shot data without backpropagation.

Key findings are:

- **Single-domain performance (training stage):** Using ES(1+1) for hyperparameter optimization (ES-HPO), our best within-domain model achieved **3.47 mL MAE** on the OR dataset (EfficientNetB0, BI), demonstrating that evolutionary search can effectively tune model/training hyperparameters.
- **Cross-domain transfer baseline (no adaptation):** Directly applying a source-trained model to the other domain can fail severely. In particular, **OR→RED** shows a large domain gap, increasing MAE to **214.59 mL** without adaptation.
- **Few-shot weight adaptation (deployment stage):** Population ES(μ+λ) few-shot weight optimization on a **65-image mixed fitness set (50 target + 15 source)** substantially improves cross-domain performance without gradient computation. The best configuration, **ES(10+80)**, reduced **OR→RED** MAE from **214.59 mL to 38.73 mL** (**81.95% reduction**) and reduced **RED→OR** MAE from **17.53 mL to 11.84 mL** (**32.47% reduction**). We also report source retention MAE to quantify catastrophic forgetting.

Overall, these results show that gradient-free ES-based adaptation can provide rapid, data-efficient cross-domain transfer for visual regression, making it a promising approach for industrial settings where labeled data is limited and gradient-based fine-tuning may be impractical.

VIII. References

[1]. J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in NeurIPS, 2014.
[2]. C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," in ICML, 2017.
[3]. J. Snell, K. Swersky, and R. Zemel, "Prototypical Networks for Few-shot Learning," in NeurIPS, 2017.
[4]. T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, "Evolution Strategies as a Scalable Alternative to Reinforcement Learning," arXiv:1703.03864, 2017.
[5]. A. E. Ezugwu et al., "Effective weight optimization strategy for precise deep learning forecasting models using EvoLearn approach," Scientific Reports, vol. 14, 2024.
[6]. I. Rechenberg, "Evolution strategy: Nature's way of optimization," in Optimization: Methods and Applications, Springer, 1989.
[7]. R. Biedrzycki, "Evolution Strategies under the 1/5 Success Rule," Mathematics, vol. 11, 2023.
[8]. S. S. Churakanti, B. Kenzheakhmetov, and C.-J. Chung, "Optimizing Hyperparameters for Deep Learning Models Using Evolutionary Algorithms," in ICMI-EIT, 2024.
[9]. J. Kirkpatrick *et al.*, "Overcoming catastrophic forgetting in neural networks," *PNAS*, vol. 114, 2017.
[10]. N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary Computation*, vol. 9, 200