International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 6, Nov – Dec 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Multi-Population Particle Swarm Optimization for Fuzzy Neural Network Video Quality Prediction: Complete Specification and Leakage-Free Evaluation

Md Nadim Hosain*, Md Daud Ibrahim**, Md Imran Hossain***
*(School of Electronics and Information Engineering, Nanjing University of Information Science & Technology, Nanjing, China
Email: mdnadimhosain@nuist.edu.cn)
** (School of Electronics and Information Engineering, Nanjing University of Information Science & Technology, Nanjing, China
Email: ibrahimmddaud@nuist.edu.cn)
***(School of Electronics and Information Engineering, Nanjing University of Information Science & Technology, Nanjing, China
Email: hossainmdimran@nuist.edu.cn)

----------**********----------

Abstract:

Monitoring the perceived quality of remotely delivered video calls for lightweight no-reference predictors. IMPMSPSO-FNN trains a fuzzy neural network (FNN) with a particle swarm optimizer that combines good-point-set initialization, five subpopulations, five randomly selected update strategies and a shared global best. It was originally reported to yield large gains on a private dataset, but its update rules, particle encoding and fitness function were incompletely specified, and its evaluation used test errors during training. In this study, we specify the method completely and evaluate it with a nested, leakage-free protocol on LIVE-VQA (compression and simulated IP and wireless transmission; 45 content-separated splits), KoNViD-1k and LIVE-VQC (20 splits each). Baselines are the same FNN trained by standard PSO, differential evolution and Adam, as well as support vector regression (SVR) and random forest on the same 60-dimensional features. IMPMSPSO-FNN reached a mean Spearman rank-order correlation (SROCC) of 0.719 on KoNViD-1k and 0.682 on LIVE-VQC; these values were not significantly different from those of the other FNN training methods and were below those of SVR (0.767 and 0.720; significant on KoNViD-1k). Ablations and repeated-seed runs showed no consistent accuracy gain from any optimizer component, although multi-strategy selection prevented occasional stalls. Using partial least squares inputs instead of principal components raised the SROCC by 0.161 on KoNViD-1k. On LIVE-VQA, every feature-based predictor generalized poorly to unseen contents (mean SROCC at most 0.277). For FNN-based video quality prediction, input representation and leakage-free validation matter more than optimizer elaboration.

Keywords — **No-reference video quality assessment, fuzzy neural network, particle swarm optimization, good point set, cooperative co-evolution.**

----------**********----------

I. INTRODUCTION

Remote video services, such as networked surveillance, video conferencing and adaptive streaming, deliver content that has been compressed and transmitted over imperfect channels. Operators of such systems need to know when the quality perceived by viewers falls below an acceptable level, but they rarely have access to the pristine source. Subjective testing according to ITU-T P.910 [1] provides the reference standard for perceived quality, yet it is slow and costly and cannot run continuously. No-reference video quality assessment (NR-VQA) addresses this gap by predicting the mean opinion

 ©IJSRED: All Rights are Reserved

score (MOS) of a video from the distorted signal alone [2], [3].

Two families of NR-VQA models dominate the literature. The first extracts handcrafted natural-scene and temporal statistics and maps them to MOS with a shallow regressor, as in BRISQUE [4], V-BLIINDS [5], TLVQM [6] and the VIDEVAL feature-fusion model [3]. The second learns representations end-to-end with deep networks, as in VSFA [7], SimpleVQA [8], FAST-VQA [9], DOVER [10] and the multimodal Q-Align model [11]. Deep models achieve the highest correlations on public benchmarks, but feature-based predictors remain attractive for embedded monitoring because they are small, fast and easier to inspect.

Fuzzy neural networks (FNNs) are a natural candidate for the regression stage of such a lightweight predictor. An FNN combines Gaussian membership functions, product inference and a weighted defuzzification layer, so its rules can be read as linguistic statements about the input features [12]–[14]. Training an FNN is, however, a non-convex problem. Gradient-based training depends on the initial membership parameters, and the product of many memberships saturates when the input dimension grows [15], [16]. Population-based optimizers such as particle swarm optimization (PSO) [17], [18] are therefore often used to search the FNN parameter space.

Li [19] recently proposed IMPMSPSO-FNN for remote video perceptual quality prediction. The method initializes the swarm with a good point set (GPS) [20], divides it into several subpopulations, lets each subpopulation pick one of five PSO update rules at random in every iteration, and shares the best position among subpopulations. The optimized parameters are then used as the FNN weights and thresholds. On 37 private video sequences (21 for training and 16 for testing), the study reported a prediction accuracy of 91.13% against 79.28% for a conventional FNN, a 24.3% lower root mean square error (RMSE) and a 58.6% lower mean relative error (MRE).

These results cannot be checked or built upon in their published form. The five update rules are not identified, the convergence factor α is defined but never used in an update equation, the particle encoding and the fitness function are only described verbally, and the dataset is not available. The flowchart of the original study also feeds the error on the test data into the fitness function, which leaks test information into training [21]. In addition, the reported mean square deviation and RMSE are mutually inconsistent ($0.0617^2 = 0.0038 \neq 0.0040$ and $0.0815^2 = 0.0066 \neq 0.0078$), "prediction accuracy" is not defined for what is a continuous prediction task, and the missed-alarm rate of the proposed method (15.80%) is worse than that of the baseline (9.31%), a fact not stated in the abstract of that study. Table I summarizes these points.

In this study, we ask a narrower and testable question: when IMPMSPSO-FNN is specified completely and evaluated without leakage on public data, does its optimizer improve video quality prediction over simpler ways of training the same FNN? We make the following contributions:

1) We give a complete and reproducible specification of IMPMSPSO-FNN: the FNN equations with corrected notation, the particle encoding of all $2md + m + 1$ parameters, the five update strategies (including the role of the convergence factor), velocity and boundary handling, the cooperation rule, the training-only fitness function and the computational complexity. We release an implementation.
2) We evaluate the method under a nested, leakage-free protocol on three public databases: LIVE-VQA, which contains compression and simulated IP and wireless transmission distortions with content-separated test sets [22], and the user-generated-content databases KoNViD-1k [23] and LIVE-VQC [24]. We compare it with the same FNN trained by Adam, standard PSO and differential evolution (DE), and with support vector regression (SVR) and random forest regressors on the same features.
3) We isolate the contribution of each component through ablation, repeated-seed runs and a uniformity analysis of the good point set, and we test differences with the corrected resampled t-test and Holm adjustment [25], [26].
4) We report alarm-oriented metrics (false-alarm rate, missed-alarm rate and the area under the ROC curve) together with run time, so that the

 ©IJSRED: All Rights are Reserved

trade-offs relevant to remote quality monitoring are visible.

Fig. 1 gives an overview of the pipeline and protocol. The results do not support the claim that the IMPMSPSO optimizer is the main source of accuracy. On the two authentic-distortion databases the full method performs on a par with standard PSO, DE and gradient-trained FNNs, none of its components produces a consistent and significant gain, and an RBF-kernel SVR on the same features achieves a higher mean correlation. The choice of input representation matters more than the choice of optimizer: replacing a supervised partial least squares (PLS) projection by principal component analysis (PCA) lowers the SROCC significantly on KoNViD-1k. On LIVE-VQA, all feature-based predictors generalize poorly to unseen contents. Section II reviews related work, Sections III and IV define the model and the optimizer, Section V describes the experimental protocol, Section VI reports the results, Section VII discusses them, and Section VIII concludes.

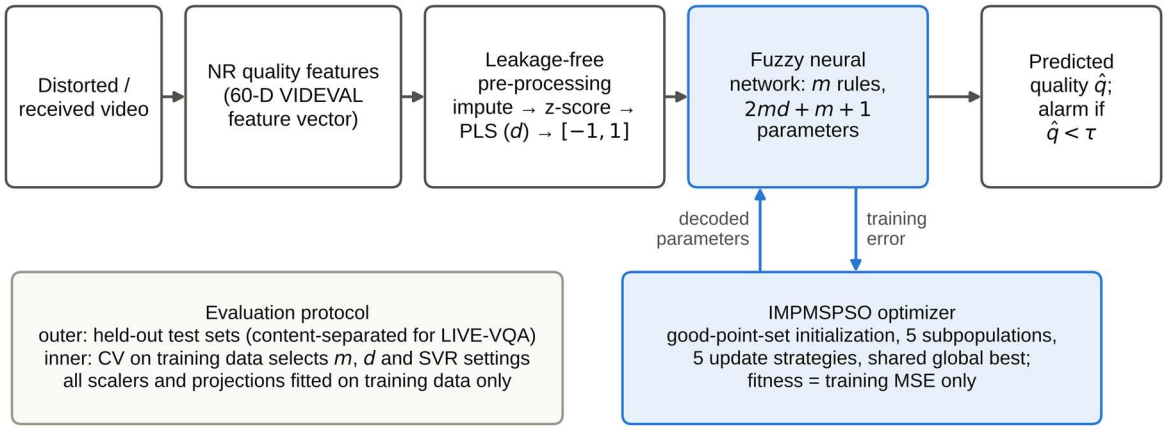


Fig. 1. Overview of the prediction pipeline and the evaluation protocol used in this study. Preprocessing, PLS projection, structure selection and the fitness function use only the training partition of each outer split.

TABLE I
REPORTED PROTOCOL OF THE ORIGINAL IMPMSPSO-FNN STUDY [19] AND THE PROTOCOL USED IN THIS STUDY

Aspect	Original study [19]	This study
Data	37 private sequences (21 train / 16 test); labels and subjective protocol not described	Three public databases with published MOS/DMOS: LIVE-VQA (150), KoNViD-1k (1200), LIVE-VQC (585)
Input features	Not specified	60-D VIDEVAL feature vector [3], projected by PLS fitted on training data
Update strategies	“Five PSO models”, not identified	Five named strategies, Eqs. (11)–(15)
Convergence factor α	Defined, not used in any update	Scales the leader-guided strategy S_5
Fitness	“Sum of squares of the classification accuracy of each class”; flowchart uses test error	Training MSE only; test data never enter training or model selection
Task and metrics	Classification accuracy reported for a continuous output; MSE and RMSE inconsistent	Regression (SROCC, KROCC, PLCC, RMSE, MAE, MRE) and alarm detection (ACC, FAR, MAR, F1, AUC)
Training details	Adam, 64×64 patches, 200 iterations, MATLAB 7.0 (no Adam in that release)	All hyper-parameters listed in Table III; Python implementation released
Validation	Single split; no repetitions or statistics	45 content-separated splits (LIVE-VQA), 20 random splits (others); nested CV; corrected t-tests

 ©IJSRED: All Rights are Reserved

II. RELATED WORK

A. Video Quality Assessment for Transmitted Video

Subjective studies define the targets that objective models learn. The LIVE Video Quality Database [22] applies MPEG-2 and H.264 compression and simulated transmission of H.264 streams over error-prone IP and wireless networks to ten reference sequences, and reports difference mean opinion scores (DMOS). Later databases shifted towards authentic distortions captured in the wild, such as KoNViD-1k [23] and LIVE-VQC [24], and towards streaming-specific impairments such as rebuffering and bitrate switching [27]. Modern codecs such as HEVC [28] change the appearance of compression artifacts, and new formats continue to raise new quality problems, for example in synthesized six-degree-of-freedom video [29]. ITU-T P.1401 [30] standardizes how objective predictors should be compared with subjective scores, using correlation, error and outlier measures.

Feature-based NR-VQA models compute statistics that change predictably with distortion and regress them onto MOS. VIDEVAL [3] selected 60 features from 763 candidates drawn from BRISQUE [4], TLVQM [6] and other models, and fed them to an SVR [31]; it remains a strong reference among handcrafted methods for user-generated content. Deep models such as VSFA [7], SimpleVQA [8], FAST-VQA [9] and DOVER [10] learn spatial and temporal representations directly and currently define the state of the art, as reviewed by Min et al. [2]. The present study does not compete with these deep models; it examines whether a particular optimizer improves a lightweight fuzzy regressor that operates on fixed features.

B. Fuzzy Neural Networks and Their Training

An FNN with Gaussian memberships, product inference and weighted-sum output is functionally a zero-order Takagi–Sugeno–Kang system, closely related to fuzzy basis function expansions [13] and to ANFIS [12]. Such models are universal approximators, and their rules are interpretable [14]. Gradient training with mini-batches, regularization and adaptive learning rates works well for moderate input dimensions [15], but the product of many membership values drives firing strengths towards zero or saturates the normalized firing levels as the dimension increases [16]. Dimension reduction before the fuzzy layer is therefore common, and we adopt supervised PLS projection [32] for this purpose.

C. Particle Swarm Optimization with Multiple Populations and Strategies

PSO moves a swarm of candidate solutions towards their personal and collective best positions [17]. The inertia weight [18] and the constriction factor [33] control the balance between exploration and exploitation, and the neighborhood topology controls how fast information spreads [34]. Many variants modify the learning rule: comprehensive learning lets each dimension learn from a different exemplar [35], time-varying acceleration coefficients shift the emphasis from the cognitive to the social term [36], and ensemble methods choose among several such rules adaptively [37]. Cooperative co-evolution splits the search across subpopulations that exchange information [38]–[40]. Recent surveys document the large number of hybrid and multi-population variants and the difficulty of comparing them fairly [41], [42], and multi-population ideas continue to be extended, for example to multitask optimization [43]. IMPMSPSO combines a random choice among five update rules with a shared global best and a GPS initialization. Its update set is similar in spirit to the ensemble in [37], but its strategies are chosen uniformly at random rather than by success rate.

D. Population Initialization

Initialization affects how the early swarm covers the search space [44], [45]. A good point set, introduced by Hua and Wang for numerical integration [20], uses the fractional parts of multiples of an irrational vector to produce points whose discrepancy decreases as $O(N^{-1+\varepsilon})$, compared with roughly $O(N^{-1/2})$ for independent uniform sampling. The constant in this bound depends on the dimension, so the practical advantage for small

 ©IJSRED: All Rights are Reserved

swarms in high dimensions has to be measured rather than assumed. Low-discrepancy sequences such as Sobol sequences [46] provide another option, and the centered L_2 discrepancy [47] offers a computable measure of uniformity.

E. Evaluation Practice

Data leakage is one of the most common causes of optimistic results in machine-learning-based science [21]. In VQA it arises when frames or patches of the same source content appear in both training and test sets, when preprocessing is fitted on the full dataset, or when test performance guides model selection. When several resampled splits are used, the training sets overlap and the usual paired t -test underestimates variance; the corrected resampled t-test of Nadeau and Bengio [25] accounts for this overlap.

III. PROBLEM FORMULATION AND FUZZY NEURAL NETWORK

A. Prediction and Alarm Tasks

Let each video be described by a feature vector $\mathbf{f} \in \mathbb{R}^F$ ($F = 60$ in this study) and a subjective score s. We map every database to a common normalized quality $q \in [0,1]$, where larger values mean better quality, using the nominal rating scale of each database (DMOS on [0, 100] for LIVE-VQA, MOS on [1, 5] for KoNViD-1k and MOS on [0, 100] for LIVE-VQC; Table II) rather than statistics of the data:

$$q = \begin{cases} 1 - s/100, & \text{LIVE-VQA} \\ (s-1)/4, & \text{KoNViD-1k} \\ s/100, & \text{LIVE-VQC} \end{cases} \tag{1}$$

The regression task is to learn a predictor $\hat{q} = g(\mathbf{f})$. The alarm task flags a video as poor when its quality falls below a threshold τ; a video is a positive case if $q < \tau$, and the predictor raises an alarm if $\hat{q} < \tau$. We set τ to the median of q in each training partition, so that the threshold never depends on test data.

B. Input Preprocessing

For each training partition we impute missing feature values with the training median, standardize every feature, project the standardized vector onto d PLS components [32] and rescale each component linearly to $[-1,1]$ with the training minimum and maximum. The test partition is transformed with the same fitted parameters. The resulting d scores form the FNN input $\mathbf{x} = (x_1, \dots, x_n)$ with $n = d$.

C. Four-Layer Fuzzy Neural Network

We use the four-layer FNN of [19], shown in Fig. 2, with corrected indices and an explicit output threshold. Layer 1 passes the inputs forward:

$$O_i^{(1)} = x_i, \qquad i = 1, \dots, n. \tag{2}$$

Layer 2 contains $n \times m$ nodes arranged in n groups. Node (i, j) evaluates a Gaussian membership function with center c_{ij} and width $\sigma_{ij} > 0$:

$$O_{ij}^{(2)} = \mu_{ij}(x_i) = \exp\left[-\left(\frac{x_i - c_{ij}}{\sigma_{ij}}\right)^2\right], i = 1, \dots, n;\ j = 1, \dots, m. \tag{3}$$

Layer 3 contains m rule nodes. Rule j combines the j-th membership of every input with the product t-norm:

$$O_j^{(3)} = \varphi_j = \prod_{i=1}^{n} \mu_{ij}(x_i), j = 1, \dots, m. \tag{4}$$

Layer 4 has a single node that forms the weighted sum of the firing strengths and adds a threshold b:

$$\hat{q} = O^{(4)} = \sum_{j=1}^{m} w_j\, \varphi_j + b, \tag{5}$$

where w_j is the weight that connects rule node j to the output node. In the original description, w_j was attributed to "fourth-layer node j", although the fourth layer has only one node, and the network was said to have "thresholds" that do not appear in its equations. Equation (5) resolves both points. We keep the unnormalized output of [19]; consequently, an input far from every rule center produces an output close to b.

 ©IJSRED: All Rights are Reserved

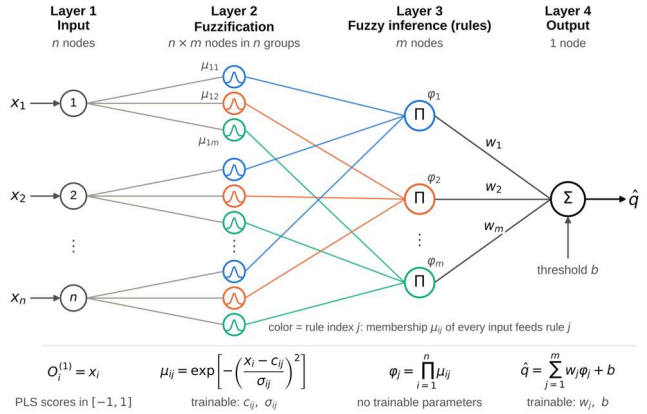


Fig. 2. Architecture of the four-layer fuzzy neural network. Each input x_i (a PLS score) feeds m Gaussian membership nodes; the j-th membership of every input feeds rule node j (shown by color); the output node forms the weighted sum of the rule firing strengths plus the threshold b. The centers c_{ij}, widths σ_{ij}, weights w_j and threshold b are the trainable parameters.

D. Parameter Encoding and Fitness

All trainable parameters are collected in one particle. With m rules and $n = d$ inputs the particle dimension is

$$D = \underset{\breve{c}_{ij}}{md} + \underset{\breve{\sigma}_{ij}}{md} + \underset{\breve{w}_j}{m} + \underset{\breve{b}}{1} = 2md + m + 1. \tag{6}$$

Each particle coordinate $p \in [-1,1]$ is decoded as follows; this mapping keeps every width positive and bounds the output-layer parameters:

$$\begin{aligned} c_{ij} &= p_c, \sigma_{ij} = \sigma_{\min} \\ &+ (\sigma_{\max} - \sigma_{\min}) \frac{p_\sigma + 1}{2}, w_j \\ &= 2\,p_w, \quad b = p_b, \end{aligned} \tag{7}$$

with $\sigma_{\min} = 0.1$ and $\sigma_{\max} = 2$. The fitness of a particle is the mean squared error of the decoded FNN on the n_{tr} training videos:

$$\mathcal{F}(\mathbf{p}) = \frac{1}{n_{tr}} \sum_{k=1}^{n_{tr}} (\hat{q}_k(\mathbf{p}) - q_k)^2. \tag{8}$$

The test partition is never used in Equation (8), in the choice of m and d, or in any stopping rule. This replaces the classification-based fitness of the original study, which is not defined for a continuous target.

IV. THE IMPMSPSO OPTIMIZER

IMPMSPSO searches $[-1,1]^D$ with N particles, each with position $\mathbf{x}_i$, velocity $\mathbf{v}_i$ and personal best $\mathbf{p}_i$. The swarm is split into K subpopulations with bests $\mathbf{s}_k$; the global best is $\mathbf{g}$. Fig. 4 and Algorithm 1 summarize the procedure; this section defines each component.

A. Good-Point-Set Initialization

Let p be the smallest prime with $p \geq 2D + 3$ and $r_k = 2\cos(2\pi k/p)$ for $k = 1, \ldots, D$. The i-th good point and the corresponding initial particle are

$$\begin{aligned} P_i &= (\{i\,r_1\}, \{i\,r_2\}, \ldots, \{i\,r_D\}), \mathbf{x}_i^{(0)} \\ &= -1 + 2P_i, \qquad i = 1, \ldots, N, \end{aligned} \tag{9}$$

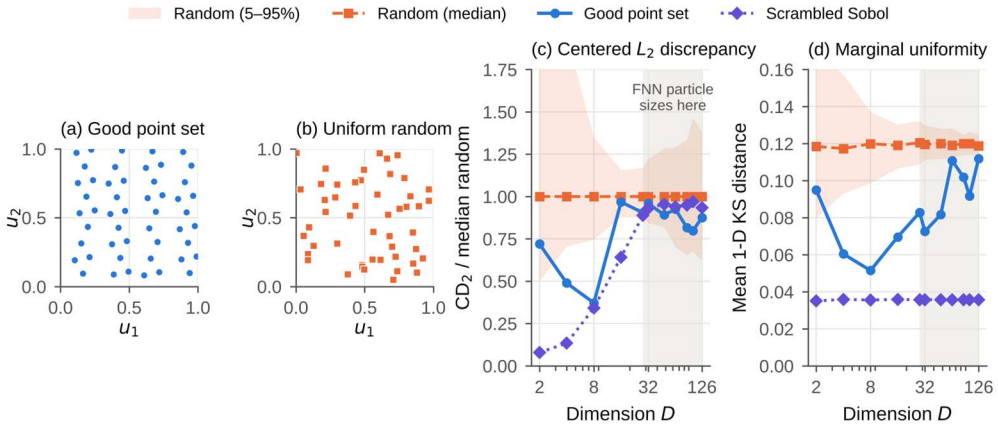


Fig. 3. Uniformity of initial populations with $N = 50$ points. (a, b) Two-dimensional good point set and uniform random sample. (c) Centered L_2 discrepancy relative to the median of 100 random samples; the band shows the 5–95% range of random samples. (d) Mean one-dimensional Kolmogorov–Smirnov distance of the coordinates. Shaded columns mark the particle dimensions used by the FNN in this study (D from 28 to 126). Scrambled Sobol points (100 seeds) are shown for reference.

 ©IJSRED: All Rights are Reserved

where $\{\cdot\}$ denotes the fractional part [20]. Initial velocities are zero. The discrepancy of a good point set satisfies $\mathcal{D}(N) \le C(\mathbf{r}, \varepsilon)\, N^{-1+\varepsilon}$ for any $\varepsilon > 0$; the rate does not depend on D, but the constant C does. The original study described the construction as "not affected by dimensionality", which overstates this property. Fig. 3 quantifies the uniformity that is actually obtained for $N = 50$.

B. Subpopulations, Update Strategies and Schedules

After initialization the swarm is randomly partitioned into $K = 5$ subpopulations of equal size, and the partition stays fixed. In every iteration t each subpopulation draws one of five strategies uniformly at random and applies it to all of its particles. The inertia weight decreases linearly and the convergence factor of [19] decreases nonlinearly:

$$w(t) = w_{\max} - (w_{\max} - w_{\min})\frac{t}{T}, \alpha(t) = 2 - 2\,\frac{e^{t/T} - 1}{e - 1}, \tag{10}$$

where T is the number of iterations. The factor α falls slowly at first and faster later, moving from 2 to 0. Let $\mathbf{a}_k$ denote the social attractor of subpopulation k (the shared global best $\mathbf{g}$ under cooperation), $\mathbf{r}_1, \mathbf{r}_2 \sim U(0,1)^D$ and $\odot$ the element-wise product. The five strategies are:

S_1, inertia-weight PSO [18]:

$$\mathbf{v}_i \leftarrow w(t)\,\mathbf{v}_i + c_1\mathbf{r}_1 \odot (\mathbf{p}_i - \mathbf{x}_i) + c_2\mathbf{r}_2 \odot (\mathbf{a}_k - \mathbf{x}_i); \tag{11}$$

S_2, ring local-best PSO [34], where $\mathbf{l}_i$ is the best personal best among particle i and its two ring neighbors within the subpopulation:

$$\mathbf{v}_i \leftarrow w(t)\,\mathbf{v}_i + c_1\mathbf{r}_1 \odot (\mathbf{p}_i - \mathbf{x}_i) + c_2\mathbf{r}_2 \odot (\mathbf{l}_i - \mathbf{x}_i); \tag{12}$$

S_3, comprehensive learning [35], where each dimension ℓ learns from its own personal best with probability $1 - P_c$ and otherwise from the winner $f_i(\ell)$ of a binary tournament among the personal bests of the subpopulation:

$$v_{i\ell} \leftarrow w(t)\,v_{i\ell} + c\,r_{1\ell}\big(p_{f_i(\ell),\ell} - x_{i\ell}\big), \ell = 1, \dots, D; \tag{13}$$

S_4, time-varying acceleration coefficients [36]:

$$\mathbf{v}_i \leftarrow w(t)\,\mathbf{v}_i + c_1(t)\mathbf{r}_1 \odot (\mathbf{p}_i - \mathbf{x}_i) + c_2(t)\mathbf{r}_2 \odot (\mathbf{a}_k - \mathbf{x}_i), c_1(t) = 2.5 - 2\,t/T\,, c_2(t) = 0.5 + 2\,t/T\,; \tag{14}$$

S_5, a leader-guided move scaled by $\alpha(t)$, which follows the encircling rule of the grey wolf optimizer [48] whose convergence factor has the same range:

$$\mathbf{v}_i \leftarrow \mathbf{a}_k - \mathbf{A} \odot |\mathbf{C} \odot \mathbf{a}_k - \mathbf{x}_i| - \mathbf{x}_i, \mathbf{A} = \alpha(t)\,(2\mathbf{r}_1 - \mathbf{1}),\ \mathbf{C} = 2\mathbf{r}_2. \tag{15}$$

For $\alpha > 1$ the step can overshoot the leader (exploration); as $\alpha \to 0$ the particle is pulled onto it (exploitation). This is the only place where α acts, which gives the convergence factor of [19] a concrete role. The original paper does not name its five "PSO models", so this set is our specification: it covers global, local, dimension-wise and time-varying learning plus one explicitly scheduled leader-guided move, all from established methods.

C. Velocity Clamping, Boundaries and Cooperation

After every update, velocities are clamped and positions are kept inside the search box; a coordinate that leaves the box is placed on the boundary and its velocity is set to zero:

$$\mathbf{v}_i \leftarrow \mathrm{clip}(\mathbf{v}_i, -v_{\max}, v_{\max}), \mathbf{x}_i \leftarrow \mathrm{clip}(\mathbf{x}_i + \mathbf{v}_i, -1, 1). \tag{16}$$

All particles are then evaluated, personal and subpopulation bests are updated, and cooperation selects the best subpopulation best as the shared attractor of the next iteration:

$$\mathbf{g} \leftarrow \arg\min_{k \in \{1,\dots,K\}} \mathcal{F}(\mathbf{s}_k), \mathbf{a}_k = \mathbf{g} \ \text{ for all } k. \tag{17}$$

Without cooperation (an ablation in Section VI) each subpopulation uses its own best, $\mathbf{a}_k = \mathbf{s}_k$, and the result is the best of the K subpopulation bests. The run stops after T iterations, and the decoded $\mathbf{g}$ is the trained FNN.

 ©IJSRED: All Rights are Reserved

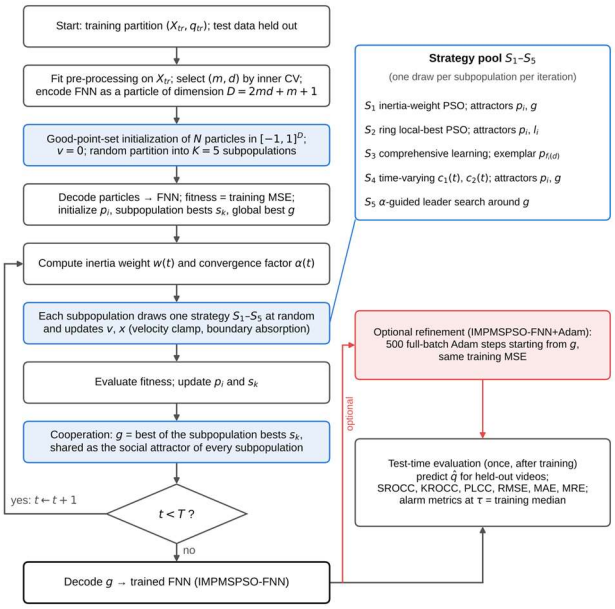


Fig. 4. Flowchart of IMPMSPSO-FNN training. Blue boxes mark the three mechanisms evaluated in the ablation study (good-point-set initialization, random strategy selection per subpopulation and cooperation through a shared global best). Test data are used only once, after training.

Algorithm 1. IMPMSPSO-FNN training (one training partition).

Input: training data $(\mathbf{X}_{tr}, \mathbf{q}_{tr})$; structure (m, d); N, K, T, $w_{\max}$, $w_{\min}$, c_1, c_2, c, P_c, $v_{\max}$
Output: FNN parameters $\theta = (\{c_{ij}\}, \{\sigma_{ij}\}, \{w_j\}, b)$

1: fit preprocessing on $\mathbf{X}_{tr}$ and compute PLS scores $\mathbf{x}_k \in [-1,1]^d$; set $D = 2md + m + 1$
2: initialize $\mathbf{x}_i^{(0)}$ with the good point set, Eq. (9); $\mathbf{v}_i \leftarrow \mathbf{0}$
3: randomly partition $\{1, \ldots, N\}$ into K subpopulations
4: evaluate $\mathcal{F}(\mathbf{x}_i)$ by Eq. (8); set $\mathbf{p}_i$, $\mathbf{s}_k$ and $\mathbf{g}$
5: **for** $t = 0$ to $T - 1$ **do**
  6: compute $w(t)$ and $\alpha(t)$ by Eq. (10)
  7: **for** each subpopulation k **do**
    8: draw $S \sim \mathcal{U}\{S_1, \ldots, S_5\}$; update $\mathbf{v}_i$ of all particles of k by Eqs. (11)–(15)
    9: clamp velocities and positions by Eq. (16)
  10: **end for**
  11: evaluate all particles; update $\mathbf{p}_i$ and $\mathbf{s}_k$
  12: cooperation: update $\mathbf{g}$ by Eq. (17)
13: **end for**
14: **return** $\theta = \text{decode}(\mathbf{g})$ by Eq. (7)

D. Computational Complexity

One fitness evaluation computes $n_{tr} \times m \times d$ membership values, so it costs $O(n_{tr} md)$. The strategy updates cost $O(ND) = O(Nmd)$ per iteration, which is negligible next to the N fitness evaluations. Training therefore costs $O(N(T + 1)\, n_{tr}\, m\, d)$ time; with the settings in Table III each run uses $N(T + 1) = 10\,050$ fitness evaluations, the same budget as standard PSO and DE. A vectorized implementation needs $O(Nn_{tr}m)$ memory. Prediction for one video costs $O(Fd)$ for the projection plus $O(md)$ for the FNN.

E. Refinement Variant

The flowchart of [19] describes the optimizer output as the "initial weights and thresholds" of the FNN, while its text treats the same output as the final model, and its experimental section mentions Adam. Both readings are plausible, so we evaluate both: IMPMSPSO-FNN uses the decoded global best directly, and IMPMSPSO-FNN+Adam continues from it with 500 full-batch Adam steps [49] on the same training MSE.

V. EXPERIMENTAL SETUP

A. Databases and Features

Table II summarizes the three databases. LIVE-VQA [22] contains 150 distorted videos generated from ten reference contents at 768×432 pixels with MPEG-2 compression, H.264 compression and simulated transmission of H.264 streams over IP and wireless channels; it is the database closest to the remote-transmission scenario of [19]. KoNViD-1k [23] (1200 videos) and LIVE-VQC [24] (585 videos) contain authentic distortions of user-generated video, including capture artifacts and the compression applied by sharing platforms.

We use the 60-dimensional VIDEVAL feature vectors released by Tu et al. [3] for all three databases, so that every regressor sees identical inputs. One feature contains missing values for 33 LIVE-VQA videos, 2 KoNViD-1k videos and 1 LIVE-VQC video; these are imputed inside each training partition. The released metadata store the LIVE-VQA scores in a column labeled "MOS". We verified their direction before use. Within the two compression ladders (file indices 9–12 and 13–16) the stored scores increase with the file index (mean within-content Spearman correlations of 1.00 and 0.98), whereas a quality predictor trained on KoNViD-1k assigns lower quality to the same videos

 ©IJSRED: All Rights are Reserved

as the index increases (−0.42 and −0.85). The column therefore holds DMOS, where larger means worse, and Equation (1) inverts it.

TABLE II
DATABASES USED IN THIS STUDY

Database	Videos	Contents	Distortions	Score (scale)	Outer splits
LIVE-VQA [22]	150	10	MPEG-2, H.264, IP and wireless transmission (simulated)	DMOS (0–100)	45, 2 contents held out
KoNViD-1k [23]	1200	1200	Authentic (Flickr videos)	MOS (1–5)	20, random 80/20
LIVE-VQC [24]	585	585	Authentic (mobile capture)	MOS (0–100)	20, random 80/20

B. Evaluation Protocol

We use a nested design (Fig. 1, bottom left). On LIVE-VQA, every outer split holds out all 30 videos of two reference contents, giving all $\binom{10}{2} = 45$ content-separated splits; no frame, distortion level or content of a test video is seen during training. On KoNViD-1k and LIVE-VQC, where every video has its own content, we draw 20 random 80/20 splits with a fixed seed. Inside each outer training partition, we select the FNN structure $(m, d) \in \{3,5\} \times \{4,8,12\}$ by the mean validation RMSE of content-grouped 4-fold cross-validation (LIVE-VQA) or 3-fold cross-validation (other databases). Each FNN method selects its own structure; the refinement variant reuses the structure selected for IMPMSPSO-FNN. SVR hyper-parameters $C \in \{0.25,1,4,16\}$ and $\gamma \in \{0.25,1,4\}/F$ are selected in the same way, following [3]. The selected model is retrained on the full outer training partition and evaluated once on the outer test partition. Every random number generator is seeded from the split index, and every method uses the same seed within a split.

C. Compared Methods and Settings

All FNN-based methods share the architecture, encoding and fitness of Section III and differ only in the optimizer. FNN-Adam starts from a random particle and runs full-batch Adam; this represents conventional gradient training of the FNN. PSO-FNN is standard inertia-weight PSO (S_1 only, random initialization, one population). DE-FNN uses DE/rand/1/bin [50]. PSO-FNN, DE-FNN and IMPMSPSO-FNN all use 10 050 fitness evaluations. SVR [31] and random forest [51] regress the full 60-dimensional standardized feature vector. For the ablation study, we keep the structure and seed of IMPMSPSO-FNN in each split and switch off one component at a time: GPS initialization (replaced by uniform random sampling), multi-strategy selection (all subpopulations use S_1), cooperation ($\mathbf{a}_k = \mathbf{s}_k$) and the schedules ($w = 0.7298$ [33] and $\alpha = 1$ throughout). A further variant replaces the PLS projection by PCA. Table III lists all settings.

TABLE III
HYPER-PARAMETERS OF ALL METHODS

Method	Setting
FNN (all variants)	$m \in \{3,5\}$, $d \in \{4,8,12\}$ (selected); $\sigma_{ij} \in [0.1,2]$; $w_j \in [-2,2]$; $b \in [-1,1]$; fitness Eq. (8)
IMPMSPSO	$N = 50$, $K = 5$, $T = 200$; $w_{\max} = 0.95$, $w_{\min} = 0.30$ (values from [19]); $c_1 = c_2 = 2.0$; $c = 1.49445$, $P_c = 0.3$ (S_3); $v_{\max} = 0.5$; search box $[-1,1]^D$
PSO-FNN	As IMPMSPSO with $K = 1$, strategy S_1 only, uniform random initialization
DE-FNN	DE/rand/1/bin; population 50; 200 generations; $F = 0.5$; $CR = 0.9$
FNN-Adam	Random initial particle; 2000 full-batch steps; learning rate 0.01; widths parameterized as $\sigma = e^u$, clipped to $[0.05,5]$
IMPMSPSO-FNN+Adam	IMPMSPSO result followed by 500 full-batch Adam steps; learning rate 0.005
SVR	RBF kernel; $C \in \{0.25,1,4,16\}$, $\gamma \in \{0.25,1,4\}/60$ (selected); $\epsilon = 0.02$
Random forest	300 trees; minimum leaf size 2; other settings at library defaults
Implementation	Python 3.13, NumPy 2.5.3, SciPy 1.18.1, scikit-learn 1.9.1; 2-core cloud CPU, single thread per run

All values were fixed before the experiments except those marked "selected", which were chosen by inner cross-validation on the training data.

D. Performance Measures and Statistical Tests

For regression we report the Spearman (SROCC) and Kendall (KROCC) rank correlations, the

 ©IJSRED: All Rights are Reserved

Pearson linear correlation (PLCC), and the RMSE and mean absolute error (MAE) on the normalized scale [30]. We compute PLCC and RMSE directly on the predictions, without the logistic mapping that is sometimes fitted on test data. MRE is computed on each database's original scale (DMOS for LIVE-VQA) as $\mathrm{MRE} = \frac{1}{n}\sum_k |\hat{s}_k - s_k| / s_k$. For the alarm task, with poor-quality videos as the positive class, we report accuracy (ACC), the false-alarm rate $\mathrm{FAR} = FP/(FP+TN)$, the missed-alarm rate $\mathrm{MAR} = FN/(FN+TP)$, the F1 score and the area under the ROC curve (AUC), which does not depend on the threshold.

Because the outer training sets overlap, we compare methods with the corrected resampled t-test [25]. For the per-split differences $\delta_j, j = 1, \ldots, J$,

$$t = \frac{\bar{\delta}}{\sqrt{\left(\frac{1}{J} + \frac{n_{te}}{n_{tr}}\right)\hat{s}_\delta^2}}, \text{with } J - 1 \text{ degrees of freedom,} \tag{18}$$

where $n_{te}/n_{tr} = 0.25$ for all databases. We adjust the p-values for the six comparisons of each table with Holm's procedure [26] and use $\alpha_{\mathrm{sig}} = 0.05$.

VI. RESULTS

A. Prediction Accuracy

Table IV reports the mean and standard deviation over outer splits, and Fig. 5 shows the distribution of the test SROCC.

TABLE IV
PREDICTION PERFORMANCE ON THE TEST PARTITIONS

Method	SROCC ↑	KROCC ↑	PLCC ↑	RMSE ↓	MAE ↓	MRE ↓
LIVE-VQA (45 splits)						
SVR (RBF)	**0.277 ± 0.200**	0.202 ± 0.144	**0.250 ± 0.230**	**0.117 ± 0.021**	0.097 ± 0.020	0.197 ± 0.040
Random forest	0.244 ± 0.206	0.177 ± 0.144	0.223 ± 0.230	0.118 ± 0.010	0.098 ± 0.009	0.202 ± 0.027
FNN-Adam	0.135 ± 0.220	0.096 ± 0.156	0.144 ± 0.213	0.195 ± 0.111	0.157 ± 0.079	0.317 ± 0.169
DE-FNN	0.213 ± 0.216	0.152 ± 0.151	0.190 ± 0.225	0.168 ± 0.101	0.133 ± 0.063	0.265 ± 0.124
PSO-FNN	0.128 ± 0.230	0.095 ± 0.159	0.122 ± 0.227	0.152 ± 0.044	0.124 ± 0.035	0.246 ± 0.078
IMPMSPSO-FNN	0.168 ± 0.275	0.124 ± 0.193	0.160 ± 0.277	0.143 ± 0.038	0.117 ± 0.030	0.227 ± 0.067
IMPMSPSO-FNN+Adam	0.132 ± 0.244	0.097 ± 0.169	0.112 ± 0.227	0.236 ± 0.100	0.196 ± 0.081	0.383 ± 0.161
KoNViD-1k (20 splits)						
SVR (RBF)	**0.767 ± 0.028**	0.569 ± 0.026	**0.758 ± 0.027**	**0.104 ± 0.005**	0.081 ± 0.004	0.121 ± 0.008
Random forest	0.710 ± 0.033	0.520 ± 0.028	0.710 ± 0.036	0.112 ± 0.005	0.089 ± 0.004	0.132 ± 0.009
FNN-Adam	0.742 ± 0.034	0.550 ± 0.031	0.723 ± 0.038	0.110 ± 0.008	0.085 ± 0.006	0.125 ± 0.011
DE-FNN	0.689 ± 0.045	0.504 ± 0.037	0.671 ± 0.043	0.118 ± 0.006	0.093 ± 0.004	0.137 ± 0.009
PSO-FNN	0.722 ± 0.041	0.531 ± 0.034	0.710 ± 0.041	0.112 ± 0.007	0.087 ± 0.005	0.129 ± 0.011
IMPMSPSO-FNN	0.719 ± 0.040	0.529 ± 0.035	0.703 ± 0.043	0.113 ± 0.007	0.089 ± 0.006	0.131 ± 0.012
IMPMSPSO-FNN+Adam	0.744 ± 0.026	0.550 ± 0.023	0.729 ± 0.030	0.109 ± 0.006	0.085 ± 0.005	0.125 ± 0.009
LIVE-VQC (20 splits)						
SVR (RBF)	**0.720 ± 0.045**	0.533 ± 0.040	**0.721 ± 0.053**	**0.117 ± 0.008**	0.089 ± 0.006	0.183 ± 0.030
Random forest	0.692 ± 0.049	0.510 ± 0.042	0.709 ± 0.053	0.119 ± 0.007	0.093 ± 0.005	0.191 ± 0.028
FNN-Adam	0.696 ± 0.044	0.512 ± 0.040	0.700 ± 0.048	0.121 ± 0.008	0.093 ± 0.007	0.187 ± 0.028
DE-FNN	0.685 ± 0.052	0.503 ± 0.046	0.685 ± 0.049	0.124 ± 0.008	0.096 ± 0.005	0.197 ± 0.030
PSO-FNN	0.685 ± 0.063	0.503 ± 0.053	0.690 ± 0.061	0.123 ± 0.009	0.095 ± 0.007	0.190 ± 0.028
IMPMSPSO-FNN	0.682 ± 0.056	0.500 ± 0.045	0.684 ± 0.053	0.124 ± 0.010	0.096 ± 0.007	0.195 ± 0.031
IMPMSPSO-FNN+Adam	0.694 ± 0.054	0.513 ± 0.046	0.690 ± 0.048	0.123 ± 0.009	0.093 ± 0.007	0.189 ± 0.032

Values are mean ± standard deviation over outer splits. PLCC and RMSE are computed without logistic mapping. The best SROCC, PLCC and RMSE values for each database are shown in bold. ↑: higher is better; ↓: lower is better.

 ©IJSRED: All Rights are Reserved

On KoNViD-1k, IMPMSPSO-FNN reached an SROCC of 0.719 ± 0.040, compared with 0.722 ± 0.041 for PSO-FNN and 0.742 ± 0.034 for FNN-Adam. On LIVE-VQC the corresponding values were 0.682 ± 0.056, 0.685 ± 0.063 and 0.696 ± 0.044. Among the FNN-based methods, the highest mean SROCC was obtained by IMPMSPSO-FNN+Adam on KoNViD-1k and by FNN-Adam on LIVE-VQC. SVR was the most accurate method on both authentic-distortion databases (SROCC 0.767 ± 0.028 and 0.720 ± 0.045). These SVR values are close to the medians of 100 random splits reported in the public release of VIDEVAL (0.7832 and 0.7522) [3], which supports the correctness of our feature handling and protocol.

On LIVE-VQA, every method generalized poorly to unseen contents. The mean SROCC ranged from 0.128 to 0.277, with SVR highest at 0.277 ± 0.200, and individual splits varied from strongly negative to above 0.7 (Fig. 5a). The FNN methods also had larger RMSE than SVR on this database (for example, 0.143 ± 0.038 for IMPMSPSO-FNN against 0.117 ± 0.021 for SVR). With only eight training contents, the features and regressors do not separate content from distortion well enough to rank the distortions of new contents; Section VII examines this in more detail.

To see where the LIVE-VQA predictions fail, we pooled the out-of-sample predictions of each video over the splits in which it was tested. Within individual contents the ranking was better than across contents: the mean within-content SROCC was 0.429 for SVR and 0.338 for IMPMSPSO-FNN. Across contents, however, the correlation within each distortion family stayed low: for videos with file indices 9–16, which form monotone bitrate ladders in every content (the H.264 and MPEG-2 versions), SROCC was 0.135 for SVR and 0.036 for IMPMSPSO-FNN; for indices 2–8 (simulated wireless and IP transmission) it was 0.002 and 0.175. The features therefore carry information about distortion severity, but their dependence on content masks it when the contents are new.

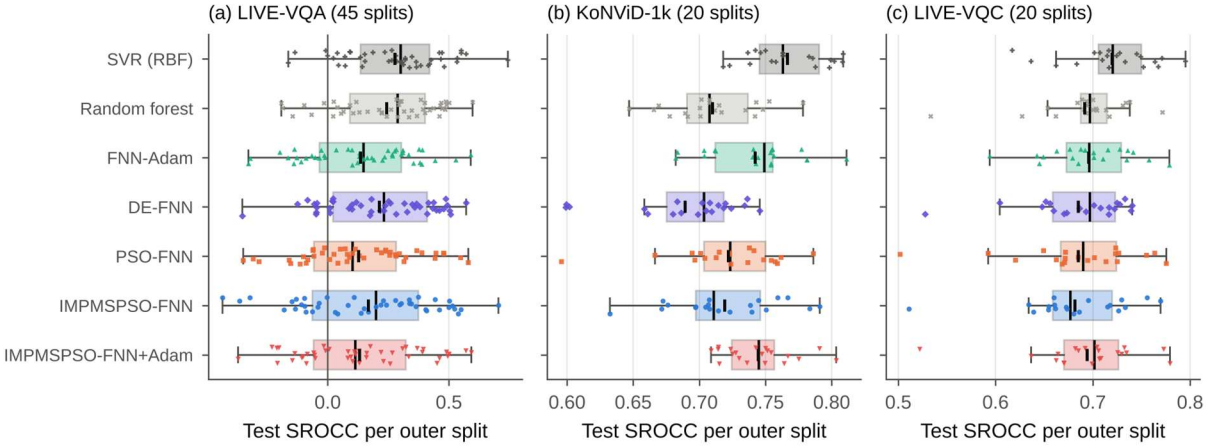


Fig. 5. Test SROCC of every method in every outer split. Boxes show the interquartile range and median, whiskers the range without outliers, points individual splits and the black vertical tick the mean. Note the different horizontal scales.

Table V tests these differences; a positive ΔSROCC or a negative ΔRMSE favors IMPMSPSO-FNN. After Holm adjustment, only one SROCC difference was significant: SVR scored higher than IMPMSPSO-FNN on KoNViD-1k (adjusted $p = 0.004$). No difference between IMPMSPSO-FNN and the other FNN training methods (FNN-Adam, PSO-FNN and DE-FNN) was significant on any database. SVR had a higher mean SROCC than IMPMSPSO-FNN on every database, but the difference did not reach significance on LIVE-VQA and LIVE-VQC, where the split-to-split variation is larger. Adding the Adam refinement changed the mean SROCC relative to IMPMSPSO-FNN by +0.025 on KoNViD-1k, +0.013 on LIVE-VQC and −0.036 on LIVE-VQA.

 ©IJSRED: All Rights are Reserved

TABLE V
PAIRED COMPARISON OF IMPMSPSO-FNN WITH EACH OTHER METHOD

Comparator	ΔSROCC	Adj. p	ΔRMSE	Adj. p
LIVE-VQA				
SVR (RBF)	−0.109	1.000	+0.0260	1.000
Random forest	−0.076	1.000	+0.0252	1.000
FNN-Adam	+0.033	1.000	−0.0517	1.000
DE-FNN	−0.045	1.000	−0.0248	1.000
PSO-FNN	+0.040	1.000	−0.0085	1.000
IMPMSPSO-FNN+Adam	+0.036	1.000	−0.0927	0.242
KoNViD-1k				
SVR (RBF)	−0.047	0.004*	+0.0092	0.004*
Random forest	+0.009	1.000	+0.0007	1.000
FNN-Adam	−0.023	0.631	+0.0029	1.000
DE-FNN	+0.030	0.624	−0.0051	0.718
PSO-FNN	−0.003	1.000	+0.0012	1.000
IMPMSPSO-FNN+Adam	−0.025	0.221	+0.0037	0.101
LIVE-VQC				
SVR (RBF)	−0.038	0.759	+0.0072	1.000
Random forest	−0.010	1.000	+0.0053	1.000
FNN-Adam	−0.014	1.000	+0.0023	1.000
DE-FNN	−0.003	1.000	−0.0001	1.000
PSO-FNN	−0.003	1.000	+0.0011	1.000
IMPMSPSO-FNN+Adam	−0.013	1.000	+0.0004	1.000

Mean difference over outer splits (IMPMSPSO-FNN minus comparator) and Holm-adjusted p-value of the corrected resampled t-test. A positive ΔSROCC or a negative ΔRMSE favors IMPMSPSO-FNN; * marks $p < 0.05$.

Fig. 6 plots the out-of-sample predictions pooled over splits (each video averaged over the splits in which it was a test video) for IMPMSPSO-FNN and SVR, and Fig. 7 shows the residual distributions. The FNN predictions are less concentrated around the diagonal than those of SVR and include a few large negative errors. These come from test videos whose PLS scores fall outside the region covered by the rules; Equation (5) then returns a value close to the threshold b.

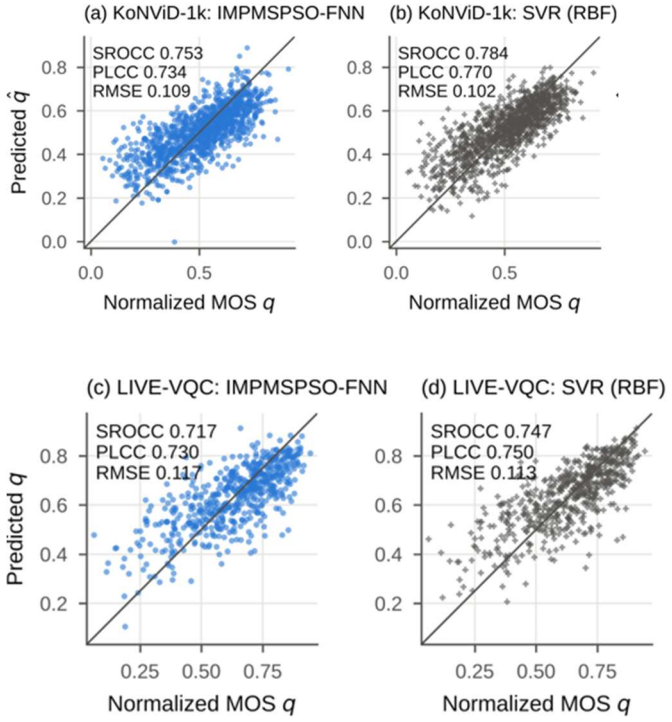


Fig. 6. Pooled out-of-sample predictions against normalized subjective scores for IMPMSPSO-FNN and SVR. Each point is one video, averaged over the outer splits in which it was in the test set. The diagonal marks perfect prediction.

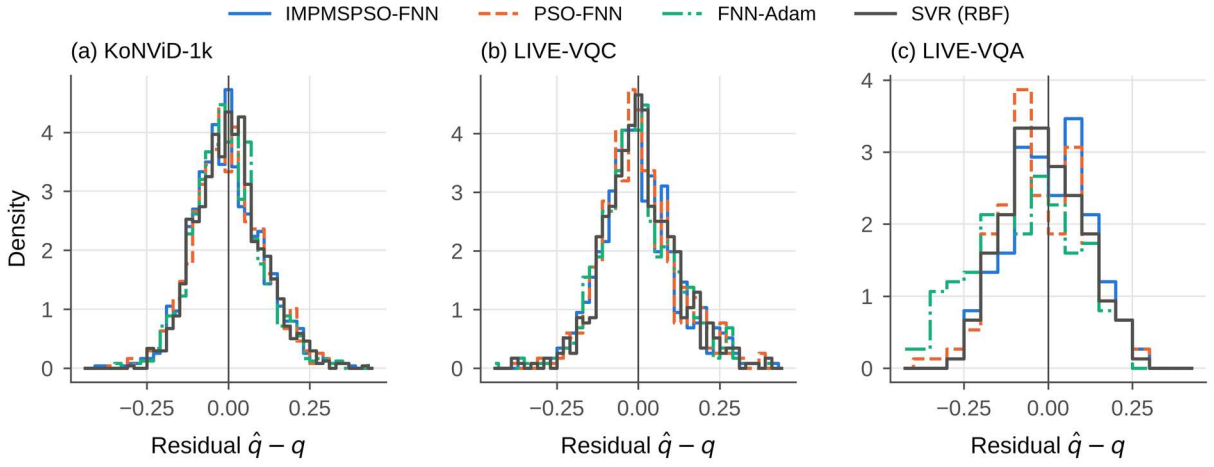


Fig. 7. Distribution of pooled out-of-sample residuals $\hat{q} - q$ for four methods on each database.

 ©IJSRED: All Rights are Reserved

B. Ablation of the Optimizer Components

Table VI and Fig. 8 report the ablations, each run with the structure and seed of the full method in the same split. Removing the GPS initialization, the multi-strategy selection or the cooperation changed the mean SROCC by at most 0.058, and none of these changes was significant after Holm adjustment on any database. Without the good point set, the optimizer even reached a slightly lower mean training MSE than the full method on all three databases. The averages, however, hide one effect of the multi-strategy update. With S_1 only, the swarm stalled (final training MSE more than 1.5 times that of the full method in the same split) in 4 of 45 LIVE-VQA splits, 1 of 20 KoNViD-1k splits and 1 of 20 LIVE-VQC splits, always at the largest structure ($D = 126$); in the KoNViD-1k case every rule was inactive on the test videos and the prediction was constant (counted as SROCC = 0). No other ablation stalled, and the full method never stalled relative to any of its ablations. Random strategy selection therefore guards against premature convergence in large particles, even though it does not raise the mean accuracy. Turning off the schedules raised the mean training MSE on every database (LIVE-VQA: 0.00798 vs. 0.00686, KoNViD-1k: 0.01278 vs. 0.01139, LIVE-VQC: 0.01360 vs. 0.01251), so the decreasing inertia weight helps the optimizer fit the training data, but this did not produce a consistent test gain. The largest single effect came from the input representation: using PCA instead of PLS changed the SROCC by −0.161 on KoNViD-1k (adjusted $p < 0.001$), −0.063 on LIVE-VQC (adjusted $p = 0.082$) and −0.106 on LIVE-VQA (adjusted $p = 1.000$).

TABLE VI
ABLATION OF IMPMSPSO-FNN

Variant	SROCC	ΔSROCC (adj. p)	RMSE	Training MSE
LIVE-VQA				
Full IMPMSPSO-FNN	0.168 ± 0.275	—	0.143 ± 0.038	0.00686
w/o good point set	0.125 ± 0.241	−0.043 (1.000)	0.140 ± 0.031	0.00672
w/o multi-strategy (S_1 only)	0.125 ± 0.259	−0.043 (1.000)	0.142 ± 0.045	0.00697
w/o cooperation	0.110 ± 0.247	−0.058 (1.000)	0.136 ± 0.028	0.00722
w/o $w(t)$, $\alpha(t)$ schedules	0.150 ± 0.222	−0.018 (1.000)	0.128 ± 0.024	0.00798
PCA instead of PLS inputs	0.062 ± 0.249	−0.106 (1.000)	0.140 ± 0.044	n/a
KoNViD-1k				
Full IMPMSPSO-FNN	0.719 ± 0.040	—	0.113 ± 0.007	0.01139
w/o good point set	0.717 ± 0.038	−0.003 (1.000)	0.114 ± 0.006	0.01133
w/o multi-strategy (S_1 only)	0.688 ± 0.166	−0.031 (1.000)	0.115 ± 0.012	0.01180
w/o cooperation	0.713 ± 0.032	−0.006 (1.000)	0.114 ± 0.005	0.01175
w/o $w(t)$, $\alpha(t)$ schedules	0.677 ± 0.057	−0.042 (0.243)	0.120 ± 0.008	0.01278
PCA instead of PLS inputs	0.559 ± 0.041	−0.161 (<0.001*)	0.131 ± 0.005	n/a
LIVE-VQC				
Full IMPMSPSO-FNN	0.682 ± 0.056	—	0.124 ± 0.010	0.01251
w/o good point set	0.683 ± 0.056	+0.001 (1.000)	0.123 ± 0.010	0.01220
w/o multi-strategy (S_1 only)	0.660 ± 0.095	−0.022 (1.000)	0.125 ± 0.010	0.01279
w/o cooperation	0.675 ± 0.041	−0.006 (1.000)	0.124 ± 0.007	0.01269
w/o $w(t)$, $\alpha(t)$ schedules	0.668 ± 0.049	−0.014 (1.000)	0.126 ± 0.009	0.01360
PCA instead of PLS inputs	0.618 ± 0.061	−0.063 (0.082)	0.130 ± 0.008	n/a

Mean ± standard deviation over outer splits. ΔSROCC is the mean difference from the full method, with the Holm-adjusted p-value in parentheses; * marks $p < 0.05$. Training MSE is the final fitness of the optimizer, averaged over splits. A constant prediction, whose correlation is undefined, is counted as SROCC = 0.

 ©IJSRED: All Rights are Reserved

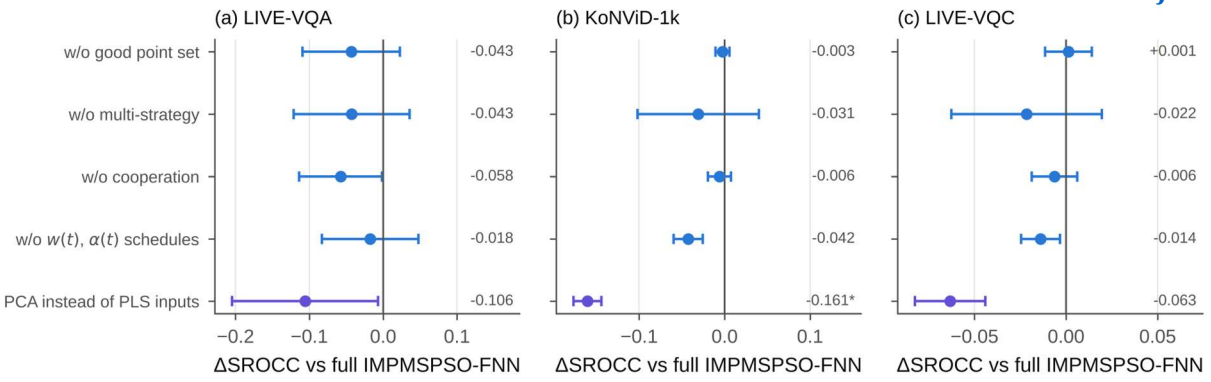


Fig. 8. Change in test SROCC when one component is removed from IMPMSPSO-FNN (mean over outer splits with 95% confidence interval of the mean). Values are printed on the right; * marks a Holm-adjusted $p < 0.05$.

C. Convergence and Run-to-Run Variability

To separate optimizer behavior from structure selection, we ran every optimizer variant with a fixed structure $(m = 5,\ d = 8,\ D = 86)$ on the first 5 outer splits of KoNViD-1k and LIVE-VQC with 5 seeds each. Fig. 9 shows the median best training MSE and the swarm diversity (mean distance of particles from the swarm centroid), and Table VII summarizes the final values.

TABLE VII
REPEATED-SEED STUDY AT A FIXED STRUCTURE

Optimizer	Train MSE	Seed SD	Full better	Test SROCC	Time (s)
KoNViD-1k					
IMPMSPSO (full)	0.01130	0.00031	—	0.731 ± 0.017	6.3
PSO	0.01122	0.00046	44% (0.474)	0.734 ± 0.016	6.3
DE	0.01162	0.00039	72% (0.001)	0.724 ± 0.019	6.6
w/o good point set	0.01121	0.00035	40% (0.325)	0.731 ± 0.015	6.2
w/o multi-strategy	0.01116	0.00047	36% (0.101)	0.734 ± 0.024	6.3
w/o cooperation	0.01159	0.00026	80% (0.004)	0.726 ± 0.011	6.3
w/o schedules	0.01253	0.00083	96% (<0.001)	0.704 ± 0.028	6.3
LIVE-VQC					
IMPMSPSO (full)	0.01177	0.00043	—	0.672 ± 0.026	3.0
PSO	0.01136	0.00040	20% (0.006)	0.676 ± 0.024	2.9
DE	0.01217	0.00026	84% (0.001)	0.664 ± 0.020	3.1
w/o good point set	0.01185	0.00038	48% (0.426)	0.665 ± 0.018	3.0
w/o multi-strategy	0.01125	0.00045	16% (<0.001)	0.685 ± 0.022	3.0
w/o cooperation	0.01235	0.00042	80% (<0.001)	0.664 ± 0.027	3.1
w/o schedules	0.01310	0.00061	96% (<0.001)	0.649 ± 0.026	3.0

Fixed structure $m = 5$, $d = 8$; 5 splits × 5 seeds per database. "Full better" is the share of paired runs in which the full method reached a lower final training MSE, with the Wilcoxon signed-rank p-value in parentheses. SROCC is the test value (mean ± mean within-split standard deviation across seeds). Time is the mean wall-clock time per run.

The lowest mean final training MSE on both databases was reached by IMPMSPSO without multi-strategy selection. The full method reached a lower training MSE than standard PSO in 44% and 20% of the paired runs, and a lower one than DE in 72% and 84%. The run-to-run standard deviation of the test SROCC within a split was 0.017 (KoNViD-1k) and 0.026 (LIVE-VQC) for the full method, compared with 0.016 and 0.024 for standard PSO. Halfway through the run the median swarm diversity of the full method was 2.28 (KoNViD-1k) and 2.33 (LIVE-VQC), against 1.65 and 1.63 without multi-

 ©IJSRED: All Rights are Reserved

strategy selection and 1.67 and 1.58 for standard PSO. The random choice among strategies therefore keeps the swarm more spread out for most of the run. In the last iterations the diversity of the full method drops below that of the S_1-only variant, because the leader-guided strategy S_5 pulls particles onto the global best as $\alpha(t) \to 0$. At this fixed structure the extra exploration did not pay off: the S_1-only variant reached a lower mean final training MSE than the full method on both databases.

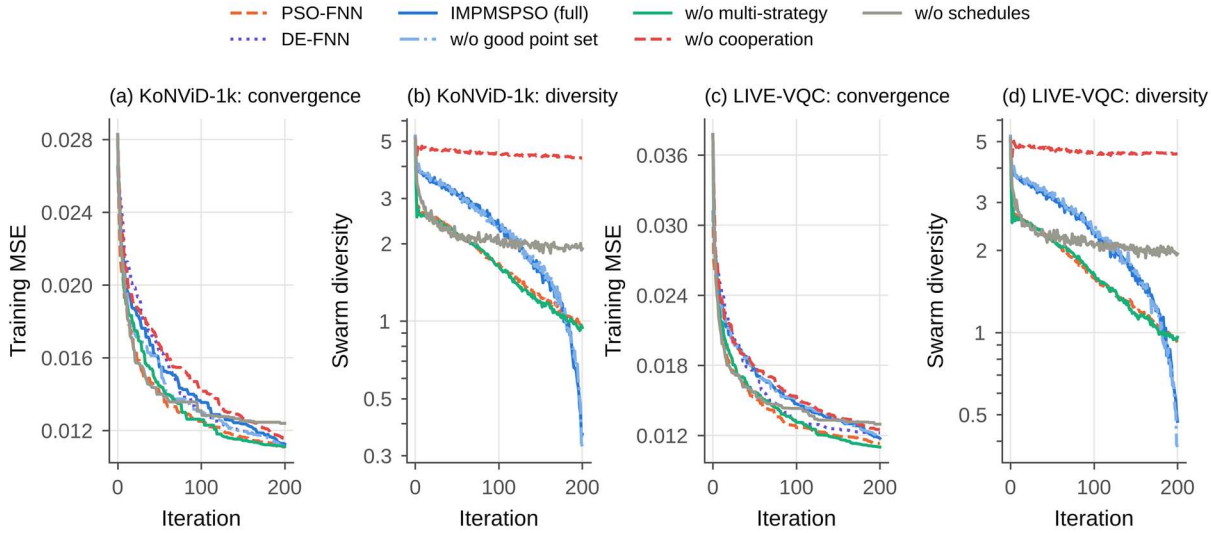


Fig. 9. Median best training MSE (convergence) and median swarm diversity over iterations for each optimizer variant at a fixed FNN structure. Medians are taken over all split × seed runs. DE has no velocity-based swarm and is omitted from the diversity panels.

D. Alarm Detection

Table VIII and Fig. 10 evaluate the methods as quality alarms. On the two authentic-distortion databases the ROC curves of the FNN methods almost coincide, and SVR had the highest mean AUC on both (0.881 ± 0.021 on KoNViD-1k and 0.859 ± 0.033 on LIVE-VQC). At the fixed threshold τ the methods trade false alarms against missed alarms differently. On LIVE-VQA, IMPMSPSO-FNN had both the lowest false-alarm rate among the FNN methods (0.340 ± 0.344) and the highest missed-alarm rate (0.635 ± 0.353), whereas SVR had FAR 0.727 ± 0.336 and MAR 0.227 ± 0.339. This mirrors the original study, in which the proposed method lowered the false-alarm rate but raised the missed-alarm rate. Such shifts at one threshold reflect calibration rather than better discrimination: the mean AUC values on LIVE-VQA (0.537 to 0.621) show that no method separates poor from acceptable videos of unseen contents reliably.

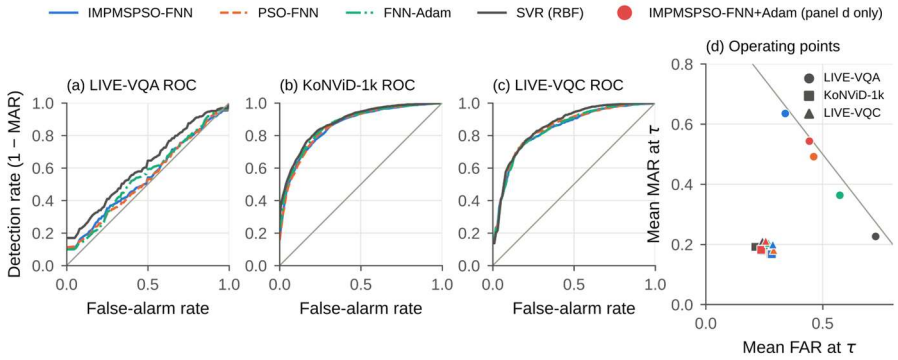


Fig. 10. Alarm detection. (a–c) ROC curves averaged over outer splits (vertical averaging on a common false-alarm grid). (d) Mean false-alarm and missed-alarm rates at the threshold τ; marker shape identifies the database and color the method.

 ©IJSRED: All Rights are Reserved

TABLE VIII
ALARM DETECTION PERFORMANCE

Method	ACC ↑	FAR ↓	MAR ↓	F1 ↑	AUC ↑
LIVE-VQA					
SVR (RBF)	0.520 ± 0.117	0.727 ± 0.336	0.227 ± 0.339	0.562 ± 0.224	0.616 ± 0.111
Random forest	0.498 ± 0.108	0.693 ± 0.304	0.269 ± 0.323	0.550 ± 0.183	0.621 ± 0.116
FNN-Adam	0.530 ± 0.108	0.573 ± 0.313	0.364 ± 0.319	0.525 ± 0.204	0.555 ± 0.119
DE-FNN	0.521 ± 0.097	0.466 ± 0.317	0.476 ± 0.319	0.462 ± 0.224	0.584 ± 0.100
PSO-FNN	0.517 ± 0.085	0.462 ± 0.339	0.492 ± 0.345	0.441 ± 0.237	0.537 ± 0.131
IMPMSPSO-FNN	0.510 ± 0.105	0.340 ± 0.344	0.635 ± 0.353	0.329 ± 0.274	0.545 ± 0.143
IMPMSPSO-FNN+Adam	0.503 ± 0.107	0.442 ± 0.310	0.544 ± 0.322	0.417 ± 0.221	0.538 ± 0.130
KoNViD-1k					
SVR (RBF)	0.799 ± 0.024	0.212 ± 0.048	0.192 ± 0.037	0.798 ± 0.026	0.881 ± 0.021
Random forest	0.759 ± 0.026	0.281 ± 0.047	0.201 ± 0.042	0.765 ± 0.032	0.848 ± 0.024
FNN-Adam	0.787 ± 0.024	0.247 ± 0.037	0.178 ± 0.045	0.791 ± 0.028	0.870 ± 0.023
DE-FNN	0.759 ± 0.026	0.272 ± 0.062	0.211 ± 0.060	0.762 ± 0.033	0.841 ± 0.026
PSO-FNN	0.778 ± 0.028	0.276 ± 0.042	0.168 ± 0.051	0.785 ± 0.033	0.861 ± 0.024
IMPMSPSO-FNN	0.775 ± 0.026	0.282 ± 0.053	0.167 ± 0.053	0.784 ± 0.030	0.860 ± 0.025
IMPMSPSO-FNN+Adam	0.791 ± 0.022	0.237 ± 0.041	0.182 ± 0.038	0.793 ± 0.025	0.873 ± 0.019
LIVE-VQC					
SVR (RBF)	0.772 ± 0.029	0.242 ± 0.044	0.212 ± 0.054	0.766 ± 0.035	0.859 ± 0.033
Random forest	0.749 ± 0.041	0.271 ± 0.062	0.227 ± 0.058	0.745 ± 0.046	0.835 ± 0.031
FNN-Adam	0.761 ± 0.033	0.267 ± 0.055	0.205 ± 0.058	0.759 ± 0.039	0.843 ± 0.030
DE-FNN	0.759 ± 0.037	0.255 ± 0.071	0.224 ± 0.077	0.752 ± 0.046	0.840 ± 0.041
PSO-FNN	0.761 ± 0.040	0.289 ± 0.065	0.181 ± 0.062	0.764 ± 0.042	0.848 ± 0.043
IMPMSPSO-FNN	0.753 ± 0.034	0.286 ± 0.075	0.199 ± 0.050	0.755 ± 0.034	0.840 ± 0.034
IMPMSPSO-FNN+Adam	0.764 ± 0.036	0.255 ± 0.063	0.212 ± 0.050	0.760 ± 0.038	0.843 ± 0.034

Threshold τ = training median of q; positive class: poor quality. Values are mean ± standard deviation over outer splits.

E. Cross-Database Generalization

To test transfer between authentic-distortion databases, we trained each method on all videos of one database (with structure and SVR settings selected by inner cross-validation on that database) and tested it on all videos of the other. Stochastic methods were run with five seeds. Every method lost accuracy relative to the within-database results (Table IX). The FNN-based methods remained below SVR in both directions. In terms of SROCC, FNN-Adam was the best FNN method in both directions.

TABLE IX
CROSS-DATABASE SROCC AND PLCC

Method	KoNViD-1k → LIVE-VQC SROCC	PLCC	LIVE-VQC → KoNViD-1k SROCC	PLCC
SVR (RBF)	0.623	0.654	0.641	0.626
FNN-Adam	0.493 ± 0.053	0.469 ± 0.064	0.614 ± 0.010	0.555 ± 0.039
PSO-FNN	0.449 ± 0.078	0.424 ± 0.094	0.574 ± 0.008	0.564 ± 0.015
IMPMSPSO-FNN	0.434 ± 0.077	0.410 ± 0.092	0.580 ± 0.026	0.577 ± 0.037
IMPMSPSO-FNN+Adam	0.420 ± 0.030	0.388 ± 0.031	0.568 ± 0.009	0.545 ± 0.037

Mean ± standard deviation over five seeds for stochastic methods; SVR is deterministic.

 ©IJSRED: All Rights are Reserved

F. Computational Cost

Table X lists the training time of the final model in each split and the prediction time per video, excluding feature extraction. All FNN optimizers trained in a few seconds or less; IMPMSPSO-FNN cost about as much as standard PSO because both use the same number of fitness evaluations, and the strategy bookkeeping adds little. Prediction with an FNN takes microseconds per video once the features are available. The VIDEVAL features themselves are far more expensive: the authors' release reports roughly 60 s per 540p video and 350 s per 1080p video [3], so feature extraction dominates the end-to-end cost for every regressor.

TABLE X
COMPUTATIONAL COST OF ALL METHODS

Method	Train LIVE-VQA	Train KoNViD-1k	Train LIVE-VQC	Predict (µs)	Params D
FNN-Adam	0.37	1.53	0.78	1.3	56
PSO-FNN	0.54	5.98	2.30	1.6	79
DE-FNN	0.69	4.57	2.74	1.3	71
IMPMSPSO-FNN	0.64	5.78	2.10	1.2	75
IMPMSPSO-FNN+Adam	0.75	6.21	2.33	1.4	75
SVR (RBF)	<0.01	0.06	0.02	44.2	—
Random forest	0.74	6.36	2.81	385.8	—

Mean training time of the final model per outer split (s), mean prediction time per video averaged over databases (µs, excluding feature extraction) and mean number of trainable FNN parameters D of the selected structures.

VII. DISCUSSION

A. Why the Optimizer Improvements Had Little Effect

Three observations help explain why the IMPMSPSO components did not translate into better quality prediction. First, the GPS initialization is only marginally more uniform than random sampling at the particle sizes used here. For $N = 50$ and D between 28 and 126, the centered L_2 discrepancy of the good point set was 0.80–0.96 times that of the median random sample, which lies inside the 5–95% range of random draws; a clear advantage appears only for $D \leq 8$ (Fig. 3c). Scrambled Sobol points also had more uniform one-dimensional marginals than the good point set at every dimension (Fig. 3d). After a few iterations the swarm has moved far from its initial positions, so a small difference in initial uniformity has little lasting effect. Second, the FNN fitness landscape contains many near-equivalent solutions, created by the $m!$ permutations of the rules and by the trade-off between rule widths and output weights, so a more uniform start or a more diverse swarm does not lead to a distinctly better region. In the repeated-seed runs, the full method had a higher median training MSE than the S_1-only variant halfway through the run on both databases (Fig. 9), so the extra diversity mainly slows convergence within a fixed budget. Third, the differences in fit between the variants are small: on KoNViD-1k and LIVE-VQC the ranking of the five optimizer variants of Table VI by mean training MSE agreed well with their ranking by test SROCC (Spearman correlation 0.90 on both), so the optimizer affects accuracy mainly through how well it fits the training data. The full method, however, did not fit better than its simpler variants; without the good point set the training MSE was even slightly lower on all three databases. On LIVE-VQA the two rankings were unrelated (correlation 0.00), which is consistent with a test error dominated by content mismatch rather than by fit.

The input representation, in contrast, had the largest effect in the study, with the same sign on all three databases. A supervised PLS projection keeps the directions of the 60-dimensional feature space that covary with quality, whereas PCA keeps the directions of largest feature variance, which need not be related to quality. For a model that can only use a handful of inputs before the product of memberships saturates [16], this choice matters more than the search strategy.

 ©IJSRED: All Rights are Reserved

B. Comparison with the Original Study

Our results do not reproduce the large advantage reported in [19]. Several differences can explain the gap. The original results come from a single split of 37 private sequences, without repetitions; its flowchart uses test error inside the fitness function, which biases the reported test performance upward [21]; its accuracy measure is not defined; and its baseline FNN training is not described. Our implementation also had to supply the five update rules and the role of α, which the original paper does not specify, so a different choice of strategies could behave differently. We therefore do not claim that the original numbers are wrong, only that the advantage of the optimizer is not visible under a complete specification and a leakage-free protocol on public data. The alarm analysis agrees with the original study in one respect: on LIVE-VQA, the database closest to the original setting, the IMPMSPSO-trained FNN lowered the false-alarm rate at the cost of more missed alarms, which is a calibration shift rather than better discrimination.

C. Implications for Remote Video Quality Monitoring

For a monitoring system, a missed alarm leaves viewers exposed to poor video, whereas a false alarm wastes operator attention or triggers an unnecessary bitrate change. The threshold-free AUC is a better basis for choosing a model than accuracy at one threshold, and the operating point should then be set from the relative cost of the two errors on validation data. On the present evidence, an SVR on VIDEVAL features is the better lightweight choice, and an FNN is worth considering mainly when its rule structure is needed for interpretation. The poor content generalization on LIVE-VQA shows that features designed for user-generated content do not transfer to synthetic transmission impairments; a deployed monitor for networked video should use features that measure packet-loss and temporal artifacts directly, or a learned model trained on representative impairments.

D. Limitations and Future Work

- We used precomputed VIDEVAL features and did not re-extract features from the videos or evaluate deep NR-VQA models, which achieve higher correlations on these databases [2], [9], [10]. Our conclusions concern the training of an FNN regressor on fixed features.
- LIVE-VQA contains only ten contents, so each split trains on eight; the large split-to-split variation limits the precision of all LIVE-VQA estimates. No database in this study uses HEVC, which the original study targeted.
- We used 20 outer splits for the authentic-distortion databases (VIDEVAL used 100) and a single evaluation budget of 10 050 fitness evaluations; larger budgets or other strategy sets might change the ranking of the optimizers.
- The structure grid was small ($m \leq 5$, $d \leq 12$) to keep nested cross-validation affordable, and the decoding ranges of Equation (7) were fixed a priori.
- The alarm threshold was set to the training median; deployment would require a threshold derived from the costs of the two error types.

Future work should test the optimizer on features designed for transmission impairments, compare adaptive strategy selection [37] with uniform random selection, and evaluate normalized or logarithmic firing strengths [16] that avoid the out-of-coverage errors observed here.

VIII. CONCLUSION

In this study, we specified the IMPMSPSO-FNN method for remote video quality prediction completely, corrected its equations, removed the test-set leakage from its fitness function and evaluated it with nested protocols on three public databases, with content-separated splits for LIVE-VQA. On KoNViD-1k and LIVE-VQC the method reached mean SROCC values of 0.719 and 0.682. These values are statistically indistinguishable from those of the same FNN trained by standard PSO, DE or Adam, and lower than those of an RBF-kernel SVR on the same features (0.767 and 0.720, significant on KoNViD-1k). Ablations showed no consistent benefit from the good-point-set initialization, the multi-strategy update or the cooperation mechanism. A supervised PLS input projection, by contrast, improved accuracy markedly compared with PCA. On LIVE-VQA, no feature-based method generalized well to unseen contents.

 ©IJSRED: All Rights are Reserved

These results suggest that, for fuzzy neural network quality predictors, effort is better spent on the input representation and on leakage-free validation than on increasingly elaborate swarm optimizers. The released code and protocol allow future optimizer proposals to be tested on the same footing.

ACKNOWLEDGMENT

The authors thank the School of Electronics and Information Engineering, Nanjing University of Information Science and Technology, for providing research facilities and academic support during this work. The authors also thank the anonymous reviewers for comments that improved the manuscript.

REFERENCES

[1] ITU-T, "Subjective video quality assessment methods for multimedia applications," Recommendation ITU-T P.910, Geneva, Switzerland, Oct. 2023.

[2] X. Min, H. Duan, W. Sun, Y. Zhu, and G. Zhai, "Perceptual video quality assessment: A survey," *Science China Information Sciences*, vol. 67, no. 11, Art. no. 211301, 2024, doi: 10.1007/s11432-024-4133-3.

[3] Z. Tu, Y. Wang, N. Birkbeck, B. Adsumilli, and A. C. Bovik, "UGC-VQA: Benchmarking blind video quality assessment for user generated content," *IEEE Transactions on Image Processing*, vol. 30, pp. 4449–4464, 2021, doi: 10.1109/TIP.2021.3072221.

[4] A. Mittal, A. K. Moorthy, and A. C. Bovik, "No-reference image quality assessment in the spatial domain," *IEEE Transactions on Image Processing*, vol. 21, no. 12, pp. 4695–4708, 2012, doi: 10.1109/TIP.2012.2214050.

[5] M. A. Saad, A. C. Bovik, and C. Charrier, "Blind prediction of natural video quality," *IEEE Transactions on Image Processing*, vol. 23, no. 3, pp. 1352–1365, 2014, doi: 10.1109/TIP.2014.2299154.

[6] J. Korhonen, "Two-level approach for no-reference consumer video quality assessment," *IEEE Transactions on Image Processing*, vol. 28, no. 12, pp. 5923–5938, 2019, doi: 10.1109/TIP.2019.2923051.

[7] D. Li, T. Jiang, and M. Jiang, "Quality assessment of in-the-wild videos," in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 2351–2359, doi: 10.1145/3343031.3351028.

[8] W. Sun, X. Min, W. Lu, and G. Zhai, "A deep learning based no-reference quality assessment model for UGC videos," in *Proc. 30th ACM Int. Conf. Multimedia*, 2022, pp. 856–865, doi: 10.1145/3503161.3548329.

[9] H. Wu, C. Chen, J. Hou, L. Liao, A. Wang, W. Sun, Q. Yan, and W. Lin, "FAST-VQA: Efficient end-to-end video quality assessment with fragment sampling," in *Computer Vision – ECCV 2022*, Lecture Notes in Computer Science, vol. 13666, 2022, pp. 538–554, doi: 10.1007/978-3-031-20068-7_31.

[10] H. Wu, E. Zhang, L. Liao, C. Chen, J. Hou, A. Wang, W. Sun, Q. Yan, and W. Lin, "Exploring video quality assessment on user generated contents from aesthetic and technical perspectives," in *Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV)*, 2023, pp. 20144–20154, doi: 10.1109/ICCV51070.2023.01843.

[11] H. Wu, Z. Zhang, W. Zhang, C. Chen, L. Liao, C. Li, Y. Gao, A. Wang, E. Zhang, W. Sun, Q. Yan, X. Min, G. Zhai, and W. Lin, "Q-Align: Teaching LMMs for visual scoring via discrete text-defined levels," in *Proc. 41st Int. Conf. Machine Learning*, PMLR, vol. 235, 2024, pp. 54015–54029.

[12] J.-S. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665–685, 1993, doi: 10.1109/21.256541.

[13] L.-X. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least-squares learning," *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp. 807–814, 1992, doi: 10.1109/72.159070.

[14] P. V. de Campos Souza, "Fuzzy neural networks and neuro-fuzzy networks: A review the main techniques and applications used in the literature," *Applied Soft Computing*, vol. 92, Art. no. 106275, 2020, doi: 10.1016/j.asoc.2020.106275.

[15] D. Wu, Y. Yuan, J. Huang, and Y. Tan, "Optimize TSK fuzzy systems for regression problems: Minibatch gradient descent with regularization, DropRule, and AdaBound (MBGD-RDA)," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 5, pp. 1003–1015, 2020, doi: 10.1109/TFUZZ.2019.2958559.

[16] Y. Cui, D. Wu, and Y. Xu, "Curse of dimensionality for TSK fuzzy neural networks: Explanation and solutions," in *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, 2021, pp. 1–8.

[17] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Networks (ICNN'95)*, vol. 4, 1995, pp. 1942–1948, doi: 10.1109/ICNN.1995.488968.

[18] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE Int. Conf. Evolutionary Computation*, 1998, pp. 69–73, doi: 10.1109/ICEC.1998.699146.

[19] J. Li, "Improved particle swarm and fuzzy neural network algorithm for remote video perceptual quality prediction" (in Chinese), *Techniques of Automation and Applications*, vol. 45, no. 9, pp. 152–155, 2026.

[20] L.-K. Hua and Y. Wang, *Applications of Number Theory to Numerical Analysis*. Berlin, Germany: Springer-Verlag; Beijing, China: Science Press, 1981.

[21] S. Kapoor and A. Narayanan, "Leakage and the reproducibility crisis in machine-learning-based science," *Patterns*, vol. 4, no. 9, Art. no. 100804, 2023, doi: 10.1016/j.patter.2023.100804.

[22] K. Seshadrinathan, R. Soundararajan, A. C. Bovik, and L. K. Cormack, "Study of subjective and objective quality assessment of video," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1427–1441, 2010, doi: 10.1109/TIP.2010.2042111.

[23] V. Hosu, F. Hahn, M. Jenadeleh, H. Lin, H. Men, T. Szirányi, S. Li, and D. Saupe, "The Konstanz natural video database (KoNViD-1k)," in *Proc. 9th Int. Conf. Quality of Multimedia Experience (QoMEX)*, 2017, pp. 1–6, doi: 10.1109/QoMEX.2017.7965673.

[24] Z. Sinno and A. C. Bovik, "Large-scale study of perceptual video quality," *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 612–627, 2019, doi: 10.1109/TIP.2018.2869673.

[25] C. Nadeau and Y. Bengio, "Inference for the generalization error," *Machine Learning*, vol. 52, no. 3, pp. 239–281, 2003, doi: 10.1023/A:1024068626366.

[26] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, 1979.

[27] C. G. Bampis, Z. Li, I. Katsavounidis, T.-Y. Huang, C. Ekanadham, and A. C. Bovik, "Towards perceptually optimized adaptive video streaming—A realistic quality of experience database," *IEEE Transactions on Image Processing*, vol. 30, pp. 5182–5197, 2021, doi: 10.1109/TIP.2021.3073294.

[28] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012, doi: 10.1109/TCSVT.2012.2221191.

[29] T. Tang, W. Zou, Z. Peng, F. Chen, and C. Jin, "Quality assessment for windowed-6DoF synthesized video based on multilayer features fusion" (in Chinese), *Acta Electronica Sinica*, vol. 53, no. 1, pp. 193–208, 2025, doi: 10.12263/DZXB.20240101.

[30] ITU-T, "Methods, metrics and procedures for statistical evaluation, qualification and comparison of objective quality prediction models," Recommendation ITU-T P.1401, Geneva, Switzerland, Jan. 2020.

 ©IJSRED: All Rights are Reserved

[31] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004, doi: 10.1023/B:STCO.0000035301.49549.88.

[32] S. Wold, M. Sjöström, and L. Eriksson, "PLS-regression: A basic tool of chemometrics," *Chemometrics and Intelligent Laboratory Systems*, vol. 58, no. 2, pp. 109–130, 2001, doi: 10.1016/S0169-7439(01)00155-1.

[33] M. Clerc and J. Kennedy, "The particle swarm—Explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002, doi: 10.1109/4235.985692.

[34] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proc. Congress on Evolutionary Computation (CEC'02)*, 2002, pp. 1671–1676, doi: 10.1109/CEC.2002.1004493.

[35] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006, doi: 10.1109/TEVC.2005.857610.

[36] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004, doi: 10.1109/TEVC.2004.826071.

[37] N. Lynn and P. N. Suganthan, "Ensemble particle swarm optimizer," *Applied Soft Computing*, vol. 55, pp. 533–548, 2017, doi: 10.1016/j.asoc.2017.02.007.

[38] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from Nature—PPSN III*, Lecture Notes in Computer Science, vol. 866, 1994, pp. 249–257, doi: 10.1007/3-540-58484-6_269.

[39] F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004, doi: 10.1109/TEVC.2004.826069.

[40] X. Ma, X. Li, Q. Zhang, K. Tang, Z. Liang, W. Xie, and Z. Zhu, "A survey on cooperative co-evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 421–441, 2019, doi: 10.1109/TEVC.2018.2868770.

[52]

[41] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh, and S. Mirjalili, "Particle swarm optimization: A comprehensive survey," *IEEE Access*, vol. 10, pp. 10031–10061, 2022, doi: 10.1109/ACCESS.2022.3142859.

[42] A. G. Gad, "Particle swarm optimization algorithm and its applications: A systematic review," *Archives of Computational Methods in Engineering*, vol. 29, no. 5, pp. 2531–2561, 2022, doi: 10.1007/s11831-021-09694-4.

[43] H. Han, X. Bai, Y. Hou, and J. Qiao, "Multitask particle swarm optimization with heterogeneous domain adaptation," *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 1, pp. 178–192, 2024, doi: 10.1109/TEVC.2023.3258491.

[44] M. Sarhani, S. Voß, and R. Jovanovic, "Initialization of metaheuristics: Comprehensive review, critical analysis, and research directions," *International Transactions in Operational Research*, vol. 30, no. 6, pp. 3361–3397, 2023, doi: 10.1111/itor.13237.

[45] J. O. Agushaka and A. E. Ezugwu, "Initialisation approaches for population-based metaheuristic algorithms: A comprehensive review," *Applied Sciences*, vol. 12, no. 2, Art. no. 896, 2022, doi: 10.3390/app12020896.

[46] I. M. Sobol', "On the distribution of points in a cube and the approximate evaluation of integrals," *USSR Computational Mathematics and Mathematical Physics*, vol. 7, no. 4, pp. 86–112, 1967.

[47] F. J. Hickernell, "A generalized discrepancy and quadrature error bound," *Mathematics of Computation*, vol. 67, no. 221, pp. 299–322, 1998.

[48] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[49] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learning Representations (ICLR)*, 2015, arXiv:1412.6980.

[50] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997, doi: 10.1023/A:1008202821328.

[51] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

 ©IJSRED: All Rights are Reserved