Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

OHRID-Net: Many-objective hyperparameter reduction in deep networks using fuzzy-preference sorting, with application to deceased-donor kidney allocation

Lirim Ashiku

lirimashiku@gmail.com

Missouri University of Science and Technology

Cihan Dagli

Missouri University of Science and Technology

Research Article

Keywords: Many-objective optimization, Neural architecture search, Fuzzy inference, Evolutionary deep learning, Decision support, Kidney allocation

Posted Date: November 13th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11517203/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

OHRID-Net: Many-objective hyperparameter reduction in deep networks using fuzzy-preference sorting, with application to deceased-donor kidney allocation

Lirim Ashiku[1]*, **Cihan Dagli**[1]

[1] Department of Engineering Management and Systems Engineering, Missouri University of Science and Technology, Rolla, MO 65409, USA

*Corresponding author: Lirim Ashiku, email: lirimashiku@gmail.com

Abstract

The selection of deep network hyperparameters and topology remains largely manual, and evolutionary architecture search usually optimizes a single accuracy objective. When one model design must support several conflicting prediction tasks while remaining compact, the search becomes a many-objective problem: most candidates are mutually non-dominated, and Pareto ranking alone cannot select among them. We propose many-**O**bjective **H**yperparameter **R**eduction **I**n **D**eep **Net**works (OHRID-Net), an evolutionary framework with two components. A fixed-length binary encoding represents the hyperparameters chosen for tuning, together with a free layer-connection topology, as one chromosome whose length follows from the search space, and which always decodes to a connected network. A fuzzy-preference sorting operator replaces crowding-distance and reference-point selection: membership functions and if–then rules convert each candidate's objectives into one preference score, so stakeholders state priorities in their own terms. Applied to national deceased-donor records from the Organ Procurement and Transplantation Network (2016–2022), OHRID-Net jointly optimized hard-to-place kidney identification, provisional acceptance, primary acceptance and trainable-parameter count. Elite networks reached median accuracies of 87.3%, 86.1% and 89.9%, with best values within 0.3–1.7 percentage points of separately optimized single-task networks, while using 8.1–24.3% of the worst-case parameter budget. In a single run, two added stakeholder rules moved the primary-acceptance distribution in the intended direction. On public benchmark problems, fuzzy sorting converged faster and more tightly than standard non-dominated sorting genetic algorithms while focusing on a preferred region, and priority rules steered it reliably toward the prioritized objective.

Keywords Many-objective optimization · Neural architecture search · Fuzzy inference · Evolutionary deep learning · Decision support · Kidney allocation

1. Introduction

Many deceased-donor kidneys recovered for transplantation in the United States are never transplanted, and national reports track this non-use as a persistent problem [1]. Kidneys declined many times are not reliably of lower quality [2], and many kidneys discarded in the United States would be transplanted elsewhere [3]. Machine-learning models that predict discard or non-recovery from donor data have accordingly become an active research area [4], [5], within a broader move towards artificial intelligence in transplantation [6]. Such decision support must serve stakeholders whose goals conflict, and its models must be designed, tuned and kept small enough to retrain as allocation policy changes. The question we address is how to search for one deep architecture family that serves several conflicting prediction tasks while keeping models small, and how stakeholders can steer that search.

Deep learning performance depends on model capacity, data quality, configuration, training technique and problem characteristics [7]. Choosing among these options is largely trial and error and requires considerable expertise [8]. Grid search is feasible for small networks but becomes computationally prohibitive as the number of hyperparameters grows. Evolutionary deep learning (EDL), in which genetic and other evolutionary algorithms search the architecture and hyperparameter space, automates this process and has produced competitive architectures [7], [9].

Most EDL studies treat architecture search as a single-objective problem in which validation accuracy is maximized. That formulation falls short for two reasons. First, wider and deeper networks need exponentially more compute, memory and energy, so capacity is itself an objective that small organizations cannot ignore [10]. Second, many real decision-support systems serve several stakeholders, each with its own prediction task. In kidney allocation, organ procurement organizations (OPOs) seek to place every recovered kidney, whereas transplant centers (TXCs) seek to match the right kidney to the right patient. Both goals aim to reduce kidney non-utilization, but they trade off against each other. A model improved for one stakeholder tends to serve another worse, and enlarging a network to recover accuracy raises its cost. Section 2.4 reviews the evidence for this conflict.

Multi-objective evolutionary algorithms such as the non-dominated sorting genetic algorithm II (NSGA-II) [11] handle two or three objectives well. As objectives increase, however, the share of non-dominated solutions grows exponentially [12]. Pareto dominance then no longer separates candidates, selection pressure weakens and convergence slows. Its successor, NSGA-III, restores diversity with a structured set of reference points, but the reference points must be specified in advance and do not naturally express *which* trade-offs a decision-maker prefers.

This paper proposes **OHRID-Net**, short for many-**O**bjective **H**yperparameter **R**eduction **I**n **D**eep **Net**works: an EDL framework that ranks non-dominated architectures with a fuzzy inference system in place of crowding distance or reference-point association. The paper makes four contributions.

1. **A search-space-driven, fixed-length encoding.** Each hyperparameter chosen for tuning, and the connections among convolution layers, are mapped to bits, so the chromosome length is determined by the problem rather than fixed by the method (Section 3.2). A designer who tunes more or fewer hyperparameters, or allows deeper blocks, obtains a longer or shorter chromosome with the same operators. The topology bits generalize VGG-, ResNet- and DenseNet-style connectivity, and a default previous-layer link guarantees that every decoded network is connected, so no evaluation budget is spent on invalid individuals.
2. **Reference-point-free fuzzy survivor selection for architecture search.** Membership functions and linguistic rules score every member of the front being split, replacing crowding distance and reference-point niching inside the generational loop. Unlike earlier fuzzy ranking schemes, which were tested on closed-form design problems or retained reference points (Section 2.5), the operator is used here where every fitness evaluation trains several networks.
3. **Model capacity as an explicit objective.** A closed-form trainable-parameter count, normalized by the worst case of the search space, lets the search trade accuracy against size without training extra models.
4. **A multi-stakeholder case study on national data.** Three allocation-decision models used by organ procurement organizations and transplant centers are optimized jointly with model size on seven years of national deceased-donor records, and the effect of stakeholder rules on the elite front is reported.

Section 2 reviews related work. Section 3 presents the encoding and the fuzzy sorting algorithm. Section 4 describes the data, objectives and experimental settings. Section 5 reports and discusses the results, and Section 6 concludes.

2. Related work

2.1 Evolutionary neural architecture search

Neural architecture search (NAS) automates network design using reinforcement learning, gradient-based relaxation or evolutionary search [13], and evolutionary NAS has been reviewed in detail [14]. Evolutionary approaches represent each architecture as a chromosome and apply selection, crossover and mutation. Genetic CNN encoded inter-layer connections of each stage as a binary string [9]. Evolutionary frameworks such as DEAP make these operators accessible [15]. Our earlier work used a single-objective genetic algorithm with a 21-bit chromosome to identify hard-to-place deceased-donor kidneys [16].

2.2 Multi-objective deep optimization

Accuracy and model size conflict, so several studies cast NAS as a multi-objective problem. NSGA-Net applied NSGA-II to trade classification error against computational cost [17] and was later extended to the multi-objective design of deep CNNs [18], and NAS has since been formulated as a family of multi-objective benchmarks [19]. LEMONADE used Lamarckian inheritance to approximate the accuracy–resource Pareto front [20], and architecture search has also targeted minimal training parameters [10]. Multi-objective hyperparameter optimization more broadly has been surveyed in [21], [22]. Most of these methods handle two or three objectives. Neuroevolution has also been combined with reinforcement learning for engineering design [23].

2.3 Many-objective optimization and preference handling

With four or more objectives, most solutions become mutually non-dominated, and dominance-based selection loses pressure [12], [24]. NSGA-III addresses diversity with reference points on a normalized hyperplane [12], [25]. Segment-based search increases the number of evaluations per generation by admitting non-elite individuals into mating selection [26]. Fuzzy dominance sorting has been used in the equitable fuzzy sorting genetic algorithm [27], and fuzzy inference has been coupled with NSGA-III to select solutions in engineering design [28]. Interactive frameworks such as DESDEO bring the decision-maker into the loop [29], and the assessment of such interactive methods has been reviewed in [30].

2.4 Conflicting objectives and the case for preference-based selection

Two objectives conflict when one can be improved only by worsening the other. Under conflict no single solution is best on every objective; the optimizer instead returns the Pareto-optimal set, in which each solution trades some objectives against others [31]. Conflict is also what makes many-objective search hard: Pareto-based algorithms tuned for two or three conflicting objectives perform very poorly as conflicting objectives are added, even with large populations [32], and later surveys identify the loss of selection pressure and the difficulty of choosing among a huge non-dominated set as the central problems of the field [24].

The objectives in kidney allocation conflict for structural reasons. Organ procurement organizations aim to place every recovered kidney, whereas transplant centers accept offers that suit their own candidates and outcome expectations. These views diverge in practice. In 78,940 U.S. transplants, about 13% of kidneys had been declined at least 100 times before acceptance, yet after adjustment for the Kidney

Donor Risk Index most outcome differences between the hardest- and easiest-to-place kidneys disappeared [2]. French centers would have transplanted more than 60% of the roughly 28,000 kidneys discarded in the United States between 2004 and 2014 [3], and U.S. discard is associated with donor risk indices, procurement biopsy findings and regional practice [33], while procurement-biopsy practices and their interpretation vary widely between transplant centers [34]. A model tuned to flag hard-to-place kidneys for the procurement organization thus pulls against models tuned to reproduce transplant-center acceptance, a tension our earlier systems analysis documented across allocation stakeholders [35]. Model size conflicts with all three tasks, because accuracy generally rises with capacity [10], [17], [20]. Improving one objective should therefore degrade at least one other, and for this reason OHRID-Net treats the problem as many-objective instead of optimizing each model separately.

When objectives conflict, a decision-maker must still choose one solution from the trade-off set [31]. Preferences can enter before the search (a priori weighting), after it (a posteriori selection) or during it (progressive articulation) [31], [24]. A priori weights must be fixed before the trade-offs are known, and a weighted sum cannot reach solutions in non-convex parts of the front [31]. A posteriori selection is impractical with four or more objectives, because most of the population is non-dominated and the front is too large to inspect [12], [32]. OHRID-Net articulates preferences progressively, through fuzzy rules applied at every generation. We chose this route for three reasons.

1. **Stakeholder preferences are linguistic and imprecise.** Statements such as “primary acceptance should be very high while model size stays acceptable” describe vague categories, and fuzzy sets were introduced to represent such categories [36]. Our rules combine such conditions with the min operator, so a candidate scores well on a rule only if it satisfies every condition of that rule.
2. **Rules convert preferences into a ranking transparently.** Mamdani inference turns if–then rules into a crisp output through fuzzification, rule firing and defuzzification [37]. Each rule can be read and audited by clinicians, unlike a weight vector whose meaning shifts with objective scaling.
3. **Fuzzy grading restores discrimination among non-dominated solutions.** Grading solutions by degree rather than by binary dominance separates members of the same front [27], and fuzzy inference has been used to rank Pareto solutions by designer preference [28].

The fuzzy superfunction picks out the best compromise for a stated set of stakeholder preferences instead of an arbitrary member of the front. When the preferences change, the compromise changes, but the algorithm does not.

2.5 Research gap

Table 1 compares survivor-selection operators for the front that does not fit into the next population. The two closest methods are the equitable fuzzy sorting genetic algorithm, which ranks solutions by an overall fuzzy activation score [27], and fuzzy-inference NSGA-III, which adds a Mamdani inference system to NSGA-III so designer preferences rank Pareto solutions [28]. Both were demonstrated on benchmark functions or closed-form engineering design. Many-objective evolutionary algorithms are rarely applied to deep architecture search, where each evaluation trains networks and objectives come from separate learning tasks. OHRID-Net brings fuzzy preference selection into that setting without reference points and adds an encoding and a capacity objective designed for it.

Table 1 Survivor-selection operators for the last admitted front

Operator	How survivors are chosen	Where preferences enter	Extra settings	Behavior as objectives increase
NSGA-II [11]	Crowding distance in objective space	Not supported; diversity only	None	Selection pressure weakens beyond three objectives [24]
NSGA-III [12], [25]	Niching around reference points on a normalized hyperplane	Indirectly, by placing reference points	Divisions p; H reference points	H grows combinatorially with the number of objectives
Equitable fuzzy sorting [27]	Fuzzy activation score defines fuzzy dominance ranks	Membership functions	Membership-function parameters	Score stays discriminating; shown on benchmarks
Fuzzy-inference NSGA-III [28]	Mamdani inference ranks Pareto solutions alongside NSGA-III	Linguistic rules from the designer	Rules plus reference points	Inherits NSGA-III reference-point growth
OHRID-Net (this work)	Mamdani superfunction ranks the split front each generation	Linguistic rules, editable between runs	Membership functions and a sparse rule base	No reference points; cost linear in population (Section 3.6)

3. The OHRID-Net framework

3.1 Search space

OHRID-Net searches over convolutional networks built from repeated optimization blocks (O-blocks) (Fig. 1). An input convolution layer feeds n identical O-blocks. Each O-block contains several convolution layers, each with batch normalization, activation, kernel size, hidden-layer width and dropout; a geometry gene sets how those layers connect. Optional deep aggregation joins the final convolution layers of successive O-blocks. Dense final layers and the chosen optimizer complete the network. Table 2 lists the hyperparameters and their encodings.

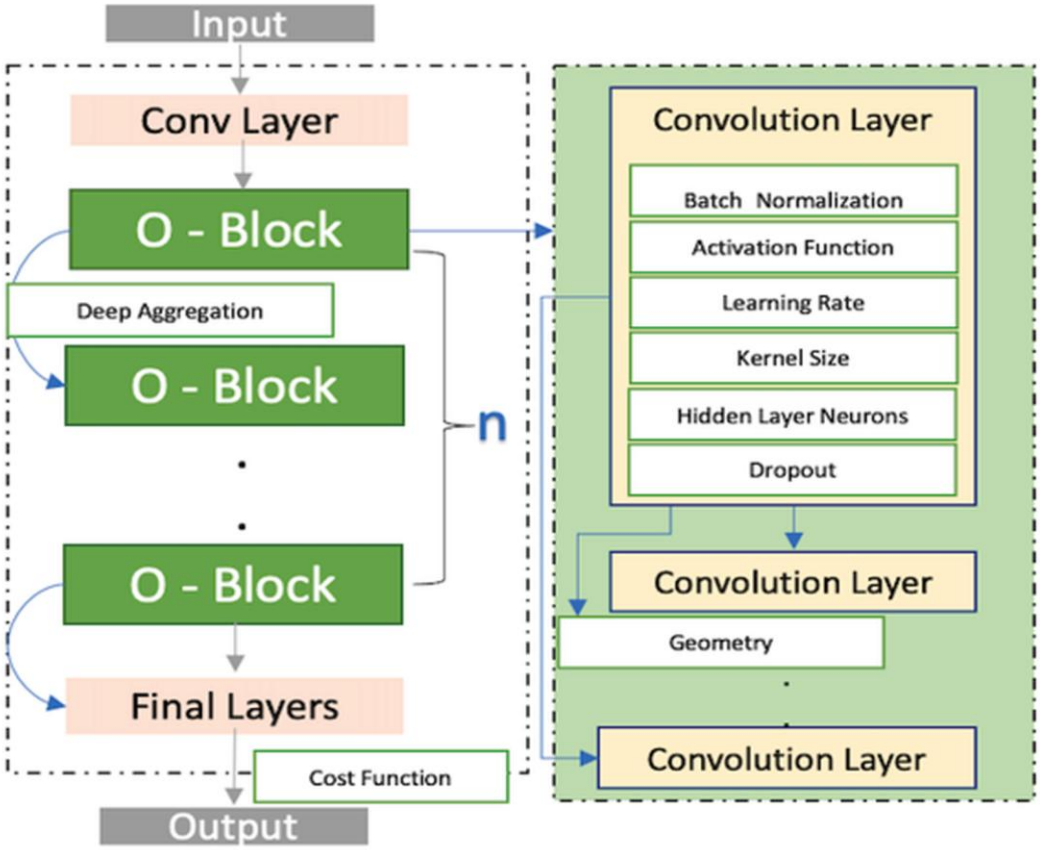


Fig. 1 O-block representation of the deep architecture. Convolution-layer hyperparameters are shared within an O-block; geometry sets the intra-block connections

Table 2 Hyperparameter search space and chromosome encoding

Gene	Values	Bits
Deep aggregation	off, on	1
Stacked convolution layers	off, on	1
Mutation type	bit-flip, swap	1
Crossover type	one-point, two-point	1
O-blocks	1–8	3
Activation	linear, sigmoid, tanh, ReLU, LeakyReLU, hard-sigmoid, swish, softplus	3
Convolution layers per block	4, 6, 8, 10	2
Learning rate	10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}	2

Gene	Values	Bits
Kernel size	3, 5	1
Optimizer	SGD, SGD-momentum, AdaGrad, RMSProp, AdaDelta, Adam, Adamax, Nadam	3
Dropout	0.2, 0.3, 0.4, 0.5	2
Hidden-layer neurons	16, 24, 32, 64	2
Geometry (layer connections)	free topology, padded to 10 layers	36
Total		**58**

3.2 Topology encoding

Our earlier single-objective study restricted geometry to four named topologies (VGG, ResNet, DenseNet, GoogLeNet) in two bits. OHRID-Net encodes connectivity directly instead, so the search can find topologies between and beyond these templates.

In an L-layer block, layer j (for $j = 3 \ldots L$) receives a group of $j - 2$ bits. Bit i of the group is 1 when layer i feeds layer j. Every layer always receives the output of the layer immediately before it, so that connection needs no bit. Because of this default, no network can be disconnected; an all-zero string decodes to a plain sequential stack. A six-layer block needs 1 + 2 + 3 + 4 = 10 bits, and the ResNet-style pattern is written 1-01-001-0001. The geometry gene needs 3, 10, 21 and 36 bits for 4, 6, 8 and 10 layers. Shorter genes are zero-padded to 36 bits, and padding bits are ignored by crossover and mutation, which keeps every chromosome at 58 bits (Fig. 2).

The chromosome length is thus set by the search space and not by the method. If Q hyperparameters are tuned, hyperparameter q has a set $V(q)$ of candidate values, and blocks may have up to $L_{\max}$ layers, the length is

$$\ell = \sum_{q=1}^{Q} b\,(q) + \frac{(L_{\max}-1)(L_{\max}-2)}{2} \quad (1)$$

Here b(q) is the number of bits needed to index the candidate values of hyperparameter q, rounded up to a whole bit (3 bits for eight optimizers, 2 bits for four learning rates). The first term covers the hyperparameter genes and the second the topology genes. For the search space of Table 2, $\ell = 22 + 36 = 58$ bits. Adding or removing a hyperparameter, widening a value range or changing $L_{\max}$ changes ℓ, while the encoding rules, crossover, mutation and decoding stay the same.

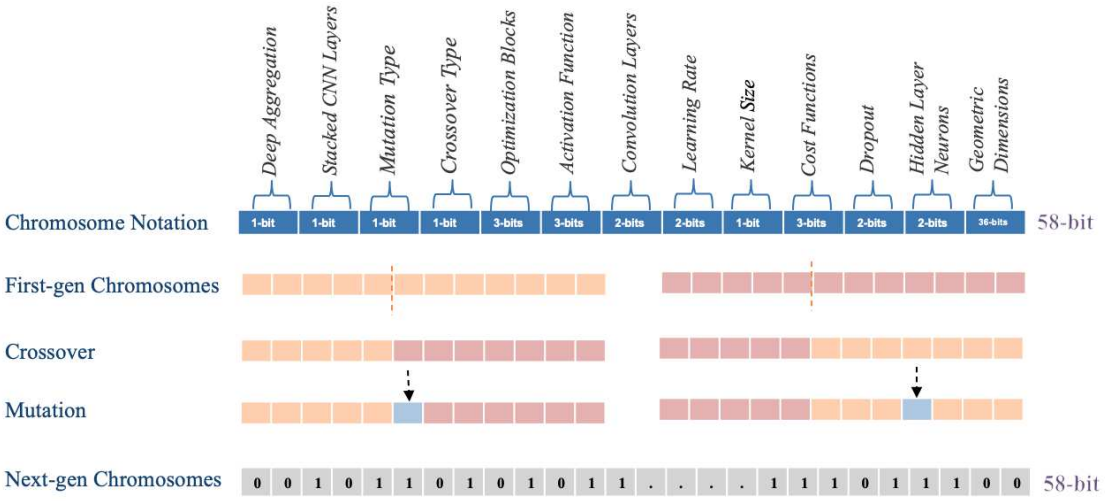


Fig. 2 The 58-bit chromosome with single-point crossover and bit-flip mutation

3.3 Objectives

OHRID-Net minimizes k objectives. For the $k-1$ prediction tasks, each objective is the test error of the decoded architecture trained on that task:

$$f_m(\mathbf{x}) = 1 - \mathrm{Acc}_m(\mathbf{x}), \quad m = 1, \ldots, k-1 \tag{2}$$

The last objective is model capacity, the trainable parameter count $P(\mathbf{x})$ normalized by the worst case $P_{\max}$ in the search space:

$$f_k(\mathbf{x}) = \frac{P(\mathbf{x})}{P_{\max}} \tag{3}$$

The count in Eq. (4) follows the layer structure. With hidden width h, kernel size K, B O-blocks of L convolution layers, stacking factor $s \in \{1, 2\}$, input width d, and c output classes:

$$P(\mathbf{x}) = (hK + h) + BsL(hK + 1)h + (dh^2 + h) + (h^2 + h) + (hc + c) \tag{4}$$

The five terms count, in order, the input convolution, the B × s × L convolution layers inside the O-blocks, the flatten-to-dense layer, the hidden dense layer and the output layer. The worst case (stacked layers, 8 O-blocks, 10 layers, $K = 5$, $h = 64$, $d = 34$, $c = 2$) gives $P_{\max} = 384 + 8 \times 2 \times 10 \times 20{,}544 + 139{,}328 + 4{,}160 + 130 = 3{,}431{,}042$ trainable parameters.

3.4 Non-dominated sorting

OHRID-Net keeps the elitist generational loop of NSGA-II [11]. A parent population P_t of size N produces an offspring population Q_t by tournament selection, crossover and mutation. The combined population $R_t = P_t \cup Q_t$ of size $2N$ is sorted into non-dominated fronts $F_1, F_2, \ldots$, and fronts are copied into P_{t+1} until one no longer fits. NSGA-II fills the remaining places by crowding distance; NSGA-III fills them by associating normalized members with H reference points, where for M objectives and p divisions

$$H = \frac{(M+p-1)!}{p!(M-1)!} \tag{5}$$

so four objectives with four divisions already need 35 reference points [12]. With many objectives, F_1 alone often exceeds N, and the choice of which first-front members survive dominates the search.

3.5 Fuzzy sorting

OHRID-Net replaces crowding distance and reference-point association with a fuzzy inference system (FIS) that scores every member of the front being split (Fig. 3). The FIS has three steps.

Fuzzification. Each objective is converted to a desirability score $u_m = 100(1 - f_m)$ on the universe of discourse [0, 100], so that larger is better. The score is described by G overlapping Gaussian membership functions:

$$\mu_{m,g}(u_m) = \exp\left(-\frac{(u_m - c_{m,g})^2}{2\sigma_{m,g}^2}\right), \quad g = 1, \ldots, G \tag{6}$$

Centers c and widths σ may be set by the decision-maker or generated from the population's minimum and maximum. We use $G = 5$ terms per objective (Low, Med, Med-High, High, Very-High; for parameters, Undesired, OK, Desired, Great, Excellent) and eight output terms (Not Feasible, Acceptable, Tolerable, Mediocre, Good, Better, Great, Perfect).

Rule evaluation. Rules of the form *IF* u_1 *is* A_1 *AND … AND* u_k *is* A_k *THEN overall is* B encode how combinations of objective levels map to overall desirability. Each rule fires with strength equal to the minimum of its antecedent memberships. Stakeholder preference enters through the rules. For example, a rule that rewards Very-High primary acceptance regardless of the other objectives makes that objective dominant.

Defuzzification. The aggregated output fuzzy set is defuzzified by centroid into a crisp superfunction value $S(\mathbf{x}) \in [0, 100]$. Members of the split front are ranked by S and the highest-ranked fill P_{t+1}.

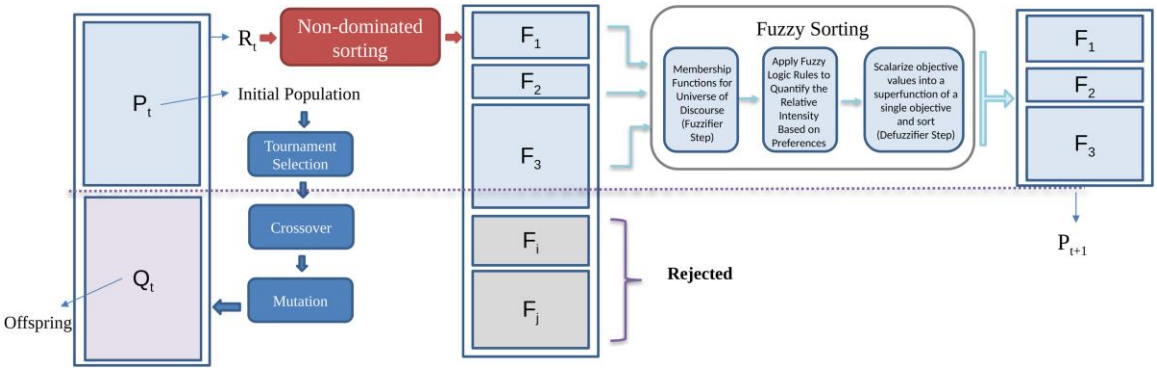


Fig. 3 OHRID-Net generation loop. Fronts enter P_{t+1} in rank order; the superfunction ranks the members of the front that does not fit entirely (here F_3), replacing crowding distance (NSGA-II) and reference-point association (NSGA-III)

Because the rule base scores every candidate on a common preference scale, it also lets non-elite individuals with a preferred profile enter mating selection, a variation that helps the search escape local optima [26]. No reference points need to be specified. Algorithm 1 summarizes the procedure.

Algorithm 1. OHRID-Net fuzzy-sorting genetic algorithm
Input: population size N; maximum generations T; objectives $f_1, \ldots, f_k$; membership functions and rule base
Output: elite population and its objective values
1: $P_0 \leftarrow N$ random chromosomes; decode, train and evaluate each

Algorithm 1. OHRID-Net fuzzy-sorting genetic algorithm

$t \leftarrow 0$
repeat
  $Q_t \leftarrow$ tournament selection, crossover and mutation on P_t
  decode, train and evaluate each member of Q_t
  $R_t \leftarrow P_t \cup Q_t$
  $(F_1, F_2, \ldots) \leftarrow$ non-dominated sorting of R_t
  $P_{t+1} \leftarrow \{\}$; $i \leftarrow 1$
  while $|P_{t+1}| + |F_i| \leq N$ **do**
    $P_{t+1} \leftarrow P_{t+1} \cup F_i$; $i \leftarrow i + 1$
  end while
  for each $\mathbf{x}$ in F_i **do**
    fuzzify $f(\mathbf{x})$, fire the rules, defuzzify to $S(\mathbf{x})$
  end for
  add the $N - |P_{t+1}|$ members of F_i with the highest S to P_{t+1}
  $t \leftarrow t + 1$
until the elite population is unchanged for 10 generations **or** $t = T$
return P_t

3.6 Computational cost

Let N be the population size, M the number of objectives (M = k), G the membership functions per objective, R the number of rules and D the number of points at which the output fuzzy set is discretized for centroid defuzzification. Non-dominated sorting of the $2N$ combined members costs $O(MN^2)$ in all three algorithms [11]. Crowding distance adds $O(MN \log N)$. NSGA-III association adds $O(MNH)$ and needs H reference points, already 35 for four objectives with four divisions (Section 3.4), and H grows combinatorially with M. Fuzzy sorting fuzzifies, fires and defuzzifies each candidate once, giving the cost in Eq. (7):

$$O\big(N(MG + RM + RD)\big) \tag{7}$$

The cost is linear in N and independent of reference points. The rule base is sparse: a full grid would need $G^M = 625$ rules for four objectives, whereas 14–16 rules were sufficient here. With $N = 30$, $M = 4$, $G = 5$, $R = 16$ and $D = 101$, fuzzy sorting needs about 10^5 elementary operations per generation. This is negligible next to the $3N$ network trainings of up to 1,000 epochs each that make up fitness evaluation, so the cost of the method lies in evaluation and not in selection (Section 5.4).

4. Experimental setup

4.1 Data

We use Organ Procurement and Transplantation Network (OPTN) records of deceased donors, waitlisted candidates and match-run offers for kidneys recovered between 1 January 2016 and 31 December 2022. Earlier years were excluded because of their high proportion of missing values. Donor, kidney/pancreas and potential-transplant-recipient files were joined on donor identifiers to build three supervised tasks, each representing one stakeholder decision in kidney allocation. Feature definitions follow the Scientific

Registry of Transplant Recipients (SRTR) data dictionary, with two derived variables: a five-level clamp-time-of-day category (00:00–04:59, 05:00–10:59, 11:00–15:59, 16:00–19:59, 20:00–23:59) and the hours from cross-clamp to final offer decision. Features for each task were selected by statistical screening and recursive feature elimination. The hard-to-place task uses 63,295 recovered deceased-donor kidneys. For every task, 80% of the records were used for training and 20% were held out as a test set, on which objectives f_1–f_3 are measured. The test sets contain approximately 12,659 kidneys for the hard-to-place task and 10,707 and 12,824 offers for provisional and primary acceptance. The majority class makes up 79.5% and 63.9% of the two acceptance test sets; keeping only the last offers per donor reduces, but does not remove, class imbalance. A classifier that always predicted the majority class would reach 79.5% and 63.9% accuracy on these test sets. The median elite member in Table 6 exceeds these rates by 6.6 and 26.0 percentage points, and every elite member summarized there exceeds them, the least accurate provisional-acceptance member only narrowly (80.8%).

Table 3 Objectives of the four-objective case study

Objective	Stakeholder decision	Target	Construction	Single-objective GA error
f_1 Hard-to-place kidney	OPO: will a recovered kidney be transplanted or discarded?	KI_DISPOSITION	Donor clinical and procurement-biopsy features (vascular changes, interstitial fibrosis), clamp-time category	6.9%
f_2 Provisional acceptance	TXC: first response to an offer	INITIAL_RESPONSE	Donor and candidate features plus provisional-response counts; last ten offers per donor kept to reduce class imbalance	7.2%
f_3 Primary acceptance	TXC: final accept or decline	OFFER_ACCEPT	Donor and candidate clinical features only, clamp-to-decision hours; final offers per donor kept to reduce class imbalance	3.91%
f_4 Model capacity	Model developer	—	Trainable parameters / P_{max}	—

The single-objective errors come from a genetic algorithm that optimized each task independently with no capacity constraint, on the same data and following the approach in [16]. They serve as an unconstrained reference for the trade-offs found by the many-objective search.

4.2 Shared input layer

Every candidate architecture must be evaluated on all three prediction tasks, so all tasks share one input width. The union of the selected features gives 34 inputs. For each task, its own features occupy their positions, and the remaining positions are filled with masking features that carry no signal and cause

negligible loss in accuracy (Fig. 4). Convolution-layer parameter counts do not depend on input width, but the flatten layer does, so a common width also keeps f_4 comparable across tasks.

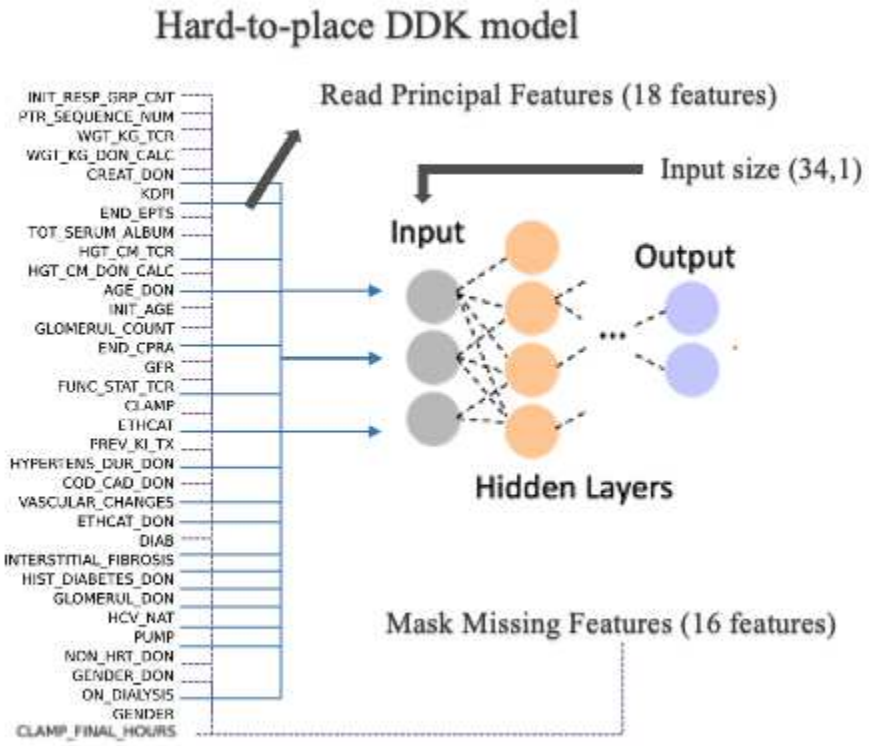


Fig. 4 Feature masking for the hard-to-place kidney task. The same scheme is applied to the two acceptance tasks

4.3 Search settings

Table 4 OHRID-Net settings

Setting	Value
Population size N	30
Maximum generations T	1,000
Objectives k	4
Stopping rule	elite population unchanged for 10 generations
Chromosome length	58 bits
Selection	binary tournament

Setting	Value
Crossover / mutation	one- or two-point / bit-flip or swap (encoded in chromosome)
Training epochs per task	up to 1,000, with early stopping and best-model checkpointing
Input membership functions	5 Gaussian per objective
Output membership functions	8 Gaussian
Rule base (run 1)	14 preference-neutral rules
Rule base (run 2)	run 1 + 2 rules prioritizing primary acceptance

The framework is written in Python. Genetic operators, encoding, decoding and fitness evaluation were implemented as custom functions, building on the DEAP toolbox [15]. The fuzzy inference system was built in Python and mirrored in MATLAB through the MATLAB Engine for visual inspection of membership functions and rules.

4.4 Use of AI-assisted tools

Claude (Anthropic), a large language model, was used to adapt the first author's dissertation chapter into journal format, edit the text, check references, extract the summary statistics in Table 6 by reading values from the published result figures (Figs. 6 and 7), and implement and run the benchmark study in Section 5.3. The authors reviewed all output and take full responsibility for the content.

5. Results and discussion

5.1 Preference-neutral run

With the 14 preference-neutral rules in Table 5, OHRID-Net converged at generation 312 and was stopped after a further ten generations without change in the elite population. Each fitness evaluation trained three networks, one per prediction task. Fig. 5 shows the fuzzified objectives, the firing of each rule for one candidate, and the output terms of the superfunction.

Table 5 Preference-neutral rule base (run 1). Inputs: f_1 hard-to-place, f_2 provisional acceptance, f_3 primary acceptance, f_4 training parameters.

Rule	f_1	f_2	f_3	f_4	Overall
1	Low	Low	Low	Undesired	Not Feasible
2	Med	Med	Med	OK	Acceptable
3	Med-High	Med-High	Med-High	Desired	Good

Rule	f_1	f_2	f_3	f_4	Overall
4	High	High	High	Great	Better
5	Very-High	Very-High	Very-High	Excellent	Perfect
6	Med-High	Med-High	Very-High	Excellent	Better
7	High	High	Very-High	Excellent	Great
8	Low	Med	Med-High	Great	Tolerable
9	Very-High	Med	Med-High	OK	Mediocre
10	Very-High	High	Med	Excellent	Better
11	Low	High	Med-High	Desired	Tolerable
12	Very-High	Very-High	Very-High	OK	Great
13	Med	High	Very-High	Excellent	Good
14	Med-High	Low	Very-High	OK	Acceptable

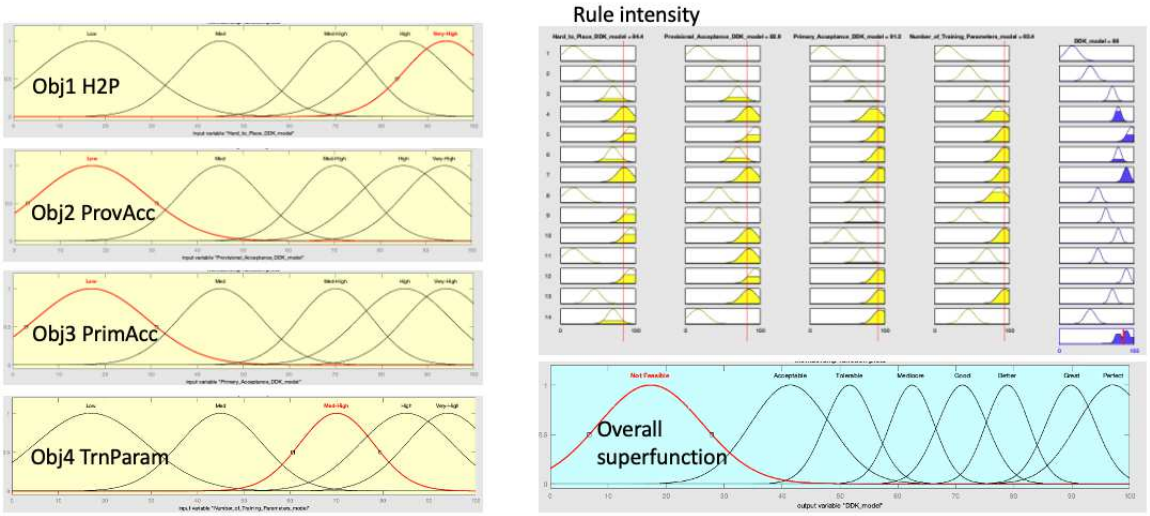


Fig. 5 Input membership functions for the four objectives (left), rule firing for one candidate (top right) and output membership functions of the superfunction (bottom right)

Fig. 6 plots the 30 elite architectures in parallel coordinates; each line is one architecture, and each axis is a desirability score (100 × accuracy for f_1–f_3; 100 × (1 − P/P_{max}) for f_4). Table 6 summarizes the front. Hard-to-place accuracy spans 80.0–92.8% (median 87.3%), provisional acceptance 81.5–91.1% (median 86.1%) and primary acceptance 85.5–94.9% (median 89.9%). Capacity scores span 75.7–91.9 (median 85.8), so elite architectures use 8.1–24.3% of the worst-case budget, about 0.28–0.83 million trainable parameters.

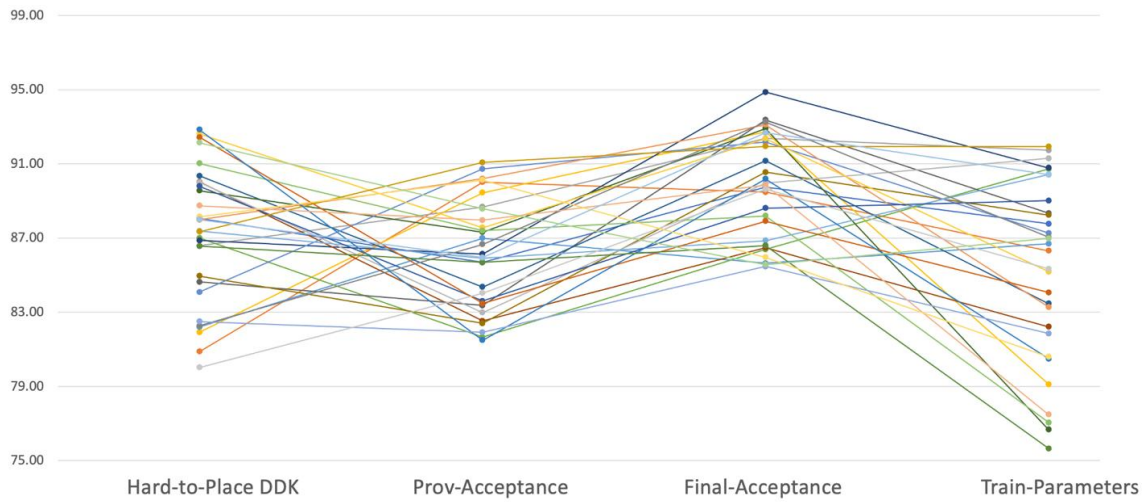


Fig. 6 Elite population after convergence with preference-neutral rules (run 1). Higher is better on every axis

The elite front shows the conflict between objectives. The best hard-to-place (92.8%), provisional-acceptance (91.1%) and primary-acceptance (94.9%) values belong to three different architectures, so each is outperformed on another axis by other members, and no member reaches the single-task optima (Table 7). Lines cross between every pair of adjacent axes in Fig. 6, which is how trade-offs appear in parallel coordinates. When run 2 favored primary acceptance, the median provisional-acceptance accuracy fell from 86.1% to 85.4% (Section 5.2). Several architectures above 92% primary-acceptance accuracy drop to the lowest capacity scores (about 76–80), while others keep both scores near or above 88, so the front offers quite different compromises. The hard-to-place and provisional-acceptance tasks also pull in opposite directions: several members above 90% on f_1 fall to about 81–84% on f_2.

Table 7 compares the best elite value on each prediction axis with the unconstrained single-objective optimum. The many-objective search comes within 0.3–1.7 points of each single-task optimum while using at most about a quarter of the worst-case parameter budget. These best values come from different architectures, and no single member attains all of them. This is where the decision-maker's preferences come in.

Table 6 Elite-population statistics, preference-neutral rules (run 1) and with rules 15–16 (run 2). Desirability scores: accuracy (%) for f_1–f_3; $100 \times (1 - P/P_{max})$ for f_4. Statistics are computed over the members resolvable in Figs. 6 and 7.

Objective	Run	Min	Q1	Median	Q3	Max	Mean
f_1 Hard-to-place	1	80.0	84.1	87.3	90.0	92.8	87.1
f_1 Hard-to-place	2	82.5	85.2	87.5	89.1	92.7	87.3
f_2 Provisional acceptance	1	81.5	83.5	86.1	88.6	91.1	86.1
f_2 Provisional acceptance	2	80.8	83.2	85.4	87.8	91.3	85.6

Objective	Run	Min	Q1	Median	Q3	Max	Mean
f_3 Primary acceptance	1	85.5	86.9	89.9	92.4	94.9	89.8
f_3 Primary acceptance	2	85.4	88.3	89.4	92.0	94.9	90.0
f_4 Capacity score	1	75.7	81.5	85.8	88.6	91.9	85.0
f_4 Capacity score	2	< 75	80.7	84.6	89.3	92.8	84.6

Table 7 Best elite accuracy (run 1) versus single-objective optimum

Task	Single-objective GA accuracy	Best OHRID-Net elite accuracy	Gap (points)
Hard-to-place	93.1%	92.8%	0.3
Provisional acceptance	92.8%	91.1%	1.7
Primary acceptance	96.1%	94.9%	1.2

5.2 Steering the search with stakeholder rules

Rules can encode stakeholder priorities. To emulate a transplant center that weighs primary acceptance most heavily, we added two rules to the base:

- Rule 15: IF f_3 is Very-High THEN overall is Acceptable.
- Rule 16: IF f_1 is Med AND f_2 is Med AND f_3 is Very-High AND f_4 is Med THEN overall is Perfect.

Rule 15 lifts architectures that excel on primary acceptance but otherwise score poorly towards "Acceptable", which pushes the search toward single-objective behavior; under centroid defuzzification it can also pull already high-scoring architectures towards that level (Section 5.3). Rule 16 rewards high primary acceptance strongly but still requires moderate performance elsewhere. Fig. 7 shows the resulting elite population.

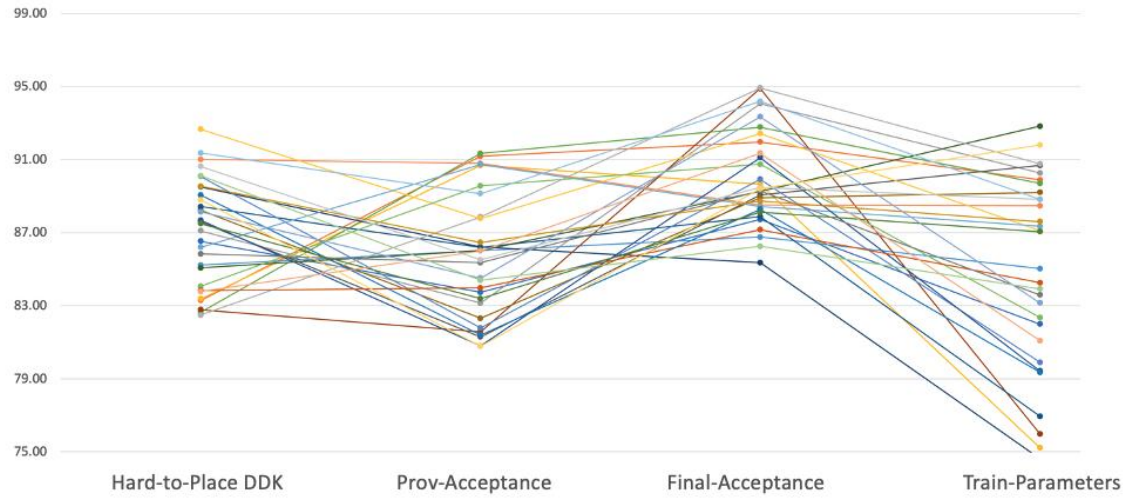


Fig. 7 Elite population after adding rules 15 and 16 (run 2)

The rules concentrated the front on the prioritized objective (Table 6). Members below 87% primary-acceptance accuracy fell from 7 to 3, members at or above 94% rose from 1 to 3, and the lower quartile rose from 86.9% to 88.3%. The maximum stayed at 94.9%, and the median moved slightly from 89.9% to 89.4% as more members clustered in the 88–90% band. The cost appeared on provisional acceptance, whose median fell from 86.1% to 85.4%; hard-to-place accuracy and capacity kept similar distributions. These shifts are only directional. A one-sided Mann–Whitney test on primary-acceptance accuracy ($p = 0.43$) and a one-sided Fisher exact test on the share of members below 87% ($p = 0.14$) do not reject the hypothesis of no change, and one run per rule base cannot separate rule effects from run-to-run variation. The change needed two rules and no new reference points, weights or code. In practice this is the main advantage of fuzzy sorting over NSGA-III association.

5.3 Benchmark check of the selection operator

The kidney data cannot be shared, and repeating the deep-network search with other selection operators was beyond the scope of this study, so fuzzy sorting was also tested on public benchmark problems. Fuzzy sorting, NSGA-II and NSGA-III were applied to the four-objective DTLZ1, DTLZ2 and DTLZ4 problems [38] with real-coded variables, simulated binary crossover and polynomial mutation, a population of 84, and 500, 300 and 700 generations respectively, over 20 independent runs per method. Fuzzy sorting used five Gaussian membership functions per objective, eight output terms and a preference-neutral rule base of 13 rules built like Table 5. Ranking the split front by an equal-weight sum of normalized objectives served as an ablation. Convergence was measured as the mean distance of the final population from the true Pareto front, and coverage of the whole front by hypervolume [24], [19]; methods were compared with the Wilcoxon rank-sum test.

Table 8 Benchmark results over 20 independent runs (medians). Convergence error: mean distance of the final population from the true Pareto front (lower is better). Hypervolume: objectives normalized by the true nadir point, reference point 1.1 (higher is better)

Problem	Metric	Fuzzy sorting	NSGA-II	NSGA-III	Weighted sum (ablation)
DTLZ1	Convergence error	8.3×10^{-4}	1.3×10^{-2}	1.5×10^{-3}	3.5×10^{-4}
DTLZ1	Hypervolume	0.13	1.33	1.37	0.13
DTLZ2	Convergence error	9.1×10^{-5}	2.7×10^{-2}	6.7×10^{-4}	5.8×10^{-8}
DTLZ2	Hypervolume	0.17	0.88	1.01	0.13
DTLZ4	Convergence error	1.3×10^{-5}	2.7×10^{-2}	2.2×10^{-4}	3.9×10^{-9}
DTLZ4	Hypervolume	0.16	0.90	1.01	0.13

Fuzzy sorting placed its solutions closer to the true Pareto front than NSGA-II on all three problems and then NSGA-III on DTLZ2 and DTLZ4 ($p < 0.001$); on DTLZ1 the difference from NSGA-III was not significant ($p = 0.06$). Ranking the split front by preference thus keeps up selection pressure where Pareto dominance alone weakens. NSGA-III, in contrast, covered far more of the front: as a preference-driven operator without a diversity mechanism, fuzzy sorting concentrates the population in a narrow region, so its hypervolume is far below that of NSGA-II and NSGA-III on every problem. The two methods serve different goals. NSGA-III approximates the whole front, whereas fuzzy sorting converges quickly on the region that stakeholders prefer, which is what the kidney study needs. The weighted-sum ablation converged even more tightly but collapsed further, covering less of the front than fuzzy sorting on DTLZ2 and DTLZ4 ($p < 10^{-5}$). A simple diversity extension, in which candidates in the same fuzzy-score band are ordered by crowding distance, recovered most of the coverage (hypervolume 1.31, 0.73 and 0.68 on DTLZ1, DTLZ2 and DTLZ4) but lost most of the convergence advantage (median convergence error 0.024, 0.017 and 0.019, worse than NSGA-II on DTLZ1 and better on DTLZ2 and DTLZ4). The balance between focus and coverage can thus be tuned, although NSGA-III remained the best choice for approximating the whole front.

Convergence speed was measured by recording the convergence error after every generation and counting the generations each run needed to reach an error of 10^{-2} and of 10^{-3}, thresholds fixed before the analysis; a run that never reached a threshold was counted at the full budget. All methods evaluate 84 new candidates per generation, so generations measure evaluation cost directly. On DTLZ2 and DTLZ4 fuzzy sorting reached both thresholds in every run, in 27–44% of the generations NSGA-III needed ($p < 10^{-6}$), while NSGA-II reached neither within the budget (Table 9, Fig. 8). On DTLZ1 fuzzy sorting was faster than NSGA-II ($p < 10^{-4}$) and reached 10^{-2} in a median of 173.5 generations against 228.5 for NSGA-III, but this difference was not significant ($p = 0.06$). Fuzzy sorting converged faster as well as more tightly. This matters most when each evaluation is expensive, as it is in deep-network search.

Table 9 Generations needed to reach a convergence error of 10^{-2} and 10^{-3} over 20 independent runs (median, with the number of runs that reached the threshold). Budget: 500, 300 and 700 generations for DTLZ1, DTLZ2 and DTLZ4

Problem	Threshold	Fuzzy sorting	NSGA-II	NSGA-III	p (fuzzy sorting vs NSGA-III)
DTLZ1	10^{-2}	173.5 (20/20)	496 (12/20)	228.5 (20/20)	0.06
DTLZ1	10^{-3}	451.5 (13/20)	not reached (0/20)	≥ 500 (6/20)	0.06
DTLZ2	10^{-2}	39 (20/20)	not reached (0/20)	101 (20/20)	6.7×10^{-8}
DTLZ2	10^{-3}	84.5 (20/20)	not reached (0/20)	252 (19/20)	6.7×10^{-8}
DTLZ4	10^{-2}	44.5 (20/20)	not reached (0/20)	102 (20/20)	8.9×10^{-7}
DTLZ4	10^{-3}	77.5 (20/20)	not reached (0/20)	293 (18/20)	2.7×10^{-7}

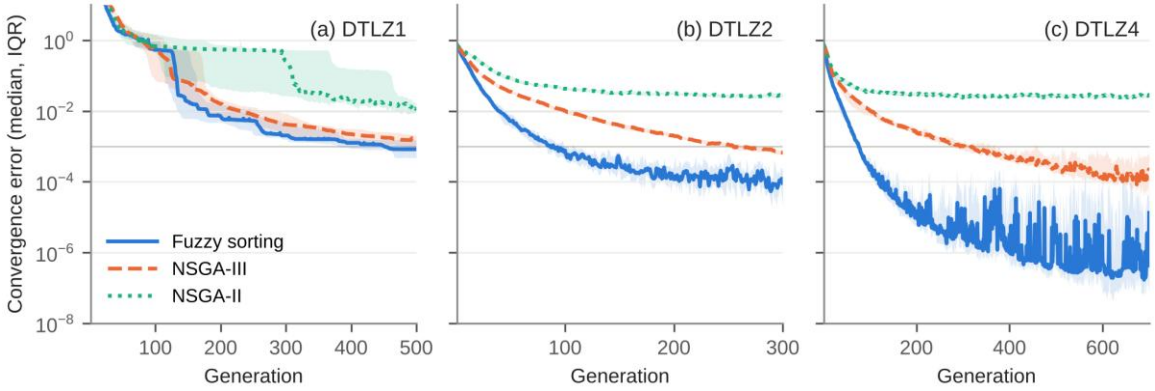


Fig. 8 Convergence error after each generation on the four-objective DTLZ problems (median over 20 runs; shaded band, interquartile range; logarithmic scale). Horizontal lines mark the thresholds 10^{-2} and 10^{-3} used in Table 9

The benchmark also clarified how priority rules behave. Rules modelled on rules 15 and 16 did not move the population towards the prioritized objective. A likely reason is that a rule whose consequent is a middle term such as “Acceptable” adds output mass at that level and, under centroid defuzzification, pulls high-scoring candidates down as well as lifting low-scoring ones. We then tested three priority rules with extreme consequents: if the prioritized objective is Very-High, the model is Perfect; if it is Very-High and the others are Med-High, the model is Perfect; and if it is Low, the model is Not Feasible. With these rules the prioritized objective moved to its best value in every test (Table 10; one-sided Wilcoxon rank-sum $p \leq 0.008$), while the median of the remaining objectives rose in three of the four tests. In the fourth (f_3 on DTLZ2) the median of the others fell as well; a median does not show how the remaining trade-off is spread across them. Steering with fuzzy rules works, provided the priority rules point to the extreme output terms.

Table 10 Effect of priority rules with extreme consequents over 20 independent runs (median across runs of the per-run median objective value; lower is better)

Problem	Prioritized objective	Neutral rules	Priority rules	Other objectives (neutral → priority)	p (one-sided)
DTLZ1	f_1	6.9×10^{-5}	≈ 0	$4.4 \times 10^{-4} \rightarrow 0.154$	1.3×10^{-5}
DTLZ2	f_1	0.579	≈ 0	$0.511 \rightarrow 0.538$	5.3×10^{-8}
DTLZ2	f_3	0.521	≈ 0	$0.540 \rightarrow 0.257$	1.2×10^{-6}
DTLZ4	f_2	0.670	≈ 0	$0.303 \rightarrow 0.706$	8.4×10^{-3}

5.4 Discussion and practical implications

In this search space, model capacity can be reduced at little cost. The best elite architectures come within 0.3–1.7 points of separately optimized single-task networks (Table 7), yet every elite member uses 8.1–24.3% of the worst-case parameter budget (Table 6). Adding capacity as a fourth objective costs little accuracy, which is useful wherever models are retrained often or run on modest hardware.

Fuzzy sorting also gives every member of the split front a distinct superfunction score, so selection still discriminates when most of the population is mutually non-dominated, and it does so without reference points (Table 1, Section 3.6). In addition, the rule base makes the selection criterion easy to read. A rule such as “if primary acceptance is very high, the model is acceptable” can be read, challenged and edited by clinicians and allocation committees without machine-learning expertise.

For kidney allocation, the framework yields one compact architecture family that serves both organ procurement organizations (identifying kidneys at risk of non-use early enough for accelerated placement) and transplant centers (anticipating provisional and primary acceptance). Sharing one design across the three tasks simplifies deployment and retraining, and stakeholders can rebalance the trade-off by changing rules rather than commissioning a new search algorithm.

5.5 Scope and limitations

The main conclusions rest on properties of a converged elite front rather than on fine-grained comparisons. Below we note what the design controls and what it leaves open.

Repeated runs. Each rule base was run once. Elitism and the stopping rule (no change in the elite population for ten generations) ensure that each reported front is converged, so the two main findings, small accuracy gaps to single-task optima and large reductions in model size, are properties of that front. Only the rule-steering result compares two runs, so we present it as an illustration of the mechanism, and Section 5.2 reports that the shift is directional rather than statistically established. The benchmark in Section 5.3, with 20 runs per setting, confirms that priority rules with extreme consequents steer the search reliably. Multiple seeds per rule base would quantify run-to-run variation. Table 6 is computed over 25–29 of the 30 elite members per objective because some members overlap in Figs. 6 and 7; the overlapping members coincide with plotted values, so their omission has little effect on the reported ranges and quartiles.

Comparison with other selection operators. OHRID-Net changes only the survivor-selection step: encoding, variation operators and non-dominated sorting are identical to NSGA-II. Any difference in behavior comes from fuzzy sorting, whose properties relative to crowding distance and reference-point

niching are established analytically in Table 1 and Section 3.6 and checked empirically on public benchmark problems in Section 5.3. Separately optimized single-task networks provide an empirical upper reference for each prediction objective (Table 7). A head-to-head comparison on the kidney search space itself, and refinement of the diversity extension tested in Section 5.3, are the natural next steps.

Performance measures. Test-set accuracy was chosen so that the many-objective results are directly comparable with the single-objective baselines, and keeping the last offers per donor reflects the decision points that matter in allocation. Because the acceptance test sets remain imbalanced, Section 4.1 reports the majority-class rates against which these accuracies should be read. The least accurate provisional-acceptance members add little predictive value over that rate; they sit at the low-accuracy end of the front, and a decision-maker would select from the more accurate part of the front. The framework does not depend on the metric. Area under the ROC curve, F1 score or calibration error can replace accuracy as an objective without any change to the algorithm. Masking features give every task the same input width, which is required for a fair capacity objective, and carry no signal by construction.

Data and generalizability. The models draw on national registry data covering every U.S. procurement organization and transplant center over seven years, including periods before and after the March 2021 shift from donation service areas to distance-based kidney allocation, so the trained models reflect both allocation regimes. Records before 2016 were excluded because of high missingness, and features were screened statistically and by recursive feature elimination; residual inconsistencies in registry coding, such as cold ischemia time for non-used kidneys, remain a known property of the data. The optimization framework is not specific to transplantation. Applying it elsewhere requires new objectives and a rule base elicited from that domain's stakeholders.

6. Conclusion and future work

OHRID-Net shows that many-objective evolutionary search can design deep networks that serve several conflicting prediction tasks while limiting model size. Its fixed-length binary encoding sizes the chromosome from the hyperparameters chosen for tuning (58 bits for the search space studied here) and always decodes to a connected network. Its fuzzy sorting operator ranks non-dominated architectures through membership functions and rules, keeping selection discriminating when most of the population is mutually non-dominated and letting stakeholders steer the search without reference points or code changes. On public four-objective benchmark problems it converged faster and more tightly than NSGA-II and NSGA-III while concentrating on a preferred region of the front, and priority rules with extreme consequents steered it reliably toward the prioritized objective. On national deceased-donor kidney data with four objectives, the elite architectures came within 0.3–1.7 points of each unconstrained single-task optimum while using 8.1–24.3% of the worst-case parameter budget, and two added rules moved the lower quartile of primary-acceptance accuracy from 86.9% to 88.3%, the objective a transplant center would priorities.

Future work will (i) refine the diversity extension of Section 5.3 and compare fuzzy sorting with NSGA-II, NSGA-III and MOEA/D on the kidney search space over repeated runs; (ii) learn membership-function parameters and rule weights from elicited stakeholder preferences; (iii) reduce search cost with surrogate fitness models or weight sharing; (iv) package the encoding, operators and fuzzy sorter as an open framework for other datasets; and (v) extend the approach to allocation models for other organs and integrate explainability into the fitness evaluation.

Statements and Declarations

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Competing interests The authors have no relevant financial or non-financial interests to disclose.

Ethics approval This study is a secondary analysis of de-identified registry data obtained on request from the Organ Procurement and Transplantation Network; institutional review board approval was not required.

Consent to participate and consent for publication Not applicable.

Data availability The data are national registry data from the Organ Procurement and Transplantation Network (OPTN) and are available to qualified researchers on request from the OPTN under a data use agreement; the authors are not permitted to redistribute them. Match-run data are subject to additional restrictions.

Code availability Code for the encoding, genetic operators and fuzzy sorting, and the code and raw results of the benchmark study (Section 5.3), are available from the corresponding author on reasonable request.

Author contributions Lirim Ashiku: conceptualization, methodology, software, data curation, formal analysis, investigation, visualization, writing – original draft. Cihan Dagli: conceptualization, methodology, supervision, writing – review and editing. All authors read and approved the final manuscript.

References

1. Lentine, K.L., Smith, J.M., Miller, J.M., Bradbrook, K., Larkin, L., Weiss, S., Handarova, D.K., Temple, K., Israni, A.K., Snyder, J.J.: OPTN/SRTR 2021 annual data report: kidney. Am. J. Transplant. **23**(2S1), S21–S120 (2023). https://doi.org/10.1016/j.ajt.2023.02.004
2. King, K.L., Chaudhry, S.G., Ratner, L.E., Cohen, D.J., Husain, S.A., Mohan, S.: Declined offers for deceased donor kidneys are not an independent reflection of organ quality. Kidney360 **2**(11), 1807–1818 (2021). https://doi.org/10.34067/KID.0004052021
3. Aubert, O., Reese, P.P., Audry, B., Bouatou, Y., Raynaud, M., Viglietti, D., Legendre, C., Glotz, D., Empana, J.-P., Jouven, X., Lefaucheur, C., Jacquelinet, C., Loupy, A.: Disparities in acceptance of deceased donor kidneys between the United States and France and estimated effects of increased US acceptance. JAMA Intern. Med. **179**(10), 1365 (2019). https://doi.org/10.1001/jamainternmed.2019.2322
4. Pettit, R.W., Marlatt, B.B., Miles, T.J., Uzgoren, S., Corr, S.J., Shetty, A., Havelka, J., Rana, A.: The utility of machine learning for predicting donor discard in abdominal transplantation. Clin. Transplant. **37**(5), e14951 (2023). https://doi.org/10.1111/ctr.14951
5. Sageshima, J., Than, P., Goussous, N., Mineyev, N., Perez, R.: Prediction of high-risk donors for kidney discard and nonrecovery using structured donor characteristics and unstructured donor narratives. JAMA Surg. **159**(1), 60–68 (2024). https://doi.org/10.1001/jamasurg.2023.4679
6. Peloso, A., Moeckli, B., Delaune, V., Oldani, G., Andres, A., Compagnon, P.: Artificial intelligence: present and future potential for solid organ transplantation. Transpl. Int. **35**, 10640 (2022). https://doi.org/10.3389/ti.2022.10640

7. Zhan, Z.-H., Li, J.-Y., Zhang, J.: Evolutionary deep learning: a survey. Neurocomputing **483**, 42–58 (2022). https://doi.org/10.1016/j.neucom.2022.01.099
8. Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, J., Ullmann, T., Becker, M., Boulesteix, A.-L., Deng, D., Lindauer, M.: Hyperparameter optimization: foundations, algorithms, best practices, and open challenges. WIREs Data Min. Knowl. Discov. **13**(2), e1484 (2023). https://doi.org/10.1002/widm.1484
9. Xie, L., Yuille, A.: Genetic CNN. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017). https://doi.org/10.1109/ICCV.2017.154
10. Gottapu, R.D., Dagli, C.H.: Efficient architecture search for deep neural networks. Procedia Comput. Sci. **168**, 19–25 (2020). https://doi.org/10.1016/j.procs.2020.02.246
11. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002). https://doi.org/10.1109/4235.996017
12. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. IEEE Trans. Evol. Comput. **18**(4), 577–601 (2014). https://doi.org/10.1109/TEVC.2013.2281535
13. Elsken, T., Metzen, J.H., Hutter, F.: Neural architecture search: a survey. J. Mach. Learn. Res. **20**(55), 1–21 (2019)
14. Liu, Y., Sun, Y., Xue, B., Zhang, M., Yen, G.G., Tan, K.C.: A survey on evolutionary neural architecture search. IEEE Trans. Neural Netw. Learn. Syst. **34**(2), 550–570 (2023). https://doi.org/10.1109/TNNLS.2021.3100554
15. Kim, J., Yoo, S.: Software review: DEAP (Distributed Evolutionary Algorithm in Python) library. Genet. Program. Evolvable Mach. **20**(1), 139–142 (2019). https://doi.org/10.1007/s10710-018-9341-4
16. Ashiku, L., Dagli, C.: Identify hard-to-place kidneys for early engagement in accelerated placement with a deep learning optimization approach. Transplant. Proc. **55**(1), 38–48 (2023). https://doi.org/10.1016/j.transproceed.2022.12.005
17. Lu, Z., Whalen, I., Boddeti, V., Dhebar, Y., Deb, K., Goodman, E., Banzhaf, W.: NSGA-Net: neural architecture search using multi-objective genetic algorithm. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '19), pp. 419–427 (2019). https://doi.org/10.1145/3321707.3321729
18. Lu, Z., Whalen, I., Dhebar, Y., Deb, K., Goodman, E., Banzhaf, W., Boddeti, V.N.: Multiobjective evolutionary design of deep convolutional neural networks for image classification. IEEE Trans. Evol. Comput. **25**(2), 277–291 (2021). https://doi.org/10.1109/TEVC.2020.3024708
19. Lu, Z., Cheng, R., Jin, Y., Tan, K.C., Deb, K.: Neural architecture search as multiobjective optimization benchmarks: problem formulation and performance assessment. IEEE Trans. Evol. Comput. **28**(2), 323–337 (2024). https://doi.org/10.1109/TEVC.2022.3233364
20. Elsken, T., Metzen, J.H., Hutter, F.: Efficient multi-objective neural architecture search via Lamarckian evolution. In: International Conference on Learning Representations (ICLR) (2019)
21. Morales-Hernández, A., Van Nieuwenhuyse, I., Rojas Gonzalez, S.: A survey on multi-objective hyperparameter optimization algorithms for machine learning. Artif. Intell. Rev. **56**(8), 8043–8093 (2023). https://doi.org/10.1007/s10462-022-10359-2
22. Karl, F., Pielok, T., Moosbauer, J., Pfisterer, F., Coors, S., Binder, M., Schneider, L., Thomas, J., Richter, J., Lang, M., Garrido-Merchán, E.C., Branke, J., Bischl, B.: Multi-objective hyperparameter optimization in machine learning—an overview. ACM Trans. Evol. Learn. Optim. **3**(4), 16 (2023). https://doi.org/10.1145/3610536

23. Radaideh, M.I., Du, K., Seurin, P., Seyler, D., Gu, X., Wang, H., Shirvan, K.: NEORL: NeuroEvolution Optimization with Reinforcement Learning—applications to carbon-free energy systems. SSRN Electron. J. (2022). https://doi.org/10.2139/ssrn.4305368
24. Li, B., Li, J., Tang, K., Yao, X.: Many-objective evolutionary algorithms: a survey. ACM Comput. Surv. **48**(1), 1–35 (2015). https://doi.org/10.1145/2792984
25. Jain, H., Deb, K.: An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: handling constraints and extending to an adaptive approach. IEEE Trans. Evol. Comput. **18**(4), 602–622 (2014). https://doi.org/10.1109/TEVC.2013.2281534
26. Li, M., Yang, S., Li, K., Liu, X.: Evolutionary algorithms with segment-based search for multiobjective optimization problems. IEEE Trans. Cybern. **44**(8), 1295–1313 (2014). https://doi.org/10.1109/TCYB.2013.2282503
27. Jamwal, P.K., Abdikenov, B., Hussain, S.: Evolutionary optimization using equitable fuzzy sorting genetic algorithm (EFSGA). IEEE Access **7**, 8111–8126 (2019). https://doi.org/10.1109/ACCESS.2018.2890274
28. Li, J., Li, Y., Wang, Y.: Fuzzy inference NSGA-III algorithm-based multi-objective optimization for switched reluctance generator. IEEE Trans. Energy Convers. **36**(4), 3578–3581 (2021). https://doi.org/10.1109/TEC.2021.3099961
29. Misitano, G., Saini, B.S., Afsar, B., Shavazipour, B., Miettinen, K.: DESDEO: the modular and open source framework for interactive multiobjective optimization. IEEE Access **9**, 148277–148295 (2021). https://doi.org/10.1109/ACCESS.2021.3123825
30. Afsar, B., Miettinen, K., Ruiz, F.: Assessing the performance of interactive multiobjective optimization methods: a survey. ACM Comput. Surv. **54**(4), 85 (2021). https://doi.org/10.1145/3448301
31. Miettinen, K.: Nonlinear Multiobjective Optimization. Kluwer Academic Publishers, Boston (1999). https://doi.org/10.1007/978-1-4615-5563-6
32. Purshouse, R.C., Fleming, P.J.: On the evolutionary optimization of many conflicting objectives. IEEE Trans. Evol. Comput. **11**(6), 770–784 (2007). https://doi.org/10.1109/TEVC.2007.910138
33. Mohan, S., Chiles, M.C., Patzer, R.E., Pastan, S.O., Husain, S.A., Carpenter, D.J., Dube, G.K., Crew, R.J., Ratner, L.E., Cohen, D.J.: Factors leading to the discard of deceased donor kidneys in the United States. Kidney Int. **94**(1), 187–198 (2018). https://doi.org/10.1016/j.kint.2018.02.016
34. Lentine, K.L., Fleetwood, V.A., Caliskan, Y., Randall, H., Wellen, J.R., Lichtenberger, M., Dedert, C., Rothweiler, R., Marklin, G., Brockmeier, D., Schnitzler, M.A., Husain, S.A., Mohan, S., Kasiske, B.L., Cooper, M., Mannon, R.B., Axelrod, D.A.: Deceased donor procurement biopsy practices, interpretation, and histology-based decision-making: a survey of US kidney transplant centers. Kidney Int. Rep. **7**(6), 1268–1277 (2022). https://doi.org/10.1016/j.ekir.2022.03.021
35. Threlkeld, R., Ashiku, L., Canfield, C., Shank, D.B., Schnitzler, M.A., Lentine, K.L., Axelrod, D.A., Battineni, A.C.R., Randall, H.B., Dagli, C.: Reducing kidney discard with artificial intelligence decision support: the need for a transdisciplinary systems approach. Curr. Transplant. Rep. **8**(4), 263–271 (2021). https://doi.org/10.1007/s40472-021-00351-0
36. Zadeh, L.A.: Fuzzy sets. Inf. Control **8**(3), 338–353 (1965). https://doi.org/10.1016/S0019-9958(65)90241-X
37. Mamdani, E.H., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. Int. J. Man-Mach. Stud. **7**(1), 1–13 (1975). https://doi.org/10.1016/S0020-7373(75)80002-2

38. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable multi-objective optimization test problems. In: Proceedings of the 2002 Congress on Evolutionary Computation (CEC'02), pp. 825–830 (2002). https://doi.org/10.1109/CEC.2002.1007032